# SIMPOL Language Reference Manual

## The Nitty Gritty

**Neil Robinson**
**Anthony K. Dyer**

# SIMPOL Language Reference Manual: The Nitty Gritty

by Neil Robinson and Anthony K. Dyer

# Table of Contents

# List of Tables

# Foreword

This reference book contains the complete information about the operators, intrinsic functions, system functions, supplied data types, and errors that can occur.

# Part I. The Core SIMPOL Language

This part of the book contains the core elements of the SIMPOL programming language. All of the items in this part of the book are built into the core SIMPOL language shared library. The next part of this book concerns itself with library components that provide additional functionality.

# Table of Contents

# Chapter 1. Intrinsic Functions

# Introduction

In SIMPOL intrinsic functions are functions that begin with the dot operator, deal with one or more value types, returns a value type, the parameters do not have names and they must all be specified, and there is no associated function object. They begin with a dot in order to ensure that no user-defined function could be named the same so as to allow for future expansion in this area without fear of colliding with existing user-defined function names.

# Compression Functions

This section contains the intrinsic functions that are used for compressing and decompressing strings and blobs.

## .compress1()

### Prototype

```
.compress1( <content> )
```

### Return types

.nul, .inf, string|blob

### Description

Returns a string or blob compressed to the same type. In the case of a string, if all of the characters are single-byte characters, then the resulting compressed string will also be single-byte characters (this is of importance if the compressed string is being output since it can be output as 1 byte per character).

### Parameters

#### *content*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.inf` |
| *blob* | The result will be a blob compressed using the compress1 algorithm |
| *string* | The result will be a string compressed using the compress1 algorithm |

## .decompress1()

### Prototype

```
.decompress1( <content> )
```

## Return types

.nul, .inf, string|blob

## Description

Returns a string or blob decompressed from the same type.

## Parameters

### *content*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.inf` |
| *blob* | The result will be the original blob that was compressed using the compress1 algorithm |
| *string* | The result will be the original string that was compressed using the compress1 algorithm |

# Conversion Functions

This section contains the intrinsic functions that are used for converting values from one type to another. This includes functions for converting strings to integers or numbers, integers and numbers to strings, integers to characters and characters to integers.

# .char()

## Prototype

`.char( <charval> )`

## Return types

.nul, .inf, string

## Description

Returns a string containing the character with the specified numeric value.

## Parameters

### *charval*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.inf` |
| *integer* | The value to get the character for |

# .charval()

## Prototype

```
.charval( <text> )
```

## Return types

.nul, .inf, integer

## Description

Returns the numeric value of the first character of a string

## Parameters

### *text*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf |
| *string* | The string for which the numeric value of the first character should be returned. If this is the empty string then the return value is .nul. |

# .deintegerize()

## Prototype

```
.deintegerize( <value> )
```

## Return types

.nul, .inf, string

## Description

This converts any non-negative integer into a unique string and is the reverse of .integerize(). If *value* is equal to .nul or .inf then the result is the same.

## Parameters

### *value*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf |
| *(non-negative) integer* | The integer will be converted to a unique string |

# .integerize()

## Prototype

```
.integerize( <value> )
```

## Return types

.nul, .inf, integer

## Description

This converts any string into a unique non-negative integer and is the reverse of `.deintegerize()`. If *`value`* is equal to `.nul` or `.inf` then the result is the same.

## Parameters

### *`value`*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *`.nul`* | The result will be `.nul` |
| *`.inf`* | The result will be `.inf` |
| *`string`* | The string will be converted to a unique non-negative integer |

# .lcase()

## Prototype

```
.lcase( <value> )
```

## Return types

.nul, .inf, string

## Description

This function converts a string to lower case. This function supports Unicode version 3.2 including support for special folding characters such as final sigma in Greek. If *`value`* is equal to `.nul` or `.inf` then the result is the same. Please note that in some situations it is possible for a string to grow in length when converted to lower case.

## Parameters

### *`value`*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *`.nul`* | The result will be `.nul` |
| *`.inf`* | The result will be `.inf` |

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *string* | The string will be converted to lower case. |

# .tcase()

## Prototype

```
.tcase( <value> )
```

## Return types

.nul, .inf, string

## Description

This function converts a string to title case. This function supports Unicode version 3.2 including support for special folding characters such as final sigma in Greek. If *value* is equal to .nul or .inf then the result is the same. Please note that in some situations it is possible for a string to increase or decrease in length when converted to title case.

## Parameters

### *value*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf |
| *string* | The string will be converted to title case. |

# .toblob()

## Prototype

```
.toblob( <string> )
```

## Return types

.nul, .inf, blob

## Description

Represents a string value as a blob.

## Parameters

### *value*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.inf* | The result will be `.inf`, unless the second parameter is `.nul` or `.inf` |
| *string* | The string value to represent as a blob |

# .tostr()

## Prototype

```
.tostr( <value> , <base> )
```

## Return types

.nul, .inf, string

## Description

Represents a numeric value as a string.

## Parameters

### *value*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.inf`, unless the second parameter is `.nul` or `.inf` |
| *integer* | The value to represent as a string |
| *number* | The value to represent as a string. If there is any fractional part then a full stop ('.') is used as a decimal point character. If there is a block of recurring digits then the second copy of this block of digits is placed in square brackets and the string then terminated, for example 1/3 is shown in base 10 as '0.3[3]'. There is no provision for terminating the fractional part if it is very long; trying to represent a number with a large number of digits in the fractional part may raise an error if the string is too long. |

### *base*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.nul` |
| *integer* | The base to use, for example the value 10 specifies a decimal representation. Values less than 2 will cause an error. There is also a platform and version specific upper limit which is not less than 36. |

# .toval()

## Prototype

```
.toval( <value> , <ignore> , <base> )
```

## Return types

.nul, .inf, integer, number

## Description

Converts a string representation of a numeric value into that value, with the option of ignoring some characters. Any invalid characters in the input string will cause an error.

## Parameters

### *value*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | the result to be .nul |
| *string* | The string to extract a numeric value from. A full stop ('.') is assumed to be a decimal point character and square brackets are assumed to be used to indicate a recurring block of digits in the fractional part of the number. |

### *ignore*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | No characters will be ignored |
| *.inf* | The result will be .nul |
| *string* | The characters to be ignored |

### *base*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .nul |
| *integer* | The base to use, for example the value 10 specifies that the string is to be interpretted as a decimal representation of a numeric value. Values less than 2 will cause an error. There is also a platform and version specific upper limit which is not less than 36. |

# .ucase()

## Prototype

.ucase( <*value*> )

## Return types

.nul, .inf, string

## Description

This function converts a string to upper case. This function supports Unicode version 3.2 including support for special folding characters such as final sigma in Greek. If *`value`* is equal to `.nul` or `.inf` then the result is the same. Please note that in some situations it is possible for a string to increase or decrease in length when converted to upper case.

## Parameters

### *`value`*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *`.nul`* | The result will be `.nul` |
| *`.inf`* | The result will be `.inf` |
| *`string`* | The string will be converted to upper case. |

# Numeric Functions

This section contains the intrinsic functions that are used for working with numbers.

# .fix()

## Prototype

`.fix( <value> , <denominator> )`

## Return types

.nul, .inf, integer, number

## Description

Returns the result of rounding a numeric value to the nearest multiple of a specified fraction. If the second parameter is 1, then the value will be rounded to the nearest whole number value, with values exactly between being rounded away from zero. That means that if the value is `0.5` and the denominator is `1`, then the return value will be `1`. If the value is `-0.5` and the denominator is `1`, then the return value will be `-1`. This function can be used to return a value to the nearest whole number as just described, or to the nearest half, using a denominator of 2, nearest fourth, using a denominator of 4, nearest 10th, using a denominator of 10, and so on.

## Parameters

### *`value`*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *`.nul`* | The result will be `.nul` |
| *`.inf`* | The result will be `.inf`, unless another parameter causes the result to be `.nul` |

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *integer* | The result will be the value of this parameter, unless another parameter causes the value to be .nul or .inf |
| *number* | The number to be rounded to the nearest fraction |

### *denominator*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be the value of the first parameter |
| *integer* | The denominator of the fraction to round to, for example the value 1000 will cause rounding to three decimal digits |

# .ipower()

## Prototype

.ipower( <*operand*> , <*power*> )

## Return types

.nul, .inf, integer

## Description

Returns the result of raising the *operand* to *power*. If either *operand* or *power* is .nul then the result is .nul. Otherwise, if either *operand* or *power* is .inf then the result is .inf. Otherwise, *power* must be non-negative. If *power* is equal to zero then the result is equal to one. If *operand* is equal to one then the result is equal to one. If *operand* is equal to zero and *power* is not equal to zero then the result is zero. Otherwise the result is *operand* to the power of *power*, in the normal way.

## Parameters

### *operand*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf, unless another parameter causes the result to be .nul |
| *integer* | The result will be the value of this parameter raised to the power of the second parameter, unless the second parameter causes the value to be .nul or .inf |

### *power*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf unless the first parameter causes it to be .nul or .inf |

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *integer* | The first parameter raised to the power represented by this parameter unless the first parameter causes it to be .nul |

# .ipowermod()

## Prototype

.ipowermod( <*operand*> , <*power*> , <*modulus*> )

## Return types

.nul, .inf, integer

## Description

Returns the remainder of result reached by raising the *operand* to *power* and then dividing by the modulus value. If any of the parameters is equal to .nul then the result is .nul. If *modulus* is .inf then the result is the same as .ipower(operand, power). Otherwise, if either *operand* or *power* is .inf then the result is .inf. Barring those cases, *power* must be non-negative and *modulus* must be positive. If *modulus* is equal to one then the result is zero. If *power* is equal to zero then the result is equal to one. If *operand* is equal to zero or one then the result is equal to zero or one respectively. Otherwise the result is *operand* to the power of *power* mod *modulus*, in the normal way.

## Parameters

### *operand*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf, unless another parameter causes the result to be .nul |
| *integer* | The result will be the value of this parameter raised to the power of the second parameter modulus the third parameter, unless another parameter causes the value to be .nul or .inf |

### *power*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf unless another parameter causes it to be .nul |
| *integer* | The first parameter raised to the power represented by this parameter modulus the third parameter unless another parameter causes it to be .nul or .inf |

### *modulus*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be the same as .ipower(operand, power) |

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *integer* | The first parameter raised to the power represented by the second parameter modulus this parameter unless another parameter causes it to be `.nul` or `.inf` |

# Selection Functions

This section contains the intrinsic functions that are used for selecting a value from a group of values.

# .if()

## Prototype

`.if( <expression> , <retvaltrue> , <retvalfalse> )`

## Return types

.nul, .inf, string, boolean, integer, number

## Description

The `.if()` function outputs the second or the third parameter, depending on the value of the first parameter.

## Parameters

### *expression*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result is the value of the third parameter |
| *.inf* | The result is the value of the second parameter |
| *string* | If the string is not the empty string then the result is the value of the second parameter, otherwise it is the value of the third parameter |
| *boolean* | If the value of the boolean is .true then the result is the value of the second parameter, otherwise it is the value of the third parameter |
| *integer* | If the value of the integer is non-zero then the result is the value of the second parameter, otherwise it is the value of the third parameter |
| *number* | If the value of the number is non-zero then the result is the value of the second parameter, otherwise it is the value of the third parameter |

### *Result if Expression is True*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result may be `.nul` |
| *.inf* | The result may be `.inf` |
| *string* | The result may be this value |
| *boolean* | The result may be this value |
| *integer* | The result may be this value |

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *number* | The result may be this value |

### *Result if Expression is False*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result may be `.nul` |
| *.inf* | The result may be `.inf` |
| *string* | The result may be this value |
| *boolean* | The result may be this value |
| *integer* | The result may be this value |
| *number* | The result may be this value |

# .min()

## Prototype

`.min( <…> )`

## Return types

.nul, .inf, string, integer, number, boolean

## Description

Returns the minimum of any number of parameters. Values are considered to be in the following order, with the greatest first. Within any item below values are compared in the normal way.

- `.inf`
- numbers greater than or equal to 1
- `.true`
- strings that are not empty
- blobs that are not empty
- numbers greater than 0 and less than 1
- the empty blob
- the empty string
- `.false`
- zero
- negative numbers
- `.nul`

### Parameters

Variable

## .max()

### Prototype

`.max( <…> )`

### Return types

integer, number, boolean, .inf, .nul

### Description

Returns the maximum of any number of parameters. Values are considered to be in the following order, with the greatest first. Within any item below values are compared in the normal way.

- `.inf`
- numbers greater than or equal to 1
- `.true`
- strings that are not empty
- blobs that are not empty
- numbers greater than 0 and less than 1
- the empty blob
- the empty string
- `.false`
- zero
- negative numbers
- `.nul`

### Parameters

Variable

# Blob Functions

This section contains the intrinsic functions that are used for working together with blob values.

## .inblob()

### Prototype

`.inblob( <sourceblob> , <searchblob> )`

## Return types

.nul, .inf, integer

## Description

Returns the one-based position of the first occurrence of one blob within another. If the subblob is not found then zero is returned.

## Parameters

### *sourceblob*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.inf`, unless the second parameter is `.nul` |
| *blob* | The blob to find the subblob in |

### *searchblob*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.inf`, unless the first parameter is `.nul` |
| *blob* | The subblob to search for. If this is the empty blob then the return value will be 0, unless the first parameter is `.nul` or `.inf` |

# .subblob()

## Prototype

`.subblob( <sourceblob> , <startpos> , <count> )`

## Return types

.nul, .inf, blob

## Description

Extracts bytes a specified number of bytes from a specified position in a blob

## Parameters

### *sourceblob*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.inf`, unless another parameter is `.nul` |

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *blob* | The blob to extract bytes from |

### *startpos*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | If the result will be a blob, then it will be the empty blob, ie. the function behaves as if byte extraction starts at the end of the blob. |
| *integer* | The one-based position to start extracting bytes from. If this is zero or negative then the start position is the start of the blob. If it is greater than the length of the blob then an empty blob is returned. |

### *count*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | If the result is a blob then all bytes to the end of the blob are extracted |
| *integer* | The number of bytes to extract. If this is negative no bytes are returned. If it is greater than the length from the start of the extraction to the end of the blob then bytes up to the end of the blob are returned. |

# String Functions

This section contains the intrinsic functions that are used for working together with string values.

# .instr()

## Prototype

.instr( <*sourcetext*> , <*searchtext*> )

## Return types

.nul, .inf, integer

## Description

Returns the one-based position of the first occurrence of one string within another. If the substring is not found then zero is returned.

## Parameters

### *sourcetext*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.inf* | The result will be .inf, unless the second parameter is .nul |
| *string* | The string to find the substring in |

### *searchtext*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf, unless the first parameter is .nul |
| *string* | The substring to search for. If this is the empty string then the return value will be 0, unless the first parameter is .nul or .inf |

# .len()

## Prototype

.len( <*sourcetext*> )

## Return types

.nul, .inf, integer

## Description

Determines the length of a string.

## Parameters

### *sourcetext*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf |
| *string* | The string to return the length of |

# .like1()

## Description

Compares a string with a pattern and returns .true if the string matches the pattern, or .false otherwise.

## Prototype

.like1 ( string , string )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | None | `string` | The string to test against the pattern. |
| | None | `string` | The pattern to test the string against. The string is matched against the pattern on a character by character basis, with the exception of the * wildcard described below. The string is considered to match the pattern is every character in it is matched by the characters in the pattern, and neither has any characters in excess. Matching can be performed either in a case sensitive mode, or a case insensitive mode. With the exception of the special pattern characters described below, a character in the string matches a character in the pattern if either the characters are the same, or if case insensitive testing is being done and the characters are the same when converted to lower case. The special pattern characters are:<br>• ^ = toggles between case sensitive and case insensitive testing. The initial case sensitivity setting is always case insensitive.<br>• ? = matches any character, but the character must exist.<br>• * = matches any number of any characters, including no characters.<br>• [ = starts a range specifier. A range specifier gives a number of characters, any of which may match a character from the string, for example the range "[abc]" matches either "a", "b" or "c". If case insensitive testing is being done, then it also matches "A" "B" and "C". Within a range there are some characters with special meanings:<br>  • ] = indicates the end of the range.<br>  • [ = indicates that the next character is to be interpreted literally as part of the range, and any special meaning is to be overlooked. For example the pattern "[[[]" is the only way to match only the string "[", and "[[]]" is the only way to match only the string "]"<br>  • - = indicates that a contiguous range is being included in the range. If '-' is the first character in the range then all characters up to the value of the next character in the range are included, for example the pattern "[-9]" matches all characters up to and including the character '9'. If '-' is the last character in the range then all characters including or greater than the previ- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ous character are included in the range, for example the pattern "[A-]" matches all characters above or equal to character 'A'. If '-' is the only character in the range then any character from the string is a match, i.e. the pattern "[-]" is equivalent to the pattern "?". If the character '-' has another character both before and after it (as would normally be the case) then the range includes every character between the lower valued and the higher valued of these characters, for example the pattern "[A-Z]" matches all characters in the upper case roman alphabet. If a range is specified and case insensitive testing is being done, then a character in the string is compared with the characters in the range and the lower case of the character from the string is compared with the lower case of each character in the range. The conversion to lower case is done AFTER contiguous ranges have been calculated, which may be different from calculating contiguous ranges after converting to lower case. |

# .lstr()

## Prototype

`.lstr( <sourcetext> , <count> )`

## Return types

.nul, .inf, string

## Description

Extracts characters from the beginning of a string

## Parameters

### *sourcetext*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.inf`, unless the second parameter is `.nul` |
| *string* | The string to extract the beginning characters from |

### *count*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.inf* | The result is the whole first parameter |
| *integer* | The number of characters to extract. If this is negative no characters are returned. If it is greater than the length of the string then the whole string is returned |

# .rstr()

## Prototype

```
.rstr( <sourcetext> , <count> )
```

## Return types

.nul, .inf, string

## Description

Extracts characters from the end of a string

## Parameters

### *sourcetext*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result will be `.inf`, unless the second parameter is `.nul` |
| *string* | The string to extract the ending characters from |

### *count*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be `.nul` |
| *.inf* | The result is the whole first parameter |
| *integer* | The number of characters to extract. If this is negative no characters are returned. If it is greater than the length of the string then the whole string is returned |

# .substr()

## Prototype

```
.substr( <sourcetext> , <startpos> , <count> )
```

## Return types

.nul, .inf, string

## Description

Extracts characters a specified number of characters from a specified position in a string

# Parameters

## *sourcetext*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | The result will be .inf, unless another parameter is .nul |
| *string* | The string to extract characters from |

## *startpos*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | If the result will be a string, then it will be the empty string, ie. the function behaves as if character extraction starts at the end of the string. |
| *integer* | The one-based position to start extracting characters from. If this is zero or negative then the start position is the start of the string. If it is greater than the length of the string then an empty string is returned. |

## *count*

| Type name | Result for an expression of this type in this parameter |
|---|---|
| *.nul* | The result will be .nul |
| *.inf* | If the result is a string then all characters to the end of the string are extracted |
| *integer* | The number of characters to extract. If this is negative no characters are returned. If it is greater than the length from the start of the extraction to the end of the string then characters up to the end of the string are returned. |

# Chapter 2. System Functions

## Introduction

In SIMPOL system functions are functions that begin with the exclamation point (!) operator and take a variable number of potentially named parameters. They begin with an exclamation point for the same reasons as intrinsic functions begin with a dot; in order to ensure that no user-defined function could be named the same so as to allow for future expansion in this area without fear of colliding with existing user-defined function names.

## !beginthread()

### Prototype

`!beginthread(`<*`startfunction`*>, <…>)

### Return value

.nul

### Description

The first parameter is a function object for the function to start the new thread at. The following parameters must all be objects, not values, but there can be any number of them and they can be of any type, or `.nul`. These are passed to the function when the new thread starts. A program only ends when all threads have ended, ie. reached the end of their start function.

### Parameters

| Name | Default | Type | Description |
|---|---|---|---|
| *startfunction* | `.nul` | function | The function object where the thread should start. |
| … | none | any object | Zero or more parameters as objects to pass to the start function |

## !execute()

### Prototype

`!execute` ( string *`command`*, boolean *`block`*, integer *`error`* )

### Return value

.nul

### Description

Executes a program on the native OS, and optionally waits for it to complete.

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| command | None | string | The command line to be executed. The full command line should be given, i.e. the program name and space-delimited command line parameters. |
| block | `.false` | boolean | A flag to indicate whether the call should wait in the function until the program being executed has finished. |
| error | `.nul` | integer | Specifies an object which is used to output any error code which is raised during the starting of the external program. Errors which occur in the program after it has successfully started may halt the program, but will not be returned by this function. |

# !getproperty()

## Prototype

`!getproperty` ( type(*) *`object`*, property *`property`* )

## Return value

This function returns the value of the property which may be of any type: type(*).

## Description

Retrieves the value of a property of an object, using the property object for the type, and the object itself. This can be useful if you are working with an unknown object, and wish to retrieve the values of its properties. Using the type object the properties can be retrieved and now with this function the values of those properties can be accessed. Without this function, the name of the property must be known at compile time in order to retrieve a property's value from an object.

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| object | `.nul` | type(*) | The object that contains the property for which the value is to be retrieved. |
| property | `.nul` | property | The property object for the property whose value is to be retrieved. |

# !getvariable()

## Prototype

`!getvariable` ( string *`variable`* )

## Return value

This function returns a string containing the value of the environment variable matching the string argument, or .nul if no match was found.

## Description

Retrieves the value of an environment variable that matches the name of the argument passed. If no match is found, it returns `.nul`.

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| variable | None | string | The name of the environment variable to be retrieved. |

# !loadmodulefile()

## Prototype

`!loadmodulefile`( string <*`filename`*>, integer <*`error`*>)

## Return value

.nul

## Description

The first parameter is the name of the compiled SIMPOL module to be loaded. The second parameter must be an integer object where any error upon loading the module will be placed. If it is not included, any error will cause the program to halt. Once loaded, a module cannot currently be unloaded. Modules that are loaded in this way do not, by default, expose their functions or data types. In order to access a function or data type from an external module, the `export` keyword must have been used in their definition.

## Parameters

| Name | Default | Type | Description |
|---|---|---|---|
| *`filename`* | `.nul` | string | The name of the module file that is to be loaded. |
| *`error`* | `.nul` | integer | This parameter must be an object, not an integer value. If an error (other than a syntax or parameter passing error) occurs because the new object position is not completely within the form, then the error number will be output to the specified object. If no error object is provided and an error occurs then the program will exit with an unhandled exception. |

# !osinfo()

## Prototype

`!osinfo`( string <*`ostype`*>, integer <*`architecture`*>)

## Return value

.nul

## Description

Both parameters must be initialized to the empty string. The first parameter will be filled with the name of the operating system and the second with the architecture of the operating system. Currently these support "windows" and "linux" for the first parameter. The second parameter will currently return "x86" since we only provide a 32-bit implementation. The architecture value does not tell you if you are running on a 64-bit system, but rather for which type of system the program was compiled for. This becomes important when you need to make a call to an external library, since the library must have been compiled for the correct architecture. This information would also affect the parameter and return value entries in any declaration of an external function call.

## Parameters

| Name | Default | Type | Description |
|---|---|---|---|
| *`ostype`* | `.nul` | string | This must be a pre-initalized string object and will contain the operating system for which the SIMPOL runtime system was compiled when the function returns. Currently this will be either "windows" or "linux". |
| *`architec-ture`* | `.nul` | string | This must be a pre-initalized string object and will contain the architecture of the operating system for which the SIMPOL runtime system was compiled when the function returns. Currently this will only be "x86" until we release a 64-bit version. |

# !setproperty()

## Prototype

`!setproperty` ( type(*) *`object`*, property *`property`*, type(*) *`value`* )

## Return value

.nul

## Description

Sets the value of a property of an object, using the property object for the type, and the object itself. This can be useful if you are working with an unknown object, and wish to assign a value to one of its

properties. Using the type object the properties can be retrieved and now with this function the values of those properties can be assigned. Without this function, the name of the property must be known at compile time in order to assign a property's value. This is also the only way that a property that has been marked `readonly` can be set. In such a case, the read only property must be set in a member function of the type and it can only be set using this system function.

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| object | `.nul` | type(*) | The object that contains the property for which the value is to be assigned. |
| property | `.nul` | property | The property object for the property whose value is to be assigned. |
| value | `.nul` | type(*) | The value that is to be assigned to the property. In the case of reference properties, this should be an object. |

# !wait()

## Description

Suspends execution of the current SIMPOL thread for a specified period of time.

## Prototype

`!wait` ( integer )

## Return value

.nul

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | `.nul` | integer | The number of microseconds to wait for, which can be any value from 0 to 315537811200000000. A value of `.nul` means there will be no delay. A value of `.inf` means that the thread will be suspended indefinitely. |

# Chapter 3. Operators

# Introduction

This section provides a reference to the various operators that are built into the SIMPOL programming language. Most of the operators should appear fairly familiar, although there are some significant differences between a few of the ones common in most BASIC-language derivatives and those found in SIMPOL, although these differences will not surprise users of other languages such as C, Java, or Pascal. There are also a few operators that are specific to SIMPOL alone, as you will find below. This appendix is divided into five sections: Assignment Operators, Comparison Operators, Arithmetic Operators, Logical Operators, and Bitwise Operators. Object operators are the property operator and the member operator and are covered in detail in the Programming Guide.

# Assignment Operators

Assignment operators are different than logical or arithmetic operators in that the latter two can be characterized as performing an operation on one or more values and then returning a result, whereas the assignment operator takes the value of something on the right side of the operator (a value or a reference) and assigns it to the item on the left side of the operator.

## Assignment operator (=)

### Operand count

2

### Types

`.nul`, `.inf`, `blob`, `boolean`, `integer`, `number`, `string`

### Description

The value of the right side of the assignment operator is assigned to the item on the left side of the operator. The assignment operator differs from the equals operator in BASIC in that it can only be used for assignment in SIMPOL, not for comparisons. For comparisons the == operator must be used. Numbers can be assigned to integers but will lose their fractional component and strings can be assigned to blobs but they must consist only of characters that are less than 256 in value.

## Reference assignment operators (=@, @=)

### Operand count

2

### Types

`.nul`, *`any object`*,

### Description

A reference to the object on the right side of the assignment operator is assigned to the item on the left side of the operator.

# Arithmetic Operators

Arithmetic operators are commonly used to apply mathematical calculations to the items on either side of the operator and to return some result. In SIMPOL, the arithmetic operators do not necessarily need to apply to numbers or integers. There are a number of useful ways that strings can be used together with arithmetic operators as will be seen from the documentation that follows.

## Unary Minus (–)

### Operand count

1

### Types

`.nul`, `.inf`, `blob`, `boolean`, `integer`, `number`, `string`

### Description

Negates a value

#### Behavior by type

| Type name | Result when unary minus (–) is applied to this type |
|---|---|
| `.nul` | The result will be `.nul` |
| `.inf` | The result will be `.inf` |
| blob | The result is the reverse of the operand, ie. the bytes of the result are the bytes of the input operand but in the opposite order |
| boolean | The result is the same as the operand, ie. **–.true** is `.true` and **–.false** is `.false` |
| integer | The result is the negation of the operand |
| number | The result is the negation of the operand |
| string | The result is the reverse of the operand, ie. the characters of the result are the characters of the input operand but in the opposite order |

## Subtraction (–)

### Operand count

2

### Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

### Description

Subtracts one operand from the other. The operands can be of any type, but an error will result if string, integer, number or boolean types are mixed.

### Behavior by type

| Type name | Result when subtraction (–) is applied to this type |
|---|---|
| `.nul` | If any operand is `.nul` then the result will be `.nul` |
| `.inf` | If any operand is `.inf` and no operand is `.nul` then the result will be `.inf` |
| boolean | If the operands are the same then the result is `.false`, otherwise it is `.true` |
| integer | The result is the difference between the operands |
| number | The result is the difference between the operands |
| string | The result is the first string with all copies of the second string removed, for example "abcdefabc" - "bc" is "adefa" |

# Addition (+)

## Operand count

2 or more

## Types

`.nul`, `.inf`, `blob`, `boolean`, `integer`, `number`, `string`

## Description

Calculates the sum of the operands. The operands can be of any type, but an error will result if string, blob, integer, number or boolean types are mixed.

### Behavior by type

| Type name | Result when addition (+) is applied to this type |
|---|---|
| `.nul` | If any operand is `.nul` then the result will be `.nul` |
| `.inf` | If any operand is `.inf` and no operand is `.nul` then the result will be `.inf` |
| blob | If all operands are blobs then the result is the concatenation of them |
| boolean | If all the operands are boolean then the result is `.true` if any operand is `.true`, otherwise the result is `.false` |
| integer | If all operands are integers then the result is their sum |
| number | If all operands are integers or numbers then the result is their sum |
| string | If all operands are strings then the result is the concatenation of them |

# Multiplication (*)

## Operand count

2 or more

## Types

`.nul`, `.inf`, `blob`, `boolean`, `integer`, `number`, `string`

## Description

Calculates the product of the operands. The operands can be of any type, but an error will result if more than one operand is a string or a blob.

### Behavior by type

| Type name | Result when multiplication (*) is applied to this type |
|---|---|
| `.nul` | If any operand is `.nul` then the result will be `.nul` |
| `.inf` | If any operand is `.inf` and no operand is `.nul` then the result will be `.inf` |
| blob | Only one operand can be a blob. If one of the operands is a blob then the result will be a blob. If all the other operands are integers or numbers or the `.true` boolean value then the result is the blob operand replicated the number of times given by the product of the integer and number operands. If the product of the integer operands is negative then the blob result is reversed. |
| boolean | If all operands are boolean then the result will be boolean. If any of the operands are `.false` then the result will be `.false`, the empty string or zero. Any operands which are `.true` do not affect the result unless all operands are `.true`, in which case the result will be `.true` |
| integer | If there are no string, number, `.nul` or `.inf` operands, and there is at least one integer operand, then the result will be an integer. The result of multiplying integer operands is their numerical product. |
| number | If there are no string, number, `.nul` or `.inf` operands, and there is at least one number operand, then the result will be an integer or a number. The result of multiplying integer or number operands is their numerical product. |
| string | Only one operand can be a string. If one of the operands is a string then the result will be a string. If all the other operands are integers or numbers or the `.true` boolean value then the result is the string operand replicated the number of times given by the product of the integer and number operands. If the product of the integer operands is negative then the string result is reversed. |

# Division (/)

## Operand count

2

## Types

`.nul`, `.inf`, `integer`, `number`

## Description

Divides one operand by another.

### Behavior by type

| Type name | Result when division (/) is applied to this type |
|---|---|
| `.nul` | If either operand is `.nul` then the result will be `.nul` |

| Type name | Result when division (/) is applied to this type |
|---|---|
| `.inf` | Dividing any number or `.inf` by `.inf` gives the result `.nul`. Dividing `.inf` by any number except `.nul` gives the result `.inf`. |
| integer | The result is the quotient of the operands |
| number | The result is the quotient of the operands |

# Modulus (mod)

## Operand count

2

## Types

`.nul`, `.inf`, `integer`, `number`

## Description

Calculate the remainder when one number is divided by another.

### Behavior by type

| Type name | Result when modulus (mod) is applied to this type |
|---|---|
| `.nul` | If either operand is `.nul` then the result will be `.nul` |
| `.inf` | Using modulus on any number or `.inf` by `.inf` gives the result `.nul`. Using modulus on `.inf` with any number except `.nul` gives the result `.inf`. |
| integer | The result is the remainder when the largest possible number of second operands are taken from the first operand. If the result is non-zero then it will be of the same sign as the first operand. |
| number | The result is the remainder when the largest possible number of second operands are taken from the first operand. If the result is non-zero then it will be of the same sign as the first operand. |

# Comparison Operators

Comparison operators are used for doing comparisons for the purpose of flow-control within the language. They differ from arithmetic operators in that they typically only return `.true` or `.false` which is then used to make a decision about the flow of the program.

# Less than (<)

## Operand count

2

## Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

## Description

The result is .true if the first operand is less than the second operand, otherwise it is .false. Values of any types can be compared. Values are considered to be in the following order, with the greatest first. Within any item below values are compared in the normal way.

- `.inf`
- numbers greater than or equal to 1
- `.true`
- strings that are not empty
- blobs that are not empty
- numbers greater than 0 and less than 1
- the empty blob
- the empty string
- `.false`
- zero
- negative numbers
- `.nul`

# Less than or equal (<=)

## Operand count

2

## Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

## Description

The result is .true if the first operand is less than or equal to the second operand, otherwise it is .false. Values of any types can be compared. Values are considered to be in the following order, with the greatest first. Within any item below values are compared in the normal way.

- `.inf`
- numbers greater than or equal to 1
- `.true`
- strings that are not empty
- blobs that are not empty
- numbers greater than 0 and less than 1

- the empty blob
- the empty string
- `.false`
- zero
- negative numbers
- `.nul`

# Equal to (==)

## Operand count

2

## Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

## Description

The result is .true if the first operand is equal to the second operand, otherwise it is .false. Values of any types can be compared. Values are considered to be in the following order, with the greatest first. Within any item below values are compared in the normal way.

- `.inf`
- numbers greater than or equal to 1
- `.true`
- strings that are not empty
- blobs that are not empty
- numbers greater than 0 and less than 1
- the empty blob
- the empty string
- `.false`
- zero
- negative numbers
- `.nul`

# Not equal to (<>, !=)

## Operand count

2

## Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

## Description

The result is .true if the first operand is not equal to the second operand, otherwise it is .false. Values of any types can be compared. Values are considered to be in the following order, with the greatest first. Within any item below values are compared in the normal way.

- `.inf`
- numbers greater than or equal to 1
- `.true`
- strings that are not empty
- blobs that are not empty
- numbers greater than 0 and less than 1
- the empty blob
- the empty string
- `.false`
- zero
- negative numbers
- `.nul`

# Greater than or equal (>=)

## Operand count

2

## Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

## Description

The result is .true if the first operand is greater than or equal to the second operand, otherwise it is .false. Values of any types can be compared. Values are considered to be in the following order, with the greatest first. Within any item below values are compared in the normal way.

- `.inf`
- numbers greater than or equal to 1
- `.true`

- strings that are not empty
- blobs that are not empty
- numbers greater than 0 and less than 1
- the empty blob
- the empty string
- `.false`
- zero
- negative numbers
- `.nul`

# Greater than (>)

## Operand count

2

## Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

## Description

The result is .true if the first operand is greater than the second operand, otherwise it is .false. Values of any types can be compared. Values are considered to be in the following order, with the greatest first. Within any item below values are compared in the normal way.

- `.inf`
- numbers greater than or equal to 1
- `.true`
- strings that are not empty
- blobs that are not empty
- numbers greater than 0 and less than 1
- the empty blob
- the empty string
- `.false`
- zero
- negative numbers
- `.nul`

# Refer to same object (=@=)

## Operand count

2

## Types

`.nul`, *`any object`*

## Description

The result is .true if both operands are references to the same object or are both nul object references, otherwise it is .false.

# Not refer to same object (<@>, !@=)

## Operand count

2

## Types

`.nul`, *`any object`*

## Description

The result is .false if both operands are references to the same object or are both nul object references, otherwise it is .true.

# Logical Operators

Logical operators are used for doing boolean operations for the purpose of evaluating multiple conditions that are based on true and false values. They differ from arithmetic operators in that they typically only return `.true` or `.false` which is then used to make a decision about the flow of the program. They are commonly used together with comparison operators and arithmetic operators in branching and looping statements to evaluate whether the loop should continue or decide which branch to take.

# and

## Operand count

2 or more

## Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

## Description

Gives a boolean result indicating whether or not all the operands are not `.nul`, the empty string (`""`), zero (`0`)or `.false`.

### Behavior by type

| Type name | Result for and when applied to this type |
|---|---|
| `.nul` | If any operand is `.nul` then the result is `.false` |
| `.inf` | If all operands are `.inf` then the result is `.true`, otherwise the result depends on the value of the other operands |
| boolean | If any operand is `.false` then the result is `.false` |
| integer | If any operand is zero (`0`) then the result is `.false` |
| number | If any operand is zero (`0`) then the result is `.false` |
| string | If any operand is the empty string then the result is `.false` |

# or

## Operand count

2 or more

## Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

## Description

Gives a boolean result indicating whether or not any operand is not `.nul`, the empty string (`""`), zero (`0`)or `.false`.

### Behavior by type

| Type name | Result for or when applied to this type |
|---|---|
| `.nul` | If all operands are `.nul` then the result is `.false` otherwise the result depends on the value of the other operands |
| `.inf` | If any operand is `.inf` then the result is `.true` |
| boolean | If any operand is `.true` then the result is `.true` |
| integer | If any operand is non-zero then the result is `.true` |
| number | If any operand is non-zero then the result is `.true` |
| string | If any operand is a non-empty string then the result is `.true` |

# not

## Operand count

1

## Types

`.nul`, `.inf`, `boolean`, `integer`, `number`, `string`

## Description

Creates a result which is a logical opposite of the input operand

### Behavior by type

| Type name | Result when not is applied to this type |
|---|---|
| `.nul` | The result will be `.nul` |
| `.inf` | The result will be `.inf` |
| boolean | `.true` becomes `.false`, and `.false` becomes `.true` |
| integer | `0` becomes `1`, and any other value becomes `0` |
| number | `0` becomes the integer `1`, and any other value becomes the integer `0` |
| string | The empty string becomes `"."`, and any other string becomes the empty string |

# Bitwise Operators

Bitwise operators are used for applying boolean operations at the individual bit level of a value. They are commonly used to set and test flag values where a single 32-bit integer may carry 32 different values in one integer. This is a very efficient way to pass a number of options in a single value. Bitwise operators are also commonly used in hashing, compression, and encryption algorithms, among others. Use of these operators is generally the province of advanced programmers and they should not be used in place of the Boolean equivalents, since although they may sometimes work, they could produce unexpected results. The bitwise operators are `AND`, `OR`, and `XOR`. These should *not* be confused with the lowercase operators of the same name in the case of `AND` and `OR`, which are the Boolean operators.

For those unfamiliar with bitwise operators, they operate on the individual bits where each bit contains only 0 or 1. The `AND` operator compares two values and only returns the value 1 if both bits contain a 1, otherwise it returns 0. The `OR` operator compares two values and returns 1 if *either* of the bits contain 1, otherwise it returns 0. The `XOR` operator compares two values and returns 1 if one of the two values has a bit set and the other does not. If both values are the same (both 1's or both 0's), then it returns 0. The following tables may help clarify things:

### Table 3.1. Bitwise AND Table

| Op1 | Op2 | Result |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |

A common use for the `AND` operator is as a test to see if a bit is set in a value.

### Table 3.2. Bitwise OR Table

| Op1 | Op2 | Result |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |

| Op1 | Op2 | Result |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 1 | 1 |

A common use for the `OR` operator is to set a bit in a value independent of its current state.

### Table 3.3. Bitwise XOR Table

| Op1 | Op2 | Result |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 0 |

A common use for the `XOR` operator is to toggle a bit in a value. Using `XOR` twice with the same operands will return the value back to its starting point.

# AND

## Operand count

2 or more

## Types

`.nul`, `.inf`, `blob`, `boolean`, `integer`, `number`, `string`

## Description

All operands other than `.nul` and `.inf` must be of the same type. If any operand is equal to `.nul` then the result is `.nul`. Otherwise if any operand is equal to `.inf` then the result is `.inf`. Otherwise the result is calculated according to the type of the operands.

### Table 3.4. Behavior by type for AND

| Type name | Result for AND when applied to this type |
|---|---|
| `.nul` | If any operand is `.nul` then the result is `.nul` |
| `.inf` | If any operand is `.inf` and no operand is `.nul` then the result is `.inf`. |
| blob | The result is as long as the longest operand where operands that are not that long behave as if they have been extended with zero bytes. Each byte in the result is the bitwise combination of the corresponding bytes in the operands, where bytes are treated as unsigned values. |
| boolean | The result is calculated using the obvious meanings of `AND`, `OR`, and `XOR`. |
| integer \| number | Operands that are of type number are converted to an integer by taking the integer part. The result is the bitwise combination of the inte- |

| Type name | Result for AND when applied to this type |
|---|---|
| | ger operand values, where negative values are treated as if they represented the two's complement of their absolute value. |
| string | The result is as long as the longest operand and each character is the bitwise combination of the corresponding characters in each operand string, where characters are considered to be unsigned values. Where operands are not as long as the result the operands behave as if they are extended with characters of Unicode code point zero (`.char(0)`). |

# OR

## Operand count

2 or more

## Types

`.nul`, `.inf`, `blob`, `boolean`, `integer`, `number`, `string`

## Description

All operands other than `.nul` and `.inf` must be of the same type. If any operand is equal to `.nul` then the result is `.nul`. Otherwise if any operand is equal to `.inf` then the result is `.inf`. Otherwise the result is calculated according to the type of the operands.

### Table 3.5. Behavior by type for OR

| Type name | Result for OR when applied to this type |
|---|---|
| `.nul` | If any operand is `.nul` then the result is `.nul` |
| `.inf` | If any operand is `.inf` and no operand is `.nul` then the result is `.inf`. |
| blob | The result is as long as the longest operand where operands that are not that long behave as if they have been extended with zero bytes. Each byte in the result is the bitwise combination of the corresponding bytes in the operands, where bytes are treated as unsigned values. |
| boolean | The result is calculated using the obvious meanings of `AND`, `OR`, and `XOR`. |
| integer \| number | Operands that are of type number are converted to an integer by taking the integer part. The result is the bitwise combination of the integer operand values, where negative values are treated as if they represented the two's complement of their absolute value. |
| string | The result is as long as the longest operand and each character is the bitwise combination of the corresponding characters in each operand string, where characters are considered to be unsigned values. Where operands are not as long as the result the operands behave as if they are extended with characters of Unicode code point zero (`.char(0)`). |

# XOR

## Operand count

2 or more

## Types

`.nul`, `.inf`, `blob`, `boolean`, `integer`, `number`, `string`

## Description

All operands other than `.nul` and `.inf` must be of the same type. If any operand is equal to `.nul` then the result is `.nul`. Otherwise if any operand is equal to `.inf` then the result is `.inf`. Otherwise the result is calculated according to the type of the operands.

### Table 3.6. Behavior by type for XOR

| Type name | Result for XOR when applied to this type |
|---|---|
| `.nul` | If any operand is `.nul` then the result is `.nul` |
| `.inf` | If any operand is `.inf` and no operand is `.nul` then the result is `.inf`. |
| blob | The result is as long as the longest operand where operands that are not that long behave as if they have been extended with zero bytes. Each byte in the result is the bitwise combination of the corresponding bytes in the operands, where bytes are treated as unsigned values. |
| boolean | The result is calculated using the obvious meanings of `AND`, `OR`, and `XOR`. |
| integer \| number | Operands that are of type number are converted to an integer by taking the integer part. The result is the bitwise combination of the integer operand values, where negative values are treated as if they represented the two's complement of their absolute value. |
| string | The result is as long as the longest operand and each character is the bitwise combination of the corresponding characters in each operand string, where characters are considered to be unsigned values. Where operands are not as long as the result the operands behave as if they are extended with characters of Unicode code point zero (`.char(0)`). |

# Part II. C-Language Components

SIMPOL uses a component-based architecture that allows components written in languages such as C to be added to the language. These components are added to the project by the user as required. If a component is required then it must be listed in the components section of the compiled SIMPOL program or library. In addition to the types and functions provided directly by SIMPOL as part of the basic language, a growing number of components are provided. These components currently include: forms, windows, sockets, user-interface utilities, operating system utilities, PPCS, and SBME. Not all types and/or functions are available on all platforms, though the intention is that the majority of types and functions are eventually available for all supported platforms where their existence makes sense.

## Note

The WIN1 and FRM1 components are currently deprecated. Please use the WXWN components in their place. The new components have greater capabilities and will very quickly become cross-platform, as they are based on the cross-platform wxWidgets toolkit. The basic design to each component is similar, so conversion shouldn't take very long if you have already used them. A new utility for generating code from SBVs based on the WXWN components has already been created.

# Table of Contents

# Chapter 4. SIMPOL Built-In Types

This section provides a reference into the various types and functions that are built into SIMPOL. These types and functions are always present and do not need to be added as a component when creating a project.

# anyvalue

## Description

Variables of type anyvalue can be assigned a value from any of the value types in SIMPOL. This includes blob, boolean, integer, number, and string. For specifics about the values of each data type, see their descriptions. They can also be assigned a value from a literal of any type that can be assigned within a SIMPOL source code file.

### Warning

It is possible to pass an object of type anyvalue to a function that is expecting a string if the anyvalue object contains a string value. What will happen is that the value of the string contained in the anyvalue object will be passed into a new string object upon arrival in the function but any changes to the string will not be passed back to the anyvalue object in the calling function, since the two objects are not of the same type.

To return a value of type anyvalue to a calling function and have it assigned to a variable that is one of the standard value types, return the anyvalue object as the return value from the function and assign it using the equals (=) operator.

The anyvalue object is not considered to be a value type and does require a call to its `new()` method in order to be initialized.

## Type Tags

None

## Object Value

The value of an anyvalue object is its content.

# anyvalue.new()

## Description

Creates a new anyvalue object and sets its initial value. The return value should normally be assigned using the `=@` operator to avoid unnecessary creation and destruction of an object to hold the interim value.

## Prototype

*anyvalue*.new ( blob|boolean|integer|number|string *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | blob\|boolean\|integer\|number\|string | The value to be assigned to the new object. |

## Properties

| Property | Type | Description |
|---|---|---|
| datatype | type | Contains a reference to the type object that represents the data type of the content of the anyvalue object. |
| type | type | Specifies the anyvalue type object. |

# array

## Description

Array objects are actually objects in their own right, they are not arrays of some single other type. The array implementation in SIMPOL is extremely flexible. Typical array implementation can be considered to be a subset of the SIMPOL array type. SIMPOL array objects do not need to be sized in advance, nor are they of one particular type. They can also house elements at every level of the array (in the case of a multidimensional array). The subscript of an array element can also be a string. An array can hold values and references to objects. Assigning a value type to an array element assigns the *value* to the element. It does not also contain a reference to a value type. To assign a reference to an object to an array element it is necessary to use the reference assignment operator (`@=` or `=@`). To clear an element it is necessary to assign a reference to `.nul`.

## Type Tags

None

## Object Value

The value of an object of type array is undefined and it is an error to attempt to get or to set it.

## array.new()

### Description

Creates a new array object.

### Prototype

*`array`*`.new ()`

### Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the array type object. |

# Methods

## count()

### Description

Returns the count of items that are hierarchically below the item referenced by the subscript including the subscript itself. For example: `a.count(1,2)` if there is an item located at `a[1,2]` then this will count as 1, and all items subscripted below `a[1,2]`, such as `a[1,2,1]` or `a[1,2,1,3]` will also count assuming that those points actually contain a value or a reference. The `count()` method is not particularly useful unless a certain degree of discipline and design are used with this type. If the array is used in a more typical way, then this method will work as expected, assuming that the subscripts begin with the value `1` and are sequential from there.

### Prototype

*`arrayvar`*`.count` ( integer|string *`count`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| count | None | integer\|string | If no argument is passed then all of the elements will be counted. Otherwise it is as described above. There is no way to detect the subscripts of elements if they are out of sequence or based on strings. If the value passed is equal either to `.nul` or `.inf` then the return value will be `.nul`. |

# array[]

## Get

### Subscripts

An integer or string value giving the index position of an item in the list. This can be a single integer, a string, the empty value (`[]`), or a hierarchical position, such as `[1,2]`. Even a combination of index values is permitted, such as: `[1, "blue"]`.

### Description

Retrieves the value or object at the specified index position.

### Set

#### Subscripts

An integer or string value giving the index position of an item in the list. This can be a single integer, a string, the empty value ([]), or a hierarchical position, such as [1,2]. Even a combination of index values is permitted, such as: [1, "blue"].

#### Description

Assigns a value to the specified index position in the array.

### Set Reference

#### Subscripts

An integer or string value giving the index position of an item in the list. This can be a single integer, a string, the empty value ([]), or a hierarchical position, such as [1,2]. Even a combination of index values is permitted, such as: [1, "blue"].

#### Description

Assigns a reference to an object to the specified index position in the array.

# blob

## Description

Blob types are value types and can contain either .nul, .inf, the empty blob or a blob value of any size (limited by memory). Blobs are made up of bytes. To provide for ease of creation, it is possible to assign literal strings and string variables to a blob using either the equals = symbol, the .toblob() intrinsic function, or as part of the blob.new() method.

## Type Tags

None

## Object Value

The value of a blob object is its content.

## blob.new()

### Description

Resizes an existing blob. If the blob's size is increased, it is padded with 0's. If it is decreased, the content is truncated.

### Prototype

*blob*.new ( blob | string *value*, integer *size* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | blob \| string | The value to be assigned to the new object. In value types the parameter to the new function is not a named parameter. Specifically in this object the initial parameter can be either a blob, a string that contains no characters greater than character value 255, or the named parameter that follows, which is an initialization size. |
| size | None | integer | This parameter can be used in place of an initialization value in order to create a blob with an initial size that has been initialized with 0 in each byte. This can be useful when manipulating items that will be eventually placed in a single buffer and can be far more efficient both in memory and speed than doing so in other ways. To use this parameter, it is necessary use it by name in the `new()` method. |

# Properties

| Property | Type | Description |
|---|---|---|
| size | integer | Contains the size of the blob in bytes (octets). |
| type | type | Specifies the integer type object. |

# Methods

## getblob()

### Description

Returns a blob from a blob starting at position *`start`* for length bytes.

### Prototype

*`blobvar`*`.getblob` ( integer *`start`*, integer *`length`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| start | `1` | integer | Contains the starting position for extracting the blob. If the value of |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | this is `.nul` or `.inf` then the return value will be `.nul`. |
| length | `1` | integer | Contains the number of bytes to read from the blob. If the value of this is `.nul` then the return value will also be `.nul`. If it is `.inf` then all of the bytes to the end of the blob will be read. |

# getinteger()

## Description

Returns an integer from a blob starting at the byte for position *`start`*. If any of the four parameters are equal to `.nul` then the result will also be `.nul`. If any of the following parameters: *`start`*, *`lbo`*, or *`signed`* is equal to `.inf` then the result will be `.nul`.

## Prototype

*`blobvar`*`.getinteger` ( integer *`start`*, integer *`length`*, boolean *`lbo`*, boolean *`signed`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| start | `1` | integer | Returns an integer starting at the position named in this parameter. |
| length | `1` | integer | The number of bytes to read from the blob to return as an integer value. If this is equal to `.inf`, then all of the bytes will be read to the end of the blob and interpreted as an integer. |
| lbo | `.true` | boolean | Indicates whether the integer value should be interpreted as having been stored with the least significant byte first (`.true`) or with the most significant byte first (`.false`). |
| signed | `.false` | boolean | Indicates whether the integer will be interpreted as a signed or unsigned value. |

# getstring()

## Description

Returns a string from a blob starting at the byte for position *`start`* for *`length`* characters and taking into consideration the settings for character size and logical byte order as well as the optional terminator characters. If any of the first four parameters are equal to `.nul` then the result will also be `.nul`. If any of the following parameters: *`start`*, *`charsize`*, *`lbo`*, or *`terminator`* is equal to `.inf` then the result will be `.nul`.

### Prototype

*blobvar*.getstring ( integer *start*, integer *length*, integer *charsize*, boolean *lbo*, string *terminator*, string *terminatedby* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| start | 1 | integer | Returns a string starting at the position named in this parameter. |
| length | 1 | integer | The number of characters to read from the blob. If this is equal to .inf, then characters will be read to the end of the blob, or until a terminator is found (see *terminator* parameter). |
| charsize | 2 | integer | The number of bytes per character, which must be greater than 0. |
| lbo | .true | boolean | Indicates whether characters should be interpreted as having been stored with the least significant byte first (.true) or with the most significant byte first (.false). |
| terminator | "" | string | A string made up of all the characters that can terminate the string being read. As soon as any one character in the string of terminators is read then reading stops and the string is returned. if the value of this is .nul then no characters will terminate the string being read. |
| terminatedby | no object | string | This parameter must be an object, not a string value such as a literal or expression. If the reading of the string is terminated by finding a character in the *terminator* string then that character is returned in the *terminatedby* object. |

## putblob()

### Description

Outputs a blob into a blob starting at position *start*. It is an error to try and write beyond the end of the blob.

### Prototype

*blobvar*.putblob ( integer *start*, blob *blob* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| start | `1` | integer | Writes a blob beginning at the starting position named in this parameter. If the value of this is `.nul` or `.inf` then the return value will be `.nul`. |
| blob | None | blob | Contains the blob to be output. If the value of this is either `.nul` or `.inf`, then no bytes will be output. |

## putinteger()

### Description

Outputs an integer to a blob starting at position *`start`* for length bytes. It is an error to write past the end of the blob. If either the *`start`* or *`length`* parameter is equal to either `.nul` or `.inf` then a runtime error, "number out of range" will occur. If the *`lbo`* parameter is equal to either `.nul` or `.inf` or the *`integer`* parameter is equal to either `.nul` or `.inf` then no bytes will be output.

### Prototype

*`blobvar`*`.putinteger` ( integer *`start`*, integer *`integer`*, integer *`length`*, boolean *`lbo`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| start | `1` | integer | Writes an integer starting at the position named in this parameter. |
| integer | None | integer | The value to output. |
| length | `1` | integer | The number of bytes to write to the blob. |
| lbo | `.true` | boolean | Indicates whether the integer value should be stored with the least significant byte first (`.true`) or with the most significant byte first (`.false`). |

## putstring()

### Description

Outputs a string to a blob starting at the byte for position *`start`* for the length of the string and taking into consideration the settings for character size and logical byte order. It is an error to write past the end of the blob. If either of the following parameters: *`charsize`* or *`lbo`* is equal to either `.nul` or `.inf` or if the *`string`* parameter is equal to either `.nul` or `.inf` then no bytes will be output. If the *`start`* is equal to either `.nul` or `.inf` then an error, number out of range, will be generated.

### Prototype

*blobvar*.putstring ( integer *start*, string *string*, integer *charsize*, boolean *lbo* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| start | 1 | integer | Writes a string to the blob starting at the position named in this parameter. |
| string | None | string | The string to be output. |
| charsize | 2 | integer | The number of bytes per character, which must be greater than 0. |
| lbo | .true | boolean | Indicates whether characters have been stored with the least significant byte first (.true) or with the most significant byte first (.false). |

## setsize()

### Description

Resizes an existing blob. If the blob's size is increased, it is padded with 0's. If it is decreased, the content is truncated.

### Prototype

*blobvar*.setsize ( integer *size* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| size | None | integer | Contains the new size for the blob. |

# blob[]

## Get

### Subscripts

An integer giving the index position of the byte within the blob.

### Description

Retrieves the integer value of the byte at the specified index position.

## Set

### Subscripts

An integer giving the index position of the byte within the blob.

### Description

Assigns an integer value in the range of 0 to 255 to the specified index position in the blob.

## Set Reference

Attempting to set a reference to an object is not supported.

# boolean

## Description

The boolean object is considered to be a value type and does not require a call to a `new()` method in order to be initialized; normal assignment using the assignment operator `=` is sufficient.

## Type Tags

None

## Object Value

The value of a boolean object is its content.

## Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the boolean type object. |

# date

## Description

Objects of type date contain a value which represents a date, and can be accessed either as a whole or by component, such as day, month and year. Valid dates are in the range 1/1/0001 to 31/12/9999, and the Gregorian calendar is assumed throughout that period.

## Type Tags

None

## Object Value

The value of an object of type date is the number of days since 1/1/0001, and can be read or written.

## date.new()

### Description

Creates a new date object and sets its initial value.

## Prototype

*date*.new ( integer *date* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| date | .nul | integer | The day count to be assigned to the new object. |

# Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the date type object. |

# Methods

## dayinmonth()

### Description

Returns the day of the month for the date in the object.

### Prototype

*datevar*.dayinmonth ()

### Parameters

None

## dayinweek()

### Description

Returns the day of the week for the date in the object, where 1 is Monday and 7 is Sunday.

### Prototype

*datevar*.dayinweek ()

### Parameters

None

## dayinyear()

### Description

Returns the day in the year for the date in the object, where January 1st is day 1.

### Prototype

*datevar*.dayinyear ()

### Parameters

None

## month()

### Description

Returns the month number for the date in the object.

### Prototype

*datevar*.month ( boolean *total* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | .true | boolean | Indicates whether months are to be counted from the beginning of the current year, or from the beginning of year 1. If .false then the returned value will range from 1 for January to 12 for December. If .true then the returned value will range from 1 for January 0001 to 119988 for December 9999. If the value of this parameter is either .nul or .inf then the result will be .nul. |

## set()

### Description

Sets the value in a date object from the components specified. The year component must always be given. Also, either the dayinyear (and not the month or dayinmonth) or the month and dayinmonth (and not the dayinyear) components must be given.

### Prototype

*datevar*.set ( integer *year*, integer *month*, integer *dayinmonth*, integer *dayinyear* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| year | None | integer | Specifies the year component of the new date value, which must be in the range 1 to 9999. If the value of this parameter is .nul then the date will be set to .nul and if the |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | value of this parameter is `.inf` then the date will be set to `.inf`. |
| month | `.nul` | integer | Specifies the month component of the new date value, which must be in the range 1 to 12. If the value of this parameter is `.nul` then the date will be set to `.nul` and if the value of this parameter is `.inf` then the date will be set to `.inf`. |
| dayinmonth | `.nul` | integer | Specifies the day of the month of the new date value, which must be in the range 1 to 28, 29, 30 or 31 depending on the year and month that are specified. If the value of this parameter is `.nul` then the date will be set to `.nul` and if the value of this parameter is `.inf` then the date will be set to `.inf`. |
| dayinyear | `.nul` | integer | Specifies the day in the year of the new date value, which must be in the range 1 to 365 or 366 depending on whether or not the year specified is a leap year. If the value of this parameter is `.nul` then the date will be set to `.nul` and if the value of this parameter is `.inf` then the date will be set to `.inf`. |

# setnow()

## Description

Sets the specified object to now according to the OS and returns the value of now for the date type.

## Prototype

*datevar*.setnow ()

## Parameters

None

# year()

## Description

Returns the year for the date in the object.

## Prototype

*datevar*.year ()

### Parameters

None

# datetime

## Description

Objects of type datetime contain a value which represents a date and a time of day, and can be accessed either as a whole or by component, such as month, dayinmonth, hour and minute. Valid datetimes are from midnight at the beginning of 1/1/0001 to 23:59:59.999999 on 31/12/9999.

## Type Tags

None

## Object Value

The value of an object of type datetime is the number of microseconds since midnight at the beginning of 1/1/0001 and can be read or written.

## datetime.new()

### Description

Creates a new datetime object and sets its initial value.

### Prototype

*`datetime`*.new ( integer *`datetime`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datetime | `.nul` | integer | The microsecond count to be assigned to the new object. |

## Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the datetime type object. |

## Methods

### dayinmonth()

#### Description

Returns the day of the month for the date in the object.

### Prototype

*datetimevar*.dayinmonth ()

### Parameters

None

## dayinweek()

### Description

Returns the day of the week for the date in the object, where 1 is Monday and 7 is Sunday.

### Prototype

*datetimevar*.dayinweek ()

### Parameters

None

## dayinyear()

### Description

Returns the day in the year for the date in the object, where January 1st is day 1.

### Prototype

*datetimevar*.dayinyear ()

### Parameters

None

## hours()

### Description

Returns the number of hours in the datetime value.

### Prototype

*datetimevar*.hours ( boolean *total* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | .true | boolean | Indicates whether hours are to be counted from time zero or from |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | the last whole day. If `.false` then the hours component of the datetime value is returned. If `.true` then the total number of hours in the datetime value is returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# microseconds()

## Description

Returns the number of microseconds in the datetime value.

## Prototype

*`datetimevar`*`.microseconds` ( boolean *`total`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | `.true` | boolean | Indicates whether microseconds are to be counted from time zero or from the last whole millisecond. If `.false` then the microseconds component of the datetime value is returned. If `.true` then the total number of microseconds in the datetime value is returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# milliseconds()

## Description

Returns the number of milliseconds in the datetime value.

## Prototype

*`datetimevar`*`.milliseconds` ( boolean *`total`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | `.true` | boolean | Indicates whether milliseconds are to be counted from time ze- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ro or from the last whole second. If `.false` then the milliseconds component of the datetime value is returned. If `.true` then the total number of milliseconds in the datetime value is returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# minutes()

## Description

Returns the number of minutes in the datetime value.

## Prototype

*`datetimevar`*`.minutes` ( boolean *`total`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | `.true` | boolean | Indicates whether minutes are to be counted from time zero or from the last whole hour. If `.false` then the minutes component of the datetime value is returned. If `.true` then the total number of minutes in the datetime value is returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# month()

## Description

Returns the month number for the date in the object.

## Prototype

*`datetimevar`*`.month` ( boolean *`total`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | `.true` | boolean | Indicates whether months are to be counted from the beginning |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | of the current year, or from the beginning of year 1. If `.false` then the returned value will range from 1 for January to 12 for December. If `.true` then the returned value will range from 1 for January 0001 to 119988 for December 9999. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# seconds()

## Description

Returns the number of seconds in the datetime value.

## Prototype

*`datetimevar`*`.seconds` ( boolean *`total`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | `.true` | boolean | Indicates whether seconds are to be counted from time zero or from the last whole minute. If `.false` then the seconds component of the datetime value is returned. If `.true` then the total number of seconds in the datetime value is returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# set()

## Description

Sets the value in a datetime object from the components specified. If the `year` component is specified then either the `dayinyear` or the `dayinmonth`) and the `month` components must be specified, but not both. If the year component is not specified then none of the month, dayinmonth or dayinyear components may be specified. Any or all or the microseconds, milliseconds, seconds, minutes and hours components may be specified. To specify the time components, it is necessary to supply the parameter name of the component, such as `dt.set(hours=5)`.

## Prototype

*`datetimevar`*`.set` ( integer *`year`*, integer *`month`*, integer *`dayinmonth`*, integer *`dayinyear`*, integer *`microseconds`*, integer *`milliseconds`*, integer *`seconds`*, integer *`minutes`*, integer *`hours`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| year | `.nul` | integer | Specifies the year component of the new datetime value, which must be in the range 1 to 9999. If the value of this parameter is `.nul` then the datetime will be set to `.nul` and if the value of this parameter is `.inf` then the datetime will be set to `.inf`. |
| month | `.nul` | integer | Specifies the month component of the new datetime value, which must be in the range 1 to 12. If the value of this parameter is `.nul` then the datetime will be set to `.nul` and if the value of this parameter is `.inf` then the datetime will be set to `.inf`. |
| dayinmonth | `.nul` | integer | Specifies the day of the month of the new datetime value, which must be in the range 1 to 28, 29, 30 or 31 depending on the year and month that are specified. If the value of this parameter is `.nul` then the datetime will be set to `.nul` and if the value of this parameter is `.inf` then the datetime will be set to `.inf`. |
| dayinyear | `.nul` | integer | Specifies the day in the year of the new datetime value, which must be in the range 1 to 365 or 366 depending on whether or not the year specified is a leap year. If the value of this parameter is `.nul` then the datetime will be set to `.nul` and if the value of this parameter is `.inf` then the datetime will be set to `.inf`. |
| microseconds | `.nul` | integer | Specifies the microseconds component of the new datetime value, which must be in the range 0 to 999. If the value of this parameter is `.nul` then the datetime will be set to `.nul` and if the value of this parameter is `.inf` then the datetime will be set to `.inf`. |
| milliseconds | `.nul` | integer | Specifies the milliseconds component of the new datetime value, which must be in the range 0 to 999. If the value of this parame- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ter is `.nul` then the datetime will be set to `.nul` and if the value of this parameter is `.inf` then the datetime will be set to `.inf`. |
| seconds | `.nul` | integer | Specifies the seconds component of the new datetime value, which must be in the range 0 to 59. If the value of this parameter is `.nul` then the datetime will be set to `.nul` and if the value of this parameter is `.inf` then the datetime will be set to `.inf`. |
| minutes | `.nul` | integer | Specifies the minutes component of the new datetime value, which must be in the range 0 to 59. If the value of this parameter is `.nul` then the datetime will be set to `.nul` and if the value of this parameter is `.inf` then the datetime will be set to `.inf`. |
| hours | `.nul` | integer | Specifies the hours component of the new datetime value, which must be in the range 0 to 87649415. If the value of this parameter is `.nul` then the datetime will be set to `.nul` and if the value of this parameter is `.inf` then the datetime will be set to `.inf`. |

# setnow()

## Description

Sets the specified object to now according to the OS and returns the value of now for the datetime type.

## Prototype

*`datetimevar`*`.setnow` ()

## Parameters

None

# year()

## Description

Returns the year for the date in the object.

## Prototype

*`datetimevar`*`.year` ()

### Parameters

None

# event

## Description

The event object is a type that provides a place for the user to assign a function reference and an optional reference to any type. This event can be named whatever the user wishes and is then called as required. For example, if a type is created that contains a database record, then the creator of the type could add an event object called `onsave`. Whenever the `save()` method is called, the code can check to see if the `onsave.function` is assigned and if it is, call that first, passing the appropriate parameters (as defined by the type designer).

## Type Tags

None

## Object Value

The value of an object of type event is undefined and it is an error to attempt to get or to set it.

## Properties

| Property | Type | Description |
|---|---|---|
| function | function | This should be assigned a reference to the function that should be called when the event is fired. |
| reference | type(*) | This can optionally be assigned a reference to any type. It will be passed as the final parameter to the function if it is assigned to an object. |
| type | type | Specifies the event type object. |

# fixedpoint

## Description

The fixedpoint object is a version of a point containing an x and a y coordinate and which can be read but not altered. This type is one of the base types of the wxgraphic drawing support.

## Type Tags

point

## Object Value

The value of a fixedpoint object is a unique value that can be used to assign the coordinates of the point to another point object. The fixedpoint object cannot have its value changed by assignment. The value is compatible between point and fixedpoint objects.

# fixedpoint.new()

## Description

Creates a new fixedpoint object.

## Prototype

*fixedpoint*.new ( integer *x*, integer *y* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| x | None | integer | The value that is used for the x coordinate of the object. |
| y | None | integer | The value that is used for the y coordinate of the object. |

# Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the fixedpoint type object. |
| x | integer | Specifies the x coordinate of the point. |
| y | integer | Specifies the y coordinate of the point. |

# fsfileinputstream

# Description

Provides an interface to open a file in a file system for sequential reading.

# Type Tags

None

# Object Value

The value of an object of type fsfileinputstream is undefined and it is an error to attempt to get or to set it.

# fsfileinputstream.new()

## Description

Opens the specified file system file for reading and creates a new object.

## Prototype

*fsfileinputstream*.new ( string *filename*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | .nul | string | The name of the file to open. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns .nul. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| endofdata | boolean | Contains the value .false until reading has failed because the end of the file has been found, at which point it is set to .true. |
| filename | string | Contains the name of the file that is being read. |
| type | type | Specifies the fsfileinputstream type object. |

# Methods

## getblob()

### Description

Returns a blob containing bytes read from the file.

### Prototype

*fsfileinputstreamvar*.getblob ( integer *length* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| length | `1` | integer | The number of bytes to read from the file. If this value is `.nul` then the result will also be `.nul`. If the value is `.inf`, then bytes will be read until the end of the file. |

## getinteger()

### Description

Returns an integer read from the file.

### Prototype

*`fsfileinputstreamvar`*`.getinteger` ( integer *`length`*, boolean *`lbo`*, boolean *`signed`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| length | `1` | integer | The number of bytes to read from the file. If this value is `.nul` then the result will also be `.nul`. If the value is `.inf`, then bytes will be read until the end of the file and then interpreted as a single integer value. |
| lbo | `.true` | boolean | Indicates whether integers have been stored with the least significant byte first (`.true`) or with the most significant byte first (`.false`). If this value is `.nul` or `.inf` then the result will also be `.nul`. |
| signed | `.false` | boolean | Indicates whether the bytes will be interpreted as a signed or unsigned integer. If this value is either `.nul` or `.inf` then the result will also be `.nul`. |

## getstring()

### Description

Returns a string containing characters read from the file. If the string is stopped by a terminator (see the 'terminator' parameter) then the terminating character is not included in the returned string.

### Prototype

*`fsfileinputstreamvar`*`.getstring` ( integer *`length`*, integer *`charsize`*, boolean *`lbo`*, string *`terminator`*, string *`terminatedby`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| length | `1` | integer | The number of characters to read from the file. If this value is `.nul` then the result will also be `.nul`. If the value is `.inf`, then characters will be read to the end of the file, or until a terminator is found (see the `terminator` parameter). |
| charsize | `2` | integer | The number of bytes per character, which must be greater than 0. If this value is either `.nul` or `.inf` then the result will be `.nul`. |
| lbo | `.true` | boolean | Indicates whether characters have been stored with the least significant byte first (`.true`) or with the most significant byte first (`.false`). If this value is `.nul` or `.inf` then the result will be `.nul`. |
| terminator | `""` | string | A string made up of all the characters which can terminate the string being read. As soon as any one character in the string of terminators is read then reading stops and the string is returned. If this value is `.nul` then no character will terminate the reading. If it is `.inf` then the result will be `.nul`. |
| terminatedby | `.nul` | string | This parameter must be an object, not a string value such as a literal or expression. If the reading of the string is terminated by finding a character in the *`terminator`* string then that character is returned in the *`terminatedby`* object. |

# fsfileoutputstream

## Description

Provides an interface to open a file in a file system for sequential writing. If the file already exists then all existing contents are deleted unless the *`append`* parameter is set to `.true`.

# Type Tags

None

# Object Value

The value of an object of type fsfileoutputstream is undefined and it is an error to attempt to get or to set it.

# fsfileoutputstream.new()

## Description

Opens the specified file system file for writing and creates a new object.

## Prototype

*`fsfileoutputstream`*.new ( string *`filename`*, integer *`error`*, boolean *`append`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | `.nul` | string | The name of the file to open or create. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |
| append | `.false` | boolean | A boolean value, the default of which is `.false`, that indicates whether or not to start output at the end of existing data, or to truncate and start at the new beginning. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| filename | string | Contains the name of the file to which is being written. |
| type | type | Specifies the fsfileoutputstream type object. |

# Methods

## putblob()

### Description

Outputs a blob to the file and returns the number of bytes actually written to the file.

### Prototype

*`fsfileoutputstreamvar`*`.putblob` ( blob *`blob`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| blob | None | blob | The blob to be output. If this value is `.nul` or `.inf` then no bytes will be written. |

## putinteger()

### Description

Writes an integer to a file and returns the number of bytes written.

### Prototype

*`fsfileoutputstreamvar`*`.putinteger` ( integer *`integer`*, integer *`length`*, boolean *`lbo`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| integer | None | integer | The integer value to be output. If this value is `.nul` or `.inf` then no bytes will be written. |
| length | `1` | integer | The number of bytes to write to the file. If this value is `.nul` or |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | `.inf` then no bytes will be written. |
| lbo | `.true` | boolean | Indicates whether integers are to be stored with the least significant byte first (`.true`) or with the most significant byte first (`.false`). If this value is `.nul` or `.inf` then no bytes will be written. |

## putstring()

### Description

Outputs a string to the file and returns the number of bytes actually written to the file.

### Prototype

*`fsfileoutputstreamvar`*`.putstring` ( string *`string`*, integer *`charsize`*, boolean *`lbo`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| string | None | string | The string value to be output. If this value is `.nul` or `.inf` then no characters will be output. |
| charsize | `2` | integer | The number of bytes per character, which must be greater than 0. If this value is `.nul` or `.inf` then no characters will be written. |
| lbo | `.true` | boolean | Indicates whether characters are to be stored with the least significant byte first (`.true`) or with the most significant byte first (`.false`). If this value is `.nul` or `.inf` then no characters will be written. |

# function

# Description

The function object contains a reference to the module in which it is defined as well as to the type of which it is a part if it is a method of a type.

# Type Tags

None

## Object Value

The value of a function object is the name of the function.

## Properties

| Property | Type | Description |
|---|---|---|
| information | string | Contains the string that was associated with the function object when it was compiled. Although this can contain anything, it is recommended to watch for guidelines on how to use this capability. Initially, it will be used for defining the return value of a function (if any). This will be in the format: `[simpol::return::string]` that is formatted to use a name space style, where simpol is the name space, return is the type of information and string is the actual value. |
| intype | type | Contains a reference to the type object of which this is a method. This relationship is similar to that of a form control with a reference to the form that it is on, or a database field with a refence to the table of which the field is a part. If the function is a free function and not a method of a type, then this will be equal to `.nul`. |
| module | module | Contains a reference to the module within which the function is defined. |
| next | function | Contains a reference to the next function in the ring of functions. |
| type | type | Specifies the function type object. |

# integer

## Description

The integer object is considered to be a value type and does not require a call to a `new()` method in order to be initialized; normal assignment using the assignment operator = is sufficient.

## Type Tags

None

## Object Value

The value of an integer object is its content.

## Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the integer type object. |

# lock1

## Description

`lock1` is an intrinsic object type, where objects can be created using **lock1.new()** or by embedding. `lock1` objects are locked by threads, either as an exclusive lock or a shared lock. This object is used to control access to a resource (an object referenced via a variable) that may be shared by more than one thread concurrently.

## Type Tags

None

## Object Value

The value of an object of type lock1 is undefined and it is an error to attempt to get or to set it.

## lock1.new()

### Description

A lock created using `lock1.new()` can be *pre-locked*. This lock is exclusive and anonymous in the sense that no thread holds the lock, but any thread can release it. The return value is a reference to the new object. Embedded lock1 objects are created without a lock.

### Prototype

*`lock1`*.new ( boolean *`locked`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| locked | `.false` | boolean | If this is equal to `.true` then the object is locked upon creation. |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| type | type | Specifies the lock1 type object. |

# Methods

## getexclusive()

### Description

The lock will be acquired if no other thread has any lock on this object and there is no anonymous lock on it. A thread can have any number of exclusive locks on each `lock1` object, but a single exclusive lock by one thread prevents any sort of lock by any other thread. A thread can have both exclusive and shared locks at the same time. If no lock can be acquired immediately and the value of the `block` parameter is `.true` then the thread will not continue until the lock can be acquired. If the lock can never be acquired because one or more other locks are held by threads that cannot release them then the program halts with the error number 73: deadly embrace. The return value is `.true` or `.false`, indicating whether or not the lock was acquired.

### Prototype

*`lock1var`*`.getexclusive` ( boolean *`block`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| block | `.true` | boolean | If the value of this parameter is `.true` then the thread will block until it can acquire the exclusive lock. |

## getshared()

### Description

The lock will be acquired if no other thread has an exclusive lock on this object and there is no anonymous lock on it. A thread can have any number of shared locks on each `lock1` object, and any number of threads can also hold shared locks on the same object at the same time, but a single exclusive lock by one thread prevents any shared lock by any other thread. A thread can have both exclusive and shared locks at the same time. If no lock can be acquired immediately and the value of the `block` parameter is `.true` then the thread will not continue until the lock can be acquired. If the lock can never be acquired because one or more exclusive locks are held by threads that cannot release them then the program halts with the error number 73: deadly embrace. The return value is `.true` or `.false`, indicating whether or not the lock was acquired.

### Prototype

*`lock1var`*`.getshared` ( boolean *`block`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| block | `.true` | boolean | If the value of this parameter is `.true` then the thread will block |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | until it can acquire the shared lock. |

## release()

### Description

This method releases the most recently acquired lock for this thread, or releases the anonymous lock granted at creation time of the lock1 object. If no lock exists that the current thread can release then a fatal error is raised.

### Prototype

*lock1var*.release()

### Parameters

None

# module

# Description

The module object contains a reference to all of the types and free functions that are defined within the module.

# Type Tags

None

# Object Value

Module objects do not have a value and it is an error to attempt to get or set it.

# Properties

| Property | Type | Description |
|---|---|---|
| firstfunction | function | Contains a reference to the first function in the ring of free functions. |
| firsttype | type | Contains a reference to the first type object in the ring of type objects. |
| next | module | Contains a reference to the next module in the ring of modules. Each C-language component normally represents a single module, as does each SIMPOL-language compiled library or program. |
| type | type | Specifies the module type object. |

| Property | Type | Description |
|---|---|---|
| version | string | Contains the version string for the module. |

# number

## Description

The number object is considered to be a value type and does not require a call to a `new()` method in order to be initialized; normal assignment using the assignment operator `=` is sufficient.

## Type Tags

None

## Object Value

The value of a number object is its content.

## Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the number type object. |

# point

## Description

The point object is a version of a point containing an x and a y coordinate and which can be both read and modified. This type is one of the base types of the wxgraphic drawing support.

## Type Tags

point

## Object Value

The value of a point object is a unique value that can be used to assign the coordinates of the point to another point object. The value is compatible between point and fixedpoint objects.

## point.new()

### Description

Creates a new fixedpoint object.

### Prototype

*`point`*`.new` ( integer *`x`*, integer *`y`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| x | None | integer | The value that is used for the x coordinate of the object. |
| y | None | integer | The value that is used for the y coordinate of the object. |

## Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the point type object. |
| x | integer | Specifies the x coordinate of the point. |
| y | integer | Specifies the y coordinate of the point. |

# property

## Description

The property object is another core feature of the SIMPOL programming language. Every type has properties that are of the type property and which have a contenttype that describes the content of the property.

## Type Tags

None

## Object Value

The value of a property object is the name of the property.

## Properties

| Property | Type | Description |
|---|---|---|
| contenttype | type | Contains a reference to the type object that describes the data type of the content of the property (such as string, or boolean). |
| intype | type | Contains a reference to the type object of which this is a property. This relationship is similar to that of a form control with a reference to the form that it is on, or a database field with a refence to the table of which the field is a part. |
| next | property | Contains a reference to the next property in the ring of properties within a given type. This also includes methods. The properties are held in a ring. |
| type | type | Specifies the property type object. |

# set

## Description

An object of this type provides the ability to store values and objects in a way which can be managed using traditional set operations

## Type Tags

None

## Object Value

The value of an object of type set is undefined and it is an error to attempt to get or to set it.

## set.new()

### Description

Creates a new set object.

### Prototype

*set*.new ( string *duplicates* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| duplicates | `"allow"` | string | Specifies how the new set will respond to attempts to add more than one copy of a value or object to it. If set to "allow" then more than one copy of a value or object can be stored, and a count of how many copies there are is retained. If set to "ignore" then an attempt to add a second or subseqent copy of a value or object is ignored, with the count of the number of copies of that element being kept at one. If set to "prevent" it is an error to attempt to add a second or subsequent copy of a value or object to the set. |

## Properties

| Property | Type | Description |
|---|---|---|
| duplicates | string | Specifies how the set responds to attempts to add more than one copy of a value or object to it. If set to "allow" then more than one copy of a value |

| Property | Type | Description |
|---|---|---|
| | | or object can be stored, and a count of how many copies there are is retained. If set to "ignore" then an attempt to add a second or subseqent copy of a value or object is ignored, with the count of the number of copies of that element being kept at one. If set to "prevent" it is an error to attempt to add a second or subsequent copy of a value or object to the set. |
| type | type | Specifies the set type object. |

# Methods

## addobject()

### Description

Adds an object to the set, and returns the set object itself.

### Prototype

*setvar*.addobject ( type(*) *object*, integer *count*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| object | None | type(*) | This is the object to be added to the set. |
| count | 1 | integer | The number of copies of the object to add to the set. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

## addvalue()

### Description

Adds a value to the set, and returns the set object itself.

### Prototype

*setvar*.addvalue ( integer|string|blob|boolean|number *value*, integer *count*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | integer\|string\|blob\|boolean\|number | This is the value to be added to the set. The values `.nul` and `.inf` are also permitted. |
| count | `1` | integer | The number of copies of the value to add to the set. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

## count()

### Description

Returns the number of unique elements in the set.

### Prototype

*`setvar`*.`count` ()

### Parameters

None

## difference()

### Description

Establishes the difference between the set for which the method is called and the set passed as an argument. The set for which the method is called will then contain only elements (and copies of elements) which previously had been in one set but not both. The set itself is returned.

### Prototype

*`setvar`*.`difference` ( set *`set`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| set | None | set | The set that is to be differenced with the set for which the method is called. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

# intersect()

## Description

Intersects the set with a set passed as an argument. The set for which the method is called will then only contain elements (and copies of elements) that are found in both sets. The set itself is returned.

## Prototype

*`setvar`*`.intersect` ( set *`set`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| set | None | set | The set that is to be intersected with the set for which the method is called. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

# objectcount()

## Description

Returns the number of copies of a specified object in the set. If the object is not in the set then `0` is returned.

## Prototype

*`setvar`*`.objectcount` ( type(*) *`object`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| object | None | type(*) | The object to return the number of copies of. |

## removeobject()

### Description

Removes an object from the set, and returns the set object itself.

### Prototype

*setvar*.removeobject ( type(*) *object*, integer *count*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| object | None | type(*) | This is the object to be removed from the set. |
| count | 1 | integer | The number of copies of the object to be removed from the set. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

## removevalue()

### Description

Removes a value from the set, and returns the set object itself.

### Prototype

*setvar*.removevalue ( integer|string|blob|boolean|number *value*, integer *count*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | integer\|string\|blob\|boolean\|number | This is the value to be removed from the set. The values .nul and .inf are also permitted. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| count | `1` | integer | The number of copies of the value to be removed from the set. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

# subtract()

## Description

Subtracts a set passed as an argument from the set for which the method is called. The set for which the method is called will then contain only elements (and copies of elements) which were in the set before calling the method, but were not in the set passed as an argument. The set itself is returned.

## Prototype

*`setvar`*`.subtract` ( set *`set`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| set | None | set | The the set that is to be subtracted from the set for which the method is called. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

# totalcount()

## Description

Returns the total number of elements in the set, including all copies of elements.

### Prototype

*setvar*.totalcount ()

### Parameters

None

## unite()

### Description

Unites the set with a set passed as an argument. The set itself is returned.

### Prototype

*setvar*.unite ( set *set*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| set | None | set | The set that is to be united with the set for which the method is called. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

## valuecount()

### Description

Returns the number of copies of a specified value in the set. If the value is not in the set then 0 is returned.

### Prototype

*setvar*.valuecount ( integer|string|blob|boolean|number *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | integer\|string\|blob\|boolean\|number | The value to return the number of copies of. The values .nul and .inf are also permitted. |

# set[]

## Get

### Subscripts

An integer value giving the index position of an element in the set.

### Description

Retrieves the value or object at the specified index position.

## Set

Attempting to set the value is not supported.

## Set Reference

Attempting to set a reference to an object is not supported.

# string

# Description

The string object is considered to be a value type and does not require a call to a `new()` method in order to be initialized; normal assignment using the assignment operator = is sufficient.

# Type Tags

None

# Object Value

The value of a string object is its content.

# Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the string type object. |

# time

# Description

Objects of type time contain a value which represents a time of day or a time interval, and can be accessed either as a whole or by component, such as hour, minute and second. Valid times are from zero to 9999 years.

# Type Tags

None

# Object Value

The value of an object of type time is the number of microseconds in the time value, and can be read or written.

# time.new()

## Description

Creates a new time object and sets its initial value.

## Prototype

*time*.new ( integer *time* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| time | .nul | integer | The microsecond count to be assigned to the new object. |

# Properties

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the time type object. |

# Methods

## hours()

### Description

Returns the number of hours in the time value.

### Prototype

*timevar*.hours ( boolean *total* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | .true | boolean | Indicates whether hours are to be counted from time zero or from the last whole day. If .false |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | then the hours component of the time value is returned. If `.true` then the total number of hours in the time value is returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# microseconds()

## Description

Returns the number of microseconds in the time value.

## Prototype

*timevar*.microseconds ( boolean *total* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | `.true` | boolean | Indicates whether microseconds are to be counted from time zero or from the last whole millisecond. If `.false` then the microseconds component of the time value is returned. If `.true` then the total number of microseconds in the time value is returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# milliseconds()

## Description

Returns the number of milliseconds in the time value.

## Prototype

*timevar*.milliseconds ( boolean *total* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | `.true` | boolean | Indicates whether milliseconds are to be counted from time zero or from the last whole second. If `.false` then the milliseconds component of the time value is |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | returned. If `.true` then the total number of milliseconds in the time value is returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# minutes()

## Description

Returns the number of minutes in the time value.

## Prototype

*`timevar`*`.minutes` ( boolean *`total`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | `.true` | boolean | Indicates whether minutes are to be counted from time zero or from the last whole hour. If `.false` then the minutes component of the time value is returned. If `.true` then the total number of minutes in the time value is returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# seconds()

## Description

Returns the number of seconds in the time value.

## Prototype

*`timevar`*`.seconds` ( boolean *`total`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| total | `.true` | boolean | Indicates whether seconds are to be counted from time zero or from the last whole minute. If `.false` then the seconds component of the time value is returned. If `.true` then the total number of seconds in the time value is |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | returned. If the value of this parameter is either `.nul` or `.inf` then the result will be `.nul`. |

# set()

## Description

Sets or changes the value in a time object from the components specified. Any or all components can be specified.

## Prototype

*`timevar`*`.set` ( integer *`microseconds`*, integer *`milliseconds`*, integer *`seconds`*, integer *`minutes`*, integer *`hours`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| microseconds | `.nul` | integer | Specifies the microseconds component of the new time value, which must be in the range 0 to 999. If the value of this parameter is `.nul` then the time will be set to `.nul` and if the value of this parameter is `.inf` then the time will be set to `.inf`. |
| milliseconds | `.nul` | integer | Specifies the milliseconds component of the new time value, which must be in the range 0 to 999. If the value of this parameter is `.nul` then the time will be set to `.nul` and if the value of this parameter is `.inf` then the time will be set to `.inf`. |
| seconds | `.nul` | integer | Specifies the seconds component of the new time value, which must be in the range 0 to 59. If the value of this parameter is `.nul` then the time will be set to `.nul` and if the value of this parameter is `.inf` then the time will be set to `.inf`. |
| minutes | `.nul` | integer | Specifies the minutes component of the new time value, which must be in the range 0 to 59. If the value of this parameter is `.nul` then the time will be set to `.nul` and if the value of this parameter is `.inf` then the time will be set to `.inf`. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| hours | `.nul` | integer | Specifies the hours component of the new time value, which must be in the range 0 to 87649415. If the value of this parameter is `.nul` then the time will be set to `.nul` and if the value of this parameter is `.inf` then the time will be set to `.inf`. |

### setnow()

#### Description

Sets the specified object to now according to the OS and returns the value of now for the time type.

#### Prototype

*timevar*.setnow ()

#### Parameters

None

# type

## Description

The type object is the core of SIMPOL. Every other type has a type object that represents it and which is of the data type type. Even properties and functions are represented by type objects.

## Type Tags

None

## Object Value

The value of an type object is the name of the type.

## type.new()

### Description

Creates a new instance of the type for which the method is called. If this is a user-defined type and the user has created a `new()` method for the type, then after the object has been created it will be passed together with the other arguments to the user's `new()` method. The basic `new()` method has no parameters.

### Prototype

*type*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| firstfunction | function | Contains a reference to the first method of the type. The methods are held in a ring. |
| firstproperty | property | Contains a reference to the first property of the type. The properties are held in a ring. They include both properties and methods. |
| module | module | Contains a reference to the module within which the type is defined. |
| next | type | Contains a reference to the next type in the ring of types within a given module. The types are held in a ring. |
| type | type | Specifies the type type object. |

# Chapter 5. The Peer-to-Peer Client/ Server (PPCS) Component

Provides access to database files via the Superbase Peer-to-Peer Client/Server protocol. This access is read/write/delete but does not provide the ability to modify the database definition or create or modify database tables.

# ppcstype1

## Description

Objects of type ppcstype1 are a base from which data available from some PPCS server can be accessed. Each object of type ppcstype1 uses one communication medium such as UDP or a serial port.

## Type Tags

db1base

## Object Value

Objects of type ppcstype1 have no value, and it is an error to try to get or set this value.

# ppcstype1.new()

## Description

Creates a new ppcstype1 object and specifies the nature of the comunication method it will use. Parameters must be specified for exactly one of the possible media.

## Prototype

*ppcstype1*.new ( integer *udpport*, integer *txfactor*, integer *serialport*, integer *baudrate*, integer *parity*, integer *stopbits*, integer *error*, string *username* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| udpport | None | integer | The UDP port number to use for the new ppcstype1 object. If this is specified then no parameters for other communication media should be given. If *udpport* is assigned the value `.nul` then a random port will be assigned. |
| txfactor | `0` | integer | The transmission delay to use when communicating. Generally this does not need to be specified. |
| serialport | None | integer | The serial port to use for the new ppcstype1 object. If this is speci- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | fied then no parameters for other communication media should be given. |
| baudrate | 19200 | integer | The baud rate to use if the new ppcstype1 object is to use serial communications. |
| parity | 0 | integer | The parity to use for serial communications. Permitted values are: 0 (no parity), 1 (odd parity) and 2 (even parity). |
| stopbits | 1 | integer | The number of stop bits to use in serial communications. Permitted values are 1 and 2. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |
| username | platform dependent | string | The username must be at least one character in length and cannot be greater than eighteen characters in length. The username for a ppcstype1 object is the username used for all operations performed on files opened using the ppcstype1 object. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| parity | integer | Contains the parity setting being used, if the object uses serial communications, or `.nul` if not. |

| Property | Type | Description |
|---|---|---|
| serialport | integer | Contains the number of the serial port in use, if the object uses communications, or `.nul` if not. |
| stopbits | integer | Contains the number of stop bits being used, if the object uses serial communications, or `.nul` if not. |
| txfactor | integer | Contains a transmission factor value that can be used to slow down the communication with the other end of the PPCS connection. |
| type | type | Specifies the ppcstype1 type object. |
| udpport | integer | Contains the UDP port number if the object uses UDP, or `.nul` if not. |
| username | string | Contains the username for all PPCS operations performed using the `ppcstype1` object. |

# Methods

## gettablenames()

### Description

Gets the names of the PPCS files which are available on a specified PPCS server. If the attempt to get the table names is successful then `.nul` is returned. If the attempt is not successful and neither the *`error`* nor *`errortext`* parameters are provided then an error is raised. If the attempt is not successful and at least one of the *`error`* or *`errortext`* parameters are provided then `.nul` is returned and error information is output in the `error` or `errortext` objects provided.

### Prototype

*`ppcstype1var`*.`gettablenames` ( string *`address`*, array *`tablenames`*, string *`pattern`*, integer *`codepage`*, integer *`error`*, string *`errortext`*, integer *`retry`*, integer *`timeout`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| address | None | string | The internet address, machine name or IP address and port number of the PPCS server providing the file names to be retrieved. |
| tablenames | None | array | Specifies an object of type array which is used to output the names of the files which are available from the specified server and which match the specified pattern, if any. The first table name, if any, is placed in the array in the position '[1]', the second in position '[2]', and so on. The number of table names retrieved in placed in position '[]'. |
| pattern | "" | string | A pattern used to restrict the table names which will be retrieved. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | Only names which match the pattern will be output. The matching of a name to a pattern follows the rules for the intrinsic function `.like1`. |
| codepage | 850 | integer | The number of the OEM codepage that will be assumed to be in use by the PPCS server providing the file. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during name retrieval will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during name retrieval will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# openudpfile()

## Description

Opens a database file on a PPCS server using UDP. The ppcstype1 object for which the method is called must be using UDP. If the attempt to open the file is successful then a reference to an object of type ppcstype1file is returned. If the attempt is not successful and neither the *error* nor *errortext* parameters are provided then an error is raised. If the attempt is not successful and at least one of the *error* or *errortext* parameters are provided then .nul is returned and error information is output in the error or errortext objects provided.

## Prototype

*ppcstype1var*.openudpfile ( string *address*, string *filename*, string *password*, integer *codepage*, integer *error*, string *errortext*, integer *retry*, integer *timeout*, string *recordid-fieldname* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| address | None | string | The internet address, machine name or IP address and port number of the PPCS server providing the file to be opened. |
| filename | None | string | The name of the file to open. |
| password | None | string | The password to be used to access the file. |
| codepage | 850 | integer | The number of the OEM codepage that will be assumed to be in use by the PPCS server providing the file. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns .nul. |
| errortext | .nul | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *error* is not specified or is |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |
| recordidfield-name | `.nul` | string | The name to use for the exposed internal unique record ID. If this is supplied then the internal record ID will be shown as an indexed unique field. The internal record id is read-only and guaranteed for the life of the record in the table. If the table is reorganized, the record ID is likely to change.<br>  <br>**Warning**<br>Do not store the record ID for future use! The record ID will certainly change if the table is reorganized so it should not be used for any long term purpose. Within the same session and assuming the same database table it can be used for safely re-selecting the same record, which makes it an excellent candidate for routines that search for duplicate records and such. |

## settxfactor()

### Description

Sets the transmission factor. This is used to moderate the speed at which the packets are transmitted. A factor of 0 means that the packets will be sent at the maximum speed (this may be too fast for Internet connectivity and result in difficulty reading data and excessive retries and timeouts).

### Prototype

*ppcstype1var*.settxfactor ( integer *txfactor* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| txfactor | None | integer | The factor to use to delay the server. The value 0 means no delay and the higher the number the longer the delay will become. Also each increment is a greater delay than the one before, so a setting of 3 is slower by a greater amount than just the total of the delay at 1 plus 2. |

# ppcstype1field

# Description

Objects of type ppcstype1field represent a field in a database file in a PPCS database file, and provide information that may be needed about the field.

# Type Tags

db1field

# Object Value

Objects of type ppcstype1field have no value, and it is an error to try to get or set this value.

# Properties

| Property | Type | Description |
|---|---|---|
| SBLdisplayformat | string | For a non-text field this contains the format string for the field. If the field is a text (string) field, then the boolean value is .nul. |
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of |

| Property | Type | Description |
|---|---|---|
| | | being marked with the resolve keyword, so that object resolution can continue down this property. |
| allowreturns | boolean | Contains a boolean indicator showing whether or not the field is intended to contain carriage returns in its contents. If the field is not a text (string) field, then the boolean value is `.nul`. |
| comment | string | Contains the comment for the field. |
| datatype | type | Contains a reference to the type object that describes the type of data that the field contains. |
| file | ppcstype1file | Contains a reference to the ppcstype1file object for the file that contains this field. This is the same as the table property. |
| help | string | Contains the custom help for the field. |
| index | ppcstype1index | If this field is indexed then this property contains a reference to the ppcstype1index object for that index, otherwise it contains `.nul`. |
| logical | boolean | Contains a boolean indicator showing whether or not the field is intended to contain a logical flag. If the field is not a text (string) field, then the boolean value is `.nul`. |
| maxlength | integer | Contains the maximum length of text that the field can contain. If the field is not a text (string) field, then the value is `.nul`. |
| name | string | Contains the name of the field. |
| next | ppcstype1field | Contains a reference to the next ppcstype1field object for the file. The ppcstype1field objects for a table form a closed loop that is linked by this property. |
| readonly | boolean | Contains a boolean indicator showing whether or not the field is read only. A field that is read only can have values assigned to it but is not intended to allow input by the user. |
| required | boolean | Contains a boolean indicator showing whether or not the field has to contain a value before the record can be saved. |
| table | ppcstype1file | Contains a reference to the ppcstype1file object for the file that contains this field. This is the same as the file property. |
| type | type | Specifies the ppcstype1field type object. |

# ppcstype1file

## Description

Objects of type ppcstype1file represent a database file on a PPCS server, and provide the necessary interface to access the data in the database. Objects of this type also do not have a `new`() method, since they are created by a call to the `openudpfile`() or some other file opening method of the ppcstype1 object.

## Type Tags

db1table

## Object Value

Objects of type ppcstype1file have no value, and it is an error to try to get or set this value.

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| codepage | integer | Contains the OEM codepage number that the PPCS server is assumed to be using. |
| filename | string | Contains the name of the table. |
| firstfield | ppcstype1field | Contains a reference to the ppcstype1field object that represents the first field in the file. |
| firstindex | ppcstype1index | Contains a reference to the ppcstype1index object that represents the first index in the file. If there are no indexes on the file then the reference is `.nul`. |
| locked | boolean | Contains a boolean indicator showing whether or not all records in the file have been locked using this ppcstype1file object. |
| ppcstype | ppcstype1 | Contains a reference to the ppcstype1 object used to open the table. |
| recordidfield-name | string | Contains the name of the field (if any) that is the internal record ID field. |
| tablename | string | Contains the name of the table. |
| type | type | Specifies the ppcstype1file type object. |

# Methods

## lock()

### Description

Locks all records in a file. If the attempt to lock all records is successful then `.nul` is returned. If the attempt is not successful and neither the *`error`* nor *`errortext`* parameters are provided then an error is raised. If the attempt is not successful and at least one of the *`error`* or *`errortext`* parameters are provided then `.nul` is returned and error information is output in the `error` or `errortext` objects provided.

### Prototype

*`ppcstype1filevar`*`.lock` ( integer *`error`*, string *`errortext`*, integer *`retry`*, integer *`timeout`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# newrecord()

## Description

Creates a ppcstype1record object to represent a new (as yet not persistently stored) record in the database table.

## Prototype

*ppcstype1filevar*.newrecord ()

## Parameters

None

# recordcount()

## Description

Returns the number of records in a file. If the attempt to get the current record count is successful then that count is returned. If the attempt is not successful and neither the *error* nor the *errortext* parameter is provided then an error is raised. If the attempt is not successful and at least one of the *error* or *errortext* parameters are provided then .nul is returned and error information is output in the error or errortext objects provided.

## Prototype

*ppcstype1filevar*.recordcount ( boolean *get*, integer *error*, string *errortext*, integer *retry*, integer *timeout* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| get | .false | boolean | If .false then the most recent value that the object already has is returned. If *get* is .true then a PPCS operation is performed to determined the very latest value for the record count. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during object creation will halt the program. If an |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# select()

## Description

Selects either the first or the last record in a PPCS database file. The selection does not depend on any index or on the position of any other record. If the attempt to select a record is successful then a reference to an object of type ppcstype1record is returned. If the attempt is not successful and neither the *`error`* nor the *`errortext`* parameter is provided then an error is raised. If the attempt is not successful and the *`error`* or the *`errortext`* parameter is provided then `.nul` is returned and error information is output in the `error` and/or `errortext` object(s) provided. Providing the *`lock`* parameter will allow the record to be selected with a lock.

## Prototype

*`ppcstype1filevar`*`.select` ( boolean *`lastrecord`*, boolean *`lock`*, integer *`error`*, string *`errortext`*, integer *`retry`*, integer *`timeout`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lastrecord | `.false` | boolean | If `.false` then the first record in the file is selected. If `.true` then the last record in the file is selected. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lock | `.false` | boolean | If `.true` then the record is selected with a lock, which is necessary if the record is to be modified or deleted, or if modification or deletion by others is to be prevented. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# serial()

## Description

Returns the current serial number for a file. If the attempt to get the serial number is successful then that count is returned. If the attempt is not successful and neither the *`error`* nor the *`errortext`* parameter is provided then an error is raised. If the attempt is not successful and the *`error`* or the *`errortext`*

parameter is provided then `.nul` is returned and error information is output in the `error` and/or `errortext` object(s) provided.

## Prototype

*`ppcstype1filevar`*`.serial` ( boolean *`increment`*, integer *`error`*, string *`errortext`*, integer *`retry`*, integer *`timeout`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| increment | `.false` | boolean | If `.false` then the serial number is returned without being modified. If increment is `.true` then the value is incremented before it is retrieved, thus ensuring that the number retrieved is not the same as any serial number that has been retrieved previously. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# unlock()

## Description

Unlocks all records in a file after they have been locked using `ppcstype1file.lock()`. If the attempt to unlock all records is successful then `.nul` is returned. If the attempt is not successful and neither the *`error`* nor the *`errortext`* parameter is provided then an error is raised. If the attempt is not successful and the *`error`* or the *`errortext`* parameter is provided then `.nul` is returned and error information is output in the `error` and/or `errortext` object(s) provided.

## Prototype

*`ppcstype1filevar`*`.unlock` ( integer *`error`*, string *`errortext`*, integer *`retry`*, integer *`timeout`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# ppcstype1file!

## Get

The member operator followed by the name of the field will return a reference to the ppcstype1field object. If the name provided is not correct (including case-sensitivity), then an error will occur.

## Set

Attempting to assign a reference or a value to a ppcstype1file object using the ! operator is not supported.

# ppcstype1index

## Description

Objects of type ppcstype1index represent an index for a database file on a PPCS server, and provide information that may be needed about the index.

## Type Tags

db1index

## Object Value

Objects of type ppcstype1index have no value, and it is an error to try to get or set this value.

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |

| Property | Type | Description |
|---|---|---|
| field | ppcstype1field | Contains a reference to the ppcstype1field object upon which this index is based. For a ppcstype1file all indexes are on exactly one field. |
| file | ppcstype1file | Contains a reference to the ppcstype1file object for the file that contains this index. This is the same as the table property. |
| next | ppcstype1index | Contains a reference to the next `ppcstype1index` object for the table. The ppcstype1index objects for a file form a closed loop that is linked by this property. |
| table | ppcstype1file | Contains a reference to the ppcstype1file object for the file that contains this index. This is the same as the file property. |
| type | type | Specifies the ppcstype1index type object. |
| unique | boolean | If `.true` then the index contains only unique key values, otherwise duplicate values are allowed and the property value is `.false`. |

# Methods

## select()

### Description

Selects either the first or the last record in a PPCS database file according to this index. The selection does not depend on the position of any other record. If the attempt to select a record is successful then a reference to an object of type ppcstype1record is returned. If the attempt is not successful and neither the *`error`* nor the *`errortext`* parameter is provided then an error is raised. If the attempt is not successful and the *`error`* or the *`errortext`* parameter is provided then `.nul` is returned and error information is output in the `error` and/or `errortext` object(s) provided. Providing the *`lock`* parameter will allow the record to be selected with a lock.

### Prototype

*`ppcstype1indexvar`*.`select` ( boolean *`lastrecord`*, boolean *`lock`*, integer *`error`*, string *`errortext`*, integer *`retry`*, integer *`timeout`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lastrecord | `.false` | boolean | If `.false` then the first record in the index order is selected. If `.true` then the last record in the index order is selected. |
| lock | `.false` | boolean | If `.true` then the record is selected with a lock, which is necessary if the record is to be modified or deleted, or if modification or deletion by others is to be prevented. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during the PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# selectkey()

## Description

Selects a record according to the location of a specified value in the index. The selection does not depend on the position of any other record. If the attempt to select a record is successful then a reference to an object of type ppcstype1record is returned. If the attempt is not successful and neither the *`error`* nor the *`errortext`* parameter is provided then an error is raised. If the attempt is not successful and the *`error`* or the *`errortext`* parameter is provided then `.nul` is returned and error information is output in the `error` object provided. If an exact match is not found, but the *`found`* parameter and the *`error`* or *`errortext`* parameters are provided, then no error will be generated, the *`found`* parameter will be set to `.false` and the next closest matching record in the index will be returned. Providing the *`lock`* parameter will allow the record to be selected with a lock.

## Prototype

*ppcstype1indexvar*.selectkey ( string | integer | number *value*, boolean *lock*, integer *error*, string *errortext*, integer *retry*, integer *timeout*, boolean *found* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | .nul | string \| integer \| number | If .nul then a search will be made for the first record in the index order where the value is empty. If the index is a string then a value passed of this type will be sought for in the index. If the index is of type integer, date, or time then an integer value passed in this parameter will be sought in the index. If the index is of type number then a numeric value passed in this parameter will be sought in the index.<br>  <br>**Warning**<br>It is important to be aware that tables with numeric indices may not always resolve reliably when accessed from different machines on different operating systems or hardware because of the differences in the way that floating point values may be calculated. For this reason it would be wise to migrate this type of index to string or integer. |
| lock | .false | boolean | If .true then the record is selected with a lock, which is necessary if the record is to be modified or deleted, or if modification or deletion by others is to be prevented. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | an object, not an integer value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during the PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns .nul. |
| errortext | .nul | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns .nul. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |
| found | .nul | boolean | This must be a boolean object that will be set to .true if an exact match is found and if an exact match is not found, then it will be set to .false. |

# ppcstype1record

## Description

Objects of type ppcstype1record represent a record in a database file on a PPCS database file. Using objects of this type a SIMPOL program may interrogate or modify the values in a record, or create or delete records.

## Type Tags

db1record

# Object Value

Objects of type ppcstype1record have no value, and it is an error to try to get or set this value.

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| file | ppcstype1file | Contains a reference to the ppcstype1file object for the database file that contains this record. This is the same as the table property. |
| index | ppcstype1index | Contains a reference to the ppcstype1index object that was used to locate this record, that is the ppcstype1index object that was used to select the record, or the ppcstype1index object associated with the ppcstype1record object used to select the record. |
| locked | boolean | This is a read-only property indicating whether or not the record has been locked by, or during the creation of, this record object. This is not a foolproof way of determining whether or not the record in the file is locked, since any other record object for the same record may have it locked. |
| stored | boolean | This is a read-only property indicating whether or not the record has been read from or stored in the file by, or during the creation of, this record object. It is not foolproof as the record may have been deleted by a different record object or user. |
| table | ppcstype1file | Contains a reference to the ppcstype1file object for the database file that contains this record. This is the same as the file property. |
| type | type | Specifies the ppcstype1record type object. |

# Methods

## delete()

### Description

Deletes a record that has previously been selected with a lock. This can only be done to a record that has been selected with a lock or where all records in the file are locked. After being deleted a record can be saved again; this will create a new record in the file. If the attempt to delete a record is successful then

`.nul` is returned. If the attempt is not successful and neither the *`error`* nor the *`errortext`* parameter is provided then an error is raised. If the attempt is not successful and the *`error`* or the *`errortext`* parameter is provided then `.nul` is returned and error information is output in the `error` object provided.

## Prototype

*`ppcstype1recordvar`*`.delete` ( integer *`error`*, string *`errortext`*, integer *`retry`*, integer *`timeout`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during the PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# get()

## Description

Returns the value for the specified field in a record.

## Prototype

*ppcstype1recordvar*.get ( ppcstype1field *field* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | ppcstype1field | The field for which to return the record data. The data is returned as a type that is compatible with the datatype of the field. Fields of the data types date and time are treated as integers. |

# put()

## Description

Assigns a value to a specified field in a record.

## Prototype

*ppcstype1recordvar*.put ( ppcstype1field *field*, string | integer | number *value* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | ppcstype1field | The field to which to assign record data. |
| value | .nul | string | integer | number | If .nul then an empty value will be assigned to the field in the record. In the case of the normal value types, they will be assigned to the field if they are compatible to the data type of the field. |

# save()

## Description

Saves a record that may have been modified or may have been new. If the record is a modified record then it must have been selected with a lock before the modifications were made or all records in the file must be locked. If the attempt to save a record is successful then .nul is returned. If the attempt is not successful

and neither the *error* nor the *errortext* parameter is provided then an error is raised. If the attempt is not successful and the *error* or the *errortext* parameter is provided then .nul is returned and error information is output in the error object provided.

## Prototype

*ppcstype1recordvar*.save ( boolean *lock*, integer *error*, string *errortext*, integer *retry*, integer *timeout* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lock | .false | boolean | If .true then the record remains locked after being saved. If .false then the record will not be locked after being saved. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during the PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns .nul. |
| errortext | .nul | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns .nul. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to com- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | plete before it is assumed that it will fail. |

# select()

## Description

Selects either the next or the previous record in a PPCS database file according to the same index, if any, used to select this record. If the attempt to select a record is successful then a reference to an object of type ppcstype1record is returned. If the attempt is not successful and neither the *error* nor the *errortext* parameter is provided then an error is raised. If the attempt is not successful and the *error* or the *errortext* parameter is provided then `.nul` is returned and error information is output in the `error` and/or `errortext` object(s) provided. Providing the *lock* parameter will allow the record to be selected with a lock.

## Prototype

*ppcstype1recordvar*.`select` ( boolean *previousrecord*, boolean *lock*, integer *error*, string *errortext*, integer *retry*, integer *timeout* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| previousrecord | `.false` | boolean | If `.false` then the next record in the relevant order is selected. If `.true` then the previous record in the relevant order is selected. |
| lock | `.false` | boolean | If `.true` then the record is selected with a lock, which is necessary if the record is to be modified or deleted, or if modification or deletion by others is to be prevented. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is `.nul` and *errortext* is not specified or is `.nul` then any error that occurs during the PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text gener- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ated during the execution of the method. This parameter must be an object, not a string value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns .nul. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# selectcurrent()

## Description

Reselects a record in a PPCS database file and optionally allows the new record object to be based on a different index. If the attempt to select a record is successful then a reference to an object of type ppcstype1record is returned. If the attempt is not successful and neither the *error* nor the *errortext* parameter is provided then an error is raised. If the attempt is not successful and the *error* or the *errortext* parameter is provided then .nul is returned and error information is output in the error and/or errortext object(s) provided. Providing the *lock* parameter will allow the record to be selected with a lock.

## Prototype

*ppcstype1recordvar*.selectcurrent ( ppcstype1index *index*, boolean *lock*, integer *error*, string *errortext*, integer *retry*, integer *timeout* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| index | .nul | ppcstype1index | If .nul then the new record will be considered to have been positioned sequentially in the file, that is, without an index, otherwise, the record will be considered to have been positioned using the index provided. |
| lock | .false | boolean | If .true then the record is selected with a lock, which is necessary if the record is to be modi- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | fied or deleted, or if modification or deletion by others is to be prevented. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during the PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| errortext | `.nul` | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *`error`* is not specified or is `.nul` and *`errortext`* is not specified or is `.nul` then any error that occurs during PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns `.nul`. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# unlock()

## Description

Unlocks a record that has previously been selected with a lock. This can only be done to a record that has been selected with a lock. Any modifications in the record will not be able to be saved after it has been unlocked. If the attempt to unlock a record is successful then `.nul` is returned. If the attempt is not successful and neither the *`error`* nor the *`errortext`* parameter is provided then an error is raised. If the attempt is not successful and the *`error`* or the *`errortext`* parameter is provided then `.nul` is returned and error information is output in the `error` and/or `errortext` object(s) provided.

### Prototype

*ppcstype1recordvar*.unlock ( integer *error*, string *errortext*, integer *retry*, integer *timeout* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during the PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns .nul. |
| errortext | .nul | string | Specifies an object that is used to output any error text generated during the execution of the method. This parameter must be an object, not a string value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during PPCS operation will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns .nul. |
| retry | 1000000 | integer | The number of microseconds between retries, if the operation is not successful immediately. |
| timeout | 5000000 | integer | The number of microseconds allowed for the operation to complete before it is assumed that it will fail. |

# ppcstype1record!

## Get

The member operator followed by the name of the field will return the value of the field content in the record object. If the name provided is not correct (including case-sensitivity), then an error will occur.

## Set

If the member operator followed by the name of the field is on the left side of an assignment (=), then the value of the right side of the statement will be assigned to the field in the record that is referenced by the field name. If the name provided is not correct (including case-sensitivity), then an error will occur.

# Chapter 6. The Superbase Micro Engine (SBME) Component

Provides single-user access to and the ability to create database files in the Superbase Micro Engine database format.

# sbme1

## Description

Creates a new sbme1 object and either opens or creates a `*.SBM` file. The *`action`* parameter must be provided.

### Note

Since creating this object opens the database file exclusively, it is not possible to create multiple sbme1 objects in a single program that concurrently open the same database file. Instead, store the reference to the opened database file in a utility type and have each portion of the program use the same sbme1 object.

## Type Tags

db1base

## Object Value

Objects of type sbme1 have no value, and it is an error to try to get or set this value.

# sbme1.new()

## Description

Creates a new sbme1 object and either opens or creates a `.SBM` file. The *`action`* parameter *must* be provided.

## Prototype

*`sbme1`*.new ( string *`filename`*, string *`action`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | The path, if required, plus the name of the SBME file including the SBM extension that is to be opened or created. |
| action | `"0"` | string | The action to take on the filename which must be one of the follow- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ing characters or combinations of characters: O, R, OC, RC, C. The characters can be upper or lower case. The default value is O. The characters represent the following actions: O=open, R=replace, C=create, OC=open if exists otherwise create, RC=replace if exists otherwise create. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| cachesize | integer | Contains the size in bytes of the cache for caching read-only portions of the database. Items that have been modified or created (including indexes, files, fields, and records) are not included in this value until they have been committed since it would be inappropriate to commit them if the cache size were to be exceeded until the program calls the commit function explicitly. |
| committype | string | Contains either `"auto"` or `""`, where `"auto"` commits after every operation that writes and `""` requires an explicit call to the `commit`() method. |
| filename | string | Contains the name of the file. |
| locktype | string | Contains the type of lock currently held on the associated resource. This can be one of three values: `""` (no lock), `"shared"`, or `"exclusive"`. More |

| Property | Type | Description |
|---|---|---|
| | | than one object can hold a shared lock on a lockable entity, but only one object can hold an exclusive lock on a lockable entity. |
| type | type | Specifies the sbme1 type object. |

# Methods

## commit()

### Description

This method is used to commit any values written to the database since the last time the `commit`() method was called, assuming that the database `committype` is set to `""`. If instead the `committype` is set to `"auto"` then this method has no effect.

### Prototype

*`sbme1var`*.`commit` ( integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

## deletetable()

### Description

Deletes a database table from the sbme1 file. This may not be particularly fast depending on the record count, since each record must be deleted individually. The deletion is committed at the end so there must also be sufficient room in memory to hold the changes. This will change at a later date. In any event, if you expect to be deleting and rebuilding large data tables on a regular basis, it may be smarter to use a separate sbme1 file for those tables. That way the file can simply be deleted and the table recreated.

### Prototype

*`sbme1var`*.`deletetable` ( string *`tablename`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tablename | None | string | The name of the table to be deleted. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

## gettabledefinition()

### Description

Retrieves a database definition in the form of an sbme1newtable object. Using this the table definition can be modified. See the sbme1new* items for more information.

### Prototype

*`sbme1var`*`.gettabledefinition` ( string *`tablename`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tablename | None | string | The name of the table for which the sbme1newtable object is to be retrieved. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

## gettablenames()

### Description

Gets the names of the SBME tables which are available in a specified SBME file. If the attempt to get the table names is successful then `.nul` is returned. If the attempt is not successful and neither the *`error`* nor

*errortext* parameters are provided then an error is raised. If the attempt is not successful and at least one of the *error* or *errortext* parameters are provided then .nul is returned and error information is output in the error or errortext objects provided.

### Prototype

*sbme1var*.gettablenames ( array *tablenames*, string *pattern*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tablenames | None | array | Specifies an object of type array which is used to output the names of the files which are available in the specified SBME file and which match the specified pattern, if any. The first table name, if any, is placed in the array in the position '[1]', the second in position '[2]', and so on. The number of table names retrieved in placed in position '[]'. |
| pattern | "" | string | A pattern used to restrict the table names which will be retrieved. Only names which match the pattern will be output. The matching of a name to a pattern follows the rules for the intrinsic function .like1. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul and *errortext* is not specified or is .nul then any error that occurs during name retrieval will halt the program. If an error object or errortext object is specified and an error occurs during execution then the error code or text is output into that object and the method returns .nul. |

## lock()

### Description

The lock() method is used to acquire a lock on the sbme1 object. The type of lock can be either shared or exclusive. It is necessary to hold a shared lock on the sbme1 object if a new table is to be created. To prevent the creation of a table, an exclusive lock must be held. To release a lock, call the unlock() method.

### Prototype

*sbme1var*.lock ( string *locktype*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| locktype | None | string | Contains the type of lock to be acquired on the associated resource. This must be either "shared" or "exclusive". It is an error to not specify the *locktype* parameter.<br><br>**Important**<br><br>This method *acquires* a lock, it does not change an existing locktype. Therefore it is imperative that the object not be locked before calling this method. It is currently not possible to atomically change a lock between the shared type and the exclusive type; one lock must be released and the next one acquired. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns .nul. |

## newtable()

### Description

Creates a new database table owned by the database file associated with the sbme1 object for which the method is called. The return value of this call *must* be assigned to a variable of type sbme1newtable (using

the =@ operator). The resulting table is not written back into the database file, that is done by calling the `create`() method of the sbme1newtable object.

### Prototype

*`sbme1var`*.`newtable` ( string *`tablename`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tablename | None | string | The name of the table to be created. This is not the name of the SBME file, but of the data table within. |

## opentable()

### Description

Opens a database table in the database file associated with the sbme1 object for which the method is called. If the attempt to open the table is successful then a reference to an object of type sbme1table is returned. If the attempt is not successful and the error parameter is not provided then an error is raised. If the attempt is not successful and the error parameter was provided then `.nul` is returned and error information is output in the error object provided.

### Prototype

*`sbme1var`*.`opentable` ( string *`tablename`*, string *`recordidfieldname`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tablename | None | string | The name of the table to be opened. This is not necessarily the name of the SBME file, but of the data table within. |
| recordidfieldname | `.nul` | string | The name to use for the exposed internal unique record ID. If this is supplied then the internal record ID will be shown as an indexed unique field where the index is using the algorithm `"recordid"`. The internal record id is read-only and guaranteed for the life of the record in the table. If the table is reorganized, the record ID is likely to change.<br>  <br>**Warning**<br>Do not store the record ID for future use! The record ID |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | will certainly change if the table is reorganized so it should not be used for any long term purpose. Within the same session and assuming the same database table it can be used for safely reselecting the same record, which makes it an excellent candidate for routines that search for duplicate records and such. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# renamefield()

## Description

Renames a field in a named database table to a new name.

## Prototype

*`sbme1var`*`.renamefield` ( string *`tablename`*, string *`fieldname`*, string *`newfieldname`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tablename | None | string | The name of the data table that contains the field to be renamed. |
| fieldname | None | string | The name of the field that is to be renamed. |
| newfieldname | None | string | The new name of the field. |
| error | `.nul` | integer | Specifies an object that is used to output any error code gener- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

# renametable()

## Description

Renames a database table to a new name.

## Prototype

*sbme1var*.renametable ( string *tablename*, string *newtablename*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tablename | None | string | The name of the data table to be renamed. |
| newtablename | None | string | The new name of the data table. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

# rollback()

## Description

This method is used to rollback any previously written values to the database that have not yet been committed using the commit() method assuming that the database committype is set to "". If instead the committype is set to "auto" then this method has no effect.

## Prototype

*sbme1var*.rollback ( integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

## setcachesize()

### Description

This method is used to set the value of the `cachesize` property. The value *must* be supplied. This determines the amount of memory that is used for caching portions of the SBM file.

### Prototype

*`sbme1var`*`.setcachesize` ( integer *`cachesize`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| cachesize | None | integer | The cache size desired. This should not be too small or too large and should only override the default value if truly required. This is limited to a non-negative integer that is less than the total amount of addressable memory for the operating system. On Windows and Linux this is 4GB (Win64 and other 64-bit operating systems will allow considerably higher values but this still should not exceed a reasonable percentage of physical memory). |

## setcommittype()

### Description

This method is used to assign the value of the `committype` property. The value *must* be supplied, even if it is the empty string.

### Prototype

*sbme1var*.setcommittype ( string *committype* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| committype | None | string | The commit method desired. The permitted values are "auto" and "" and one of these values *must* be supplied to the method. |

## settabledefinition()

### Description

Sets the definition of the named table to be that represented by the *definition* parameter. Note that fields are matched by name, so if the old table definition has a field called "myfield", and the new definition also has a field called "myfield" then they are assumed to be the same field, so the data in the field will be retained. Any fields not found in the new definition are considered deleted and are lost forever. Any fields in the definition which weren't there before are new fields, which start with no data.

### Prototype

*sbme1var*.settabledefinition ( string *tablename*, sbme1newtable *definition*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tablename | None | string | The name of the table for which the file definition is being set. |
| definition | .nul | sbme1newtable | The sbme1newtable object that was retrieved to modify the table definition using the gettabledefinition() method. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during execution will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

## unlock()

### Description

Unlocks the `sbme1` object that has previously been locked with either a shared or exclusive lock using the `lock`() method. If the attempt to unlock the database file is successful then `.nul` is returned. If the attempt is not successful and the `error` parameter is not provided then an error is raised. If the attempt is not successful and the `error` parameter is provided then `.nul` is returned and error information is output in the `error` object provided.

### Prototype

*`sbme1var`*.`unlock` ( integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# sbme1field

# Description

Objects of type sbme1field represent a field in a database table in an SBME database file, and provide information that may be needed about the field.

# Type Tags

db1field

# Object Value

Objects of type sbme1field have no value, and it is an error to try to get or set this value.

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| datatype | type | Contains a reference to the type object that describes the type of data that the field contains. |
| index | sbme1index | If the field is part of an index then this property contains a reference to the most precise sbme1index object for that index that has this field as the first field, otherwise it contains `.nul`. |
| name | string | Contains the name of the field. |
| next | sbme1field | Contains a reference to the next sbme1field object for the table. The sbme1field objects for a table form a closed loop that is linked by this property. |
| table | sbme1table | Contains a reference to the sbme1table object for the table that contains this field. |
| type | type | Specifies the sbme1field type object. |

# sbme1index

## Description

Objects of type sbme1index represent an index for a table in a SBME database file, and provide information that may be needed about the index.

## Type Tags

db1index

## Object Value

Objects of type sbme1index have no value, and it is an error to try to get or set this value.

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |

| Property | Type | Description |
|---|---|---|
| algorithm | string | String indicating the algorithm used in the generation of the index. Currently the only valid values are `.nul` and `SB Compatible`. The `.nul` algorithm is in the same sort order as that used by the comparison operators. The `SB Compatible` order is compatible to the sort order used by Superbase and is also extended to support a fairly large number of additional Unicode characters. |
| field | sbme1field | Contains a reference to the first sbme1field object that this index is based on. Currently that is the only one. |
| next | sbme1index | Contains a reference to the next `sbme1index` object for the table. The sbme1index objects for a table form a closed loop that is linked by this property. |
| precision | integer | For full precision this value must be `.nul`, otherwise it is a positive integer value that indicates the maximum number of characters taken from a string value or the maximum number of bytes of data taken from the most significant end of an integer value when sorting. For indexes on number fields, this value must be greater than `0` and defaults to `1`. The value in this case implies the degree of fraction that is supported: the default is $^{1}/_{256}^{th}$ and `2` permits up to the nearest $^{1}/_{65536}^{th}$ and so on. |
| table | sbme1table | Contains a reference to the sbme1table object for the table that contains this index. |
| type | type | Specifies the sbme1index type object. |
| unique | boolean | If this value is equal to `.true`, then the index is guaranteed to be unique and will cause an error if an attempt is made to save a record with a duplicate index value. |

# Methods

## select()

### Description

Selects either the first or the last record in an SBME database table according to this index. The selection does not depend on the position of any other record. If the attempt to select a record is successful then a reference to an object of type sbme1record is returned. If the attempt is not successful and the *`error`* parameter is not provided then an error is raised. If the attempt is not successful and the *`error`* parameter is provided then `.nul` is returned and error information is output in the `error` object provided. Providing the *`locktype`* parameter will allow the record to be selected with either a shared or exclusive lock.

### Prototype

*`sbme1indexvar`*`.select` ( boolean *`lastrecord`*, string *`locktype`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lastrecord | `.false` | boolean | If `.false` then the first record in the index order is selected. If `.true` then the last record in the index order is selected. |
| locktype | `""` | string | If `"shared"` or `"exclusive"` then the record is selected with a lock of the associated type. An exclusive lock is required if the record is to be modified or deleted, and at least a shared lock is required if modification or deletion by others is to be prevented. These can also be accomplished by virtue of holding an exclusive lock on the sbme1table or the sbme1. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

## selectkey()

### Description

Selects a record according to the location of a specified value in the index. The selection does not depend on the position of any other record. If the attempt to select a record is successful then a reference to an object of type sbme1record is returned. If the attempt is not successful and the *`error`* parameter is not provided then an error is raised. If the attempt is not successful and the *`error`* parameter is provided then `.nul` is returned and error information is output in the `error` object provided. If an exact match is not found, but the *`found`* parameter and the *`error`* parameter are provided, then no error will be generated, the *`found`* parameter will be set to `.false` and the next closest matching record in the index will be returned. Providing the *`locktype`* parameter will allow the record to be selected with either a shared or exclusive lock.

### Prototype

*`sbme1indexvar`*`.selectkey` ( string | integer | boolean | blob *`value`*, string *`locktype`*, integer *`error`*, boolean *`found`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | `.nul` | string \| integer \| boolean \| blob | If `.nul` then a search will be made for the first record in the index order where the value is empty. If the index is a string or a blob then a value passed of this type will be sought for in the index. If the index is of type integer, date, time, or datetime then an integer value passed in this parameter will be sought in the index. |
| locktype | `""` | string | If `"shared"` or `"exclusive"` then the record is selected with a lock of the associated type. An exclusive lock is required if the record is to be modified or deleted, and at least a shared lock is required if modification or deletion by others is to be prevented. These can also be accomplished by virtue of holding an exclusive lock on the sbme1table or the sbme1. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |
| found | `.nul` | boolean | This must be a boolean object that will be set to `.true` if an exact match is found and if an exact match is not found, then it will be set to `.false`. |

# sbme1newfield

## Description

Objects of type sbme1newfield represent a field in a new database table associated with an SBME database file, and provide information that may be needed about the field.

# Type Tags

None

# Object Value

Objects of type sbme1newfield have no value, and it is an error to try to get or set this value.

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| datatype | type | Contains a reference to the type object that describes the type of data that the field contains. |
| index | sbme1newindex | If the field is part of an index then this property contains a reference to the most precise sbme1newindex object for that index that has this field as the first field, otherwise it contains `.nul`. |
| name | string | Contains the name of the field. |
| next | sbme1newfield | Contains a reference to the next sbme1newfield object for the table. The sbme1newfield objects for a table form a closed loop that is linked by this property. |
| table | sbme1newtable | Contains a reference to the sbme1newtable object for the table that contains this field. |
| type | type | Specifies the sbme1newfield type object. |

# Methods

## remove()

### Description

Removes the field from its table definition.

### Prototype

*`sbme1newfieldvar`*`.remove` ( integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

## setdatatype()

### Description

Changes the data type of the associated field.

#### Warning

Doing this to an existing database table can result in data loss! If the data type is changed from a string to an integer, then upon accessing the data it will be converted to an integer value. The contents of date, time, and datetime fields are stored as type integer. If this is changed to a string type, then when read it will not contain a date formatted according to some date format convention, instead it will contain an integer value that has been converted to a string.

### Prototype

*`sbme1newfieldvar`*`.setdatatype` ( string | integer | number | boolean | blob | date | time | datetime *`datatype`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datatype | None | string \| integer \| number \| boolean \| blob \| date \| time \| datetime | This parameter receives the actual datatype of the target field. Technically any object that has a value that can be read and written can be passed here, such as the date, time, and datetime types. |

# sbme1newindex

# Description

Objects of type sbme1newindex represent an index in a new database table associated with an SBME database file, and provide information that may be needed about the index.

# Type Tags

None

# Object Value

Objects of type sbme1newindex have no value, and it is an error to try to get or set this value.

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| algorithm | string | Contains the name of the algorithm for this index. If this value is the empty string (`""`) then the index is in order by character value. |
| field | sbme1newfield | Contains a reference to the first sbme1newfield object that this index is based on. Currently that is the only one. |
| next | sbme1newindex | Contains a reference to the next sbme1newindex object for the table. The sbme1newindex objects for a table form a closed loop that is linked by this property. |
| precision | integer | For full precision this value must be `.nul`, otherwise it is a positive integer value that indicates the maximum number of characters taken from a string value, the maximum number of bytes of data taken from the most significant end of an integer value, or the maximum number of bytes taken from a blob when sorting. |
| table | sbme1newtable | Contains a reference to the sbme1newtable object for the table that contains this index. |
| type | type | Specifies the sbme1newindex type object. |
| unique | boolean | If this value is equal to `.true`, then the index is guaranteed to be unique and will cause an error if an attempt is made to save a record with a duplicate index value. |

# Methods

## remove()

### Description

Removes the index from its table definition.

### Prototype

*sbme1newindexvar*.remove ( integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns .nul. |

# sbme1newtable

# Description

Objects of type sbme1newtable are used to create a new database table in an SBME database file and provide the necessary interface to create and modify the components of the table. Objects of this type also do not have a new() method, since they are created by a call to the newtable() method of the sbme1 object.

# Type Tags

None

# Object Value

Objects of type sbme1newtable have no value, and it is an error to try to get or set this value.

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| firstfield | sbme1newfield | Contains a reference to the sbme1newfield object that represents the first field in the table. |
| firstindex | sbme1newindex | Contains a reference to the sbme1newindex object that represents the first index on the file. If there are no indexes on the file then the reference is `.nul`. |
| sbme | sbme1 | Contains a reference to the sbme1 object that owns the new table. |
| tablename | string | Contains the name of the table. |
| type | type | Specifies the sbme1newtable type object. |

# Methods

## create()

### Description

Creates the new table in the sbme1 object's file. If the attempt is not successful and the *`error`* parameter is not provided then an error is raised. If the attempt is not successful and the *`error`* parameter is provided then error information is output in that `error` object.

### Prototype

*`sbme1newtablevar`*`.create` ( integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# newfield()

## Description

This method adds a field to a table definition. The *name* and *datatype* parameters are required. If the *next* parameter is .nul then the field will be added before the current firstfield.

## Prototype

*sbme1newtablevar*.newfield ( string *name*, string | integer | number | boolean | blob | date | time | datetime *datatype*, sbme1newfield *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The name of the field. |
| datatype | None | string \| integer \| number \| boolean \| blob \| date \| time \| datetime | This parameter receives the actual datatype of the target field. Technically any object that has a value that can be read and written can be passed here, such as the date, time, and datetime types. |
| next | .nul | sbme1newfield | This is the new field before which this new field is to be added. If this parameter is not supplied, then the new field will be added before the current firstfield. |

# newindex()

## Description

Creates a new index based on the field passed as the first parameter. The *field* parameter *must* be supplied. The default *precision* is .nul (full precision) except for number fields where it is 1 and must be at greater than 0. The default *algorithm* is the empty string "" (character value ordering), anotehr alternative is the Superbase-compatible index algorithm: "SB Compatible".

## Prototype

*sbme1newtablevar*.newindex ( sbme1newfield *field*, integer *precision*, boolean *unique*, string *algorithm* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | sbme1newfield | A reference to a sbme1newfield object for which the index will be built. This is a required parameter. |
| precision | .nul | integer | The default value for this is .nul (full precision). Otherwise it represents the number of bytes of a string or the number of significant |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | digits from the left of a value to use when sorting. |
| unique | `.false` | boolean | This determines whether the index is limited to unique values or not. If this is `.true`, then attempting to save a record that contains a duplicate value for an index will cause an error. |
| algorithm | `""` | string | This is the name of the algorithm to be used for sorting the index. It defaults to the empty string (character value ordering). |

# sbme1record

## Description

Objects of type sbme1record represent a record in a database table in an SBME database file. Using objects of this type a SIMPOL program may interrogate or modify the values in a record, or create or delete records.

## Type Tags

db1record

## Object Value

Objects of type sbme1record have no value, and it is an error to try to get or set this value.

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| index | sbme1index | Contains a reference to the sbme1index object that was used to locate this record, that is the sbme1index object that was used to select the record, or the sbme1index object associated with the sbme1record object used to select the record. |
| locktype | string | Contains the type of lock currently held on the associated resource. This can be one of three values: `""` (no lock), `"shared"`, or `"exclusive"`. More than one object can hold a shared lock on a lock- |

| Property | Type | Description |
|---|---|---|
| | | able entity, but only one object can hold an exclusive lock on a lockable entity. |
| stored | boolean | This is a read-only property indicating whether or not the record has been read from or stored in the file by, or during the creation of, this record object. It is not foolproof as the record may have been deleted by a different record object or user. |
| table | sbme1table | Contains a reference to the sbme1table object for the table that contains this record. |
| type | type | Specifies the sbme1record type object. |

# Methods

## delete()

### Description

Deletes a record that has previously been selected with a lock. This can only be done to a record that has been selected with a lock or where all records in the file are locked. After being deleted a record can be saved again; this will create a new record in the file. If the attempt to delete a record is successful then `.nul` is returned. If the attempt is not successful and the `error` parameter is not provided then an error is raised. If the attempt is not successful and the `error` parameter is provided then `.nul` is returned and error information is output in the `error` object provided.

### Prototype

*`sbme1recordvar`*`.delete` ( integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

## get()

### Description

Returns the value for the specified field in a record.

### Prototype

*sbme1recordvar*.get ( sbme1field *field* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | sbme1field | The field for which to return the record data. The data is returned as a type that is compatible with the datatype of the field. Fields of the data types date, time, and datetime are all treated as integers. |

## put()

### Description

Assigns a value to a specified field in a record.

### Prototype

*sbme1recordvar*.put ( sbme1field *field*, string | integer | number | boolean | blob *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | sbme1field | The field to which to assign record data. |
| value | .nul | string | integer | number | boolean | blob | If .nul then an empty value will be assigned to the field in the record. In the case of the normal value types, they will be assigned to the field if they are compatible to the data type of the field. |

## save()

### Description

Saves a record that may have been modified or may have been new. If the record is a modified record then it must have been selected with an exclusive lock before the modifications were made or all records in the file must be locked. If the attempt to save a record is successful then .nul is returned. If the attempt is not successful and the *error* parameter is not provided then an error is raised. If the attempt is not successful and the *error* parameter is provided then .nul is returned and error information is output in the error object provided.

### Prototype

*sbme1recordvar*.save ( string *locktype*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| locktype | `""` | string | If `"shared"` or `"exclusive"` then the record is saved with a lock of the associated type. An exclusive lock is required if the record is to be modified or deleted, and at least a shared lock is required if modification or deletion by others is to be prevented. These can also be accomplished by virtue of holding an exclusive lock on the sbme1table or the sbme1. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during the SBME operation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

## select()

### Description

Selects either the next or the previous record in an SBME database table according to the same index, if any, used to select this record. If the attempt to select a record is successful then a reference to an object of type sbme1record is returned. If the attempt is not successful and the *`error`* parameter is not provided then an error is raised. If the attempt is not successful and the *`error`* parameter is provided then `.nul` is returned and error information is output in the `error` object provided. Providing the *`locktype`* parameter will allow the record to be selected with either a shared or exclusive lock.

### Prototype

*`sbme1recordvar`*`.select` ( boolean *`previousrecord`*, string *`locktype`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| previousrecord | `.false` | boolean | If `.false` then the next record in the relevant order is selected. If `.true` then the previous record in the relevant order is selected. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| locktype | `""` | string | If `"shared"` or `"exclusive"` then the record is selected with a lock of the associated type. An exclusive lock is required if the record is to be modified or deleted, and at least a shared lock is required if modification or deletion by others is to be prevented. These can also be accomplished by virtue of holding an exclusive lock on the sbme1table or the sbme1. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# selectcurrent()

## Description

Reselects a record in an SBME database table and optionally allows the new record object to be based on a different index. If the attempt to select a record is successful then a reference to an object of type sbme1record is returned. If the attempt is not successful and the *`error`* parameter is not provided then an error is raised. If the attempt is not successful and the *`error`* parameter is provided then `.nul` is returned and error information is output in the `error` object provided. Providing the *`locktype`* parameter will allow the record to be selected with either a shared or exclusive lock.

## Prototype

*`sbme1recordvar`*`.selectcurrent` ( sbme1index *`index`*, string *`locktype`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| index | `.nul` | sbme1index | If `.nul` then the new record will be considered to have been positioned sequentially in the file, that is, without an index, otherwise, the record will be considered to have been positioned using the index provided. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| locktype | `""` | string | If `"shared"` or `"exclusive"` then the record is selected with a lock of the associated type. An exclusive lock is required if the record is to be modified or deleted, and at least a shared lock is required if modification or deletion by others is to be prevented. These can also be accomplished by virtue of holding an exclusive lock on the sbme1table or the sbme1. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# unlock()

## Description

Unlocks a record that has previously been selected with either a shared or exclusive lock. This can only be done to a record that has been selected with a lock. Any modifications in the record will not be able to be saved after it has been unlocked. If the attempt to unlock a record is successful then `.nul` is returned. If the attempt is not successful and the *`error`* parameter is not provided then an error is raised. If the attempt is not successful and the *`error`* parameter is provided then `.nul` is returned and error information is output in the `error` object provided.

## Prototype

*`sbme1recordvar`*`.unlock` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# sbme1record!

## Get

The member operator followed by the name of the field will return the value of the field content in the record object. If the name provided is not correct (including case-sensitivity), then an error will occur.

## Set

If the member operator followed by the name of the field is on the left side of an assignment (=), then the value of the right side of the statement will be assigned to the field in the record that is referenced by the field name. If the name provided is not correct (including case-sensitivity), then an error will occur.

# sbme1table

# Description

Objects of type sbme1table represent a database table in an SBME database file and provide the necessary interface to access the data in the database. Objects of this type also do not have a `new`() method, since they are created by a call to the `opentable`() method of an sbme1 object.

# Type Tags

db1table

# Object Value

Objects of type sbme1table have no value, and it is an error to try to get or set this value.

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |

| Property | Type | Description |
|---|---|---|
| firstfield | sbme1field | Contains a reference to the sbme1field object that represents the first field in the table. |
| firstindex | sbme1index | Contains a reference to the sbme1index object that represents the first index on the table. If there are no indexes on the table then the reference is `.nul`. |
| locktype | string | Contains the type of lock currently held on the associated resource. This can be one of three values: `""` (no lock), `"shared"`, or `"exclusive"`. More than one object can hold a shared lock on a lockable entity, but only one object can hold an exclusive lock on a lockable entity. |
| recordidfield-name | string | Contains the name of the field that is being used to show the internal record ID. If the record ID is not being shown, this field will be equal to `.nul`. |
| sbme | sbme1 | Contains a reference to the sbme1 object used to open the table. |
| tablename | string | Contains the name of the table. |
| type | type | Specifies the sbme1table type object. |

# Methods

## lock()

### Description

The `lock`() method is used to acquire a lock on the sbme1table object. The type of lock can be either shared or exclusive. It is necessary to hold a shared lock on the sbme1table object or an exclusive lock on the sbme1 object to add a record to the table. To prevent the addition of records to a table, an exclusive lock must be held on either the table or the sbme1. To release a lock, call the `unlock`() method.

### Prototype

*`sbme1tablevar`*`.lock` ( string *`locktype`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| locktype | None | string | Contains the type of lock to be acquired on the associated resource. This must be either `"shared"` or `"exclusive"`. It is an error to not specify the *`locktype`* parameter.<br><br>**Important**<br><br>This method *acquires* a lock, it does |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | not change an existing `locktype`. Therefore it is imperative that the object not be locked before calling this method. It is currently not possible to atomically change a lock between the shared type and the exclusive type; one lock must be released and the next one acquired. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# newrecord()

## Description

Creates a sbme1record object to represent a new (as yet not persistently stored) record in the database table.

## Prototype

*`sbme1tablevar`*`.newrecord` ()

## Parameters

None

# recordcount()

## Description

Returns the number of records in the table.

## Prototype

*`sbme1tablevar`*`.recordcount` ()

### Parameters

None

## select()

### Description

Selects either the first or the last record in an SBME database table. The selection does not depend on any index or on the position of any other record. If the attempt to select a record is successful then a reference to an object of type sbme1record is returned. If the attempt is not successful and the *error* parameter is not provided then an error is raised. If the attempt is not successful and the *error* parameter is provided then .nul is returned and error information is output in the error object provided. Providing the *locktype* parameter will allow the record to be selected with either a shared or exclusive lock.

### Prototype

*sbme1tablevar*.select ( boolean *lastrecord*, string *locktype*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lastrecord | .false | boolean | If .false then the first record in the table is selected. If .true then the last record in the table is selected. |
| locktype | "" | string | If "shared" or "exclusive" then the record is selected with a lock of the associated type. An exclusive lock is required if the record is to be modified or deleted, and at least a shared lock is required if modification or deletion by others is to be prevented. These can also be accomplished by virtue of holding an exclusive lock on the sbme1table or the sbme1. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns .nul. |

## unlock()

### Description

Unlocks the sbme1table object that has previously been locked with either a shared or exclusive lock using the `lock` method. If the attempt to unlock the table is successful then `.nul` is returned. If the attempt is not successful and the *`error`* parameter is not provided then an error is raised. If the attempt is not successful and the *`error`* parameter is provided then `.nul` is returned and error information is output in the `error` object provided.

### Prototype

*`sbme1tablevar`*.`unlock` ( integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# sbme1table!

## Get

The member operator followed by the name of the field will return a reference to the sbme1field object. If the name provided is not correct (including case-sensitivity), then an error will occur.

## Set

Attempting to assign a reference or a value to a sbme1table object using the ! operator is not supported.

# Chapter 7. The PPCS Server for SBME Databases (PPSR) Component

Contains a server to allow SBME databases to be accessible to a PPCS client. The server is an object which is configured and controlled by SIMPOL

# ppcstype1server

## Description

A ppcstype1server object provides access to SBME databases to clients using PPCS type 1 over UDP.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| firstsbme | ppcstype1serversbme | Specifies the object which represents the first SBME file used by the server. |
| firstudpport | ppcstype1serverudpport | Specifies the object which represents the first UDP port used by the server. |
| serving | boolean | Indicates whether or not the there is currently a pending call to the .serve() method of the ppcstype1server object. |
| type | type | Specifies the ppcstype1server type object. |

## Methods

### addsbme()

#### Description

Adds an SBME file to the ring of SBME files in use by the server.

## Prototype

*ppcstype1servervar*.addsbme ( string *filename*, string *committype*, integer *locktimeout*, integer *cachesize*, boolean *recordcount*, integer *codepage*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | Specified the SBME file to add to the server, by giving its path and name. |
| committype | "" | string | Specifies when changes to the SBME file are to be committed. If this is "" then changes are committed only when the `commit` method is called for the SBME object. If the committype is `"auto"` then changes are committed after every change is complete. Committing after every change means that the SBME file is more secure against system or programming errors, but there may be a performance penalty. |
| locktimeout | `.inf` | integer | Gives the length of time that a record lock is respected for, measured in microseconds. When a client locks a record (or all records for a table) then that lock must be released by the client within the `locktimeout` time period. If not then the server will release the lock, and the client may encounter an error if it attempts an operation which assumes that the lock is still in place. A `locktimeout` of `.inf` means that the server will never automatically release records locks for tables in the new SBME file. |
| cachesize | Platform dependent | integer | Specifies the amount of memory to reserve for caching for the new SBME file. |
| recordcount | `.true` | boolean | Specifies whether a record count should be performed when the tables are added. If the record count is not done, it will be set to 4294967280 (0xfffffff0). This can be an advantage when sharing very large tables, since the record count may be seldom needed, but |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | the time it takes to calculate it can be 7 minutes or more for large tables (more than 1GB in size). |
| codepage | `850` | integer | Specifies the code page to be used for sending and receiving data. The default code page is 850, the DOS Latin 1 code page. Other supported code page values include: 437 (US ASCII), 720 (Arabic), 737 (Greek), 775 (Baltic Rim – Estonian, Lithuanian, and Latvian), 852 (Latin 2 – Eastern European languages), 855 (Cyrillic), 857 (Turkish), 862 (Hebrew), 866 (Cyrillic), 874 Thai, and 1258 (Vietnamese). It is important that the client also be set to open the table using the same code page. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the ppcstype1serversbme object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the method returns `.nul`. |

# addudpport()

## Description

Adds a port to the ring of ports in use by the server.

## Prototype

*`ppcstype1servervar`*.`addudpport` ( integer *`udpport`*, integer *`txfactor`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| udpport | None | integer | Gives the number of the UDP port which the ppcstype1server object is to start using. |
| txfactor | 0 | integer | This value is used to throttle the speed at which packets are sent. The value 0 means no throttling. If a value is specified that is too high to be useful, then the high- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | est most meaningful value will be used. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the ppcstype1serverudpport object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the method returns `.nul`. |

# break()

## Description

Breaks out of all pending calls to the `server` method.

## Prototype

*`ppcstype1servervar`*`.break` ()

## Parameters

None

# serve()

## Description

Instructs the ppcstype1server object to process requests from PPCS clients.

## Prototype

*`ppcstype1servervar`*`.serve` ( integer *`timeout`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| timeout | `0` | integer | Specifies the length of time in microseconds to wait for client requests and process them. If this argument is `0` then the object will process pending requests but not wait for any length of time. If `timeout` is not `0` then this method will block for the specified length of time, or until it is explicitly broken, by the `break` method. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during the processing of client requests. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object. |

# ppcstype1serverfield

## Description

A ppcstype1serverfield object represents a field (sometimes called a 'column') in a table from an SBME database.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| SBLdisplayformat | string | The format string which SBL applications should use for the data from the field. |
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| datatype | type | Specifies the type object which represents the type of the data associated with the field in the SBME database file. |
| fieldname | string | Gives the name of the field as recorded in the SBME database file. |
| next | ppcstype1serverfield | Gives the next ppcstype1serverfield object for the same table. |
| shareable | boolean | Indicates whether or not the field will be included when clients access the table, i.e. whether or not |

| Property | Type | Description |
|---|---|---|
| | | it is shared via PPCS. By default, fields in a table ARE shareable. |
| sharename | string | Gives the field name which PPCS clients must use. By default, this is the same as the name recorded in the SBME database file. |
| sharetype | string | Indicates the SBL data type which the client should assume for the field. `sharetype` must be one of the following values.<br><br>'txt' - text data<br><br>'log' - single character text data indicating 'true' or 'false'<br><br>'num' - decimal numeric data<br><br>'nml' - integer numeric data between -2G and +2G<br><br>'nmi' - integer numeric data between -32k and +32k<br><br>'dat' - a date<br><br>'tim' - a time |
| table | ppcstype1servertable | Specifies the table to which the field belongs. |
| type | type | Specifies the ppcstype1serverfield type object. |

# Methods

## setshareability()

### Description

Determines the sharing characteristics of a field.

### Prototype

*`ppcstype1serverfieldvar`*`.setshareability` ( boolean *`shareable`*, string *`sharename`*, string *`sharetype`*, string *`SBLdisplayformat`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| shareable | The current value of the `shareable` property | boolean | Determines whether or not the field can be accessed by PPCS clients, i.e. whether or not it is shared via PPCS. |
| sharename | The current value of the `sharename` property | string | Determines the name by which the field is accessed by PPCS clients. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sharetype | The current value of the `sharetype` property | string | Indicates the SBL data type which the client should assume for the field. `sharetype` must be one of the following values.<br><br>'txt' - text data<br><br>'log' - single character text data indicating 'true' or 'false'<br><br>'num' - decimal numeric data<br><br>'nml' - integer numeric data between -2G and +2G<br><br>'nmi' - integer numeric data between -32k and +32k<br><br>'dat' - a date<br><br>'tim' - a time |
| SBLdisplayformat | The current value of the `SBLdisplayformat` property | string | The format string which SBL applications should use for the data from the field. |

# ppcstype1serversbme

## Description

A ppcstype1serversbme object represents an SBME database file which a ppcstype1server object makes available to PPCS clients.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of |

| Property | Type | Description |
|---|---|---|
| | | being marked with the resolve keyword, so that object resolution can continue down this property. |
| cachesize | integer | Specifies the amount of memory currently available for caching the new SBME file. |
| committype | string | Specifies when changes to the SBME file are to be committed. If this is "" then changes are committed only when the `commit` method is called for the SBME object. If the committype is `"auto"` then changes are committed after every change is complete. Committing after every change means that the SBME file is more secure against system or programming errors, but there may be a performance penalty. |
| filename | string | Gives the name of the SBME file. |
| firsttable | ppcstype1servertable | Specifies the object which represents the first table in the SBME file. |
| locktimeout | integer | Gives the length of time that a record lock is respected for, measured in microseconds. When a client locks a record (or all records for a table) then that lock must be released by the client within the `locktimeout` time period. If not then the server will release the lock, and the client may encounter an error if it attempts an operation which assumes that the lock is still in place. A `locktimeout` of `.inf` means that the server will never automatically release records locks for tables in the SBME file. |
| next | ppcstype1serversbme | Gives the next SBME file for the same ppcstype1server object. |
| server | ppcstype1server | Specifies the server to which this SBME file belongs. |
| type | type | Specifies the ppcstype1serversbme type object. |

# Methods

## commit()

### Description

Commits all changes to the SBME file.

### Prototype

*`ppcstype1serversbmevar`*`.commit` ( integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object which is used to output any error code generat- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ed during the committing of data. If `error` is not specified or is `.nul` then any error which occurs during committing of data will halt the program. If an error object is specified and an error occurs during the committing of data then the error code is output into that object and the method returns `.nul`. |

# remove()

## Description

Removes a SBME file from the ring of SBME files for the server. The file is then available for use by other servers, although some operating systems will not allow it to be used until a platform specific short time period has elapsed.

## Prototype

*`ppcstype1serversbmevar`*`.remove` ()

## Parameters

None

# setcachesize()

## Description

Determines how much memory is used for caching data from the SBME file.

## Prototype

*`ppcstype1serversbmevar`*`.setcachesize` ( integer *`cachesize`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| cachesize | None | integer | Specifies the amount of memory to reserve for caching for the SBME file. |

# setcommittype()

## Description

Determines when changes to the SBME file are committed.

### Prototype

*ppcstype1serversbmevar*.setcommittype ( string *committype* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| committype | None | string | Specifies when changes to the SBME file are to be committed. If this is "" then changes are committed only when the `commit` method is called for the SBME object. If the committype is `"auto"` then changes are committed after every change is complete. Committing after every change means that the SBME file is more secure against system or programming errors, but there may be a performance penalty. |

## setlocktimeout()

### Description

Determines how long client's record locks are respected for.

### Prototype

*ppcstype1serversbmevar*.setlocktimeout ( integer *locktimeout* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| locktimeout | None | integer | Gives the length of time that a record lock is respected for, measured in microseconds. When a client locks a record (or all records for a table) then that lock must be released by the client within the `locktimeout` time period. If not then the server will release the lock, and the client may encounter an error if it attempts an operation which assumes that the lock is still in place. A `locktimeout` of `.inf` means that the server will never automatically release records locks for tables in the SBME file. |

# ppcstype1servertable

## Description

A ppcstype1servertable object represents an table in an SBME database.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| firstfield | ppcstype1serverfield | Specifies the object which represents the first field in the table. |
| hidden | boolean | Indicates whether the table is hidden when a PPCS client attempts to retrieve the list of available tables. By default, tables that are SBME shared are NOT hidden. |
| next | ppcstype1servertable | Gives the next ppcstype1servertable object for the same SBME file. |
| recordidfield-name | string | Contains the name of the field that is being used to show the internal record ID. If the record ID is not being shown, this field will be equal to `.nul`. If this is not equal to `.nul` then PPCS clients of the server will see a `TXT RDO IXU 12` (12 character length, text, read-only, uniquely-indexed) field called whatever the recordidfieldname property has been set to for that table. |
| sbme | ppcstype1serversbme | Specifies the SBME database file to which the table belongs. |
| shareable | boolean | Indicates whether the table can be accessed by PPCS clients, i.e. whether or not it is shared via PPCS. By default, tables in an SBME database file are NOT shareable. |
| sharename | string | Gives the table name which PPCS clients must use when accessing the table. By default, this is the |

| Property | Type | Description |
|---|---|---|
| | | same as the name recorded in the SBME database file. |
| tablename | string | Gives the name of the table as recorded in the SBME database file. |
| type | type | Specifies the ppcstype1servertable type object. |

# Methods

## getpassword()

### Description

Gets the access rights associated with the specified password for that table. The access mode string will not necessarily be the same as the access string given in the `setpassword()` method, but it will always return the same string for the same access rights (e.g. if you specify `'LR'` in the `setpassword()` method, you will get `'rl'` back from `getpassword()`, but you will always get `'rl'` for a password that gives only read and lock rights.

### Prototype

*`ppcstype1servertablevar`*`.getpassword` ( string *`password`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| password | None | string | The password for which the access rights are being queried. |

## setpassword()

### Description

Sets the access mode associated with a password for a table. If the empty password is not specifically set to something other than "" then it gives full access to a table which has no passwords set, and no access to a table which has at least one password set.

### Prototype

*`ppcstype1servertablevar`*`.setpassword` ( string *`password`*, string *`access`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| password | None | string | The password to be associated with the set of access modes. |
| access | None | string | An access mode of `""` removes the password from the table. Valid access strings are made up of any combination of `'r'` (read), |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | `'c'` (create), `'l'` (lock), `'w'` (write), `'d'` (delete) where 'w' and 'd' require that 'r' and 'l' are also included. The characters are not case sensitive. |

# setrecordidfieldname()

## Description

Call this function to both set the name to be used and to expose using that name the internal record ID from the `*.sbm` file. If this is not equal to `.nul` then PPCS clients of the server will see a `TXT RDO IXU 12` (12 character length, text, read-only, uniquely-indexed) field called whatever the *`recordidfieldname`* parameter has been set to for that table.

## Prototype

*`ppcstype1servertablevar`*`.setrecordidfieldname` ( string *`recordidfieldname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| recordidfield-name | None | string | The name to be used for exposing the internal read-only record ID. |

# setshareability()

## Description

Determines the sharing characteristics of a table.

## Prototype

*`ppcstype1servertablevar`*`.setshareability` ( boolean *`shareable`*, string *`sharename`*, boolean *`hidden`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| shareable | The current value of the `shareable` property | boolean | Determines whether or not the table can be accessed by PPCS clients, i.e. whether or not it is shared via PPCS. |
| sharename | The current value of the `sharename` property | string | Determines the name by which the table is accessed by PPCS clients. |
| hidden | The current value of the `hidden` property | boolean | Determines whether the table can be seen in table lists retrieved from the server by PPCS clients. |

# ppcstype1serverudpport

## Description

A ppcstype1serverudpport object represents a UDP port, which a ppcstype1server object uses to provide PPCS support to clients using UDP.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| next | ppcstype1serverudpport | Gives the next UDP port for the same ppcstype1server object. |
| server | ppcstype1server | Specifies the server to which this port belongs. |
| txfactor | integer | Contains the transmission factor used by the ppcstype1serverudpport object. This value is used to throttle the speed at which packets are sent. The value 0 means no throttling. If a value is specified that is too high to be useful, then the highest most meaningful value will be used. |
| type | type | Specifies the ppcstype1serverudpport type object. |
| udpport | integer | Specifies the number of the UDP port. |

## Methods

### remove()

#### Description

Removes a port from the ring of UDP ports for the server. The port is then available for use by other servers, although some operating systems will not make the port available until a platform specific time period has elapsed.

### Prototype

*ppcstype1serverudpportvar*.remove ()

### Parameters

None

## settxfactor()

### Description

Sets the transmission factor. This is used to moderate the speed at which the packets are transmitted. A factor of 0 means that the packets will be sent at the maximum speed (this may be too fast for Internet connectivity and result in difficulty reading data and excessive retries and timeouts).

### Prototype

*ppcstype1serverudpportvar*.settxfactor ( integer *txfactor* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| txfactor | None | integer | This is the factor to be used in slowing the sending of the packets. |

# Chapter 8. The CGI/ISAPI/FastCGI Component

## cgicall

### Description

An object of type cgicall is passed to the entry function of a CGI (or ISAPI) smpol program. The object is used to get variable values, get and set cookies and perform other CGI style I/O tasks.

### Object Value

Objects of type cgicall have no value, and it is an error to try to get or set this value.

### Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| type | `type` | Specifies the cgicall type object. |

### Methods

#### getvariable()

##### Prototype

*`cgicallvar`*`.getvariable` ( string *`name`* )

##### Description

Gets the value of the environment variable or server variable of the specified name, if it exists. If the variable cannot be found then `.nul` is returned.

##### Parameters

| Name | Default | Type | Description |
|---|---|---|---|
| `name` | None | string | The name of the variable for which to retrieve the value |

#### output()

##### Prototype

*`cgicallvar`*`.output` ( string *`string`*, integer *`charsize`*, boolean *`lbo`* )

### Description

Outputs data from a CGI program. The output can be any string but normally this will be ASCII text such as HTML.

### Parameters

| Name | Default | Type | Description |
|---|---|---|---|
| `string` | None | string | The text (or other data) to be output |
| `charsize` | `2` | integer | The number of bytes per character, which must be greater than 0 |
| `lbo` | `.true` | boolean | Indicates whether characters have been stored with the least significant byte first (`.true`) or with the most significant byte first (`.false`). |

## input()

### Prototype

*`cgicallvar`*`.input` ( integer *`length`* )

### Description

Gets data from the standard input to a CGI program. It is always assumed that one byte of input is one character.

### Parameters

| Name | Default | Type | Description |
|---|---|---|---|
| `length` | `.inf` | integer | The whole of the available input will be read |
| | | integer | The maximum number of bytes of input to read. If this is greater than the number of bytes available then a shorter string than specified will be returned. If there is no more available input then the empty string will be returned. |

## keyvalue()

### Prototype

*`cgicallvar`*`.keyvalue` ( string *`name`*, integer *`index`* )

### Description

Gets the value of a key from the submitted input. An index can be specified to retrieve values for keys where more than one value is submitted.

### Parameters

| Name | Default | Type | Description |
|---|---|---|---|
| `name` | `None` | | The name of the key value to retrieve |

| Name | Default | Type | Description |
|---|---|---|---|
| index | None | integer | The 1-based index of the value to retrieve for the specified key |

# getcookie()

## Prototype

*cgicallvar*.getcookie ( string *name* )

## Description

Gets the value a cookie sent with the HTTP request. If no cookie of the specified name has been sent, then .nul is returned.

## Parameters

| Name | Default | Type | Description |
|---|---|---|---|
| name | None | string | The name of the cookie whose value is to be retrieved |

# outputcookie()

## Prototype

*cgicallvar*.outputcookie ( string *name*, string *value*, string *path*, integer *expires* )

## Description

Outputs a string which is intended to be a cookie in an HTTP header. It is the responsibility of the programmer to ensure that the string is output at the appropriate place to be part of the HTTP header information.

## Parameters

| Name | Default | Type | Description |
|---|---|---|---|
| name | None | string | The name of the cookie to create |
| value | None | .nul | The value part of the cookie will be empty |
| | | string | The value to put in the cookie |
| path | .nul | .nul | No path part will be added to the cookie |
| | | string | The contents of the path part of the cookie. If the empty string is specified then no path part will be output |
| expires | .nul | .nul | No expiry information will be added to the cookie, so it will expire at the end of the session |
| | | integer | The expiry date and time of the cookie, specified in microseconds since midnight at the beginning of 1/1/0001, as with the value of a datetime object. |

# Chapter 9. The Sockets (SOCK) Component

Provides the types for working with the Internet Protocol (TCP/IP and UDP/IP) for both client and server.

# tcpsocket

## Description

A tcpsocket object can be created via a call to the `new`() of the tcpsocket type or else because of a connection to a tcpsocketserver object. This object provides TCP/IP socket support for communication.

## Type Tags

None

## Object Value

Objects of type tcpsocket have no value, and it is an error to try to get or set this value.

# tcpsocket.new()

## Description

Opens a TCP/IP connection to the requested address on the requested port using the local port that was specified (if none was provided then one will be selected automatically).

## Prototype

*`tcpsocket`*.`new` ( string *`destination`*, integer *`port`*, integer *`timeout`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| destination | None | string | The name or IP address of the destination to which to connect. This must include the port number, ie. "192.168.0.1:80" or "www.superbase.com:25". |
| port | `.nul` | integer | The local port number to use. If none is specified then a port will automatically be selected. |
| timeout | `.inf` | integer | The time in microseconds to wait before deciding that the operation has failed. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generat- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ed during creation of the tcpsocket object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `new` method returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| destination | string | A read-only string giving the destination in "a.b.c.d:p" format. |
| port | integer | A read-only integer giving the local port number being used (even if it was an automatically generated one). |
| type | type | Specifies the tcpsocket type object. |

# Methods

## receiveblob()

### Description

Returns a blob or `.nul`. If the error value returned is equal to 64 (normal end of data), then it may not be safe to continue using the socket (the other end may have closed it).

### Prototype

*`tcpsocketvar`*`.receiveblob` ( integer *`timeout`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| timeout | `.inf` | integer | The time in microseconds to wait before deciding that the operation has failed. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# receivestring()

## Description

Returns a string or `.nul`. If the error value returned is equal to 64 (normal end of data), then it may not be safe to continue using the socket (the other end may have closed it).

## Prototype

*`tcpsocketvar`*`.receivestring` ( integer *`charsize`*, boolean *`lbo`*, integer *`timeout`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| charsize | `2` | integer | The number of bytes per character, which must be greater than 0. If this value is equal to `.nul` or `.inf` then the result will be `.nul`. |
| lbo | `.true` | boolean | Indicates whether characters have been stored with the least significant byte first (`.true`) or with the most significant byte first (`.false`). If this value is equal to `.nul` or `.inf` then the result will be `.nul`. |
| timeout | `.inf` | integer | The time in microseconds to wait before deciding that the operation has failed. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# sendblob()

## Description

Sends the contents of blob to the destination.

## Prototype

*`tcpsocketvar`*`.sendblob` ( blob *`blob`*, integer *`timeout`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| blob | `.inf` | blob | This parameter is required. The blob contains what is being sent to the destination. |
| timeout | `.inf` | integer | The time in microseconds to wait before deciding that the operation has failed. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# sendstring()

## Description

Sends the contents of string to the destination.

## Prototype

*`tcpsocketvar`*`.sendstring` ( string *`string`*, integer *`charsize`*, boolean *`lbo`*, integer *`timeout`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| string | None | string | This parameter is required. The string contains what is being sent to the destination. |
| charsize | 2 | integer | The number of bytes per character, which must be greater than 0. If this value is equal to .nul or .inf then the result will be .nul. |
| lbo | .true | boolean | Indicates whether characters have been stored with the least significant byte first (.true) or with the most significant byte first (.false). If this value is equal to .nul or .inf then the result will be .nul. |
| timeout | .inf | integer | The time in microseconds to wait before deciding that the operation has failed. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns .nul. |

# tcpsocketserver

## Description

A tcpsocketserver object can only be created via a call to the new() method. This object provides TCP/IP socket support for communication as a server. It can be the basis of any TCP/IP-based server, such as a web server, mail server, etc.

## Type Tags

None

## Object Value

Objects of type tcpsocketserver have no value, and it is an error to try to get or set this value.

# tcpsocketserver.new()

## Description

Creates a TCP/IP socket that is sitting on the requested port. It doesn't actually listen on the port until the `listen`() method is called. When a connection occurs on the port where the object is listening, then a new thread will be started at the function passed as a reference to the `listen`() method. That function must take a tcpsocket object as the first parameter. A single object can have multiple listeners in different threads.

## Prototype

*`tcpsocketserver`*.`new` ( integer *`port`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| port | None | integer | The local port number to use. This parameter is required and must be in the range 1-65535. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the tcpsocketserver object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `new` method returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| listening | boolean | A read-only boolean showing whether or not the server is currently listening. |
| type | type | Specifies the tcpsocketserver type object. |

# Methods

## break()

### Description

Breaks out of all listens associated with the object.

### Prototype

*tcpsocketservervar*.break ()

### Parameters

None

## listen()

### Description

This starts the object listening on the socket. When a connection is made a new thread is started at the function passed in the *function* parameter. The function must accept a tcpsocket as the first parameter and it may have a second parameter of any type that matches the type passed to the *reference* parameter (if specified).

### Prototype

*tcpsocketservervar*.listen ( function *function*, type(*) *reference*, integer *timeout*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| function | None | function | The reference to the function to be called upon a successful connection. |
| reference | .nul | type(*) | This is an optional object reference that will be passed to the function upon a successful connection. This can be a reference to an object of user-defined type to pass multiple pieces of information to the function. |
| timeout | 0 | integer | The time in microseconds to listen before returning. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# Chapter 10. The Operating System Utilities (UTOS) Component

Provides the first version of types to interact with the operating system for the purpose of manageing files and directories

# UTOSdirectory

## Description

A UTOSdirectory object represents a directory in a file system, and gives basic access to the entries it contains. More complex management of the items represented by the entries in the directory is provided through the UTOSdirectoryentry object type. UTOSdirectory objects should be considered to be a snapshot of a directory as it was at the time the object was created as opposed than an active directory listing, since the objects are not updated when the directory contents are changed.

## Type Tags

None

## Object Value

The value of a UTOSdirectory object is the full specification of the directory it represents.

# UTOSdirectory.new()

## Description

Creates a new UTOSdirectory object for the specified directory.

## Prototype

*`UTOSdirectory`*`.new` ( string *`name`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | Specifies the directory to create an object for. This may be either partially or fully specified, the interpretation of partially specified names being dependent on the operating system. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the UTOSdirectory object. If `error` is not specified or is `.nul` then any error which occurs during object cre- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `new` method returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| entrycount | integer | Gives the number of entries in the directory. An entry is either a file or a subdirectory. |
| name | string | Gives just the name of the directory, rather than the full specification of its location. |
| type | type | Specifies the UTOSdirectory type object. |

# Methods

## createdirectory()

### Description

Creates a new subdirectory in the directory represented by the UTOSdirectory object for which the method is called. A UTOSdirectoryentry object representing the new subdirectory is returned.

### Prototype

*`UTOSdirectoryvar`*`.createdirectory` ( string *`name`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | Specifies the name of the new subdirectory. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the subdirectory and the corresponding UTOSdirectoryentry object. If `error` is not specified or is `.nul` then any error which occurs |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `createdirectory` method returns `.nul`. |

# deleteentry()

## Description

Deletes an entry from the directory for which the method is called.

## Prototype

*`UTOSdirectoryvar`*`.deleteentry` ( integer *`entry`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| entry | None | integer | Specifies the entry within the UTOSdirectory object to delete. This is given as a 1-based index. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during deletion of the entry. If `error` is not specified or is `.nul` then any error which occurs during the deletion will halt the program. If an error object is specified and an error occurs during deletion then the error code is output into that object. |

# getdirectory()

## Description

Creates a new UTOSdirectory object for the directory which contains the directory represented by the object for which this method is called. This can be considered to be the 'parent' directory. If there is no such directory, e.g. if the object for which the method is called represents a root directory, then no error is raised but `.nul` is returned.

## Prototype

*`UTOSdirectoryvar`*`.getdirectory` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object which is used to output any error code generated |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | during creation of the UTOSdirectory object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `getdirectory` method returns `.nul`. |

## getentry()

### Description

Creates a new UTOSdirectoryentry object for a specific directory entry.

### Prototype

*`UTOSdirectoryvar`*.`getentry` ( integer *`entry`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| entry | None | integer | Specifies the entry within the UTOSdirectory object to create an object for. This is given as a 1-based index. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the UTOSdirectoryentry object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `getentry` method returns `.nul`. |

# UTOSdirectoryentry

## Description

A UTOSdirectoryentry object represents a file or directory in a file system and gives access to some entry specific information. UTOSdirectoryentry objects should be considered to be a snapshot of a file or directory as it was at the time the object was created, since the objects are not updated when the file or directory information is changed.

# Type Tags

None

# Object Value

The value of a UTOSdirectoryentry object is the full specification of the file or directory it represents.

# UTOSdirectoryentry.new()

## Description

Creates a new UTOSdirectoryentry object for the specified file or directory.

## Prototype

*UTOSdirectoryentry*.new ( string *name*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | Specifies the file or directory to create an object for. This may be either partially or fully specified, the interpretation of partially specified names being dependent on the operating system. |
| error | .nul | integer | Specifies an object which is used to output any error code generated during creation of the UTOSdirectoryentry object. If error is not specified or is .nul then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the new method returns .nul. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of |

| Property | Type | Description |
|---|---|---|
| | | being marked with the resolve keyword, so that object resolution can continue down this property. |
| created | datetime | Gives the date and time that the file or directory was created, according to the information stored in the file system. This may be `.nul` if the file system does not support this information. |
| entrytype | string | Gives a description of the type of directory entry which the object represents. This will be either "file" or "directory". |
| hidden | boolean | Shows whether or not the file system considers the file or directory to be 'hidden'. If the file system does not support this, then the property value will be `.false`. |
| lastaccessed | datetime | Gives the date and time that the file or directory was last accessed according to the information stored in the file system. This may be `.nul` if the file system does not support this information. |
| lastchanged | datetime | Gives the date and time that the file or directory was most recently changed according to the information stored in the file system. This may be `.nul` if the file system does not support this information. |
| name | string | Gives just the name of the file or directory, rather than the full specification of its location. |
| readonly | boolean | Shows whether or not the file system considers the file or directory to be marked as available for read access. If the file system does not support this, then the property value will be `.false`. |
| size | integer | If the UTOSdirectoryentry entry represents a file then this property gives it size, otherwise it is `.nul`. |
| system | boolean | Shows whether or not the file system considers the file or directory to be reserved for system use. If the file system does not support this, then the property value will be `.false`. |
| type | type | Specifies the UTOSdirectoryentry type object. |

# Methods

## copy()

### Description

Copies a file to another location. An error is raised if this method is called for a UTOSdirectoryentry object which represents a directory. The UTOSdirectoryentry object itself is returned.

### Prototype

*`UTOSdirectoryentryvar`*`.copy` ( string *`destination`*, boolean *`replace`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| destination | None | string | Specifies the new location to copy the file to. Interpretation of this is operating system dependent, so it is safer to fully specify the destination file name. |
| replace | `.false` | boolean | Specifies whether or not to replace any existing file. If this is given as `.false` or is omitted and there is already a file at the specified destination then an error will be raised, which may be operating system dependent. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during the copying of the file. If `error` is not specified or is `.nul` then any error which occurs during copying will halt the program. If an error object is specified and an error occurs during copying then the error code is output into that object and the `copy` method returns `.nul`. |

## getdirectory()

### Description

Creates a new UTOSdirectory object for the directory which contains the file or directory represented by the object for which this method is called. If there is no such directory then no error is raised but `.nul` is returned.

### Prototype

*`UTOSdirectoryentryvar`*`.getdirectory` ( integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the UTOSdirectory object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | that object and the `getdirectory` method returns `.nul`. |

# move()

## Description

Moves a file or directory to another location. It is not always possible to move a file or directory to another volume, partition or device, so the error parameter should normally be used. The UTOSdirectoryentry object itself is returned.

## Prototype

*`UTOSdirectoryentryvar`*`.move` ( string *`destination`*, boolean *`replace`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| destination | None | string | Specifies the new location to move the file or directory to. Interpretation of this is operating system dependent, so it is safer to fully specify the destination file or directory name. |
| replace | `.false` | boolean | Specifies whether or not to replace any existing file or directory. If this is given as `.false` or is omitted and there is already a file or directory at the specified destination then an error will be raised, which may be operating system dependent. The behaviour when an attempt is made to replace a directory may be operating system dependent. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during the moving of the file or directory. If `error` is not specified or is `.nul` then any error which occurs during moving will halt the program. If an error object is specified and an error occurs during moving then the error code is output into that object and the `move` method returns `.nul`. |

# rename()

## Description

Changes the name of the file or directory represented by the UTOSdirectoryentry object for which the method is called. The UTOSdirectoryentry object itself is returned.

## Prototype

*UTOSdirectoryentryvar*.rename ( string *newname*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| newname | None | string | Specifies the new name of the file or directory. This is just the name, and must not contain any path component. |
| error | .nul | integer | Specifies an object which is used to output any error code generated during the renaming of the file or directory. If error is not specified or is .nul then any error which occurs during renaming will halt the program. If an error object is specified and an error occurs during renaming then the error code is output into that object and the rename method returns .nul. |

# setattributes()

## Description

Sets the values for the 'attributes' of a file or directory. The UTOSdirectoryentry object itself is returned.

## Prototype

*UTOSdirectoryentryvar*.setattributes ( boolean *readonly*, boolean *hidden*, boolean *system*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| readonly | The current value of the readonly property | boolean | Specifies the required value of the readonly property for the file or directory. |
| hidden | The current value of the hidden property | boolean | Specifies the required value of the hidden property for the file or directory. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| system | The current value of the `system` property | boolean | Specifies the required value of the `system` property for the file or directory. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during setting of the attributes of the file or directory. If `error` is not specified or is `.nul` then any error which occurs during the attribute setting will halt the program. If an error object is specified and an error occurs during attribute setting then the error code is output into that object and the `setattributes` method returns `.nul`. |

# setdates()

## Description

Sets the values for the dates of a file or directory. The UTOSdirectoryentry object itself is returned.

## Prototype

*`UTOSdirectoryentryvar`*`.setdates` ( integer *`created`*, integer *`lastchanged`*, integer *`lastaccessed`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| created | The current value of the `created` property | integer | Specifies the required value of the `created` property for the file or directory. The actual value set may be different from this value, for example if the file system stores dates and times to a lower degree of precision than those used in SIMPOL. This parameter should not be specified if the file system does not support file creation times. |
| lastchanged | The current value of the `lastchanged` property | integer | Specifies the required value of the `lastchanged` property for the file or directory. The actual value set may be different from this value, for example if the file system stores dates and times to a lower degree of precision than those used in SIMPOL. This parameter should not be specified if the |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | file system does not support file changed times. |
| lastaccessed | The current value of the `lastaccessed` property | integer | Specifies the required value of the `lastaccessed` property for the file or directory. The actual value set may be different from this value, for example if the file system stores dates and times to a lower degree of precision than those used in SIMPOL. This parameter should not be specified if the file system does not support file access times. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during setting of the dates of the file or directory. If `error` is not specified or is `.nul` then any error which occurs during the date setting will halt the program. If an error object is specified and an error occurs during date setting then the error code is output into that object and the `setdates` method returns `.nul`. |

# Chapter 11. The wxWidgets-based (WXWN) Components

The wxwn component provides the various cross-platform GUI functionality for SIMPOL using the wxWidgets cross-platform toolkit.

# rgb

## Description

An rgb object represents a color, and is embedded in another object where that object has some property that is a color.

## Type Tags

None

## Object Value

The value of an rgb object is an integer that specifies a color. It is not advisable to try to interpret or to perform arithmetic on this value.

## Properties

| Property | Type | Description |
|---|---|---|
| blue | integer | Gives the value of the blue component of the color value contained in the rgb object. |
| green | integer | Gives the value of the green component of the color value contained in the rgb object. |
| red | integer | Gives the value of the red component of the color value contained in the rgb object. |
| type | type | Specifies the rgb type object. |

# wxautomation1

## Description

A wxautomation1 object provides access to OLE2 automation for Windows™ applications. All automation code is sent as string data to the target program.

## Type Tags

None

# Object Value

# wxautomation1.new()

## Description

Creates a new wxstatusbar object.

## Prototype

*wxautomation1*.new ( string *createinstance*, string *getinstance*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| createinstance | .nul | string | This parameter must be named. Use it to create a new instance of the OLE2 out of process server, for use in the code. |
| getinstance | .nul | string | This parameter must be named. Use it to get access to an already existing instance of the OLE2 out of process server, for use in the code. |
| error | .nul | integer | Specifies an object which is used to output any error code generated during creation of the wxautomation1 object. If error is not specified or is .nul then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the new methods returns .nul. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the wxautomation1 type object. |

# Methods

## callmethod()

### Description

Calls the specified method of the associated automation object.

### Prototype

*wxautomation1var*.callmethod ( string *method*, integer *error*, type(=) … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| method | .nul | string | A string containing the name of the method to be called. |
| error | .nul | integer | Specifies an object which is used to output any error code generated during the execution of the function. If error is not specified or is .nul then any error which occurs during function execution will halt the program. If an error object is specified and an error occurs during function execution then the error code is output into that object and the methods returns .nul. |
| … | .nul | type(=) | This will be a parameter being passed to the method. Any number of these can be included as unnamed parameters. |

## getobject()

### Description

Retrieves a property of an automation object as a new automation object.

### Prototype

*wxautomation1var*.getobject ( string *property*, integer *error*, type(=) … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| property | .nul | string | A string containing the name of the property from which a new |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | automation object should be retrieved. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during the execution of the function. If `error` is not specified or is `.nul` then any error which occurs during function execution will halt the program. If an error object is specified and an error occurs during function execution then the error code is output into that object and the methods returns `.nul`. |
| … | `.nul` | type(=) | This will be a parameter being passed to the help define the object that is being retrieved. Any number of these can be included as unnamed parameters. |

# getproperty()

## Description

Retrieves the named property value from the automation object.

## Prototype

*`wxautomation1var`*`.getproperty` ( string *`property`*, integer *`error`*, type(=) … )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| property | `.nul` | string | A string containing the name of the property to be retrieved. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during the execution of the function. If `error` is not specified or is `.nul` then any error which occurs during function execution will halt the program. If an error object is specified and an error occurs during function execution then the error code is output into that object and the methods returns `.nul`. |
| … | `.nul` | type(=) | This will be a parameter being passed to help define from where to retrieve the value. Any number |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | of these can be included as unnamed parameters. |

## putproperty()

### Description

Assigns the value passed into the target property of the associated automation object.

### Prototype

*wxautomation1var*.putproperty ( string *property*, integer *error*, type(=) … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| property | .nul | string | A string containing the name of the property to be assigned. |
| error | .nul | integer | Specifies an object which is used to output any error code generated during the execution of the function. If error is not specified or is .nul then any error which occurs during function execution will halt the program. If an error object is specified and an error occurs during function execution then the error code is output into that object and the methods returns .nul. |
| … | .nul | type(=) | This will be a parameter being passed to help define where to assign the value. Any number of these can be included as unnamed parameters. |

# wxbitmap

# Description

A wxbitmap object represents a bitmap resource that can be provided to any of the wx-based controls that allow a bitmap such as an icon. A separate type may be provided as an image control.

# Type Tags

None

# Object Value

# wxbitmap.new()

## Description

Creates a new wxbitmap object either by loading a bitmap from the supplied file name or by loading the bitmap from the blob supplied. If a filename parameter and format are provided then the blob, width and height should not be provided. The reverse is also true.

## Prototype

*wxbitmap*.new ( string *filename*, string *format*, blob *rgb*, integer *width*, integer *height*, integer *transparencyrgb*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | The name of the file conatining the bitmap to be loaded. |
| format | "xpm" | string | The format of the image to be loaded. The supported formats are: "bmp", "png", "jpg", "gif", "pcx", "pnm", "tif", "tga", "iff", "xpm", "ico", "cur", and "ani". Not all image formats may be available on all platforms. |
| rgb | None | blob | A blob containing an image in rgb format (three bytes per pixel). This allows the creation of the image by the program, or the storage of the image in a database field, for example. |
| width | `rgb.size / height` | integer | The width of the image contained in `rgb`. Either the height or the width must be provided when using the `rgb` argument. |
| height | `rgb.size / width` | integer | The height of the image contained in `rgb`. Either the height or the width must be provided when using the `rgb` argument. |
| transparencyrgb | None | integer | When creating an image from a blob, an RGB value can be provided to define the transparency mask of the image. If it is not provided, then the image will not use transparency. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the wxbitmap object. If `error` is not specified |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `new`() method returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| height | integer | Gives the height of the bitmap in pixels. |
| type | type | Specifies the wxbitmap type object. |
| width | integer | Gives the width of the bitmap in pixels. |

# Methods

## getblob()

### Description

Retrieves a blob object that represents the image that is stored in the bitmap object. The correct interpretation of the blob requires the values of the width and height properties of the bitmap object.

### Prototype

*wxbitmapvar*`.getblob` ()

### Parameters

None

# wxdialog

# Description

A wxdialog object is a dialog window on the screen, which the programmer has control over and is used for interacting with the user. Windows of type wxdialog have the appearance of dialog windows in the

native system, and can be shown modally, that is in a way which forces the user to respond to them before continuing.

# Type Tags

wxcontainer, wxdialogparent

# Object Value

# wxdialog.new()

## Description

Creates a new wxdialog object with the required properties.

## Prototype

*wxdialog*.new ( integer *left*, integer *top*, integer *outerwidth*, integer *outerheight*, integer *innerwidth*, integer *innerheight*, boolean *visible*, type(wxdialogparent) *parent*, boolean *captionbar*, string *captiontext*, boolean *menubutton*, boolean *visbutton*, wxbitmap *icon*, integer *backgroundrgb*, string *stdbuttons*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | None | integer | The position of the left side of the dialog window on the screen. |
| top | None | integer | The position of the top edge of the dialog window on the screen. |
| outerwidth | None | integer | The width of the entire dialog window, including borders, titles, and other parts which are not used by window content. Either `outerwidth` or `innerwidth` must be specified, but it is an error to specify both. |
| outerheight | None | integer | The height of the entire dialog window, including borders, titles, and other parts which are not used by window content. Either `outerheight` or `innerheight` must be specified, but it is an error to specify both. |
| innerwidth | None | integer | The width of the useable, inner part of the window, not including borders, titles, etc., or wxdialogstdbutton buttons, if any. Either `outerwidth` or `innerwidth` must be specified, but it is an error to specify both. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| innerheight | None | integer | The height of the useable, inner part of the window, not including borders, titles, etc., or wxdialogstdbutton buttons, if any. Either `outerheight` or `innerheight` must be specified, but it is an error to specify both. |
| visible | `.true` | boolean | Determines whether or not the dialog window will be visible on the screen. |
| parent | `.nul` | type(wxdialogparent) | Gives the parent of the new dialog window, if any. |
| captionbar | `.true` | boolean | Determines whether or not the dialog window is to have a caption bar, which is used to display text and to allow the user to move the window by dragging it using the mouse. If a window does not have a caption bar, then it cannot have a menu button or vis button, so these must not be specified as arguments with the value of `.true` (but they can be passed as `.false` or omitted). |
| captiontext | "" | string | The text which is to appear in any caption bar the dialog window has. |
| menubutton | `.true` | boolean | Determines whether or not the dialog window is to have a menu button, which allows the user to access a menu giving the ability to manage the window, for example to move or 'close' it. A window must have a caption bar if it to have a menu button, so the `captionbar` argument must not be passed as `.false` (though it can be passed as `.true` or omitted). |
| visbutton | `.true` | boolean | Determines whether or not the dialog window is to have a visibilty button, allowing the user to make the window invisible, which is often considered to be 'closing' it. A window must have a caption bar if it to have a vis button, so the `captionbar` argument must not be passed as `.false` (though it can be passed as `.true` or omitted). |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| icon | None | wxbitmap | The wxbitmap to use as the icon for the new dialog window. It is recommended to use an appropriately sized image for this purpose (16 x 16 pixels). |
| backgroundrgb | The system dependent default dialog window background color | integer | The background color of the new dialog window. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| stdbuttons | "" | string | Specifies which wxdialogstdbutton buttons, if any, are to be created in the dialog. The buttons will be created to appear in a manner which is appropriate for the platform, and the part of the dialog which displays these buttons will be excluded from the area used for showing dialog content. The string is typically a comma separated list of button names, such as "ok,cancel". The valid button names are "ok", "cancel", "yes", "no" and "help". The name of each button can be used together with the member operator (!) to access the button via the wxdialog object. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the wxdialog object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `new` methods returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of |

| Property | Type | Description |
|---|---|---|
| | | being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Shows the color of the inner part of the dialog window when no content is visible. |
| captionbar | boolean | Indicates whether or not the dialog window has a caption bar. The caption bar is used to display a title for the window, and is also used to move the window by dragging with the mouse. |
| captiontext | string | Contains the text which is displayed in the dialog window's caption bar. A dialog window which doesn't have a caption bar can still have caption text. |
| content | type(wxcontent) | The current content of the dialog window. |
| firststdbutton | wxdialogstdbutton | The first button of type wxdialogstdbutton which was created for this dialog window |
| icon | wxbitmap | The wxbitmap object that is used as the source of the icon for the dialog. |
| innerheight | integer | Gives the height of the useable, inner part of the dialog window, not including borders, titles, etc., and also excluding the area containing any wxdialogstdbutton buttons. |
| innerwidth | integer | Gives the width of the useable, inner part of the dialog window, not including borders, titles, etc., and also excluding the area containing any wxdialogstdbutton buttons. |
| left | integer | Gives the x-position of the left hand side of the dialog window on the screen. |
| menubutton | boolean | Indicates whether or not the dialog window has a button in the caption bar which provides the user with access to a menu that allows the window to be managed, for example to be moved or 'closed'. Not all combinations of `.true` and `.false` settings for `menubutton` and `visbutton` are possible on all platforms. |
| modal | boolean | Indicates whether or not the dialog window is currently being shown modally. |
| onmove | event | An event which is triggered when the position of the dialog window is changed by the user. |
| onvisibility-change | event | An event which is triggered when the user changes the visibility of the dialog window, for example by 'closing' the dialog by clicking the visibility button in the caption bar. |
| outerheight | integer | Gives the height of the entire dialog window, including borders, titles, and other parts which are not used by window content or child windows. |

| Property | Type | Description |
|---|---|---|
| outerwidth | integer | Gives the width of the entire dialog window, including borders, titles, and other parts which are not used by window content or child windows. |
| parent | type(wxdialogparent) | The parent of this dialog window. If a dialog window has a parent then it is always shown on top of the parent, though it is not contained within the parent. |
| top | integer | Gives the y-position of the top edge of the dialog window on the screen. |
| type | type | Specifies the wxdialog type object. |
| visbutton | boolean | Indicates whether or not the dialog window has a button in the caption bar with which the user can make the window invisible. This is normally thought of as 'closing' the window. Not all combinations of `.true` and `.false` settings for `menubutton` and `visbutton` are possible on all platforms. |
| visible | boolean | Indicates whether or not the dialog window is currently visible on the screen. |

# Methods

## processmodal()

### Description

Makes the dialog window visible on the screen, and processes user input to it modally, i.e. without allowing input to other dialogs or windows in the application. It is an error to call this method when the dialog is already visible, either modally or modelessly. The wxdialog object itself is returned.

### Prototype

*`wxdialogvar`*`.processmodal` ( integer *`timeout`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| timeout | `0` | integer | Specifies an amount of time (in microseconds) for which the method will continue to show the dialog and process user interaction, unless the dialog window is made not visible by calling the `setvisible()` method with a parameter value of `.false`. If a value of `.inf` is specified, then user interaction will be processed indefinitely, or until the dialog is made not visible using the `setvisible()` method. |

# setbackgroundrgb()

## Description

Sets the background color of the dialog window. The wxdialog object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

## Prototype

*`wxdialogvar`*.`setbackgroundrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `backgroundrgb` property | integer | The new background color of the dialog. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the window. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the window. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the window. This must be between 0 and 255 inclusive. |

# setcaptiontext()

## Description

Sets the text in the caption bar, if any, of the dialog window. The wxdialog object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxdialogvar`*.`setcaptiontext` ( string *`captiontext`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captiontext | None | string | The text which is to appear in any caption bar the dialog has. |

# seticon()

## Description

Sets the icon for the dialog. This must be a dialog that has a caption bar, otherwise an error is returned in the *error* parameter. The wxdialog object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxdialogvar*.seticon ( wxbitmap *icon*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| icon | None | wxbitmap | The wxbitmap that should be used as the dialog icon. This should be a bitmap that is 16x16 pixels normally. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during execution of the function. If error is not specified or is .nul then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

# setposition()

## Description

Sets the size and/or position of a dialog window. The wxdialog object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxdialogvar*.setposition ( integer *left*, integer *top*, integer *outerwidth*, integer *outerheight*, integer *innerwidth*, integer *innerheight* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the dialog window on the screen |
| top | The current value of the top property | integer | The new position of the top edge of the dialog window on the screen |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| outerwidth | The current value of the `outerwidth` property | integer | The new width of the entire dialog window, including borders, titles, and other parts which are not used by dialog content or wxdialogstdbutton buttons. It is an error to specify both `outerwidth` and `innerwidth` arguments. |
| outerheight | The current value of the `outerheight` property | integer | The new height of the entire dialog window, including borders, titles, and other parts which are not used by dialog content or wxdialogstdbutton buttons. It is an error to specify both `outerheight` and `innerheight` arguments. |
| innerwidth | The current value of the `innerwidth` property | integer | The new width of the useable, inner part of the dialog, not including borders, titles, etc., and also excluding the area containing any wxdialogstdbutton buttons. It is an error to specify both `outerwidth` and `innerwidth` arguments. |
| innerheight | The current value of the `innerheight` property | integer | The new height of the useable, inner part of the dialog, not including borders, titles, etc., and also excluding the area containing any wxdialogstdbutton buttons. It is an error to specify both `outerheight` and `innerheight` arguments. |

# setvisible()

## Description

Sets whether or not the dialog window is visible on the screen. The wxdialog object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxdialogvar*.`setvisible` ( boolean *`visible`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | Determines whether or not the window will be visible on the screen. |

# wxdialog!

## Get

The member operator (!) is used to access the objects of type wxdialogstdbutton that may have been specified during the creation of the dialog window. These can be any of `yes`, `no`, `ok`, `cancel`, or `help`. The member operator followed by the button name returns a reference to the button. Attempting to access a button that has not been defined is an error. As an example, if the standard buttons parameter contains the string `"ok"`, then the code might look like: `wxdialogvar!ok`.

## Set

Attempting to assign a reference or a value to a wxdialog object using the ! operator is not supported.

# wxdialogstdbutton

# Description

A wxdialogstdbutton object is a button on a dialog window which is created when the dialog is created, which is placed outside of the area within the window which is used to display window content, and which will have an appearance which is appropriate for the platform.

# Type Tags

None

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| dialog | wxdialog | Specifies the wxdialog object which this button belongs to. |
| name | string | The name of the wxdialogstdbutton object. |
| next | wxdialogstdbutton | Specifies the next wxdialogstdbutton button on the same dialog. |
| onclick | event | An event that is triggered every time the user clicks the button. |
| type | type | Specifies the wxdialogstdbutton type object. |

# wxfont

## Description

A wxfont object represents a font definition that can be referenced by any of the wxformcontrol objects that provide text.

## Type Tags

None

## Object Value

## wxfont.new()

### Description

Creates a new wxfont object with the specified face name, point size and features.

### Prototype

*wxfont*.new ( string *facename*, integer *pointsize*, string *style*, string *weight*, string *decoration*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| facename | None | string | The face name of the new font, such as "Arial", or "Times". |
| pointsize | None | integer | The size of the new font, in points. |
| style | `.nul` | string | The style of the new font. Use `"i"` for italic. |
| weight | `.nul` | string | The weight of the new font. Use `"b"` for bold. |
| decoration | `.nul` | string | The decoration of the new font. Use `"u"` for underlined and `"s"` for strikethrough. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the wxfont object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | object and the `new`() method returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| decoration | string | Gives the decoration of the font. The letter `u` stands for underlined. The letter `s` stands for strikethrough. The empty string implies no decoration. This could contain either `s`, `u`, both or the empty string. |
| facename | string | Gives the face name of the font. |
| pointsize | integer | Gives the size of the font, in points. |
| style | string | Gives the style of the font. The letter `i` stands for italic and the letter `n` stands for normal. |
| type | type | Specifies the wxfont type object. |
| weight | string | Gives the weight of the font. The letter `b` stands for bold, the letter `l` for light and the letter `n` for normal. |

# wxform

# Description

A wxform object is a single 'page' on which a program creates controls for user interaction. The form is displayed to the user by setting it to be the content of some wxwindow, wxdialog, or other appropriate container.

# Type Tags

wxcontent, wxgraphiccontainer

# Object Value

# wxform.new()

## Description

Creates a new wxform object of the specified size and color.

## Prototype

*wxform*.new ( integer *width*, integer *height*, integer *backgroundrgb*, type(*) *cursor*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | None | integer | The width of the new form, in pixels. |
| height | None | integer | The height of the new form, in pixels. |
| backgroundrgb | White | integer | The background color of the new form. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| cursor | None | type(*) | The cursor that should be used when the mouse is over the form. |
| error | .nul | integer | Specifies an object which is used to output any error code generated during creation of the form1 object. If error is not specified or is .nul then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the new() method returns .nul. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the background of the form. |
| container | type(wxcontainer) | Specifies the container object in which this form is currently contained, such as a window or dialog. |
| controlscaptured | boolean | Indicates whether the mouse is currently captured or not. |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td>cursor</td><td>type(*)</td><td>The current type of mouse pointer that is in use. If this is .nul, then the defaults are in use.</td></tr>
<tr><td>firstcontrol</td><td>type(wxformcontrol)</td><td>Gives the first control on the form, or is <code>.nul</code> if there are no controls on the form.</td></tr>
<tr><td>firstgraphic</td><td>type(wxgraphic)</td><td>Gives the first graphic item on the form, or is <code>.nul</code> if there are no graphic items on the form. These include lines, triangles, rectangles, arcs, and ellipses.</td></tr>
<tr><td>focuscontrol</td><td>type(wxformcontrol)</td><td>Gives the control on this form which receives user input. For input to reach the form from the user, it must be in some container (such as a window) which is active. If no control on the form has focus then this property is set to <code>.nul</code>.</td></tr>
<tr><td>height</td><td>integer</td><td>Gives the height of the form, in pixels.</td></tr>
<tr><td>mousecaptured</td><td>boolean</td><td>Indicates whether the mouse is currently captured or not.</td></tr>
<tr><td>onmouse</td><td>event</td><td>An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: <code>function onmouse(wxform form, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<ul><li>etype is the bit (defined as for onmousemask) showing what sort of event it is.</li><li>keys is a collection of bits showing key positions:<table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr><tr><td>alt down</td><td>0x00040000</td></tr><tr><td>meta down</td><td>0x00080000</td></tr></table></li><li>x and y are the position of the mouse event</li><li>reference is the (optional) reference object in the wxform.onmouse.reference</li></ul></td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>type</td><td>type</td><td>Specifies the wxform type object.</td></tr>
<tr><td>width</td><td>integer</td><td>Gives the width of the form, in pixels.</td></tr>
</table>

# Methods

## addcontrol()

### Description

Creates a new `wxformcontrol` object of the specified type, using the given arguments for many of the control's properties.

### Prototype

*wxformvar*.`addcontrol` ( type *`type`*, integer *`left`*, integer *`top`*, integer *`width`*, integer *`height`*, string *`text`*, boolean *`enabled`*, boolean *`visible`*, wxbitmap *`bitmap`*, string *`scaling`*, wxbitmap *`selectedbitmap`*, wxbitmap *`disabledbitmap`*, wxbitmap *`focusbitmap`*, integer *`backgroundrgb`*, integer *`textrgb`*, integer *`rgb`*, string *`edittype`*, string *`selectiontype`*, integer *`rowcount`*, integer *`colcount`*, integer *`rowheight`*, integer *`colwidth`*, boolean *`rowheightdraggable`*, boolean *`colwidthdraggable`*, integer *`rowlabelwidth`*, integer *`collabelheight`*, string *`rowlabelalignment`*, string *`collabelalignment`*, string *`alignment`*, string *`editstyle`*, string *`orientation`*, integer *`range`*, integer *`position`*, integer *`pagesize`*, integer *`thumbsize`*, wxfont *`font`*, wxfont *`labelfont`*, string *`tooltip`*, integer *`onmousemask`*, string *`name`*, type(wxformcontrol) *`next`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| type | None | type | The type of the new control. This must be of a type that is tagged as being a `wxformcontrol`. |
| left | None | integer | The position of the left side of the new control on the form, in pixels. The control must be fully contained within the bounds of the form. |
| top | None | integer | The position of the top edge of the new control on the form, in pixels. The control must be fully contained within the bounds of the form. |
| width | None | integer | The width of the new control on the form, in pixels. The control must be fully contained within the form. |
| height | None | integer | The height of the new control on the form, in pixels. The control must be fully contained within the form. |
| text | "" | string | The text for the new control. This does not apply to wxformlist controls; it is an error to supply this argument when creating a wxformlist. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | `.true` | boolean | Whether the new control is enabled or not. |
| visible | `.true` | boolean | Whether the new control is visible or not. |
| bitmap | None | wxbitmap | The standard bitmap for a wxformbitmapbutton control. |
| scaling | "" | string | Determines the type of scaling used on the bitmap. This can be one of: `"hfit"`, `"vfit"`, `"handvfit"`, `"preserveaspect"`, or the empty string. If no scaling is specified, then the image will be cropped to fit the container as needed. Otherwise the image will be stretched horizontally to fill the container, vertically to fill the container, or in both directions. |
| selectedbitmap | None | wxbitmap | The bitmap for a wxformbitmapbutton control that will be shown when the button is selected. |
| disabledbitmap | None | wxbitmap | The bitmap for a wxformbitmapbutton control that will be shown when the button is disabled. |
| focusbitmap | None | wxbitmap | The bitmap for a wxformbitmapbutton control that will be shown when the button has focus. |
| backgroundrgb | The current background color of the form | integer | The background color of the new control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| textrgb | Black | integer | The color of the text in the new control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| rgb | Black | integer | The color of the new wxformsizebox control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| edittype | "droplist" | string | The style for a new wxformcombo control. It is an error to supply this argument when creating any control type other than a wxformcombo control. Possible values are: "dropedit" and "droplist". |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| selectiontype | "single" | string | The style for a new wxformlist control. It is an error to supply this argument when creating any control type other than a wxformlist control. Possible values are: "single", "multiple" and "extended". |
| rowcount | None | integer | The number of rows for a new wxformgrid control. |
| colcount | None | integer | The number of columns for a new wxformgrid control. |
| rowheight | None | integer | The default height for a row in a new wxformgrid control. |
| colwidth | None | integer | The default width for a column in a new wxformgrid control. |
| rowheightdraggable | `.true` | boolean | Toggle that decides if the user can change the height of the rows. |
| colwidthdraggable | `.true` | boolean | Toggle that decides if the user can change the width of the columns. |
| rowlabelwidth | 80 | integer | The width of the row label column for a new wxformgrid control. |
| collabelheight | 20 | integer | The height of the column label row for a new wxformgrid control. |
| rowlabelalignment | `right` | string | The alignment for grid row labels. This defaults to `"right"`, which is centered vertically and then right-aligned. |
| collabelalignment | `""` | string | The alignment for grid column labels. This defaults to the empty string, which is centered both horizontally and vertically. |
| alignment | `""` | string | The default alignment for grid cells as well as the alignment for wxformtext and wxformedittext controls. In the case of a wxformgrid this is normally the empty string, which is centered both horizontally and vertically. In the case of a wxformtext or a wxformedittext control, the default is `.nul`, which will be left aligned. Valid values are: `left`, `right`, `top`, and `bottom`. For wxformtext and wxformedittext controls, this must be set at creation time and only the horizontal alignment |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | values are valid. To center something, use the empty string. For complete information on alignment for grids, see: the section called “wxformgrid”. |
| editstyle | "" | string | This describes the style of wxformedittext control that is created. This can only be specified at creation time. It can have any combination of the values: `"hscroll"`, `"multiline"`, `"password"`, and `"readonly"` whereby password and multiline together will not provide a password style, but it does not cause an error. The value `"hscroll"` by itself may make little difference, since values that are too large normally scroll horizontally anyway, and without a definitive size, they appear to have no function. The `"multiline"` value will show a scrollbar if it is needed. It also allows the entering of new line characters into the textbox. As usual, the `"password"` style shows the content as a series of asterisks and the `"readonly"` style prevents modification of the content, but it can be marked and copied to clipboard. |
| orientation | None | string | The orientation for a scrollbar control. Valid values are: `"v"` for vertical and `"h"` for horizontal scrollbars. |
| range | 1 | integer | The range for a scrollbar control. |
| position | 0 | integer | The starting position for a scrollbar control. |
| pagesize | None | integer | The size of a page for a scrollbar control. |
| thumbsize | 1 | integer | The size of the thumb in a scrollbar control. |
| font | `.nul` | wxfont | A valid wxfont object that will be used as the font for associated controls. |
| labelfont | `.nul` | wxfont | A valid wxfont object that will be used for the row and column labels of a grid control |

<table>
<tr><th>Parameter</th><th>Default value</th><th>Type name</th><th>Description</th></tr>
<tr><td>tooltip</td><td>.nul</td><td>string</td><td>The text to be used as the tooltip for the control.</td></tr>
<tr><td>onmousemask</td><td>None</td><td>integer</td><td>The definition of which events should be handled by the on-mouse event handler. The onmousemask is made up of bits:<br><table><tr><th>Event Type</th><th>Bit Value</th></tr><tr><td>left button down</td><td>0x00000001</td></tr><tr><td>left button up</td><td>0x00000002</td></tr><tr><td>left button double click</td><td>0x00000004</td></tr><tr><td>middle button down</td><td>0x00000010</td></tr><tr><td>middle button up</td><td>0x00000020</td></tr><tr><td>middle button dbl click</td><td>0x00000040</td></tr><tr><td>right button down</td><td>0x00000100</td></tr><tr><td>right button up</td><td>0x00000200</td></tr><tr><td>right button dbl click</td><td>0x00000400</td></tr><tr><td>motion</td><td>0x00001000</td></tr></table></td></tr>
<tr><td>name</td><td>.nul</td><td>string</td><td>The name of the control. This can be used together with the member operator (!) to access the form control via the wxform object.</td></tr>
<tr><td>next</td><td>.nul</td><td>type(wxformcontrol)</td><td>This determines where in the ring of controls the new control will be placed. The new control will be created so that the control specified as next will be after the new control in the ring. If the next control is the current firstcontrol for the form, then the new control will become the new firstcontrol. If there are already some controls on the form and next is specified to be .nul then the next control will be the current firstcontrol for the form, but the new control will not become a new firstcontrol (i.e. it becomes a 'last' control).</td></tr>
<tr><td>error</td><td>.nul</td><td>integer</td><td>Specifies an object which is used to output any error code generated during creation of the wxformcontrol object. If error is not specified or is .nul then any error which occurs during object</td></tr>
</table>

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `addcontrol` method returns `.nul`. |

# addgraphic()

## Description

Creates a new type(wxgraphic) object of the specified type, using the given arguments for the graphic's properties. Graphics are always behind all controls in the z-order of the form.

## Prototype

*`wxformvar`*`.addgraphic` ( type *`type`*, point *`point1`*, point *`point2`*, point *`point3`*, point *`midpoint`*, integer *`rgb`*, integer *`borderrgb`*, integer *`width`*, integer *`borderwidth`*, boolean *`visible`*, boolean *`bordervisible`*, string *`name`*, type(wxgraphic) *`next`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| type | None | type | The type of the new graphic. This must be of a type that is tagged as being a `wxgraphic`. |
| point1 | None | point | The starting point for the graphic. This can be created inline using: `point.new(10, 10)` for example. |
| point2 | None | point | The second point of the graphic. This can be created inline using: `point.new(10, 10)` for example. Most graphic types only have two points, such as the line, rectangle, ellipse, and arc. |
| point3 | None | point | This is the third vertex point for a triangle. Currently no other graphic type makes use of this parameter. This can be created inline using: `point.new(10, 10)` for example. |
| midpoint | None | point | This point represents the center of an ellipse or arc. This can be created inline using: `point.new(10, 10)` for example. |
| rgb | White | integer | The fill color of the new wxgraphic object. In the case of the line object, this is the color of the line, in all others the color of the bor- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | der lines is handled by the borderrgb parameter. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| borderrgb | Black | integer | The color of the lines representing the border of the new wxgraphic object. In the case of the line object, this parameter is invalid, use the rgb parameter instead. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| width | 1 | integer | This is the width in pixels of a line object. The minimum value is `1`. To make the line invisible, set its visibility to `.false`. |
| borderwidth | 1 | integer | This is the width in pixels of the border of a graphic object. The minimum value is `1`. If no border is desired, set its visibility to `.false`. |
| visible | `.true` | boolean | Whether the content of the new graphic is visible or not. In the case of a line object, this decides if the line is visible, in the case of the other objects, this decides if the object is filled. If this portion is set to invisible, then objects behind it will be visible. To set any fillable object (triangle, rectangle, arc, or ellipse) to be completely invisible, both this and the bordervisible properties must be set to `.false`. |
| bordervisible | `.true` | boolean | Whether the border of the new graphic is visible or not. This parameter is invalid for line objects. If this portion is set to invisible, then no border will be shown. If the visible parameter is also set to `.false`, then the entrie graphic will be invisible. |
| name | `.nul` | string | The name of the graphic. This can be used together with the member operator (!) to access the graphic on a form via the wxform object. |
| next | `.nul` | type(wxgraphic) | This determines where in the ring of graphics the new graphic |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | will be placed. The new graphic will be created so that the graphic specified as `next` will be after the new graphic in the ring. If the `next` graphic is the current `firstgraphic` for the form, then the new graphic will become the new `firstgraphic`. If there are already some graphics on the form and `next` is specified to be `.nul` then the next graphic will be the current `firstgraphic` for the form, but the new graphic will not become a new `firstgraphic` (i.e. it becomes the 'last' graphic). |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the `wxgraphic` object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `addgraphic` method returns `.nul`. |

# clearfocus()

## Description

Removes focus from all controls on a form.

## Prototype

*wxformvar*.`clearfocus` ()

## Parameters

None

# controlcapture()

## Description

Internal. Reserved.

## Prototype

*wxformvar*.`controlcapture` ()

### Parameters

None

## controlrelease()

### Description

Internal. Reserved.

### Prototype

*wxformvar*.controlrelease ()

### Parameters

None

## freeze()

### Description

Freezes all painting updates of the form until the thaw() method is called. Multiple calls to both methods are counted. A freeze of the paint messages to the form will occur on the first call. Additional calls will increment/decrement the counter. When the counter reaches zero again following a call to thaw(), the form will be redrawn and normal painting will occur once more.

**Note**

USE OF THIS FUNCTION SHOULD ONLY BE DONE WITH CARE, IT SHOULD ALWAYS BE FOR A BRIEF TIME, AND BOTH THE FREEZE() AND THAW() SHOULD BE IN THE SAME FUNCTION!

### Prototype

*wxformvar*.freeze ()

### Parameters

None

## mousecapture()

### Description

Internal. Reserved.

### Prototype

*wxformvar*.mousecapture ()

### Parameters

None

# mouserelease()

## Description

Internal. Reserved.

## Prototype

*wxformvar*.mouserelease ()

## Parameters

None

# popupmenu()

## Description

Displays a menu object as a popup menu. The menu object may not already be used as part of an existing menubar. Optionally an x and a y parameter may be passed to establish where the menu should appear. If not specified, it appears at the mouse pointer position.

## Prototype

*wxformvar*.popupmenu ( wxmenu *menu*, integer *x*, integer *y* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| menu | None | wxmenu | The menu object to be shown. |
| x | Current x position of the mouse pointer | integer | The desired x-coordinate where the menu should appear. |
| y | Current y position of the mouse pointer | integer | The desired y-coordinate where the menu should appear. |

# rubberbandarc()

## Description

Draws a rubber band style arc on the form. Which part of the arc is flexible depends on which point is passed, *point1* or *point2*. Only one of those two parameters may be passed! When the left mouse button is released, the method returns. The position of the mouse will be returned in the *point* parameter. The *midpoint* is required. If the output result is "buttonup" or "lostmouse" then the *point1* and *point2*, parameters will contain the position of the rubber band line on termination, otherwise their contents should not be used (currently will be .nul). The *result* parameter could also contain "notready" or "cancel". The coordinates are relative to the origin of the form.

## Prototype

*wxformvar*.rubberbandarc ( point|fixedpoint *midpoint*, point|fixedpoint *point1*, point|fixedpoint *point2*, point *point*, string *results* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| midpoint | `.nul` | point\|fixedpoint | The midpoint of the arc. This must be a preinitialized object! |
| point1 | `.nul` | point\|fixedpoint | The first point of the line. This must be a preinitialized object! |
| point2 | `.nul` | point\|fixedpoint | The second point of the line. This must be a preinitialized object! |
| point | `.nul` | point | The second point of the line. This must be a preinitialized object! |
| results | `.nul` | string | The results of the function. This can be one of: `"buttonup"`, `"lostmouse"`, `"notready"` or `"cancel"`. If one of the first two values is returned, then the other parameters will be filled with the coordinates of the selection, otherwise they are invalid (and currently will be `".nul"`. |

## rubberbandbox()

### Description

Draws a rubber band style box on the form. When the left mouse button is released, the method returns. If the output result is `"buttonup"` or `"lostmouse"` then the *`left`*, *`top`*, *`width`*, and *`height`* parameters will contain the position of the rubber band box on termination, otherwise their contents should not be used (currently will be .nul). The *`result`* parameter could also contain `"notready"` or `"cancel"`. The coordinates are relative to the origin of the form.

### Prototype

*`wxformvar`*`.rubberbandbox` ( integer *`left`*, integer *`top`*, integer *`width`*, integer *`height`*, string *`results`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | `.nul` | integer | The left coordinate of the selection box. This must be a preinitialized object! |
| top | `.nul` | integer | The top coordinate of the selection box. This must be a preinitialized object! |
| width | `.nul` | integer | The width of the selection box. This must be a preinitialized object! |
| height | `.nul` | integer | The height of the selection box. This must be a preinitialized object! |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| results | `.nul` | string | The results of the function. This can be one of: `"buttonup"`, `"lostmouse"`, `"notready"` or `"cancel"`. If one of the first two values is returned, then the other parameters will be filled with the coordinates of the selection box, otherwise they are invalid (and currently will be `".nul"`. |

# rubberbandellipsebox()

## Description

Draws a rubber band style ellipse on the form. When the left mouse button is released, the method returns. If the output result is `"buttonup"` or `"lostmouse"` then the *`left`*, *`top`*, *`width`*, and *`height`* parameters will contain the position of the rubber band ellipse on termination, otherwise their contents should not be used (currently will be .nul). The *`result`* parameter could also contain `"notready"` or `"cancel"`. The coordinates are relative to the origin of the form. To create an ellipse from these coordinates, the midpoint of the width at the top, the midpoint of the height at the right, and the exact midpoint of the two must be passed to the method that creates the ellipse.

## Prototype

*`wxformvar`*`.rubberbandellipsebox` ( integer *`left`*, integer *`top`*, integer *`width`*, integer *`height`*, string *`results`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | `.nul` | integer | The left coordinate of the selection box. This must be a preinitialized object! |
| top | `.nul` | integer | The top coordinate of the selection box. This must be a preinitialized object! |
| width | `.nul` | integer | The width of the selection box. This must be a preinitialized object! |
| height | `.nul` | integer | The height of the selection box. This must be a preinitialized object! |
| results | `.nul` | string | The results of the function. This can be one of: `"buttonup"`, `"lostmouse"`, `"notready"` or `"cancel"`. If one of the first two values is returned, then the other parameters will be filled |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | with the coordinates of the selection box, otherwise they are invalid (and currently will be ".nul". |

# rubberbandline()

## Description

Draws a rubber band style line on the form. When the left mouse button is released, the method returns. If the output result is "buttonup" or "lostmouse" then the *point1* and *point2*, parameters will contain the position of the rubber band line on termination, otherwise their contents should not be used (currently will be .nul). The *result* parameter could also contain "notready" or "cancel". The coordinates are relative to the origin of the form.

## Prototype

*wxformvar*.rubberbandline ( point *point1*, point *point2*, string *results* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point1 | .nul | point | The first point of the line. This must be a preinitialized object! |
| point2 | .nul | point | The second point of the line. This must be a preinitialized object! |
| results | .nul | string | The results of the function. This can be one of: "buttonup", "lostmouse", "notready" or "cancel". If one of the first two values is returned, then the other parameters will be filled with the coordinates of the selection, otherwise they are invalid (and currently will be ".nul". |

# rubberbandmultiline()

## Description

Draws one or more rubber band style lines on the form that converge from the points supplied to the current position of the mouse. When the left mouse button is released, the method returns. If the output result is "buttonup" or "lostmouse" then the *point* parameter will contain the position of the mouse pointer on termination, otherwise its contents should not be used (currently will be .nul). The *result* parameter could also contain "notready" or "cancel". The coordinates are relative to the origin of the form.

## Prototype

*wxformvar*.rubberbandmultiline ( point *point*, string *results*, point , point )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point | `.nul` | point | The position of the mouse at termination. This must be a preinitialized object! |
| results | `.nul` | string | The results of the function. This can be one of: `"buttonup"`, `"lostmouse"`, `"notready"` or `"cancel"`. If one of the first two values is returned, then the other parameters will be filled with the coordinates of the selection, otherwise they are invalid (and currently will be `".nul"`. |
| | `.nul` | point | The first point from which a line will be drawn to the position of the mouse pointer. This must be a preinitialized object! |
| | `.nul` | point | The next point from which a line will be drawn to the position of the mouse pointer. This must be a preinitialized object! Any number of points can be added to be drawn from in the multiline function. They are not named parameters. |

## setbackgroundrgb()

### Description

Sets the background color of the form. The wxform object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*`wxformvar`*`.setbackgroundrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `backgroundrgb` property | integer | The new background color of the form. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the form. This must be between 0 and 255 inclusive. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the form. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the form. This must be between 0 and 255 inclusive. |

# setcontainer()

## Description

Sets the form to be the content of some suitable container. The wxform object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformvar*.`setcontainer` ( type(wxcontainer) *`container`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| container | None | type(wxcontainer) | The object of which the form is to become the content. |

# setcursor()

## Description

Sets the current mouse pointer for use when the mouse is over the form. The wxform object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformvar*.`setcursor` ( string *`cursor`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| cursor | None | string | The name of the cursor type to use when the mouse is over the form. This can be any one of:<br>• "arrow"<br>• "rightarrow"<br>• "blank"<br>• "bullseye" |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | • "char"<br>• "cross"<br>• "hand"<br>• "ibeam"<br>• "leftbutton" — GTK+ only<br>• "magnifier"<br>• "middlebutton" — GTK+ only<br>• "noentry"<br>• "paintbrush"<br>• "pencil"<br>• "pointleft"<br>• "pointright"<br>• "questionarrow"<br>• "rightbutton" — GTK+ only<br>• "nesw"<br>• "ns"<br>• "nwse"<br>• "we"<br>• "sizing"<br>• "spraycan"<br>• "wait"<br>• "watch"<br>• "arrowwait"<br>Use the value `.nul` to return the cursor to the default behavior. |

# setkeyfocus()

## Description

Tells the form to create a special control to hold the focus and to direct keystroke input that occurs while that special control has focus to the function provided. If focus is lost, then that will be relayed to the appropriate function as defined by the user.

## Prototype

*wxformvar*.setkeyfocus ( function *onkey*, type(*) *onkeyreference*, function *onlostfocus*, type(*) *onlostfocusreference* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onkey | None | function | This is a reference to the function that will handle the key events. The function must have the prototype: `function onkey(wxform form, string char, integer keys, type(*) reference)` The *char* will contain either a single character, or else a string describing the key that was pressed. The list of currently supported strings is:<br>• "back"<br>• "tab"<br>• "return"<br>• "escape"<br>• "delete"<br>• "end"<br>• "home"<br>• "left"<br>• "up"<br>• "right"<br>• "down"<br>• "insert"<br>• "pageup"<br>• "pagedown"<br>• "f1"–"f24"<br>The *keys* is the same as in the onmouse event. It contains which additional keys were held down; *keys* is a collection of bits showing key positions: |

<table>
<tr><th>Parameter</th><th>Default value</th><th>Type name</th><th>Description</th></tr>
<tr><td></td><td></td><td></td><td>
<table>
<tr><th>Key Type</th><th>Bit Value</th></tr>
<tr><td>shift down</td><td>0x00010000</td></tr>
<tr><td>ctrl down</td><td>0x00020000</td></tr>
<tr><td>alt down</td><td>0x00040000</td></tr>
<tr><td>meta down</td><td>0x00080000</td></tr>
</table>
Finally the reference contains an optionally defined parameter of user-defined type, as is typical for event handling mechanisms in SIMPOL.
</td></tr>
<tr><td>onkeyreference</td><td>.nul</td><td>type(*)</td><td>This can be any object desired by the user. If it is defined, it will be passed to the function defined to handle the onkey event.</td></tr>
<tr><td>onlostfocus</td><td>.nul</td><td>function</td><td>This function will be called when the key event handler loses focus, either by the user clicking elsewhere on the form, alt-tabbing away, or otherwise causing focus to be lost. Once focus is lost, the key trapping functionality will end.</td></tr>
<tr><td>onlostfocusreference</td><td>.nul</td><td>type(*)</td><td>This can be any object desired by the user. If it is defined, it will be passed to the function defined to handle the onlostfocus event.</td></tr>
</table>

# setonmousemask()

## Description

Sets the mask that is used to decide which mouse events that occur on the form. This does not apply to events that involve controls, only those that occur on the form itself. The wxform object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformvar*.setonmousemask ( integer *onmousemask* )

## Parameters

<table>
<tr><th>Parameter</th><th>Default value</th><th>Type name</th><th>Description</th></tr>
<tr><td>onmousemask</td><td>None</td><td>integer</td><td>
The definition of which events should be handled by the onmouse event handler. The onmousemask is made up of bits:
<table>
<tr><th>Event Type</th><th>Bit Value</th></tr>
<tr><td>left button down</td><td>0x00000001</td></tr>
</table>
</td></tr>
</table>

| Parameter | Default value | Type name | Description | |
|---|---|---|---|---|
| | | | **Event Type** | **Bit Value** |
| | | | left button up | 0x00000002 |
| | | | left button double click | 0x00000004 |
| | | | middle button down | 0x00000010 |
| | | | middle button up | 0x00000020 |
| | | | middle button dbl click | 0x00000040 |
| | | | right button down | 0x00000100 |
| | | | right button up | 0x00000200 |
| | | | right button dbl click | 0x00000400 |
| | | | motion | 0x00001000 |

## thaw()

### Description

This restores the painting updates of the form following a call to the `freeze`() method. Multiple calls to both methods are counted. A freeze of the paint messages to the form will occur on the first call. Additional calls will increment/decrement the counter. When the counter reaches zero again following a call to `thaw`(), the form will be redrawn and normal painting will occur once more.

**Note**

USE OF THIS FUNCTION SHOULD ONLY BE DONE WITH CARE, IT SHOULD ALWAYS BE FOR A BRIEF TIME, AND BOTH THE FREEZE() AND THAW() SHOULD BE IN THE SAME FUNCTION!

### Prototype

*wxformvar*.thaw ()

### Parameters

None

# wxform!

## Get

The member operator followed by the form control name returns a reference to the form control. If the name provided is not correct (including case-sensitivity), then an error will occur.

## Set

Attempting to assign a reference or a value to a wxform object using the ! operator is not supported.

# wxformbitmap

## Description

A wxformbitmap object is an image control that can display an image on a form in any size (wxformbitmapbutton has a size limitation on some platforms of maximum 64 pixels in each dimension). Images can be unchanged, stretched or squeezed to fit the container, or scaled to fit the container while retaining the aspect ratio.

## Type Tags

wxformcontrol

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the background of the control in the case that the control is not completely filled by the associated bitmap. |
| bitmap | wxbitmap | The wxbitmap object that is used as the source of the bitmap in the control. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformbitmap objects this is always `.false` |
| form | wxform | Specifies the wxform object to which this control belongs. |
| height | integer | Gives the height of the control, in pixels. |
| left | integer | Gives the position of the left side of the control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformbitmap object. |
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td>onmouse</td><td>event</td><td>An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: <code>function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<ul><li>etype is the bit (defined as for onmousemask) showing what sort of event it is.</li><li>keys is a collection of bits showing key positions:<table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr><tr><td>alt down</td><td>0x00040000</td></tr><tr><td>meta down</td><td>0x00080000</td></tr></table></li><li>x and y are the position of the mouse event</li><li>reference is the (optional) reference object in the wxformbitmap.onmouse.reference</li></ul></td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>scaling</td><td>string</td><td>Determines the type of scaling used on the bitmap. This can be one of: <code>"hfit"</code>, <code>"vfit"</code>, <code>"handvfit"</code>, <code>"preserveaspect"</code>, or the empty string. If no scaling is specified, then the image will be cropped to fit the control as needed. Otherwise the image will be stretched horizontally to fill the container, vertically to fill the container, or in both directions.</td></tr>
<tr><td>tooltip</td><td>string</td><td>Contains the test that is displayed as a tooltip for the control.</td></tr>
<tr><td>top</td><td>integer</td><td>Gives the position of the top edge of the control relative to the top edge of the form, in pixels.</td></tr>
<tr><td>type</td><td>type</td><td>Specifies the wxformbitmap type object.</td></tr>
<tr><td>visible</td><td>boolean</td><td>Specifies whether or not the control is visible.</td></tr>
<tr><td>width</td><td>integer</td><td>Gives the width of the control, in pixels.</td></tr>
</table>

# Methods

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*wxformbitmapvar*.remove ()

### Parameters

None

## setbackgroundrgb()

### Description

Sets the color of the control. The wxformbitmap object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green`, or `blue` arguments.

### Prototype

*wxformbitmapvar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the rgb property | integer | The new color of the control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the rgb.red property | integer | The red component in the new color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the rgb.green property | integer | The green component in the new color of the control. This must be between 0 and 255 inclusive. |
| blue | The current value of the rgb.blue property | integer | The blue component in the new color of the control. This must be between 0 and 255 inclusive. |

# setbitmap()

## Description

Sets the bitmap associated with the control. The wxformbitmap object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapvar*.setbitmap ( wxbitmap *bitmap* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bitmap | None | wxbitmap | The wxbitmap object to be assigned as the source of the bitmap in the control. |

# setcaptureable()

## Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

## Prototype

*wxformbitmapvar*.setcaptureable ( boolean *captureable* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the `controlcapture()` method of the form. |

# setenabled()

## Description

Sets the enabled state of the control. The wxformbitmap object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapvar*.setenabled ( boolean *enabled* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the control. |

# setname()

## Description

Sets the name of the bitmap control. The wxformbitmap object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapvar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

## Prototype

*wxformbitmapvar*.setnext ( type(wxformcontrol) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

# setonmousemask()

## Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformbitmap object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapvar*.setonmousemask ( integer *onmousemask* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onmousemask | None | integer | The definition of which events should be handled by the on-mouse event handler. The *onmousemask* is made up of bits:<br>**Event Type** / **Bit Value**<br>left button down / 0x00000001<br>left button up / 0x00000002<br>left button double click / 0x00000004<br>middle button down / 0x00000010<br>middle button up / 0x00000020<br>middle button dbl click / 0x00000040<br>right button down / 0x00000100<br>right button up / 0x00000200<br>right button dbl click / 0x00000400<br>motion / 0x00001000 |

## setposition()

### Description

Sets the size and/or position of the control. The wxformbitmap object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformbitmapvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the control on the form. |
| top | The current value of the top property | integer | The new position of the top edge of the control on the form. |
| width | The current value of the width property | integer | The new width of the control on the form. |
| height | The current value of the height property | integer | The new height of the control on the form. |

# setscaling()

## Description

Sets the scaling method to be used for the bitmap in the control. The wxformbitmap object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapvar*.setscaling ( string *scaling* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| scaling | None | string | The scaling method to be used when adjusting the bitmap to its container. This is one of: "hfit", "vfit", "handvfit", "preserveaspect", or the empty string (""). If no scaling is specified, then the image will be cropped to fit the container as needed. Otherwise the image will be stretched horizontally to fill the container, vertically to fill the container, or in both directions. |

# settooltip()

## Description

Sets the text for the tooltip associated with the control. The wxformbitmap object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapvar*.settooltip ( string *tooltip* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

# setvisible()

## Description

Sets the visibility of the control. The wxformbitmap object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapvar*.setvisible ( boolean *visible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the control. |

# wxformbitmapbutton

## Description

A wxformbitmapbutton object is a push button on a form that the user clicks to perform some action.

## Type Tags

wxformcontrol

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the background of the button control. |
| bitmap | wxbitmap | The wxbitmap object that is used as the source of the bitmap on the button. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| disabledbitmap | wxbitmap | The wxbitmap object that is used as the source of the bitmap on the button that is displayed when the button is disabled. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformbutton objects this is always `.true` |
| focusbitmap | wxbitmap | The wxbitmap object that is used as the source of the bitmap on the button when the button has focus. |
| form | wxform | Specifies the wxform object to which this control belongs. |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td>height</td><td>integer</td><td>Gives the height of the control, in pixels.</td></tr>
<tr><td>left</td><td>integer</td><td>Gives the position of the left side of the control relative to the left side of the form, in pixels.</td></tr>
<tr><td>name</td><td>string</td><td>The name of the wxformbitmapbutton object.</td></tr>
<tr><td>next</td><td>type(wxformcontrol)</td><td>Specifies the next <code>wxformcontrol</code> on the same form.</td></tr>
<tr><td>onclick</td><td>event</td><td>An event that is triggered every time the user clicks the button. The event handling function will receive the following parameters: (wxformbitmapbutton me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>ongotfocus</td><td>event</td><td>An event that is triggered when the button receives input focus. The event handling function will receive the following parameters: (wxformbitmapbutton me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onlostfocus</td><td>event</td><td>An event that is triggered when the button loses input focus. The event handling function will receive the following parameters: (wxformbitmapbutton me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onmouse</td><td>event</td><td>An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: <code>function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<ul><li>etype is the bit (defined as for onmousemask) showing what sort of event it is.</li><li>keys is a collection of bits showing key positions:<table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr><tr><td>alt down</td><td>0x00040000</td></tr><tr><td>meta down</td><td>0x00080000</td></tr></table></li><li>x and y are the position of the mouse event</li><li>reference is the (optional) reference object in the wxformbitmapbutton.onmouse.reference</li></ul></td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
</table>

| Property | Type | Description |
|---|---|---|
| selectedbitmap | wxbitmap | The wxbitmap object that is used as the source of the bitmap on the button that is displayed when the button is pressed. |
| tooltip | string | Contains the test that is displayed as a tooltip for the control. |
| top | integer | Gives the position of the top edge of the control relative to the top edge of the form, in pixels. |
| type | type | Specifies the wxformbitmapbutton type object. |
| visible | boolean | Specifies whether or not the control is visible. |
| width | integer | Gives the width of the control, in pixels. |

# Methods

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*wxformbitmapbuttonvar*.remove ()

### Parameters

None

## setbackgroundrgb()

### Description

Sets the background color of the button control. The wxformbitmapbutton object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*wxformbitmapbuttonvar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the back- | integer | The new background color of the button control. It is inadvisable to specify any value for this ar- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | `groundrgb` property | | gument that is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the button control. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the button control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the button control. This must be between 0 and 255 inclusive. |

# setbitmaps()

## Description

Sets the various bitmaps associated with the control. It is not possible to set a bitmap to `.nul` once it has been assigned. It is possible to change its assignment though. Any of the parameters can be provided to change one or more bitmaps at the same time. The wxformbitmapbutton object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformbitmapbuttonvar`*`.setbitmaps` ( wxbitmap *`bitmap`*, wxbitmap *`selectedbitmap`*, wxbitmap *`disabledbitmap`*, wxbitmap *`focusbitmap`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bitmap | The current value of the `bitmap` property | wxbitmap | The wxbitmap object to be assigned as the source of the bitmap on the button. |
| selectedbitmap | The current value of the `selectedbitmap` property | wxbitmap | The wxbitmap object to be assigned as the source of the bitmap on the button shown when the button is selected. |
| disabledbitmap | The current value of the `disabledbitmap` property | wxbitmap | The wxbitmap object to be assigned as the source of the bitmap on the button shown when the button is disabled. |
| focusbitmap | The current value of the `focusbitmap` property | wxbitmap | The wxbitmap object to be assigned as the source of the bitmap on the button shown when the button has focus. |

# setcaptureable()

## Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

## Prototype

*wxformbitmapbuttonvar*.setcaptureable ( boolean *captureable* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the controlcapture() method of the form. |

# setenabled()

## Description

Sets the enabled state of the button. The wxformbitmapbutton object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapbuttonvar*.setenabled ( boolean *enabled* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the button control. |

# setfocus()

## Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire.

## Prototype

*wxformbitmapbuttonvar*.setfocus ()

## Parameters

None

# setname()

## Description

Sets the name of the button. The wxformbitmapbutton object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapbuttonvar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the button control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

## Prototype

*wxformbitmapbuttonvar*.setnext ( type(wxformcontrol) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

# setonmousemask()

## Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformbitmapbutton object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbitmapbuttonvar*.setonmousemask ( integer *onmousemask* )

### Parameters

<table>
<tr><th>Parameter</th><th>Default value</th><th>Type name</th><th>Description</th></tr>
<tr><td>onmousemask</td><td>None</td><td>integer</td><td>The definition of which events should be handled by the on-mouse event handler. The onmousemask is made up of bits:<br><table><tr><th>Event Type</th><th>Bit Value</th></tr><tr><td>left button down</td><td>0x00000001</td></tr><tr><td>left button up</td><td>0x00000002</td></tr><tr><td>left button double click</td><td>0x00000004</td></tr><tr><td>middle button down</td><td>0x00000010</td></tr><tr><td>middle button up</td><td>0x00000020</td></tr><tr><td>middle button dbl click</td><td>0x00000040</td></tr><tr><td>right button down</td><td>0x00000100</td></tr><tr><td>right button up</td><td>0x00000200</td></tr><tr><td>right button dbl click</td><td>0x00000400</td></tr><tr><td>motion</td><td>0x00001000</td></tr></table></td></tr>
</table>

## setposition()

### Description

Sets the size and/or position of the button. The wxformbitmapbutton object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformbitmapbuttonvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the `left` property | integer | The new position of the left side of the button on the form. |
| top | The current value of the `top` property | integer | The new position of the top edge of the button on the form. |
| width | The current value of the `width` property | integer | The new width of the button on the form. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| height | The current value of the `height` property | integer | The new height of the button on the form. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformbitmapbutton object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformbitmapbuttonvar`*`.settooltip` ( string *`tooltip`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the button. The wxformbitmapbutton object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformbitmapbuttonvar`*`.setvisible` ( boolean *`visible`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the button control. |

# wxformbutton

# Description

A wxformbutton object is a push button on a form which the user clicks to perform some action.

# Type Tags

wxformcontrol

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the background of the button control. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| default | boolean | Specifies whether or not the control is the default button on the form. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformbutton objects this is always `.true` |
| font | wxfont | This is a reference to the wxfont object that describes how the text on the button is formatted. |
| form | wxform | Specifies the wxform object to which this control belongs. |
| height | integer | Gives the height of the control, in pixels. |
| left | integer | Gives the position of the left side of the control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformbutton object. |
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |
| onclick | event | An event that is triggered every time the user clicks the button. The event handling function will receive the following parameters: (wxformbutton me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| ongotfocus | event | An event that is triggered when the button receives input focus. The event handling function will receive the following parameters: (wxformbutton me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| onlostfocus | event | An event that is triggered when the button loses input focus. The event handling function will receive |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td></td><td></td><td>the following parameters: (wxformbutton me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onmouse</td><td>event</td><td>An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: <code>function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<ul><li>etype is the bit (defined as for onmousemask) showing what sort of event it is.</li><li>keys is a collection of bits showing key positions:<table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr><tr><td>alt down</td><td>0x00040000</td></tr><tr><td>meta down</td><td>0x00080000</td></tr></table></li><li>x and y are the position of the mouse event</li><li>reference is the (optional) reference object in the wxformbutton.onmouse.reference</li></ul></td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>text</td><td>string</td><td>The current text for the wxformbutton object, which is shown in the center of the button on the form.</td></tr>
<tr><td>textrgb</td><td>rgb</td><td>Gives the color of the text on the button.</td></tr>
<tr><td>tooltip</td><td>string</td><td>Contains the test that is displayed as a tooltip for the control.</td></tr>
<tr><td>top</td><td>integer</td><td>Gives the position of the top edge of the control relative to the top edge of the form, in pixels.</td></tr>
<tr><td>type</td><td>type</td><td>Specifies the wxformbutton type object.</td></tr>
<tr><td>visible</td><td>boolean</td><td>Specifies whether or not the control is visible.</td></tr>
<tr><td>width</td><td>integer</td><td>Gives the width of the control, in pixels.</td></tr>
</table>

# Methods

## makedefault()

### Description

Makes the button the default button for the form. This also takes the default property away from any other button on the form that has it.

### Prototype

*wxformbuttonvar*.makedefault ()

### Parameters

None

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*wxformbuttonvar*.remove ()

### Parameters

None

## setbackgroundrgb()

### Description

Sets the background color of the button control. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*wxformbuttonvar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `backgroundrgb` property | integer | The new background color of the button control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the button control. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the button control. This must be between 0 and 255 inclusive. |
| blue | The current value of the | integer | The blue component in the new background color of the button |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | `backgroundrgb.blue` property | | control. This must be between 0 and 255 inclusive. |

# setcaptureable()

## Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

## Prototype

*`wxformbuttonvar`*`.setcaptureable` ( boolean *`captureable`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the `controlcapture()` method of the form. |

# setenabled()

## Description

Sets the enabled state of the button. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformbuttonvar`*`.setenabled` ( boolean *`enabled`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the button control. |

# setfocus()

## Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire.

## Prototype

*`wxformbuttonvar`*`.setfocus` ()

## Parameters

None

# setfont()

## Description

Sets the font to be used for the text in the button. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbuttonvar*.setfont ( wxfont *font* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The wxfont object that will be used to format the text. |

# setname()

## Description

Sets the name of the button. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbuttonvar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the button control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

## Prototype

*wxformbuttonvar*.setnext ( type(wxformcontrol) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately be- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | low the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

# setonmousemask()

## Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbuttonvar*.setonmousemask ( integer *onmousemask* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onmousemask | None | integer | The definition of which events should be handled by the onmouse event handler. The *onmousemask* is made up of bits: |

| Event Type | Bit Value |
|---|---|
| left button down | 0x00000001 |
| left button up | 0x00000002 |
| left button double click | 0x00000004 |
| middle button down | 0x00000010 |
| middle button up | 0x00000020 |
| middle button dbl click | 0x00000040 |
| right button down | 0x00000100 |
| right button up | 0x00000200 |
| right button dbl click | 0x00000400 |
| motion | 0x00001000 |

# setposition()

## Description

Sets the size and/or position of the button. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformbuttonvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the `left` property | integer | The new position of the left side of the button on the form. |
| top | The current value of the `top` property | integer | The new position of the top edge of the button on the form. |
| width | The current value of the `width` property | integer | The new width of the button on the form. |
| height | The current value of the `height` property | integer | The new height of the button on the form. |

## settext()

### Description

Sets the text in the button. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformbuttonvar`*.`settext` ( string *`text`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | The new text to set in the button. |

## settextrgb()

### Description

Sets the color of the text in the control. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*`wxformbuttonvar`*.`settextrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `textrgb` property | integer | The new color of the text in the control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| red | The current value of the `textrgb.red` property | integer | The red component in the new text color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the `textrgb.green` property | integer | The green component in the new text color of the control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `text.blue` property | integer | The blue component in the new text color of the control. This must be between 0 and 255 inclusive. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformbuttonvar`*`.settooltip` ( string *`tooltip`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the button. The wxformbutton object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformbuttonvar`*`.setvisible` ( boolean *`visible`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the button control. |

# wxformcheckbox

# Description

A wxformcheckbox object is a control that indicates whether it is 'checked' or not, normally by showing a square that is either empty or has a tick mark in it. The control also contains a piece of text that labels it.

# Type Tags

wxformcontrol

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the background of the checkbox control. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformcheckbox objects this is always `.true` |
| font | wxfont | This is a reference to the wxfont object that describes how the text in the option control is formatted. |
| form | wxform | Specifies the wxform object to which this checkbox control belongs. |
| height | integer | Gives the height of the checkbox control, in pixels. |
| left | integer | Gives the position of the left side of the checkbox control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformcheckbox object. |
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |
| onchange | event | An event which is triggered every time the user checks or unchecks the control. The event handling function will receive the following parameters: (wxformcheckbox me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| ongotfocus | event | An event that is triggered when the checkbox control receives input focus. The event handling function will receive the following parameters: (wxformcheckbox me[, type(*) reference]). The reference is only passed if it is provided by the user. |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td>onlostfocus</td><td>event</td><td>An event that is triggered when the checkbox control loses input focus. The event handling function will receive the following parameters: (wxformcheckbox me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onmouse</td><td>event</td><td>An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: <code>function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<ul><li>etype is the bit (defined as for onmousemask) showing what sort of event it is.</li><li>keys is a collection of bits showing key positions:<table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr><tr><td>alt down</td><td>0x00040000</td></tr><tr><td>meta down</td><td>0x00080000</td></tr></table></li><li>x and y are the position of the mouse event</li><li>reference is the (optional) reference object in the wxformcheckbox.onmouse.reference</li></ul></td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>state</td><td>string</td><td>A string which indicates whether or not the checkbox control is currently in a 'checked' state or not. If the control is not checked then the value of <code>state</code> is "". If the control is checked then the value of <code>state</code> is "on".</td></tr>
<tr><td>text</td><td>string</td><td>The current text for the wxformcheckbox object, which is displayed on the form.</td></tr>
<tr><td>textrgb</td><td>rgb</td><td>Gives the color of the text in the checkbox control.</td></tr>
<tr><td>tooltip</td><td>string</td><td>Contains the test that is displayed as a tooltip for the control.</td></tr>
<tr><td>top</td><td>integer</td><td>Gives the position of the top edge of the checkbox control relative to the top edge of the form, in pixels.</td></tr>
<tr><td>type</td><td>type</td><td>Specifies the wxformcheckbox type object.</td></tr>
<tr><td>visible</td><td>boolean</td><td>Specifies whether or not the control is visible.</td></tr>
<tr><td>width</td><td>integer</td><td>Gives the width of the checkbox control, in pixels.</td></tr>
</table>

# Methods

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*wxformcheckboxvar*.remove ()

### Parameters

None

## setbackgroundrgb()

### Description

Sets the background color of the checkbox control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*wxformcheckboxvar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `backgroundrgb` property | integer | The new background color of the checkbox control. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the checkbox control. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the checkbox control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the checkbox control. This must be between 0 and 255 inclusive. |

# setcaptureable()

## Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

## Prototype

*wxformcheckboxvar*.setcaptureable ( boolean *captureable* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the controlcapture() method of the form. |

# setenabled()

## Description

Sets the enabled state of the checkbox control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformcheckboxvar*.setenabled ( boolean *enabled* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the checkbox control. |

# setfocus()

## Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire.

## Prototype

*wxformcheckboxvar*.setfocus ()

## Parameters

None

# setfont()

## Description

Sets the font to be used for the text in the checkbox control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformcheckboxvar*.setfont ( wxfont *font* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The wxfont object that will be used to format the text in the checkbox control. |

# setname()

## Description

Sets the name of the checkbox control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformcheckboxvar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the checkbox control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

## Prototype

*wxformcheckboxvar*.setnext ( type(wxformcontrol) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

# setonmousemask()

## Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformcheckboxvar*.setonmousemask ( integer *onmousemask* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onmousemask | None | integer | The definition of which events should be handled by the onmouse event handler. The *onmousemask* is made up of bits: |

| Event Type | Bit Value |
|---|---|
| left button down | 0x00000001 |
| left button up | 0x00000002 |
| left button double click | 0x00000004 |
| middle button down | 0x00000010 |
| middle button up | 0x00000020 |
| middle button dbl click | 0x00000040 |
| right button down | 0x00000100 |
| right button up | 0x00000200 |
| right button dbl click | 0x00000400 |
| motion | 0x00001000 |

# setposition()

## Description

Sets the size and/or position of the checkbox control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformcheckboxvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the checkbox control on the form. |
| top | The current value of the top property | integer | The new position of the top edge of the checkbox control on the form. |
| width | The current value of the width property | integer | The new width of the checkbox control on the form. |
| height | The current value of the height property | integer | The new height of the checkbox control on the form. |

## setstate()

### Description

Sets the state of the checkbox control, i.e. whether or not it has the appearance of being 'checked'. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformcheckboxvar*.setstate ( string *state* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| state | None | string | The new state for the control. If the state is "" then the control is not checked. If the state is "on" then the control is checked. |

## settext()

### Description

Sets the text in the label part of the control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformcheckboxvar*.settext ( string *text* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | The new text to set in the control. |

## settextrgb()

### Description

Sets the color of the text in the label part of the control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*`wxformcheckboxvar`*.`settextrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `textrgb` property | integer | The new color of the text in the control. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| red | The current value of the `textrgb.red` property | integer | The red component in the new text color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the `textrgb.green` property | integer | The green component in the new text color of the control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `text.blue` property | integer | The blue component in the new text color of the control. This must be between 0 and 255 inclusive. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformcheckboxvar`*.`settooltip` ( string *`tooltip`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the checkbox control. The wxformcheckbox object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformcheckboxvar*.setvisible ( boolean *visible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the checkbox control. |

# wxformcombo

## Description

A wxformcombo object is a text based control on a form with a separate list of items, one of which can be selected.

## Type Tags

wxformcontrol

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the background of the combo control. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| edittype | string | A string that indicates the way in which the user can change the text in the combo. `edittype` contains one of the following values: |

| Property | Type | Description |
|---|---|---|
| | | • "droplist" — The user can choose one of the items in the list. The text of that item then becomes the text for the combo control.<br>• "dropedit" — The user can choose one of the items in the list, the text of which then becomes the text for the combo control, or the text can be directly changed as in an edit control. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformcombo objects this is always `.true` |
| font | wxfont | This is a reference to the wxfont object that describes how the text in the list control is formatted. |
| form | wxform | Specifies the wxform object to which this combo control belongs. |
| height | integer | Gives the height of the combo control, in pixels. |
| itemcount | integer | Gives the number of items in the combo control. |
| left | integer | Gives the position of the left side of the combo control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformcombo object. |
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |
| ongotfocus | event | An event that is triggered when the combo control receives input focus. The event handling function will receive the following parameters: (wxformcombo me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| onlostfocus | event | An event that is triggered when the combo control loses input focus. The event handling function will receive the following parameters: (wxformcombo me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| onmouse | event | An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: `function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)` where:<br>• etype is the bit (defined as for onmousemask) showing what sort of event it is.<br>• keys is a collection of bits showing key positions: |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><th>Key Type</th><th>Bit Value</th></tr>
<tr><td>shift down</td><td>0x00010000</td></tr>
<tr><td>ctrl down</td><td>0x00020000</td></tr>
<tr><td>alt down</td><td>0x00040000</td></tr>
<tr><td>meta down</td><td>0x00080000</td></tr>
</table>
<ul>
<li>x and y are the position of the mouse event</li>
<li>reference is the (optional) reference object in the wxformcombo.onmouse.reference</li>
</ul>
</td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>onselection-change</td><td>event</td><td>An event which is triggered when what is selected in the combo control changes. The event handling function will receive the following parameters: (wx-formcombo me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>text</td><td>string</td><td>The current text for the wxformcombo object, which is displayed on the form. During editing and also during any <code>onlostfocus</code> event for this combo control the <code>text</code> property is still set to the original text of the control, before editing started. The <code>gettext</code> method can be used to retrieve the new text which the user has entered into the control.</td></tr>
<tr><td>textrgb</td><td>rgb</td><td>Gives the color of the text in the combo control.</td></tr>
<tr><td>tooltip</td><td>string</td><td>Contains the test that is displayed as a tooltip for the control.</td></tr>
<tr><td>top</td><td>integer</td><td>Gives the position of the top edge of the combo control relative to the top edge of the form, in pixels.</td></tr>
<tr><td>type</td><td>type</td><td>Specifies the wxformcombo type object.</td></tr>
<tr><td>visible</td><td>boolean</td><td>Specifies whether or not the control is visible.</td></tr>
<tr><td>width</td><td>integer</td><td>Gives the width of the combo control, in pixels.</td></tr>
</table>

# Methods

## delete()

### Description

Deletes (removes) items from a combo control. The wxformcombo object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformcombovar*.`delete` ( integer , … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | None | integer | The 1-based index into the list of an item to delete. |
| … | None | | |

## deleteall()

### Description

Deletes (removes) all items from a combo control. The wxformcombo object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformcombovar*.deleteall()

### Parameters

None

## getselected()

### Description

Searches a combo control's list for the next selected item after a specified start point. The return value is the 1-based index of the first selected item which is equal to or greater than the specified start point, or is `.nul` if there is no such selected item.

### Prototype

*wxformcombovar*.getselected ( integer *firstitem* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| firstitem | None | integer | The 1-based index of the first item to search for selection. |

## gettext()

### Description

Gets the most recent text for the edit portion of the control. During editing and also during any `onlost-focus` event for this combo control the `gettext` method will retrieve the current text visible in the edit portion of the control. The `text` property can be used to access the original text that was in place before editing began. This primarily is of use when using the `"dropedit"` form of the control.

### Prototype

*wxformcombovar*.gettext ()

### Parameters

None

## insert()

### Description

Adds items to a combo control's list. The wxformcombo object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformcombovar*.insert ( integer *firstitem*, string , … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| firstitem | The current value of the `itemcount` property plus one, i.e. the end of the list. | integer | The 1-based index into the list to show the position to place the first of the items to be inserted. |
| | None | string | The string to be added as an item at position `firstitem`. |
| … | None | | |

## isselected()

### Description

Determines whether or not a specified item in a combo control is currently selected. The method returns `.true` if the item is selected, or `.false` otherwise.

### Prototype

*wxformcombovar*.isselected ( integer *item* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| item | None | integer | The 1-based index into the list of an item to determine the selection state of. |

# remove()

## Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

## Prototype

*wxformcombovar*.remove ()

## Parameters

None

# select()

## Description

Selects an item in a combo control. The wxformcombo object itself is returned, to allow multiple methods to be put into one expression.

## Prototype

*wxformcombovar*.select ( integer *item*, boolean *select* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| item | None | integer | The 1-based index of the item to select. |
| select | .true | boolean | .true is the only valid value for this argument. |

# setbackgroundrgb()

## Description

Sets the background color of the combo control. The wxformcombo object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the rgb argument and one or more of the red, green or blue arguments.

## Prototype

*wxformcombovar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the back- | integer | The new background color of the combo control. It is inadvisable to specify any value for this argu- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | `groundrgb` property | | ment which is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the combo control. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the combo control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the combo control. This must be between 0 and 255 inclusive. |

# setcaptureable()

## Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

## Prototype

*`wxformcombovar`*`.setcaptureable` ( boolean *`captureable`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the `controlcapture()` method of the form. |

# setenabled()

## Description

Sets the enabled state of the combo control. The wxformcombo object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformcombovar`*`.setenabled` ( boolean *`enabled`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the combo control. |

# setfocus()

## Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire.

## Prototype

*wxformcombovar*.setfocus ()

## Parameters

None

# setfont()

## Description

Sets the font to be used for the text in the combo control. The wxformcombo object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformcombovar*.setfont ( wxfont *font* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The wxfont object that will be used to format the text in the combo control. |

# setname()

## Description

Sets the name of the list control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformcombovar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the list control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

## Prototype

*wxformcombovar*.setnext ( type(wxformcontrol) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

# setonmousemask()

## Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformcombo object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformcombovar*.setonmousemask ( integer *onmousemask* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onmousemask | None | integer | The definition of which events should be handled by the onmouse event handler. The *onmousemask* is made up of bits:<br>**Event Type** / **Bit Value**<br>left button down / 0x00000001<br>left button up / 0x00000002<br>left button double click / 0x00000004<br>middle button down / 0x00000010 |

| Parameter | Default value | Type name | Description | |
|---|---|---|---|---|
| | | | **Event Type** | **Bit Value** |
| | | | middle button up | 0x00000020 |
| | | | middle button dbl click | 0x00000040 |
| | | | right button down | 0x00000100 |
| | | | right button up | 0x00000200 |
| | | | right button dbl click | 0x00000400 |
| | | | motion | 0x00001000 |

# setposition()

## Description

Sets the size and/or position of the combo control. The wxformcombo object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformcombovar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the `left` property | integer | The new position of the left side of the combo control on the form. |
| top | The current value of the `top` property | integer | The new position of the top edge of the combo control on the form. |
| width | The current value of the `width` property | integer | The new width of the combo control on the form. |
| height | The current value of the `height` property | integer | The new height of the combo control on the form. |

# settext()

## Description

Sets the text in the control. The wxformcombo object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformcombovar*.settext ( string *text* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | The new text to set in the control. |

## settextrgb()

### Description

Sets the color of the text in the combo control. The wxformcombo object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*`wxformcombovar`*`.settextrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `textrgb` property | integer | The new color of the text in the control. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| red | The current value of the `textrgb.red` property | integer | The red component in the new text color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the `textrgb.green` property | integer | The green component in the new text color of the control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `text.blue` property | integer | The blue component in the new text color of the control. This must be between 0 and 255 inclusive. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformcombo object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformcombovar`*`.settooltip` ( string *`tooltip`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the combo control. The wxformcombo object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformcombovar*.setvisible ( boolean *visible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the combo control. |

# wxformcombo[]

## Get

### Subscripts

A numeric value giving the 1-based index of an item in the list.

### Description

Retrieves the text of the specified item.

## Set

### Subscripts

A numeric value giving the 1-based index of an item in the list.

### Description

Sets the text of the specified item.

## Set Reference

Attempting to set a reference to an object is not supported.

# wxformedittext

# Description

A wxformedittext object is an area of a form where a piece of text is displayed, and the user can enter or modify that text.

# Type Tags

wxformcontrol

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| alignment | string | The alignment of the content of the control. This will either be `"left"`, `"right"`, or `""` if the content is centered. This must be set at the time the control is created. |
| backgroundrgb | rgb | Gives the color of the background of the edit control. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| editstyle | string | The edit style of the control. This will be any combination of: `""`, `"hscroll"`, `"multiline"`, `"readonly"`, and `"password"`. For details on the valid and/or usable combinations, see the section about the `addcontrol()` method of the wxform type in: the section called “wxform”. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformedittext objects this is always `.true`. |
| font | wxfont | This is a reference to the wxfont object that describes how the text in the control is formatted. |
| form | wxform | Specifies the wxform object to which this edit control belongs. |
| height | integer | Gives the height of the edit control, in pixels. |
| left | integer | Gives the position of the left side of the edit control relative to the left side of the form, in pixels. |
| maxlength | integer | Specifies the maximum length of the text allowed in the control. |
| name | string | The name of the wxformedittext object. |
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |
| onchange | event | An event that is triggered when the content of the edit control changes. The event handling function |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td></td><td></td><td>will receive the following parameters: (wxformedit-text me[, type(*) reference]). The reference is only passed if it is provided by the user. To get the current content of the control, use the <code>gettext()</code> method. The value of the text property is not modified until after the <code>onlostfocus</code> event has returned.</td></tr>
<tr><td>ongotfocus</td><td>event</td><td>An event that is triggered when the edit control receives input focus. The event handling function will receive the following parameters: (wxformedit-text me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onlostfocus</td><td>event</td><td>An event that is triggered when the edit control loses input focus. The event handling function will receive the following parameters: (wxformedit-text me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onmouse</td><td>event</td><td>An event that is triggered each time a mouse event occurs that matches the bits set in the onmouse-mask property. The onmouse event handling function should be defined as follows: <code>function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<ul><li>etype is the bit (defined as for onmousemask) showing what sort of event it is.</li><li>keys is a collection of bits showing key positions:<table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr><tr><td>alt down</td><td>0x00040000</td></tr><tr><td>meta down</td><td>0x00080000</td></tr></table></li><li>x and y are the position of the mouse event</li><li>reference is the (optional) reference object in the wxformtext.onmouse.reference</li></ul></td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>text</td><td>string</td><td>The current text for the wxformedittext object, which is displayed on the form. During editing and also during any <code>onlostfocus</code> event for this edit control the <code>text</code> property is still set to the original text of the control, before editing started. The <code>get-</code></td></tr>
</table>

| Property | Type | Description |
|---|---|---|
| | | `text` method can be used to retrieve the new text which the user has entered into the control. |
| textrgb | rgb | Gives the color of the text in the control. |
| tooltip | string | Contains the test that is displayed as a tooltip for the control. |
| top | integer | Gives the position of the top edge of the edit control relative to the top edge of the form, in pixels. |
| type | type | Specifies the wxformedittext type object. |
| visible | boolean | Specifies whether or not the control is visible. |
| width | integer | Gives the width of the edit control, in pixels. |

# Methods

## getinsertionpoint()

### Description

Gets the current position within the text contained by the control where the cursor is located.

### Prototype

*`wxformedittextvar`*`.getinsertionpoint` ( integer *`insertionpoint`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| insertionpoint | None | integer | This parameter must be a pre-initialized integer variable. It will be assigned the current position within the text in the edit control where the cursor is located. |

## getselection()

### Description

Returns the starting and ending positions of the selected text in the control.

### Prototype

*`wxformedittextvar`*`.getselection` ( integer *`start`*, integer *`end`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| start | None | integer | This must be a pre-initialized integer variable. It will contain the starting position of the selected text in the edit control. If no text |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | is selected, it will return the value `.nul`. |
| end | None | integer | This must be a pre-initialized integer variable. It will contain the ending position of the selected text in the edit control. If no text is selected, it will return the value `.nul`. |

# gettext()

## Description

Gets the most recent text for the control. During editing and also during any `onlostfocus` event for this edit control the `gettext` method will retrieve the current text visible in the control. The `text` property can be used to access the original text which was in place before editing began.

## Prototype

*`wxformedittextvar`*.`gettext` ()

## Parameters

None

# remove()

## Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

## Prototype

*`wxformedittextvar`*.`remove` ()

## Parameters

None

# setbackgroundrgb()

## Description

Sets the background color of the edit control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

## Prototype

*`wxformedittextvar`*.`setbackgroundrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `backgroundrgb` property | integer | The new background color of the edit control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the edit control. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the edit control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the edit control. This must be between 0 and 255 inclusive. |

## setcaptureable()

### Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

### Prototype

*`wxformedittextvar`*`.setcaptureable` ( boolean *`captureable`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the `controlcapture()` method of the form. |

## setenabled()

### Description

Sets the enabled state of the edit control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformedittextvar`*`.setenabled` ( boolean *`enabled`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the edit control. |

## setfocus()

### Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformedittextvar*.setfocus ()

### Parameters

None

## setfont()

### Description

Sets the font to be used for the text in the edit control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformedittextvar*.setfont ( wxfont *font* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The wxfont object that will be used to format the text. |

## setinsertionpoint()

### Description

Sets the position within the text contained by the control where the cursor should be located. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformedittextvar*.setinsertionpoint ( integer *insertionpoint* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| insertionpoint | None | integer | This is the position within the text in the edit control where the cursor is to be placed. To place the cursor at the end, enter a value that is equal to or greater than the length of the content, or use the value `.inf`. To place the cursor at the beginning, use the value `0`. |

## setmaxlength()

### Description

Sets the maximum length of the text that will be accepted by the control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformedittextvar`*`.setmaxlength` ( integer *`maxlength`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| maxlength | None | integer | This is the maximum length of text that the edit control should allow the user to enter. To allow any amount, use the value `.inf`. Setting this value does not affect any text that may already be in the control. |

## setname()

### Description

Sets the name of the edit control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformedittextvar`*`.setname` ( string *`name`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the edit control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

## Prototype

*wxformedittextvar*.setnext ( type(wxformcontrol) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

# setonmousemask()

## Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformedittextvar*.setonmousemask ( integer *onmousemask* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onmousemask | None | integer | The definition of which events should be handled by the onmouse event handler. The *onmousemask* is made up of bits:<br>**Event Type** / **Bit Value**<br>left button down / 0x00000001<br>left button up / 0x00000002<br>left button double click / 0x00000004<br>middle button down / 0x00000010 |

| Parameter | Default value | Type name | Description | |
|---|---|---|---|---|
| | | | **Event Type** | **Bit Value** |
| | | | middle button up | 0x00000020 |
| | | | middle button dbl click | 0x00000040 |
| | | | right button down | 0x00000100 |
| | | | right button up | 0x00000200 |
| | | | right button dbl click | 0x00000400 |
| | | | motion | 0x00001000 |

# setposition()

## Description

Sets the size and/or position of the edit control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformedittextvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the `left` property | integer | The new position of the left side of the edit control on the form. |
| top | The current value of the `top` property | integer | The new position of the top edge of the edit control on the form. |
| width | The current value of the `width` property | integer | The new width of the edit control on the form. |
| height | The current value of the `height` property | integer | The new height of the edit control on the form. |

# setselection()

## Description

Selects the text contained by the control within the range specified. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformedittextvar*.setselection ( integer *start*, integer *end* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| start | None | integer | This is the starting position within the text in the edit control for the text that will be selected. To start at the beginning of the text, enter the value `0`. |
| end | None | integer | This is the ending position within the text in the edit control for the text that will be selected. To mark until the end of the text, enter either a value greater than or equal to the length of the text, or the value `.inf`. |

## settext()

### Description

Sets the text in the edit control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformedittextvar`*`.settext` ( string *`text`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | The new text to set in the edit control. |

## settextrgb()

### Description

Sets the color of the text in the edit control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*`wxformedittextvar`*`.settextrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `textrgb` property | integer | The new color of the text in the edit control. It is inadvisable to specify any value for this argu- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ment that is not the value of an existing rgb object. |
| red | The current value of the `textrgb.red` property | integer | The red component in the new text color of the edit control. This must be between 0 and 255 inclusive. |
| green | The current value of the `textrgb.green` property | integer | The green component in the new text color of the edit control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `text.blue` property | integer | The blue component in the new text color of the edit control. This must be between 0 and 255 inclusive. |

# settooltip()

## Description

Sets the text for the tooltip associated with the control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformedittextvar`*`.settooltip` ( string *`tooltip`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

# setvisible()

## Description

Sets the visibility of the edit control. The wxformedittext object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformedittextvar`*`.setvisible` ( boolean *`visible`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the edit control. |

# wxformgauge

## Description

A wxformgauge object is a free-standing gauge control horizontally positioned on the form. It can be either a determinate or indeterminate gauge.

## Type Tags

wxformcontrol

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the control. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformgauge objects this is always `.false` |
| form | wxform | Specifies the wxform object to which this control belongs. |
| height | integer | Gives the height of the control, in pixels. |
| left | integer | Gives the position of the left side of the control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformgauge object. |
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |
| position | integer | This contains the current position in the gauge. Valid values are from 0 to range - 1. |
| pulse | boolean | |
| range | integer | This contains the range of values, beginning at 0 and ending at range - 1, that describe the granularity of the gauge. Valid values are from 1 through 4G - 1. |
| tooltip | string | Contains the text that is displayed as a tooltip for the control. |

| Property | Type | Description |
|---|---|---|
| top | integer | Gives the position of the top edge of the control relative to the top edge of the form, in pixels. |
| type | type | Specifies the wxformgauge type object. |
| visible | boolean | Specifies whether or not the control is visible. |
| width | integer | Gives the width of the control, in pixels. |

# Methods

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*wxformgaugevar*.remove ()

### Parameters

None

## setbackgroundrgb()

### Description

Sets the background color of the gauge control. The wxformgauge object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green`, or `blue` arguments.

### Prototype

*wxformgaugevar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the rgb property | integer | The new color of the control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the rgb.red property | integer | The red component in the new color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the | integer | The green component in the new color of the control. This must be between 0 and 255 inclusive. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | rgb.green property | | |
| blue | The current value of the rgb.blue property | integer | The blue component in the new color of the control. This must be between 0 and 255 inclusive. |

# setgauge()

## Description

Sets the type of gauge (determinate or indeterminate). If the *range* and *position* parameters are sent, the gauge is set to determinate mode. if the *pulse* is passed with a value of `.true`, then the gauge will switch to indeterminate mode and will pulse one unit. Calling this method again with a *position* parameter will return it to determinate mode. The wxformgauge object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformgaugevar*.`setgauge` ( integer *range*, integer *position*, boolean *pulse* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| range | The current value of the range | integer | The range for the gauge. Must be between `0` and `4G - 1`. |
| position | The current value of the position | integer | The current position of the gauge. Must be between `0` and the value of the range. |
| pulse | The current value of the pulse | boolean | If set to his sets the gauge to indeterminate or pulse mode. It is recommended to set the range to some useful value such as `10` and the position to `0` before calling this method with the *pulse* parameter set to `.true`. Then each time the `setgauge()` method is called with the *pulse* parameter set to `.true` the gauge will pulse one unit. |

# setname()

## Description

Sets the name of the gauge control. The wxformgauge object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformgaugevar*.`setname` ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the control. |

## setnext()

### Description

Sets the position of the control in the z-order and tab order.

### Prototype

*wxformgaugevar*.setnext ( type(wxformcontrol) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

## setposition()

### Description

Sets the size and/or position of the control. The wxformgauge object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformgaugevar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the control on the form. |
| top | The current value of the top property | integer | The new position of the top edge of the control on the form. |
| width | The current value of the width property | integer | The new width of the control on the form. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| height | The current value of the height property | integer | The new height of the control on the form. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformscrollbar object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformgaugevar*.settooltip ( string *tooltip* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the control. The wxformgauge object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformgaugevar*.setvisible ( boolean *visible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the control. |

# wxformgrid

# Description

A wxformgrid object is an area of a form where a table with rows and columns, row labels and column labels are displayed. The grid can be larger than the visible area.

# Type Tags

wxformcontrol

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| alignment | string | Specifies the default alignment for the cells. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| colcount | integer | Specifies the number of columns in the grid. |
| collabelalignment | string | Specifies the alignment for the column labels. |
| collabelheight | integer | Specifies the height of the column label row in pixels. |
| colwidthdraggable | boolean | Specifies whether the column width can be changed by the user. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformedittext objects this is always `.true`. |
| font | wxfont | This the default font for the cells. |
| form | wxform | Specifies the wxform object to which this edit control belongs. |
| height | integer | Gives the height of the grid control, in pixels. |
| labelfont | wxfont | This the font used for the labels. |
| left | integer | Gives the position of the left side of the grid control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformgrid object. |
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |
| oncellchange | event | An event that is triggered when the user changes the content of a cell. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer col[, type(*) reference]) the reference is only passed if it is provided by the user. |
| oncellleftclick | event | An event that is triggered when the user clicks with the left mouse button on a cell. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer col[, type(*) ref- |

| Property | Type | Description |
|---|---|---|
| | | erence]) the reference is only passed if it is provided by the user. |
| oncellleftd-blclick | event | An event that is triggered when the user double clicks with the left mouse button on a cell. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer col[, type(*) reference]) the reference is only passed if it is provided by the user. |
| oncell-rightclick | event | An event that is triggered when the user clicks with the right mouse button on a cell. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer col[, type(*) reference]) the reference is only passed if it is provided by the user. |
| oncellrightd-blclick | event | An event that is triggered when the user double clicks with the right mouse button on a cell. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer col[, type(*) reference]) the reference is only passed if it is provided by the user. |
| oncellselect | event | An event that is triggered when the user selects a cell. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer col[, type(*) reference]) the reference is only passed if it is provided by the user. |
| oncol-widthchange | event | An event that is triggered when the user resizes a column. The event handling function will receive the following parameters: (wxformgrid me, integer col[, type(*) reference]). The reference is only passed if it is provided by the user. |
| ongotfocus | event | An event that is triggered when the grid control receives input focus. The event handling function will receive the following parameters: (wxformgrid me[, type(*) reference]) the reference is only passed if it is provided by the user. |
| onlabelclick | event | An event that is triggered when the user clicks on a row or column label or the corner label. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer col[, type(*) reference]) the reference is only passed if it is provided by the user. If the row is equal to -1 and the col is equal to -1, then the user clicked the corner label, otherwise if the row is equal to -1, they clicked a column label. If the col is equal to -1 they clicked a row label. |
| onlabelleftd-blclick | event | An event that is triggered when the user double clicks with the left mouse button on a label. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer |

| Property | Type | Description |
|---|---|---|
| | | col[, type(*) reference]) the reference is only passed if it is provided by the user. |
| onlabel-rightclick | event | An event that is triggered when the user clicks with the right mouse button on a label. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer col[, type(*) reference]) the reference is only passed if it is provided by the user. |
| onlabelrightd-blclick | event | An event that is triggered when the user double clicks with the right mouse button on a label. The event handling function will receive the following parameters: (wxformgrid me, integer row, integer col[, type(*) reference]) the reference is only passed if it is provided by the user. |
| onlostfocus | event | An event that is triggered when the grid control loses input focus. The event handling function will receive the following parameters: (wxformgrid me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| on-rowheightchange | event | An event that is triggered when the user resizes a row. The event handling function will receive the following parameters: (wxformgrid me, integer row[, type(*) reference]). The reference is only passed if it is provided by the user. |
| rowcount | integer | Specifies the number of rows in the grid. |
| rowheightdrag-gable | boolean | Specifies whether the row height can be changed by the user. |
| rowlabelalign-ment | string | Specifies the alignment for the row labels. |
| rowlabelwidth | integer | Specifies the width of the row label column in pixels. |
| tooltip | string | Contains the test that is displayed as a tooltip for the control. |
| top | integer | Gives the position of the top edge of the grid control relative to the top edge of the form, in pixels. |
| type | type | Specifies the wxformgrid type object. |
| visible | boolean | Specifies whether or not the control is visible. |
| width | integer | Gives the width of the grid control, in pixels. |

# Methods

## enablecelleditcontrol()

### Description

This method displays the edit control for the currently selected cell if the parameter passed is true, and hides the edit control (if currently displayed) if the parameter is false.

### Prototype

*wxformgridvar*.enablecelleditcontrol ( boolean *enabled* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | .true | boolean | If this parameter is equal to .true, then the edit control for the current cell will be shown (if it is not currently). If it is equal to .false, then it will hidden if it is currently displayed. |

## getcellalignment()

### Description

Retrieves the current alignment for the desired cell as a string in the form described for setting the alignment.

### Prototype

*wxformgridvar*.getcellalignment ( integer *row*, integer *col* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is located. |
| col | None | integer | The column in which the cell is located. |

## getcellbackgroundrgb()

### Description

Retrieves the current background color for a specific grid cell.

### Prototype

*wxformgridvar*.getcellbackgroundrgb ( integer *row*, integer *col* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | Contains the row of the cell for which the background color is being retrieved. |
| col | None | integer | Contains the column of the cell for which the background color is being retrieved. |

# getcellchoice()

## Description

Retrieves the value of the choice for the desired cell and index value or `.nul` if the cell is not a choice cell or if the index value is too large.

## Prototype

*wxformgridvar*.`getcellchoice` ( integer *`row`*, integer *`col`*, integer *`item`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is located. |
| col | None | integer | The column in which the cell is located. |
| item | 1 | integer | The index of the choice in the list. |

# getcellchoiceallowothers()

## Description

Retrieves `.true` if the desired cell allows other entries than the ones from its list of choices, `.false` if it does not, or `.nul` if the cell is not a choice cell.

## Prototype

*wxformgridvar*.`getcellchoiceallowothers` ( integer *`row`*, integer *`col`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is located. |
| col | None | integer | The column in which the cell is located. |

# getcellchoicecount()

## Description

Retrieves the number of choices for the desired cell or `.nul` if the cell is not a choice cell.

## Prototype

*wxformgridvar*.`getcellchoicecount` ( integer *`row`*, integer *`col`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is located. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| col | None | integer | The column in which the cell is located. |

# getcellfont()

## Description

Retrieves the current font for the desired cell as a wxfont object, or `.nul` if no font is assigned to the cell or as the default for the grid.

## Prototype

*wxformgridvar*.getcellfont ( integer *row*, integer *col* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is located. |
| col | None | integer | The column in which the cell is located. |

# getcelltextrgb()

## Description

Retrieves the current text color for a specific grid cell.

## Prototype

*wxformgridvar*.getcelltextrgb ( integer *row*, integer *col* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | Contains the row of the cell for which the text color is being retrieved. |
| col | None | integer | Contains the column of the cell for which the text color is being retrieved. |

# getcellvalue()

## Description

Retrieves the value for the desired cell.

## Prototype

*wxformgridvar*.getcellvalue ( integer *row*, integer *col* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is located. |
| col | None | integer | The column in which the cell is located. |

## getcollabel()

### Description

Retrieves the column label for the desired column.

### Prototype

*wxformgridvar*.getcollabel( integer *col* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| col | None | integer | The index of the column for which the label is retrieved. |

## getcolwidth()

### Description

Gets the width of the designated column.

### Prototype

*wxformgridvar*.getcolwidth( integer *col* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| col | None | integer | The column for which to retrieve the width value. |

## getcursorcol()

### Description

Retrieves the col where the grid cursor is currently located.

### Prototype

*wxformgridvar*.getcursorcol()

### Parameters

None

# getcursorrow()

## Description

Retrieves the row where the grid cursor is currently located.

## Prototype

*wxformgridvar*.getcursorrow ()

## Parameters

None

# getrowheight()

## Description

Gets the height of the designated row.

## Prototype

*wxformgridvar*.getrowheight ( integer *row* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row for which to retrieve the height value. |

# getrowlabel()

## Description

Retrieves the row label for the desired row.

## Prototype

*wxformgridvar*.getrowlabel ( integer *row* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The index of the row for which the label is retrieved. |

# getselectedcol()

## Description

Retrieves the next selected column following or equal to the start column. If not supplied it defaults to 1. This only works if the form has been placed into a container. Otherwise it always returns 0. To find all of the selected columns, start at 1 and continue until you get a return value of 0. There is a known issue

with the wxWidgets component. If you are Ctrl-clicking labels, any label preceding an earlier one does not appear in the internal list of selected labels.

### Prototype

*wxformgridvar*.getselectedcol ( integer *selectedcol*, integer *startcol* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| selectedcol | None | integer | This must be a pre-initialized integer object. The return value will be placed in the variable passed. If no column is selected (or none is selected equal to or following the startcol parameter), then the return value is 0. |
| startcol | 1 | integer | Contains the column from which to start checking for selected columns. |

## getselectedrow()

### Description

Retrieves the next selected row following or equal to the start row. If not supplied it defaults to 1. This only works if the form has been placed into a container. Otherwise it always returns 0. To find all of the selected rows, start at 1 and continue until you get a return value of 0. There is a known issue with the wxWidgets component. If you are Ctrl-clicking labels, any label preceding an earlier one does not appear in the internal list of selected labels.

### Prototype

*wxformgridvar*.getselectedrow ( integer *selectedrow*, integer *startrow* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| selectedrow | None | integer | This must be a pre-initialized integer object. The return value will be placed in the variable passed. If no row is selected (or none is selected equal to or following the startrow parameter), then the return value is 0. |
| startrow | 1 | integer | Contains the row from which to start checking for selected rows. |

## iscellreadonly()

### Description

Retrieves whether or not the desired cell is read only.

### Prototype

*wxformgridvar*.iscellreadonly ( integer *row*, integer *col* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is located. |
| col | None | integer | The column in which the cell is located. |

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*wxformgridvar*.remove ()

### Parameters

None

## setalignments()

### Description

This method assigns the alignments for the row and column labels and the default cell alignment. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression. The alignment is for both horizontal and vertical alignment. The string can be made up of two comma separated string values. The valid values are: `left`, `right`, `top`, and `bottom`. Providing two values for the same vector of alignment is an error, such as `"left,right"`. If an element is not supplied, it is assumed to be middle (in either direction). Supplying an empty string would create an alignment both horizontally and vertically centered.

### Prototype

*wxformgridvar*.setalignments ( string *rowlabelalignment*, string *collabelalignment*, string *alignment* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rowlabelalignment | The current setting of the rowlabelalignment property | string | The alignment string for both horizontal and vertical alignment for row labels. |
| collabelalignment | The current setting of the col- | string | The alignment string for both horizontal and vertical alignment for column labels. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | labelalignment property | | |
| alignment | The current setting for default alignment for cells that have no specifically defined alignment. | string | The default alignment string for both horizontal and vertical alignment for cells. |

# setcaptureable()

## Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

## Prototype

*wxformgridvar*.setcaptureable ( boolean *captureable* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the `controlcapture()` method of the form. |

# setcellalignment()

## Description

Sets the alignment for the cell or cells (if used with a range). The wxformgrid object itself is returned, to allow multiple methods to be put into one expression. It is not possible to set a cell back to being controlled by the default grid alignment setting. This limitation is part of the underlying grid control used for the implementation.

## Prototype

*wxformgridvar*.setcellalignment ( integer *row*, integer *col*, string *alignment*, integer *startrow*, integer *endrow*, integer *startcol*, integer *endcol* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is found. If not specified, then the *startrow* and *endrow* parameters are used. |
| col | None | integer | The column in which the cell is found. If not specified, then the |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | *startcol* and *endcol* parameters are used. |
| alignment | The current cell alignment if already assigned else the default alignment | string | The alignment to be assigned to the cell. The alignment is for both horizontal and vertical alignment. The string can be made up of two comma separated string values. The valid values are: `left`, `right`, `top`, and `bottom`. Providing two values for the same vector of alignment is an error, such as `"left,right"`. If an element is not supplied, it is assumed to be middle (in either direction). Supplying an empty string would create an alignment both horizontally and vertically centered. |
| startrow | 1 | integer | The starting row of a range for which to assign the alignment. |
| endrow | Number of rows in the grid | integer | The ending row of a range for which to assign the alignment. |
| startcol | 1 | integer | The starting column of a range for which to assign the alignment. |
| endcol | Number of columns in the grid | integer | The ending column of a range for which to assign the alignment. |

# setcellbackgroundrgb()

## Description

Sets the background color for the cell or cells (if used with a range). The wxformgrid object itself is returned, to allow multiple methods to be put into one expression. It is not possible to set a cell back to being controlled by the default grid background color setting. This limitation is part of the underlying grid control used for the implementation.

## Prototype

*wxformgridvar*.`setcellbackgroundrgb` ( integer *row*, integer *col*, integer *rgb*, integer *startrow*, integer *endrow*, integer *startcol*, integer *endcol* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is found. If not specified, then the *startrow* and *endrow* parameters are used. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| col | None | integer | The column in which the cell is found. If not specified, then the *startcol* and *endcol* parameters are used. |
| rgb | None | integer | The background color to be assigned to the cell. |
| startrow | 1 | integer | The starting row of a range for which to assign the background color. |
| endrow | Number of rows in the grid | integer | The ending row of a range for which to assign the background color. |
| startcol | 1 | integer | The starting column of a range for which to assign the background color. |
| endcol | Number of columns in the grid | integer | The ending column of a range for which to assign the background color. |

# setcellchoices()

## Description

Sets the currently selected value plus the list of choices for the desired cell or range. The choices are not named parameters, they are simply listed as separate string parameters for as many as there are choices. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression. Calling this method with no choices values and the *allowothers* parameter set to `.true` will set the cell back to being a normal string cell. Calling this method with no choices values and the *allowothers* parameter set to `.false` will result in an error. To set or get a choice (or other value if allowed) for a choice cell using the `setcellvalue()` and `getcellvalue()` methods.

### Note

Any unnamed parameter used when calling this method will be considered to be a choice! It is *essential* that the *row* and *col* parameters or the *startrow*/*endrow* and *startcol*/*endcol* parameters are specified by name when calling this method!

## Prototype

*wxformgridvar*.`setcellchoices` ( integer *row*, integer *col*, string *value*, boolean *allowothers*, integer *startrow*, integer *endrow*, integer *startcol*, integer *endcol*, string *choices*, string , … )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is found. If not specified, then the *startrow* and *endrow* parameters are used. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| col | None | integer | The column in which the cell is found. If not specified, then the *startcol* and *endcol* parameters are used. |
| value | The existing value in the cell | string | The value to be assigned to the cell. |
| allowothers | .true | boolean | If this value is .true then the combo box will be a dropedit style, if it is .false then it will be a droplist style. |
| startrow | 1 | integer | The starting row of a range for which to assign the value. |
| endrow | Number of rows in the grid | integer | The ending row of a range for which to assign the value. |
| startcol | 1 | integer | The starting column of a range for which to assign the value. |
| endcol | Number of columns in the grid | integer | The ending column of a range for which to assign the value. |
| choices | None | string | This string parameter can be used to set the choices by placing them into the string separated by the character value 0 ("{0}"). This allows programmatic/dynamic creation of the list of choices (for example by reading records from a database and creating the string dynamically). |
| | None | string | The string to be added as a choice. |
| … | None | | |

# setcellfont()

## Description

Sets the font to be used for the cell or cells (if using a range) in the grid control. The wxformgrid object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformgridvar*.setcellfont ( integer *row*, integer *col*, wxfont *font*, integer *startrow*, integer *endrow*, integer *startcol*, integer *endcol* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is found. If not specified, then the |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | *startrow* and *endrow* parameters are used. |
| col | None | integer | The column in which the cell is found. If not specified, then the *startcol* and *endcol* parameters are used. |
| font | None | wxfont | The wxfont object that will be used to format the text in the cell(s). |
| startrow | 1 | integer | The starting row of a range for which to assign the font. |
| endrow | Number of rows in the grid | integer | The ending row of a range for which to assign the font. |
| startcol | 1 | integer | The starting column of a range for which to assign the font. |
| endcol | Number of columns in the grid | integer | The ending column of a range for which to assign the font. |

# setcellreadonly()

## Description

Sets the desired cell or range to read only. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression. Calling this method with no parameters will clear all values from the grid.

## Prototype

*wxformgridvar*.setcellreadonly ( integer *row*, integer *col*, boolean *readonly*, integer *startrow*, integer *endrow*, integer *startcol*, integer *endcol* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is found. If not specified, then the *startrow* and *endrow* parameters are used. |
| col | None | integer | The column in which the cell is found. If not specified, then the *startcol* and *endcol* parameters are used. |
| readonly | .true | boolean | Whether the cell will be set to read only or not. |
| startrow | 1 | integer | The starting row of a range for which to set cells to read only. |
| endrow | Number of rows in the grid | integer | The ending row of a range for which to set cells to read only. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| startcol | 1 | integer | The starting column of a range for which to set cells to read only. |
| endcol | Number of columns in the grid | integer | The ending column of a range for which to set cells to read only. |

# setcelltextrgb()

## Description

Sets the text color for the cell or cells (if used with a range). The wxformgrid object itself is returned, to allow multiple methods to be put into one expression. It is not possible to set a cell back to being controlled by the default grid text color setting. This limitation is part of the underlying grid control used for the implementation.

## Prototype

*wxformgridvar*.setcelltextrgb ( integer *row*, integer *col*, integer *rgb*, integer *startrow*, integer *endrow*, integer *startcol*, integer *endcol* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is found. If not specified, then the *startrow* and *endrow* parameters are used. |
| col | None | integer | The column in which the cell is found. If not specified, then the *startcol* and *endcol* parameters are used. |
| rgb | None | integer | The text color to be assigned to the cell. |
| startrow | 1 | integer | The starting row of a range for which to assign the text color. |
| endrow | Number of rows in the grid | integer | The ending row of a range for which to assign the text color. |
| startcol | 1 | integer | The starting column of a range for which to assign the text color. |
| endcol | Number of columns in the grid | integer | The ending column of a range for which to assign the text color. |

# setcellvalue()

## Description

Sets the value for the desired cell or range. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression. Calling this method with no parameters will clear all values from the grid.

### Prototype

*wxformgridvar*.setcellvalue ( integer *row*, integer *col*, string *value*, array *valuearray*, integer *startrow*, integer *endrow*, integer *startcol*, integer *endcol* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row in which the cell is found. If not specified, then the *startrow* and *endrow* parameters are used. |
| col | None | integer | The column in which the cell is found. If not specified, then the *startcol* and *endcol* parameters are used. |
| value | None | string | The value to be assigned to the cell. |
| valuearray | None | array | This is an array of values to be assigned to multiple cells. The array is structured as follows: a[row,col], so to assign a group of values in one operation, for each row an array element is assigned the corresponding string value for each column. The row number is not a relative offset, it and the column number should directly reflect the position in the grid itself. Use the *startrow*, *endrow*, *startcol*, and *endcol* parameters to restrict the updates to only a certain range. |
| startrow | 1 | integer | The starting row of a range for which to assign the value. |
| endrow | Number of rows in the grid | integer | The ending row of a range for which to assign the value. |
| startcol | 1 | integer | The starting column of a range for which to assign the value. |
| endcol | Number of columns in the grid | integer | The ending column of a range for which to assign the value. |

## setcollabelheight()

### Description

Sets the height of the column labels. By setting this to 1, the column labels can be hidden. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformgridvar*.setcollabelheight ( integer *collabelheight* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| collabelheight | None | integer | The height in pixels of the column label row. |

## setcollabels()

### Description

Assigns labels to the columns in a grid control. The starting column is used to determine where the assignments begin. Multiple labels can be assigned in one call and each one will follow the next sequentially. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformgridvar*.setcollabels ( integer *startcol*, string , … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| startcol | 1 | integer | The column to which to assign the first of the labels. The remaining strings (if any) will be assigned sequentially following the first one. |
| | None | string | The string to be assigned to the label at the position described by the *startcol*. |
| … | None | | |

## setcolwidths()

### Description

Sets the width of the designated column or columns. If the *col* parameter is specified, then the *startcol* and *endcol* parameters cannot be specified and only one width value can be specified, either named or unnamed. If the *colwidth* parameter is specified, then no unnamed parameters can be used. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformgridvar*.setcolwidths ( integer *col*, integer *colwidth*, integer *startcol*, integer *endcol*, integer , … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| col | None | integer | The column to which to apply the width value. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| colwidth | The initial width for all columns | integer | The width of the specified column. |
| startcol | 1 | integer | The starting column to which to apply the width value(s). |
| endcol | Number of columns in the grid | integer | The starting column to which to apply the width value(s). |
| | None | integer | The width in pixels of the first column in sequence. |
| … | None | | |

# setcursorposition()

## Description

Sets the position of the grid cursor. The position must be within the defined rows and columns of the grid.

## Prototype

*wxformgridvar*.`setcursorposition` ( integer *`row`*, integer *`col`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row into which the grid cursor should be placed. |
| col | None | integer | The column into which the grid cursor should be placed. |

# setdraggability()

## Description

Determines whether the row heights and/or column widths can be modified by the user. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression.

## Prototype

*wxformgridvar*.`setdraggability` ( boolean *`rowheightdraggable`*, boolean *`colwidthdraggable`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rowheightdraggable | The current setting of the rowheightdraggable property | boolean | If this is set to `.true`, then the heights of the rows can be modified by the user. |
| colwidthdraggable | The current setting of the col- | boolean | If this is set to `.true`, then the widths of the columns can be modified by the user. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | widthdraggable property | | |

# setenabled()

## Description

Sets the enabled state of the grid control. The wxformgrid object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformgridvar*.setenabled ( boolean *enabled* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the grid control. |

# setfocus()

## Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire.

## Prototype

*wxformgridvar*.setfocus ()

## Parameters

None

# setfont()

## Description

Sets the default font to be used for the text in the grid control. The wxformgrid object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformgridvar*.setfont ( wxfont *font* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The wxfont object that will be used to format the text in the cells and labels. |

# setlabelfont()

## Description

Sets the font to be used for the row and column labels in the grid control. The wxformgrid object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformgridvar*.setlabelfont ( wxfont *font* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The wxfont object that will be used to format the text in the labels. |

# setname()

## Description

Sets the name of the grid control. The wxformgrid object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformgridvar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the grid control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

## Prototype

*wxformgridvar*.setnext ( type(wxformcontrol) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

# setposition()

## Description

Sets the size and/or position of the grid control. The wxformgrid object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformgridvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the grid control on the form. |
| top | The current value of the top property | integer | The new position of the top edge of the grid control on the form. |
| width | The current value of the width property | integer | The new width of the grid control on the form. |
| height | The current value of the height property | integer | The new height of the grid control on the form. |

# setrowheights()

## Description

Sets the height of the designated row or rows. If the *row* parameter is specified, then the *startrow* and *endrow* parameters cannot be specified and only one height value can be specified, either named or unnamed. If the *rowheight* parameter is specified, then no unnamed parameters can be used. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression.

## Prototype

*wxformgridvar*.setrowheights ( integer *row*, integer *rowheight*, integer *startrow*, integer *endrow*, integer )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| row | None | integer | The row to which to apply the height value. |
| rowheight | The initial height for all rows | integer | The height of the specified row. |
| startrow | 1 | integer | The starting row to which to apply the height value(s). |
| endrow | Number of rows in the grid | integer | The starting row to which to apply the height value(s). |
| | None | integer | The height in pixels of the first row in sequence. |

## setrowlabels()

### Description

Assigns labels to the rows in a grid control. The starting row is used to determine where the assignments begin. Multiple labels can be assigned in one call and each one will follow the next sequentially. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformgridvar*.setrowlabels ( integer *startrow*, string , … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| startrow | 1 | integer | The row to which to assign the first of the labels. The remaining strings (if any) will be assigned sequentially following the first one. |
| | None | string | The string to be assigned to the label at the position described by the *startrow*. |
| … | None | | |

## setrowlabelwidth()

### Description

Sets the width of the row labels. By setting this to 1, the row labels can be hidden. The wxformgrid object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformgridvar*.setrowlabelwidth ( integer *rowlabelwidth* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rowlabelwidth | None | integer | The width in pixels of the row label column. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformgrid object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformgridvar*.settooltip ( string *tooltip* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the grid control. The wxformgrid object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformgridvar*.setvisible ( boolean *visible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the grid control. |

# wxformlist

# Description

A wxformlist object is a box on a form that contains a list separate items, one or more of which can be selected.

# Type Tags

wxformcontrol

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the background of the list control. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformlist objects this is always `.true` |
| font | wxfont | This is a reference to the wxfont object that describes how the text in the list control is formatted. |
| form | wxform | Specifies the wxform object to which this list control belongs. |
| height | integer | Gives the height of the list control, in pixels. |
| itemcount | integer | Gives the number of items in the list control. |
| left | integer | Gives the position of the left side of the list control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformlist object. |
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |
| ondoubleclick | event | An event that is triggered when the list control receives a double-click. The event handling function will receive the following parameters: (wxformlist me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| ongotfocus | event | An event that is triggered when the list control receives input focus. The event handling function will receive the following parameters: (wxformlist me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| onlostfocus | event | An event that is triggered when the list control loses input focus. The event handling function will receive the following parameters: (wxformlist me[, type(*) reference]). The reference is only passed if it is provided by the user. |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td>onmouse</td><td>event</td><td>An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: <code>function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<ul><li>etype is the bit (defined as for onmousemask) showing what sort of event it is.</li><li>keys is a collection of bits showing key positions:<table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr><tr><td>alt down</td><td>0x00040000</td></tr><tr><td>meta down</td><td>0x00080000</td></tr></table></li><li>x and y are the position of the mouse event</li><li>reference is the (optional) reference object in the wxformlist.onmouse.reference</li></ul></td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>onselection-change</td><td>event</td><td>An event which is triggered when what is selected in the list control changes. The event handling function will receive the following parameters: (wxformlist me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>selectiontype</td><td>string</td><td>A string which indicates the way in which the user selects items in the list. <code>selectiontype</code> contains one of the following values:<ul><li>"single" — The user can select one item from the list. Selecting an item automatically deselects any previously selected item.</li><li>"multiple" — The user can select any number of items from the list, where clicking an item toggles its selection state.</li><li>"extended" — The user can select any number of items from the list, where combining mouse control with special keys (such as shift or control) allows multiple items to be selected in one action.</li></ul></td></tr>
<tr><td>textrgb</td><td>rgb</td><td>Gives the color of the text in the list control.</td></tr>
</table>

| Property | Type | Description |
|---|---|---|
| tooltip | string | Contains the test that is displayed as a tooltip for the control. |
| top | integer | Gives the position of the top edge of the list control relative to the top edge of the form, in pixels. |
| type | type | Specifies the wxformlist type object. |
| visible | boolean | Specifies whether or not the control is visible. |
| width | integer | Gives the width of the list control, in pixels. |

# Methods

## delete()

### Description

Deletes (removes) items from a list control. The wxformlist object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformlistvar*.`delete` ( integer , … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | None | integer | The 1-based index into the list of an item to delete. |
| … | None | | |

## deleteall()

### Description

Deletes (removes) all items from a list control. The wxformlist object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformlistvar*.`deleteall` ()

### Parameters

None

## getselected()

### Description

Searches a list control for the next selected item after a specified start point. The return value is the 1-based index of the first selected item which is equal to or greater than the specified start point, or is `.nul` if there is no such selected item.

### Prototype

*wxformlistvar*.getselected ( integer *firstitem* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| firstitem | None | integer | The 1-based index of the first item to search for selection. |

## insert()

### Description

Adds items to a list control. The wxformlist object itself is returned, to allow multiple methods to be put into one expression.

### Prototype

*wxformlistvar*.insert ( integer *firstitem*, string , … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| firstitem | The current value of the `itemcount` property plus one - i.e. the end of the list. | integer | The 1-based index into the list to show the position to place the first of the items to be inserted. |
| | None | string | The string to be added as an item at position `firstitem`. |
| … | None | | |

## isselected()

### Description

Determines whether or not a specified item in a list control is currently selected. The method returns `.true` if the item is selected, or `.false` otherwise.

### Prototype

*wxformlistvar*.isselected ( integer *item* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| item | None | integer | The 1-based index into the list of an item for which the selection state should be determined. |

# remove()

## Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

## Prototype

*wxformlistvar*.remove ()

## Parameters

None

# select()

## Description

Selects or deselects an item in a list control. The wxformlist object itself is returned, to allow multiple methods to be put into one expression.

## Prototype

*wxformlistvar*.select ( integer *item*, boolean *select* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| item | None | integer | The 1-based index of the item to select or deselect. |
| select | .true | boolean | Determines whether to select (.true) the item, or deselect (.false) it. It is an error to try to deselect an item in a list control with a selectiontype setting of "single" |

# setbackgroundrgb()

## Description

Sets the background color of the list control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the rgb argument and one or more of the red, green or blue arguments.

## Prototype

*wxformlistvar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of | integer | The new background color of the list control. It is inadvisable to |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | the `backgroundrgb` property | | specify any value for this argument which is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the list control. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the list control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the list control. This must be between 0 and 255 inclusive. |

# setcaptureable()

## Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

## Prototype

*`wxformlistvar`*`.setcaptureable` ( boolean *`captureable`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the `controlcapture()` method of the form. |

# setenabled()

## Description

Sets the enabled state of the list control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformlistvar`*`.setenabled` ( boolean *`enabled`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the list control. |

# setfocus()

## Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire.

## Prototype

*wxformlistvar*.setfocus ()

## Parameters

None

# setfont()

## Description

Sets the font to be used for the text in the listbox control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformlistvar*.setfont ( wxfont *font* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The wxfont object that will be used to format the text in the list-box control. |

# setname()

## Description

Sets the name of the list control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformlistvar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the list control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

### Prototype

*wxformlistvar*.setnext ( type(wxformcontrol) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

## setonmousemask()

### Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformlistvar*.setonmousemask ( integer *onmousemask* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onmousemask | None | integer | The definition of which events should be handled by the on-mouse event handler. The *onmousemask* is made up of bits: |

| Event Type | Bit Value |
|---|---|
| left button down | 0x00000001 |
| left button up | 0x00000002 |
| left button double click | 0x00000004 |
| middle button down | 0x00000010 |
| middle button up | 0x00000020 |
| middle button dbl click | 0x00000040 |
| right button down | 0x00000100 |
| right button up | 0x00000200 |
| right button dbl click | 0x00000400 |

| Parameter | Default value | Type name | Description | |
|---|---|---|---|---|
| | | | **Event Type** | **Bit Value** |
| | | | motion | 0x00001000 |

# setposition()

## Description

Sets the size and/or position of the list control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformlistvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the `left` property | integer | The new position of the left side of the list control on the form. |
| top | The current value of the `top` property | integer | The new position of the top edge of the list control on the form. |
| width | The current value of the `width` property | integer | The new width of the list control on the form. |
| height | The current value of the `height` property | integer | The new height of the list control on the form. |

# settextrgb()

## Description

Sets the color of the text in the list control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

## Prototype

*wxformlistvar*.settextrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `textrgb` property | integer | The new color of the text in the control. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| red | The current value of the `textrgb.red` property | integer | The red component in the new text color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the `textrgb.green` property | integer | The green component in the new text color of the control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `text.blue` property | integer | The blue component in the new text color of the control. This must be between 0 and 255 inclusive. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformlistvar`*`.settooltip` ( string *`tooltip`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the list control. The wxformlist object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformlistvar`*`.setvisible` ( boolean *`visible`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the list control. |

# wxformlist[]

## Get

### Subscripts

A numeric value giving the 1-based index of an item in the list.

### Description

Retrieves the text of the specified item.

## Set

### Subscripts

A numeric value giving the 1-based index of an item in the list.

### Description

Sets the text of the specified item.

## Set Reference

Attempting to set a reference to an object is not supported.

# wxformoption

# Description

A wxformoption object is a control that indicates whether it is 'checked' or not, normally by showing a circle that is either empty or has a dot in it. The control also contains a piece of text that labels it.

# Type Tags

wxformcontrol

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the background of the option control. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| enabled | boolean | Specifies whether or not the control is enabled. |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td>focusable</td><td>boolean</td><td>Specifies whether or not the control can have the input focus. For wxformoption objects this is always <code>.true</code>.</td></tr>
<tr><td>font</td><td>wxfont</td><td>This is a reference to the wxfont object that describes how the text in the option control is formatted.</td></tr>
<tr><td>form</td><td>wxform</td><td>Specifies the wxform object to which this option control belongs.</td></tr>
<tr><td>height</td><td>integer</td><td>Gives the height of the option control, in pixels.</td></tr>
<tr><td>left</td><td>integer</td><td>Gives the position of the left side of the option control relative to the left side of the form, in pixels.</td></tr>
<tr><td>name</td><td>string</td><td>The name of the wxformoption object.</td></tr>
<tr><td>next</td><td>type(wxformcontrol)</td><td>Specifies the next <code>wxformcontrol</code> on the same form.</td></tr>
<tr><td>onchange</td><td>event</td><td>An event which is triggered every time the user checks the the control. Normally the user cannot uncheck an option control. The event handling function will receive the following parameters: (wxformoption me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>ongotfocus</td><td>event</td><td>An event that is triggered when the option control receives input focus. The event handling function will receive the following parameters: (wxformoption me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onlostfocus</td><td>event</td><td>An event that is triggered when the option control loses input focus. The event handling function will receive the following parameters: (wxformoption me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onmouse</td><td>event</td><td>An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: <code>function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<ul><li>etype is the bit (defined as for onmousemask) showing what sort of event it is.</li><li>keys is a collection of bits showing key positions:<table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr></table></li></ul></td></tr>
</table>

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><th>Key Type</th><th>Bit Value</th></tr>
<tr><td>alt down</td><td>0x00040000</td></tr>
<tr><td>meta down</td><td>0x00080000</td></tr>
</table>
<ul>
<li>x and y are the position of the mouse event</li>
<li>reference is the (optional) reference object in the wxformoption.onmouse.reference</li>
</ul>
</td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>state</td><td>string</td><td>A string which indicates whether or not the option control is currently in a 'checked' state or not. If the control is not checked then the value of <code>state</code> is "". If the control is checked then the value of <code>state</code> is "on".</td></tr>
<tr><td>text</td><td>string</td><td>The current text for the wxformoption object, which is displayed on the form.</td></tr>
<tr><td>textrgb</td><td>rgb</td><td>Gives the color of the text in the option control.</td></tr>
<tr><td>tooltip</td><td>string</td><td>Contains the test that is displayed as a tooltip for the control.</td></tr>
<tr><td>top</td><td>integer</td><td>Gives the position of the top edge of the option control relative to the top edge of the form, in pixels.</td></tr>
<tr><td>type</td><td>type</td><td>Specifies the wxformoption type object.</td></tr>
<tr><td>visible</td><td>boolean</td><td>Specifies whether or not the control is visible.</td></tr>
<tr><td>width</td><td>integer</td><td>Gives the width of the option control, in pixels.</td></tr>
</table>

# Methods

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*wxformoptionvar*.remove ()

### Parameters

None

# setbackgroundrgb()

## Description

Sets the background color of the option control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

## Prototype

*`wxformoptionvar`*`.setbackgroundrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `backgroundrgb` property | integer | The new background color of the option control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the option control. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the option control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the option control. This must be between 0 and 255 inclusive. |

# setcaptureable()

## Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

## Prototype

*`wxformoptionvar`*`.setcaptureable` ( boolean *`captureable`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | form using the `controlcapture()` method of the form. |

# setenabled()

## Description

Sets the enabled state of the option control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformoptionvar`*`.setenabled` ( boolean *`enabled`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the option control. |

# setfocus()

## Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire.

## Prototype

*`wxformoptionvar`*`.setfocus` ()

## Parameters

None

# setfont()

## Description

Sets the font to be used for the text in the option control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformoptionvar`*`.setfont` ( wxfont *`font`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The wxfont object that will be used to format the text. |

# setname()

## Description

Sets the name of the option control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformoptionvar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the option control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

## Prototype

*wxformoptionvar*.setnext ( type(wxformcontrol) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value .nul sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

# setonmousemask()

## Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformoptionvar*.setonmousemask ( integer *onmousemask* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onmousemask | None | integer | The definition of which events should be handled by the on-mouse event handler. The *onmousemask* is made up of bits: |

| Event Type | Bit Value |
|---|---|
| left button down | 0x00000001 |
| left button up | 0x00000002 |
| left button double click | 0x00000004 |
| middle button down | 0x00000010 |
| middle button up | 0x00000020 |
| middle button dbl click | 0x00000040 |
| right button down | 0x00000100 |
| right button up | 0x00000200 |
| right button dbl click | 0x00000400 |
| motion | 0x00001000 |

## setposition()

### Description

Sets the size and/or position of the option control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformoptionvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the option control on the form. |
| top | The current value of the top property | integer | The new position of the top edge of the option control on the form. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | The current value of the `width` property | integer | The new width of the option control on the form. |
| height | The current value of the `height` property | integer | The new height of the option control on the form. |

# setstate()

## Description

Sets the state of the option control, i.e. whether or not it has the appearance of being 'checked'. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformoptionvar`*`.setstate` ( string *`state`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| state | None | string | The new state for the control. If the state is "" then the control is not checked. If the state is "on" then the control is checked. |

# settext()

## Description

Sets the text in the label part of the control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformoptionvar`*`.settext` ( string *`text`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | The new text to set in the control. |

# settextrgb()

## Description

Sets the color of the text in the label part of the control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*wxformoptionvar*.settextrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the textrgb property | integer | The new color of the text in the control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the textrgb.red propery | integer | The red component in the new text color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the textrgb.green property | integer | The green component in the new text color of the control. This must be between 0 and 255 inclusive. |
| blue | The current value of the text.blue property | integer | The blue component in the new text color of the control. This must be between 0 and 255 inclusive. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformoptionvar*.settooltip ( string *tooltip* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the option control. The wxformoption object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformoptionvar*.setvisible ( boolean *visible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the option control. |

# wxformscrollbar

## Description

A wxformscrollbar object is a free-standing scrollbar control either vertically or horizontally positioned on the form.

## Type Tags

wxformcontrol

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Gives the color of the control. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformscrollbar objects this is always `.true` |
| form | wxform | Specifies the wxform object to which this control belongs. |
| height | integer | Gives the height of the control, in pixels. |
| left | integer | Gives the position of the left side of the control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformscrollbar object. |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td>next</td><td>type(wxformcontrol)</td><td>Specifies the next <code>wxformcontrol</code> on the same form.</td></tr>
<tr><td>ongotfocus</td><td>event</td><td>An event that is triggered when the scrollbar control receives input focus. The event handling function will receive the following parameters: (wxformscrollbar me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onlostfocus</td><td>event</td><td>An event that is triggered when the scrollbar control loses input focus. The event handling function will receive the following parameters: (wxformscrollbar me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>onmouse</td><td>event</td><td>An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: <code>function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<ul><li>etype is the bit (defined as for onmousemask) showing what sort of event it is.</li><li>keys is a collection of bits showing key positions:<table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr><tr><td>alt down</td><td>0x00040000</td></tr><tr><td>meta down</td><td>0x00080000</td></tr></table></li><li>x and y are the position of the mouse event</li><li>reference is the (optional) reference object in the wxformscrollbar.onmouse.reference</li></ul></td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>onscroll</td><td>event</td><td>An event that is triggered every time the scroll position of control changes. The event handling function will receive the following parameters: (wxformscrollbar me, string scrolltype[, type(*) reference]). The reference is only passed if it is provided by the user. The scrolltype will be one of: <code>"top"</code>, <code>"bottom"</code>, <code>"thumbdrop"</code>, <code>"thumbdrag"</code>, <code>"pageup"</code>, <code>"pagedown"</code>, <code>"lineup"</code>, or <code>"linedown"</code>.</td></tr>
<tr><td>orientation</td><td>string</td><td>This contains either <code>"v"</code> if the scroll bar is vertical or <code>"h"</code> if the scroll bar is horizontal.</td></tr>
</table>

| Property | Type | Description |
|---|---|---|
| pagesize | integer | This contains the number of values from the overall range that represent a single page (when clicking in the scroll bar area outside of the thumb itself). This then decides how far the thumb actually moves when a click in these areas occurs. Valid values are from 1 through the size of the range. |
| position | integer | This contains the current position of the thumb in the scroll bar. Valid values are from 0 to range minus thumbsize. |
| range | integer | This contains the range of values, beginning at 0 and ending at range - 1, that describe the granularity of the scroll bar. Valid values are from 1 through 4G. |
| thumbsize | integer | This contains the size of the thumb. The thumb size is normally automatically managed by the relationship between the range, the scrollbar size, and the page size, but this can be overridden. Valid values are from 1 up to the size of the range. |
| tooltip | string | Contains the test that is displayed as a tooltip for the control. |
| top | integer | Gives the position of the top edge of the control relative to the top edge of the form, in pixels. |
| type | type | Specifies the wxformscrollbar type object. |
| visible | boolean | Specifies whether or not the control is visible. |
| width | integer | Gives the width of the control, in pixels. |

# Methods

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*`wxformscrollbarvar`*`.remove` ()

### Parameters

None

## setbackgroundrgb()

### Description

Sets the color of the control. The wxformscrollbar object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green`, or `blue` arguments.

### Prototype

*wxformscrollbarvar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the rgb property | integer | The new color of the control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the rgb.red property | integer | The red component in the new color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the rgb.green property | integer | The green component in the new color of the control. This must be between 0 and 255 inclusive. |
| blue | The current value of the rgb.blue property | integer | The blue component in the new color of the control. This must be between 0 and 255 inclusive. |

## setcaptureable()

### Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

### Prototype

*wxformscrollbarvar*.setcaptureable ( boolean *captureable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the `controlcapture()` method of the form. |

## setenabled()

### Description

Sets the enabled state of the control. The wxformscrollbar object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformscrollbarvar*.setenabled ( boolean *enabled* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the control. |

## setfocus()

### Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire.

### Prototype

*wxformscrollbarvar*.setfocus ()

### Parameters

None

## setname()

### Description

Sets the name of the scrollbar control. The wxformscrollbar object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformscrollbarvar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the control. |

## setnext()

### Description

Sets the position of the control in the z-order and tab order.

### Prototype

*wxformscrollbarvar*.setnext ( type(wxformcontrol) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value `.nul` sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

## setonmousemask()

### Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformscrollbar object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformscrollbarvar`*`.setonmousemask` ( integer *`onmousemask`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onmousemask | None | integer | The definition of which events should be handled by the onmouse event handler. The *`onmousemask`* is made up of bits:<br>**Event Type** / **Bit Value**<br>left button down — 0x00000001<br>left button up — 0x00000002<br>left button double click — 0x00000004<br>middle button down — 0x00000010<br>middle button up — 0x00000020<br>middle button dbl click — 0x00000040<br>right button down — 0x00000100<br>right button up — 0x00000200<br>right button dbl click — 0x00000400<br>motion — 0x00001000 |

# setposition()

## Description

Sets the size and/or position of the control. The wxformscrollbar object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformscrollbarvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the control on the form. |
| top | The current value of the top property | integer | The new position of the top edge of the control on the form. |
| width | The current value of the width property | integer | The new width of the control on the form. |
| height | The current value of the height property | integer | The new height of the control on the form. |

# setscroll()

## Description

Sets various properties of the control. The wxformscrollbar object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformscrollbarvar*.setscroll ( integer *range*, integer *position*, integer *pagesize*, integer *thumbsize* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| range | Current value of the range property | integer | The desired range for the control. |
| position | Current value of the position property | integer | The desired position of the thumb in the scrollbar control. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| pagesize | Current value of the pagesize property | integer | The desired pagesize for the scrollbar control. |
| thumbsize | Current value of the thumbsize property | integer | The desired thumbsize for the scrollbar control. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformscrollbar object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformscrollbarvar*.settooltip ( string *tooltip* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the control. The wxformscrollbar object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformscrollbarvar*.setvisible ( boolean *visible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the control. |

# wxformsizebox

# Description

A wxformsizebox object is a control that has an outline and sizing handles that can be moved by selecting and dragging the center of the box or that can be resized by selecting and dragging one of the 8 sizing boxes (sides and corners).

# Type Tags

wxformcontrol

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformsizebox objects this is always `.true` |
| form | wxform | Specifies the wxform object to which this control belongs. |
| height | integer | Gives the height of the control, in pixels. |
| left | integer | Gives the position of the left side of the control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformsizebox object. |
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |
| ongotfocus | event | An event that is triggered when the size box control receives input focus. The event handling function will receive the following parameters: (wxformsizebox me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| onlostfocus | event | An event that is triggered when the size box control loses input focus. The event handling function will receive the following parameters: (wxformsizebox me[, type(*) reference]). The reference is only passed if it is provided by the user. |
| onmouse | event | An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: `function onmouse(type(wxformcontrol) con-` |

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td></td><td></td><td><code>trol, integer etype, integer keys, integer x, integer y, type(*) reference)</code> where:<br><br>• etype is the bit (defined as for onmousemask) showing what sort of event it is.<br><br>• keys is a collection of bits showing key positions:<br><br><table><tr><th>Key Type</th><th>Bit Value</th></tr><tr><td>shift down</td><td>0x00010000</td></tr><tr><td>ctrl down</td><td>0x00020000</td></tr><tr><td>alt down</td><td>0x00040000</td></tr><tr><td>meta down</td><td>0x00080000</td></tr></table><br>• x and y are the position of the mouse event<br><br>• reference is the (optional) reference object in the wxformsizebox.onmouse.reference</td></tr>
<tr><td>onmousemask</td><td>integer</td><td>Holds the mask that decides which mouse events are captured and sent to the onmouse event handler.</td></tr>
<tr><td>onsize</td><td>event</td><td>An event that is triggered every time the control is moved or resized. The event handling function will receive the following parameters: (wxformsizebox me[, type(*) reference]). The reference is only passed if it is provided by the user.</td></tr>
<tr><td>rgb</td><td>rgb</td><td>Gives the color of the control.</td></tr>
<tr><td>top</td><td>integer</td><td>Gives the position of the top edge of the control relative to the top edge of the form, in pixels.</td></tr>
<tr><td>type</td><td>type</td><td>Specifies the wxformsizebox type object.</td></tr>
<tr><td>visible</td><td>boolean</td><td>Specifies whether or not the control is visible.</td></tr>
<tr><td>width</td><td>integer</td><td>Gives the width of the control, in pixels.</td></tr>
</table>

# Methods

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*wxformsizeboxvar*.remove ()

### Parameters

None

# setcaptureable()

## Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

## Prototype

*wxformsizeboxvar*.setcaptureable ( boolean *captureable* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the `controlcapture()` method of the form. |

# setenabled()

## Description

Sets the enabled state of the control. The wxformsizebox object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformsizeboxvar*.setenabled ( boolean *enabled* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the control. |

# setfocus()

## Description

Sets focus to the control, provided that it is not invisible or disabled. If this is called for a form that is not yet in a container, then nothing will happen except that the focuscontrol of the wxform object will be set. Once the form is placed in a container, the control will get focus and the ongotfocus event will fire.

## Prototype

*wxformsizeboxvar*.setfocus ()

## Parameters

None

# setname()

## Description

Sets the name of the button. The wxformsizebox object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformsizeboxvar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the control. |

# setnext()

## Description

Sets the position of the control in the z-order and tab order.

## Prototype

*wxformsizeboxvar*.setnext ( type(wxformcontrol) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value `.nul` sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

# setonmousemask()

## Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformsizebox object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxformsizeboxvar*.setonmousemask ( integer *onmousemask* )

### Parameters

<table>
<tr><th>Parameter</th><th>Default value</th><th>Type name</th><th>Description</th></tr>
<tr><td>onmousemask</td><td>None</td><td>integer</td><td>The definition of which events should be handled by the on-mouse event handler. The onmousemask is made up of bits:
<table>
<tr><th>Event Type</th><th>Bit Value</th></tr>
<tr><td>left button down</td><td>0x00000001</td></tr>
<tr><td>left button up</td><td>0x00000002</td></tr>
<tr><td>left button double click</td><td>0x00000004</td></tr>
<tr><td>middle button down</td><td>0x00000010</td></tr>
<tr><td>middle button up</td><td>0x00000020</td></tr>
<tr><td>middle button dbl click</td><td>0x00000040</td></tr>
<tr><td>right button down</td><td>0x00000100</td></tr>
<tr><td>right button up</td><td>0x00000200</td></tr>
<tr><td>right button dbl click</td><td>0x00000400</td></tr>
<tr><td>motion</td><td>0x00001000</td></tr>
</table></td></tr>
</table>

## setposition()

### Description

Sets the size and/or position of the control. The wxformsizebox object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformsizeboxvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the control on the form. |
| top | The current value of the top property | integer | The new position of the top edge of the control on the form. |
| width | The current value of the width property | integer | The new width of the control on the form. |
| height | The current value of the | integer | The new height of the control on the form. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | `height` property | | |

# setrgb()

## Description

Sets the color of the control. The wxformsizebox object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green`, or `blue` arguments.

## Prototype

*`wxformsizeboxvar`*`.setrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the rgb property | integer | The new color of the control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the rgb.red property | integer | The red component in the new color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the rgb.green property | integer | The green component in the new color of the control. This must be between 0 and 255 inclusive. |
| blue | The current value of the rgb.blue property | integer | The blue component in the new color of the control. This must be between 0 and 255 inclusive. |

# setvisible()

## Description

Sets the visibility of the control. The wxformsizebox object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformsizeboxvar`*`.setvisible` ( boolean *`visible`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the control. |

# wxformtext

## Description

A wxformtext object is an area of a form where a piece of text is displayed.

## Type Tags

wxformcontrol

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| alignment | string | The alignment of the content of the control. This will either be `"left"`, `"right"`, or `""` if the content is centered. This must be set at the time the control is created. |
| backgroundrgb | rgb | Gives the color of the background of the text control. |
| captureable | boolean | Specifies whether or not the control events can be captured by the form when the `controlcapture()` method is called. |
| enabled | boolean | Specifies whether or not the control is enabled. |
| focusable | boolean | Specifies whether or not the control can have the input focus. For wxformtext objects this is always `.false`. |
| font | wxfont | This is a reference to the wxfont object that describes how the text in the control is formatted. |
| form | wxform | Specifies the wxform object to which this text control belongs. |
| height | integer | Gives the height of the text control, in pixels. |
| left | integer | Gives the position of the left side of the text control relative to the left side of the form, in pixels. |
| name | string | The name of the wxformtext object. |

| Property | Type | Description |
|---|---|---|
| next | type(wxformcontrol) | Specifies the next `wxformcontrol` on the same form. |
| onmouse | event | An event that is triggered each time a mouse event occurs that matches the bits set in the onmousemask property. The onmouse event handling function should be defined as follows: `function onmouse(type(wxformcontrol) control, integer etype, integer keys, integer x, integer y, type(*) reference)` where:<br>• etype is the bit (defined as for onmousemask) showing what sort of event it is.<br>• keys is a collection of bits showing key positions:<br>**Key Type** / **Bit Value**<br>shift down / 0x00010000<br>ctrl down / 0x00020000<br>alt down / 0x00040000<br>meta down / 0x00080000<br>• x and y are the position of the mouse event<br>• reference is the (optional) reference object in the wxformtext.onmouse.reference |
| onmousemask | integer | Holds the mask that decides which mouse events are captured and sent to the onmouse event handler. |
| text | string | The current text for the wxformtext object, which is displayed on the form. |
| textrgb | rgb | Gives the color of the text in the control. |
| tooltip | string | Contains the test that is displayed as a tooltip for the control. |
| top | integer | Gives the position of the top edge of the text control relative to the top edge of the form, in pixels. |
| type | type | Specifies the wxformtext type object. |
| visible | boolean | Specifies whether or not the control is visible. |
| width | integer | Gives the width of the text control, in pixels. |

# Methods

## remove()

### Description

Removes the control from the form. If any object has a reference to the control, those references will no longer be valid. It is not safe to attempt to use a control after this method has been called.

### Prototype

*wxformtextvar*.remove ()

### Parameters

None

## setbackgroundrgb()

### Description

Sets the background color of the text control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the rgb argument and one or more of the red, green or blue arguments.

### Prototype

*wxformtextvar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the backgroundrgb property | integer | The new background color of the text control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the backgroundrgb.red property | integer | The red component in the new background color of the text control. This must be between 0 and 255 inclusive. |
| green | The current value of the backgroundrgb.green property | integer | The green component in the new background color of the text control. This must be between 0 and 255 inclusive. |
| blue | The current value of the backgroundrgb.blue property | integer | The blue component in the new background color of the text control. This must be between 0 and 255 inclusive. |

## setcaptureable()

### Description

This method is used to determine whether the events for the control can be captured and passed to the form or not. By default, events can be captured.

### Prototype

*wxformtextvar*.setcaptureable ( boolean *captureable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captureable | None | boolean | Sets whether or not the events for this control can be captured by the form using the `controlcapture()` method of the form. |

## setenabled()

### Description

Sets the enabled state of the text control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformtextvar`*`.setenabled` ( boolean *`enabled`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the text control. |

## setfont()

### Description

Sets the font to be used for the text in the text control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformtextvar`*`.setfont` ( wxfont *`font`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The wxfont object that will be used to format the text. |

## setname()

### Description

Sets the name of the text control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxformtextvar`*`.setname` ( string *`name`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the text control. |

## setnext()

### Description

Sets the position of the control in the z-order and tab order.

### Prototype

*wxformtextvar*.setnext ( type(wxformcontrol) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxformcontrol) | Calling the method with the value `.nul` sets a control to be the last one in the ring (the one before form.firstcontrol) and so has the highest z-order, and passing any other control as the argument puts the target control immediately below the one specified as the parameter. The visible stacking of controls on a form is such that the earliest in the ring is the furthest back in the z-order. |

## setonmousemask()

### Description

Sets the mask that is used to decide which mouse events will be trapped for the control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformtextvar*.setonmousemask ( integer *onmousemask* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| onmousemask | None | integer | The definition of which events should be handled by the onmouse event handler. The *onmousemask* is made up of bits:<br>**Event Type** \| **Bit Value**<br>left button down \| 0x00000001 |

| Parameter | Default value | Type name | Description | |
|---|---|---|---|---|
| | | | **Event Type** | **Bit Value** |
| | | | left button up | 0x00000002 |
| | | | left button double click | 0x00000004 |
| | | | middle button down | 0x00000010 |
| | | | middle button up | 0x00000020 |
| | | | middle button dbl click | 0x00000040 |
| | | | right button down | 0x00000100 |
| | | | right button up | 0x00000200 |
| | | | right button dbl click | 0x00000400 |
| | | | motion | 0x00001000 |

# setposition()

## Description

Sets the size and/or position of the text control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxformtextvar`*`.setposition` ( integer *`left`*, integer *`top`*, integer *`width`*, integer *`height`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the `left` property | integer | The new position of the left side of the text control on the form. |
| top | The current value of the `top` property | integer | The new position of the top edge of the text control on the form. |
| width | The current value of the `width` property | integer | The new width of the text control on the form. |
| height | The current value of the `height` property | integer | The new height of the text control on the form. |

# settext()

## Description

Sets the text in the text control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformtextvar*.settext ( string *text* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | The new text to set in the text control. |

## settextrgb()

### Description

Sets the color of the text in the control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*wxformtextvar*.settextrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `textrgb` property | integer | The new color of the text in the control. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the `textrgb.red` property | integer | The red component in the new text color of the control. This must be between 0 and 255 inclusive. |
| green | The current value of the `textrgb.green` property | integer | The green component in the new text color of the control. This must be between 0 and 255 inclusive. |
| blue | The current value of the `text.blue` property | integer | The blue component in the new text color of the control. This must be between 0 and 255 inclusive. |

## settooltip()

### Description

Sets the text for the tooltip associated with the control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformtextvar*.settooltip ( string *tooltip* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

## setvisible()

### Description

Sets the visibility of the text control. The wxformtext object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxformtextvar*.setvisible ( boolean *visible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the text control. |

# wxgraphicarc

# Description

A wxgraphicarc object represents an arc that can be drawn on an appropriate container. The arc has an optional border, border width and border color together with a fill color.

# Type Tags

wxgraphic

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| borderrgb | rgb | Gives the color of the arc's border. |
| bordervisible | boolean | Specifies whether or not the border is visible. |
| borderwidth | integer | Gives the width of the border, in pixels. |
| container | type(wxgraphiccontainer) | Specifies the object to which this graphic belongs. |
| midpoint | fixedpoint | Gives the position of the midpoint of the arc. |
| name | string | The name of the wxgraphicarc object. |
| next | type(wxgraphic) | Specifies the next `wxgraphic` on the same container. |
| point1 | fixedpoint | Gives the position of the first point of the arc. |
| point2 | fixedpoint | Gives the position of the second point of the arc. |
| rgb | rgb | Gives the fill color of the arc. |
| type | type | Specifies the wxgraphicarc type object. |
| visible | boolean | Specifies whether or not the content is visible. |

# Methods

## remove()

### Description

Removes the graphic from the container. If any object has a reference to the graphic, those references will no longer be valid. It is not safe to attempt to use a graphic after this method has been called.

### Prototype

*wxgraphicarcvar*.remove ()

### Parameters

None

## setname()

### Description

Sets the name of the ellipse control. The wxgraphicarc object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxgraphicarcvar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the graphic. |

## setnext()

### Description

Sets the position of the graphic in the z-order and tab order.

### Prototype

*wxgraphicarcvar*.setnext ( type(wxgraphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxgraphic) | Calling the method with the value .nul sets a graphic to be the last one in the ring (the one before container.firstgraphic) and so has the highest z-order, and passing any other graphic as the argument puts the target graphic immediately below the one specified as the parameter. The visible stacking of graphics on a container is such that the earliest in the ring is the furthest back in the z-order. |

## setposition()

### Description

Sets the size and/or position of the graphic. The wxgraphicarc object itself is returned, to allow multiple setting methods to be put into one expression. Not all parameters are available to all wxgraphic objects. It is recommended to name the parameters when using this method.

### Prototype

*wxgraphicarcvar*.setposition ( point *point1*, point *point2*, point *point3*, point *midpoint*, integer *width*, integer *borderwidth* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point1 | The current value of the point1 property | point | The new position of the starting point of the graphic. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point2 | The current value of the `point2` property | point | The new position of the second point of the graphic. |
| point3 | The current value of the `point3` property | point | The new position of the third point of the graphic. |
| midpoint | The current value of the `midpoint` property | point | The new position of the midpoint of the graphic. |
| width | The current value of the `width` property | integer | The new width of the graphic. |
| borderwidth | The current value of the `borderwidth` property | integer | The new width of the border of the graphic. |

# setrgb()

## Description

Sets the color and border color of the graphic. The wxgraphicarc object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxgraphicarcvar*.`setrgb` ( integer *rgb*, integer *borderrgb* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the rgb property | integer | The new color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| borderrgb | The current value of the borderrgb property | integer | The new border color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |

# setvisible()

## Description

Sets the visibility of the graphic. The wxgraphicarc object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxgraphicarcvar*.setvisible ( boolean *visible*, boolean *bordervisible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the content of the graphic. |
| bordervisible | None | boolean | The desired visibility of the graphic's border. |

# wxgraphicellipse

## Description

A wxgraphicellipse object represents an ellipse that can be drawn on an appropriate container. The ellipse has an optional border, border width and border color together with a fill color. If an attempt is made to create an ellipse with impossible charactersitics, an error will be generated.

## Type Tags

wxgraphic

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| borderrgb | rgb | Gives the color of the ellipse's border. |
| bordervisible | boolean | Specifies whether or not the border is visible. |
| borderwidth | integer | Gives the width of the border, in pixels. |
| container | type(wxgraphiccontainer) | Specifies the object to which this graphic belongs. |
| midpoint | fixedpoint | Gives the position of the midpoint of the ellipse. |
| name | string | The name of the wxgraphicellipse object. |
| next | type(wxgraphic) | Specifies the next `wxgraphic` on the same container. |

| Property | Type | Description |
|---|---|---|
| point1 | fixedpoint | Gives the position of the first point of the ellipse. |
| point2 | fixedpoint | Gives the position of the second and opposite point of the ellipse. |
| rgb | rgb | Gives the fill color of the ellipse. |
| type | type | Specifies the wxgraphicellipse type object. |
| visible | boolean | Specifies whether or not the content is visible. |

# Methods

## remove()

### Description

Removes the graphic from the container. If any object has a reference to the graphic, those references will no longer be valid. It is not safe to attempt to use a graphic after this method has been called.

### Prototype

*wxgraphicellipsevar*.remove ()

### Parameters

None

## setname()

### Description

Sets the name of the ellipse control. The wxgraphicellipse object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxgraphicellipsevar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the graphic. |

## setnext()

### Description

Sets the position of the graphic in the z-order and tab order.

### Prototype

*wxgraphicellipsevar*.setnext ( type(wxgraphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxgraphic) | Calling the method with the value `.nul` sets a graphic to be the last one in the ring (the one before container.firstgraphic) and so has the highest z-order, and passing any other graphic as the argument puts the target graphic immediately below the one specified as the parameter. The visible stacking of graphics on a container is such that the earliest in the ring is the furthest back in the z-order. |

## setposition()

### Description

Sets the size and/or position of the graphic. The wxgraphicellipse object itself is returned, to allow multiple setting methods to be put into one expression. Not all parameters are available to all wxgraphic objects. It is recommended to name the parameters when using this method.

### Prototype

*`wxgraphicellipsevar`*`.setposition` ( point *`point1`*, point *`point2`*, point *`point3`*, point *`midpoint`*, integer *`width`*, integer *`borderwidth`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point1 | The current value of the `point1` property | point | The new position of the starting point of the graphic. |
| point2 | The current value of the `point2` property | point | The new position of the second point of the graphic. |
| point3 | The current value of the `point3` property | point | The new position of the third point of the graphic. |
| midpoint | The current value of the `midpoint` property | point | The new position of the midpoint of the graphic. |
| width | The current value of the `width` property | integer | The new width of the graphic. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| borderwidth | The current value of the `borderwidth` property | integer | The new width of the border of the graphic. |

# setrgb()

## Description

Sets the color and border color of the graphic. The wxgraphicellipse object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxgraphicellipsevar`*`.setrgb` ( integer *`rgb`*, integer *`borderrgb`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the rgb property | integer | The new color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| borderrgb | The current value of the borderrgb property | integer | The new border color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |

# setvisible()

## Description

Sets the visibility of the graphic. The wxgraphicellipse object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxgraphicellipsevar`*`.setvisible` ( boolean *`visible`*, boolean *`bordervisible`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the content of the graphic. |
| bordervisible | None | boolean | The desired visibility of the graphic's border. |

# wxgraphicline

## Description

A wxgraphicline object represents a line that can be drawn on an appropriate container. The line has a width and a color.

## Type Tags

wxgraphic

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| container | type(wxgraphiccontainer) | Specifies the object to which this graphic belongs. |
| name | string | The name of the wxgraphicline object. |
| next | type(wxgraphic) | Specifies the next `wxgraphic` on the same container. |
| point1 | fixedpoint | Gives the position of the starting point of the line. |
| point2 | fixedpoint | Gives the position of the ending point of the line. |
| rgb | rgb | Gives the color of the line. |
| type | type | Specifies the wxgraphicline type object. |
| visible | boolean | Specifies whether or not the line is visible. |
| width | integer | Gives the width of the line, in pixels. |

## Methods

### remove()

#### Description

Removes the graphic from the container. If any object has a reference to the graphic, those references will no longer be valid. It is not safe to attempt to use a graphic after this method has been called.

#### Prototype

*`wxgraphiclinevar`*`.remove` ()

### Parameters

None

## setname()

### Description

Sets the name of the line graphic. The wxgraphicline object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxgraphiclinevar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the graphic. |

## setnext()

### Description

Sets the position of the graphic in the z-order and tab order.

### Prototype

*wxgraphiclinevar*.setnext ( type(wxgraphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxgraphic) | Calling the method with the value .nul sets a graphic to be the last one in the ring (the one before container.firstgraphic) and so has the highest z-order, and passing any other graphic as the argument puts the target graphic immediately below the one specified as the parameter. The visible stacking of graphics on a container is such that the earliest in the ring is the furthest back in the z-order. |

## setposition()

### Description

Sets the size and/or position of the graphic. The wxgraphicline object itself is returned, to allow multiple setting methods to be put into one expression. Not all parameters are available to all wxgraphic objects. It is recommended to name the parameters when using this method.

### Prototype

*wxgraphiclinevar*.setposition ( point *point1*, point *point2*, point *point3*, point *midpoint*, integer *width*, integer *borderwidth* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point1 | The current value of the point1 property | point | The new position of the starting point of the graphic. |
| point2 | The current value of the point2 property | point | The new position of the second point of the graphic. |
| point3 | The current value of the point3 property | point | The new position of the third point of the graphic. |
| midpoint | The current value of the midpoint property | point | The new position of the midpoint of the graphic. |
| width | The current value of the width property | integer | The new width of the graphic. |
| borderwidth | The current value of the borderwidth property | integer | The new width of the border of the graphic. |

## setrgb()

### Description

Sets the color and border color of the graphic. The wxgraphicline object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxgraphiclinevar*.setrgb ( integer *rgb*, integer *borderrgb* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the rgb property | integer | The new color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| borderrgb | The current value of the borderrgb property | integer | The new border color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |

## setvisible()

### Description

Sets the visibility of the graphic. The wxgraphicline object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxgraphiclinevar*.setvisible ( boolean *visible*, boolean *bordervisible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the content of the graphic. |
| bordervisible | None | boolean | The desired visibility of the graphic's border. |

# wxgraphicrectangle

# Description

A wxgraphicrectangle object represents a rectangle that can be drawn on an appropriate container. The rectangle has an optional border, border width and border color together with a fill color.

# Type Tags

wxgraphic

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| borderrgb | rgb | Gives the color of the rectangle's border. |
| bordervisible | boolean | Specifies whether or not the border is visible. |
| borderwidth | integer | Gives the width of the border, in pixels. |
| container | type(wxgraphiccontainer) | Specifies the object to which this graphic belongs. |
| name | string | The name of the wxgraphicrectangle object. |
| next | type(wxgraphic) | Specifies the next `wxgraphic` on the same container. |
| point1 | fixedpoint | Gives the position of the first point of the rectangle. |
| point2 | fixedpoint | Gives the position of the second and opposite point of the rectangle. |
| rgb | rgb | Gives the fill color of the rectangle. |
| type | type | Specifies the wxgraphicrectangle type object. |
| visible | boolean | Specifies whether or not the content is visible. |

# Methods

## remove()

### Description

Removes the graphic from the container. If any object has a reference to the graphic, those references will no longer be valid. It is not safe to attempt to use a graphic after this method has been called.

### Prototype

*wxgraphicrectanglevar*.remove ()

### Parameters

None

## setname()

### Description

Sets the name of the rectangle control. The wxgraphicrectangle object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxgraphicrectanglevar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the graphic. |

## setnext()

### Description

Sets the position of the graphic in the z-order and tab order.

### Prototype

*wxgraphicrectanglevar*.setnext ( type(wxgraphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxgraphic) | Calling the method with the value .nul sets a graphic to be the last one in the ring (the one before container.firstgraphic) and so has the highest z-order, and passing any other graphic as the argument puts the target graphic immediately below the one specified as the parameter. The visible stacking of graphics on a container is such that the earliest in the ring is the furthest back in the z-order. |

## setposition()

### Description

Sets the size and/or position of the graphic. The wxgraphicrectangle object itself is returned, to allow multiple setting methods to be put into one expression. Not all parameters are available to all wxgraphic objects. It is recommended to name the parameters when using this method.

### Prototype

*wxgraphicrectanglevar*.setposition ( point *point1*, point *point2*, point *point3*, point *midpoint*, integer *width*, integer *borderwidth* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point1 | The current value of the point1 property | point | The new position of the starting point of the graphic. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point2 | The current value of the `point2` property | point | The new position of the second point of the graphic. |
| point3 | The current value of the `point3` property | point | The new position of the third point of the graphic. |
| midpoint | The current value of the `midpoint` property | point | The new position of the midpoint of the graphic. |
| width | The current value of the `width` property | integer | The new width of the graphic. |
| borderwidth | The current value of the `borderwidth` property | integer | The new width of the border of the graphic. |

# setrgb()

## Description

Sets the color and border color of the graphic. The wxgraphicrectangle object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxgraphicrectanglevar`*`.setrgb` ( integer *`rgb`*, integer *`borderrgb`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the rgb property | integer | The new color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| borderrgb | The current value of the borderrgb property | integer | The new border color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |

# setvisible()

## Description

Sets the visibility of the graphic. The wxgraphicrectangle object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxgraphicrectanglevar*.setvisible ( boolean *visible*, boolean *bordervisible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the content of the graphic. |
| bordervisible | None | boolean | The desired visibility of the graphic's border. |

# wxgraphictriangle

## Description

A wxgraphictriangle object represents a triangle that can be drawn on an appropriate container. The triangle has an optional border, border width and border color together with a fill color.

## Type Tags

wxgraphic

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| borderrgb | rgb | Gives the color of the triangle's border. |
| bordervisible | boolean | Specifies whether or not the border is visible. |
| borderwidth | integer | Gives the width of the border, in pixels. |
| container | type(wxgraphiccontainer) | Specifies the object to which this graphic belongs. |
| name | string | The name of the wxgraphictriangle object. |
| next | type(wxgraphic) | Specifies the next `wxgraphic` on the same container. |

| Property | Type | Description |
|---|---|---|
| point1 | fixedpoint | Gives the position of the starting point of the triangle. |
| point2 | fixedpoint | Gives the position of the second point of the triangle. |
| point3 | fixedpoint | Gives the position of the third point of the triangle. |
| rgb | rgb | Gives the fill color of the triangle. |
| type | type | Specifies the wxgraphictriangle type object. |
| visible | boolean | Specifies whether or not the content is visible. |

# Methods

## remove()

### Description

Removes the graphic from the container. If any object has a reference to the graphic, those references will no longer be valid. It is not safe to attempt to use a graphic after this method has been called.

### Prototype

*wxgraphictrianglevar*.remove ()

### Parameters

None

## setname()

### Description

Sets the name of the triangle control. The wxgraphictriangle object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxgraphictrianglevar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the graphic. |

## setnext()

### Description

Sets the position of the graphic in the z-order and tab order.

### Prototype

*wxgraphictrianglevar*.setnext ( type(wxgraphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(wxgraphic) | Calling the method with the value .nul sets a graphic to be the last one in the ring (the one before container.firstgraphic) and so has the highest z-order, and passing any other graphic as the argument puts the target graphic immediately below the one specified as the parameter. The visible stacking of graphics on a container is such that the earliest in the ring is the furthest back in the z-order. |

## setposition()

### Description

Sets the size and/or position of the graphic. The wxgraphictriangle object itself is returned, to allow multiple setting methods to be put into one expression. Not all parameters are available to all wxgraphic objects. It is recommended to name the parameters when using this method.

### Prototype

*wxgraphictrianglevar*.setposition ( point *point1*, point *point2*, point *point3*, point *midpoint*, integer *width*, integer *borderwidth* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point1 | The current value of the point1 property | point | The new position of the starting point of the graphic. |
| point2 | The current value of the point2 property | point | The new position of the second point of the graphic. |
| point3 | The current value of the point3 property | point | The new position of the third point of the graphic. |
| midpoint | The current value of the midpoint property | point | The new position of the midpoint of the graphic. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | The current value of the `width` property | integer | The new width of the graphic. |
| borderwidth | The current value of the `borderwidth` property | integer | The new width of the border of the graphic. |

# setrgb()

## Description

Sets the color and border color of the graphic. The wxgraphictriangle object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxgraphictrianglevar`*`.setrgb` ( integer *`rgb`*, integer *`borderrgb`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the rgb property | integer | The new color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| borderrgb | The current value of the borderrgb property | integer | The new border color of the graphic. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |

# setvisible()

## Description

Sets the visibility of the graphic. The wxgraphictriangle object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxgraphictrianglevar`*`.setvisible` ( boolean *`visible`*, boolean *`bordervisible`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | None | boolean | The desired visibility of the content of the graphic. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bordervisible | None | boolean | The desired visibility of the graphic's border. |

# wxmenu

## Description

A wxmenu object provides a host for the various menu items and sub menus. A menu can only be used in one menu bar at a time. It also cannot be used in a menu bar and as a right-mouse pop-up menu concurrently.

## Type Tags

None

## Object Value

## wxmenu.new()

### Description

Creates a new wxmenu object.

### Prototype

*wxmenu*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| firstitem | wxmenuitem | This property holds a reference to the first menu item attached to the menu. |
| menu | wxmenu | This property holds a reference to the menu where a submenu might be hosted. |

| Property | Type | Description |
|---|---|---|
| menubar | wxmenubar | This property holds a reference to the menubar where the menu is hosted. |
| type | type | Specifies the wxmenu type object. |

# Methods

## insert()

### Description

Inserts a menu item (or submenu) into the menu. The wxmenu object itself is returned, to allow multiple insertion methods to be put into one expression.

### Prototype

*wxmenuvar*.insert ( string *itemtype*, string *label*, integer *position*, wxmenu *submenu*, boolean *enabled*, boolean *checked*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| itemtype | None | string | The type of menu item being inserted. For a normal item, use the empty string (""), for a checkable item, use "checkable", for a radio item use "radio" (a radio item is one of a group of menu items, only one of which can be selected at a given time). For a separator, use "separator" and for a sub menu, use "submenu". |
| label | "" | string | The label for the menu item. |
| position | Current entry count plus one | integer | The place where the menu item is to be inserted. By default, it will be appended to the end. |
| submenu | .nul | wxmenu | Use this to insert a submenu. It can be included even when adding items as long as it is set to .nul. |
| enabled | .true | boolean | Indicates whether the menu is enabled or disabled. |
| checked | .false | boolean | Indicates whether the menuitem is checked or not. |
| name | .nul | string | This is the name of the menu item. It is not required, but can be useful if the menu item is to be manipulated using the member (!) operator. |

# wxmenu!

## Get

The member operator followed by the name of a menu item returns a reference to the wxmenuitem object. To access a sub menu associated with a menu item using only the menu object, the code would look like this: `m!file.submenu` where `m` is the menu object and `file` is the name that was assigned when the sub menu was inserted. If the name provided is not correct (including case-sensitivity), then an error will occur.

## Set

Attempting to assign a reference or a value to a wxmenu object using the ! operator is not supported.

# wxmenubar

# Description

A wxmenubar object provides the menu bar where the various menus are located that are associated with a given window. A menu bar object can only be associated with one window at a time. It can also exist independently of being hosted within a window.

# Type Tags

None

# Object Value

# wxmenubar.new()

## Description

Creates a new wxmenubar object.

## Prototype

*wxmenubar*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| firstentry | wxmenubarentry | This property holds a reference to the first menu attached to the menu bar. |
| type | type | Specifies the wxmenubar type object. |
| window | wxwindow | This property holds a reference to the window where the menu bar is hosted. |

# Methods

## delete()

### Description

Deletes a menu from the menu bar.

### Prototype

*wxmenubarvar*.delete ( wxmenu *menu* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| menu | None | wxmenu | The menu that is to be deleted from the menu bar. |

## deleteall()

### Description

Deletes all menues from the menu bar.

### Prototype

*wxmenubarvar*.deleteall ()

### Parameters

None

## insert()

### Description

Inserts a menu into the menu bar. The wxmenubar object itself is returned, to allow multiple insertions to be put into one expression.

### Prototype

*wxmenubarvar*.insert ( wxmenu *menu*, string *label*, integer *position*, boolean *enabled*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| menu | None | wxmenu | The wxmenu object that is being inserted. |
| label | "" | string | The label for the menu. |
| position | Current entry count plus one | integer | The place where the menu is to be inserted. By default, it will be appended to the end. |
| enabled | .true | boolean | Indicates whether the menu is enabled or disabled. |
| name | .nul | string | This is the name of the menu. It is not required, but can be useful if the menu is to be manipulated using the member (!) operator. |

## setwindow()

### Description

Creates the menu bar in the target window.

### Prototype

*wxmenubarvar*.setwindow ( wxwindow *window* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| window | None | wxwindow | The window that will host the menu bar. |

# wxmenubar!

## Get

The member operator followed by the name of a menu bar entry returns a reference to the wxmenubarentry object. To access a menu associated with a menu bar entry using only the menu bar object, the code would look like this: mb!file.menu where mb is the menu bar object and file is the name that was assigned when the menu was inserted. If the name provided is not correct (including case-sensitivity), then an error will occur.

## Set

Attempting to assign a reference or a value to a wxmenubar object using the ! operator is not supported.

# wxmenubarentry

## Description

A wxmenubarentry object represents an entry in the menu bar. The actual menu that is this entry is referenced in the menu property. There is no new() method for this type. It is created by the process of inserting a menu in a menu bar.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| enabled | boolean | This indicates whether the menu bar entry is enabled or not. |
| label | string | The label that is shown representing the menu bar entry on the menu bar. |
| menu | wxmenu | This property holds a reference to the menu that is the actual content of the menu bar entry. |
| name | string | The name of the menu bar entry, to allow it to be referenced using the member operator (!). |
| next | wxmenubarentry | This property holds a reference to the menu bar entry that follows this entry. All menu bar entries form a ring, beginning with the one assigned as the first entry of the menu bar. |
| type | type | Specifies the wxmenubarentry type object. |

## Methods

### setenabled()

#### Description

Enables a menu bar entry on a menu bar.

### Prototype

*wxmenubarentryvar*.setenabled ( boolean *enabled* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | Whether to enable or disable the menu. |

# wxmenuitem

## Description

A wxmenuitem object represents an entry on a menu and when selected will normally cause an event to occur in the program. Menu items can be of various types, including separators, check items, and radio items.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| checked | boolean | Specifies whether or not the menu item is checked or in the case of a radio item, if it is selected. |
| enabled | boolean | Specifies whether or not the menu item is enabled. |
| itemtype | string | This property contains the string that describes the type of menu item: separator, checkable, radio, or empty (for normal menu items). |
| label | string | The label that is shown representing the menu item on the menu. Using the & character before another character will make the following character an accelerator. Also, hotkey accelerators can be creat- |

| Property | Type | Description |
|---|---|---|
| | | ed by inserting a tab character (0x09)followed by the accelerator, such as: `Ctrl+P` for the print menu item. |
| menu | wxmenu | This property holds a reference to the menu that contains the menu item. |
| name | string | The name of the menu item, to allow it to be referenced using the member operator (!). |
| next | wxmenuitem | This property holds a reference to the next menu item. All menu items for a given menu are in a ring. |
| onselect | event | An event that is triggered every time the user selects the menu item. |
| submenu | wxmenu | This property holds a reference to a submenu (if any) that is associated with the menu item. |
| type | type | Specifies the wxmenuitem type object. |

# Methods

## setchecked()

### Description

Sets the state of the menu item, i.e. whether or not it has the appearance of being 'checked' or selected if it is a radio type of menu item. The wxmenuitem object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxmenuitemvar`*`.setchecked` ( boolean *`checked`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| checked | None | boolean | The new state for the menu item. If the argument is `.true` then the menu item is checked, if it is `.false` then the menu item is not checked. If this is a radio type of menu item, then this will select an item in a group of items if the argument is `.true`. |

## setenabled()

### Description

Sets the enabled state of the menu item. The wxmenuitem object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxmenuitemvar*.setenabled ( boolean *enabled* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the menu item. |

# wxprintbitmap

## Description

A wxprintbitmap object contains the image content that is used by a wxprintbitmapitem object when a page is output during printing. The bitmap can be associated with either the wxprintout object, the wxprintpagetemplate object, or the wxprintpage object. The choice is dependent on whether the item to be printed is the same throughout the printout, the same on every template, or is specific to a given page.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| bitmap | wxbitmap | The bitmap for the wxprintbitmap object. |
| container | type(wxprintdatacontainer) | Specifies the object that contains the ring of which the wxprintbitmap object is a member. This will be either a printout, a template, or a page. |
| name | string | The name of the wxprintbitmap object. |
| next | wxprintbitmap | Specifies the next wxprintbitmap object in the ring of printbitmaps associated either with a printout, template, or page. |

| Property | Type | Description |
|---|---|---|
| type | type | Specifies the wxprintbitmap type object. |

# Methods

## setname()

### Description

Sets the name of the wxprintbitmap object. The wxprintbitmap object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxprintbitmapvar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the printbitmap object.<br><br>**Note**<br>The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring. |

## setnext()

### Description

Sets the position of the wxprintbitmap in the ring of printbitmap objects.

### Prototype

*wxprintbitmapvar*.setnext ( wxprintbitmap *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | wxprintbitmap | Calling the method with the value `.nul` sets a printbitmap to be the last one in the ring (the one before container.firstbitmap) and passing any other printbitmap as the argument puts the target printbitmap immediately before the one specified as the parameter. |

# wxprintbitmapitem

## Description

A wxprintbitmapitem object contains the description of where a specific bitmap will be printed on a page as well as its characteristics: the horizontal and vertical alignment, scaling, etc. These are associated with a specific template which in turn is used by a page in a printout. The bitmap used can come from the printout, the template, or the actual page being printed.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| alignment | string | Shows the alignment of the bitmap item within the rectangle that represents the boundaries of the item. The alignment is for both horizontal and vertical alignment. The string can be made up of two comma separated string values. The valid values are: `"left"`, `"right"`, `"top"`, `"bottom"`, `"left,top"`, `"right,bottom"`, `"left,bottom"`, and `"right,top"` or the empty string (`""`). The order in which the dimensions are listed is undefined. If a dimension is not present, such as with `"left"` by itself, then the other dimension will be centered. The empty string |

| Property | Type | Description |
|---|---|---|
| | | would create an alignment both horizontally and vertically centered. |
| contentname | string | Contains the name of the wxprintbitmap object that contains the actual content of the bitmap item. If this property is equal to the empty string (`""`), then nothing will be printed. |
| contentsource | string | Contains the name of the source for the wxprintbitmap object, either `"printout"`, `"template"`, `"page"`, or the empty string (`""`). There is no `"system"` source for bitmaps. If this property is equal to the empty string, then nothing will be printed. |
| height | integer | Specifies the height of the bitmap item in micrometers. |
| left | integer | Specifies the position of the left edge of the bitmap item in micrometers. |
| name | string | The name of the wxprintbitmapitem object. |
| next | wxprintbitmapitem | Specifies the next wxprintbitmapitem object in the ring of bitmap items that make up the template. |
| scaling | string | Determines the type of scaling used on the bitmap. This can be one of: `"hfit"`, `"vfit"`, `"handvfit"`, `"preserveaspect"`, or the empty string. If no scaling is specified, then the image will be cropped to fit the container as needed, and based on the alignment settings. Otherwise the image will be stretched horizontally to fill the container, vertically to fill the container, or in both directions. |
| template | wxprintpagetemplate | Specifies the wxprintpagetemplate object to which this bitmap item belongs. |
| top | integer | Specifies the position of the top edge of the bitmap item in micrometers. |
| type | type | Specifies the wxprintbitmapitem type object. |
| undergraphics | boolean | Indicates whether the bitmap item is considered to be over the graphics items or under them in the z-order. It cannot be over some but under others. Either the bitmap item is over all graphics items, or it is under all of them. |
| width | integer | Specifies the width of the bitmap item in micrometers. |

# Methods

## getcontent()

### Description

This method retrieves the bitmap from the associated bitmap object. The *page* parameter is needed for items whose source is "page" or "printout" as a text or bitmap item doesn't know what pages or printouts it is on (in fact a template can be used for pages from many different printouts simultaneously).

### Prototype

*wxprintbitmapitemvar*.getcontent ( wxprintpage *page* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| page | .nul | wxprintpage | The specific page from which to retrieve the bitmap item. |

## setalignment()

### Description

This method assigns both the horizontal and vertical alignment for the bitmap item. The wxprintbitmapitem object itself is returned, to allow multiple setting methods to be put into one expression. The string can be made up of two comma separated string values. The valid values are: "left", "right", "top", "bottom", "left,top", "right,bottom", "left,bottom", and "right,top", and the empty string (""). The order in which the dimensions are provided is irrelevant as long as only maximum one value for each dimension is supplied. Providing two values for the same dimension of alignment is an error, such as "left,right". If an element is not supplied, it is assumed to be middle (in either dimension). Supplying an empty string would create an alignment both horizontally and vertically centered.

### Prototype

*wxprintbitmapitemvar*.setalignment ( string *alignment* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| alignment | The current setting of the alignment property | string | The alignment string for both horizontal and vertical alignment for the bitmap item. |

## setcontent()

### Description

Sets the name of the contentsource and the contentname properties for the bitmap item. The wxprintbitmapitem object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxprintbitmapitemvar*.setcontent ( string *contentsource*, string *contentname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| contentsource | The current value of the contentsource property | string | This must be one of: "printout", "template", "page", or the empty string (""). If the empty string is assigned, then nothing will be output. |
| contentname | The current value of the contentname property | string | This is the name of the wxprintbitmap object that is used to provide the content for this bitmap item when it is printed. If the empty string ("") is assigned, then nothing will be output. |

## setname()

### Description

Sets the name of the bitmap item. The wxprintbitmapitem object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxprintbitmapitemvar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the bitmap item.<br><br>**Note**<br><br>The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be as- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | signed to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring. |

# setnext()

## Description

Sets the position and z-order of the bitmap item in the ring of bitmap items. The first item in the ring is the lowest in the z-order.

## Prototype

*wxprintbitmapitemvar*.setnext ( wxprintbitmapitem *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | wxprintbitmapitem | Calling the method with the value .nul sets a bitmap item to be the last one in the ring (the one before wxprintpagetemplate.firstbitmapitem) and passing any other bitmap item as the argument puts the target bitmap item immediately before the one specified as the parameter. |

# setposition()

## Description

Sets the size and/or position of the bitmap item. The wxprintbitmapitem object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxprintbitmapitemvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the bitmap item on the template. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| top | The current value of the `top` property | integer | The new position of the top edge of the bitmap item on the template. |
| width | The current value of the `width` property | integer | The new width of the bitmap item on the template. |
| height | The current value of the `height` property | integer | The new height of the bitmap item on the template. |

# setscaling()

## Description

Sets the scaling method to be used for the bitmap in the bitmap item. The wxprintbitmapitem object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*`wxprintbitmapitemvar`*`.setscaling` ( string *`scaling`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| scaling | None | string | The scaling method to be used when adjusting the bitmap to its container. This is one of: `"hfit"`, `"vfit"`, `"handvfit"`, `"preserveaspect"`, or the empty string (`""`). If no scaling is specified, then the image will be cropped to fit the container as needed, and based on the alignment settings. Otherwise the image will be stretched horizontally to fill the container, vertically to fill the container, or in both directions. |

# setunders()

## Description

Sets whether the bitmap item appears above or below graphics.

## Prototype

*`wxprintbitmapitemvar`*`.setunders` ( boolean *`undergraphics`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| undergraphics | The current value of the undergraphics property | boolean | This decides whether bitmap items are rendered as being underneath all objects of type wxgraphic on the template. |

# wxprintout

## Description

A wxprintout object represents a document that is intended to be printed and which can also be sent to the print preview window. The printout object can be used to host strings and bitmaps. The printout also has a ring of pages that are to be printed. Positioning is absolute relative to the left top corner of the paper and is in micrometers (millionths of a meter). Each page contains the bitmaps and strings that are specific to that page, plus a reference to a template. The template can also contain strings and bitmaps, but these would only be part of the template if they do not change and are used on every page where the template is also used. The template also contains the graphical items, and the text and bitmap items. The text and bitmap items describe the physical representation of the text or bitmap, including position, alignment, scaling for bitmaps, and font, text color, and background color for text items. They also contain a reference to the host of the source of their string or bitmap, plus the name of that source. This design allows the layout of items on a page to be part of a template, but the data that changes is associated with the page, and information that is consistent with each template is associated with itself, while information that is used across the printout or in more than one template is associated with the printout object.

### Warning

On Windows, when printing from the print preview if a different printer is selected from the default printer and does not have the same paper size and resolution as the default printer, the resulting print out may not be correct in its proportions. Printing directly to printer, or printing always to the default printer from print preview or to the printer passed in the dialogdata parameter resolves the problem.

## Type Tags

wxprintdatacontainer

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this |

| Property | Type | Description |
|---|---|---|
| | | property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| firstbitmap | wxprintbitmap | Specifies the first wxprintbitmap object in the ring of bitmaps that contain the data associated with the printout. This is `.nul` if there are no bitmaps associated with the printout. |
| firstpage | wxprintpage | Specifies the first wxprintpage object in the ring of pages that make up the printout. This is `.nul` if there are no pages associated with the printout. |
| firststring | wxprintstring | Specifies the first wxprintstring object in the ring of strings that contain the data associated with the printout. This is `.nul` if there are no strings associated with the printout. |
| type | type | Specifies the wxprintout type object. |

# Methods

## addbitmap()

### Description

Adds a wxprintbitmap object to the ring of printbitmaps. An object of this type is added to hold the content for a wxprintbitmapitem object that is placed in a template. If the content is the same everywhere in the printout, then the bitmap should be added to the printout, if it is the same on each page that uses a specific template, then it should be added to the template, and if it is specific to a given page, then it should be added to the page.

#### Note

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

### Prototype

*wxprintoutvar*.`addbitmap` ( wxbitmap *`bitmap`*, string *`name`*, wxprintbitmap *`next`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bitmap | `.nul` | wxbitmap | The content for this wxprintbitmap object. |
| name | `.nul` | string | The name of the wxprintbitmap object. This value is used to link the object with the wxprintbitmapitem that will use it to pro- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | vide the content for an item on a printed page. |
| next | `.nul` | wxprintbitmap | This determines where in the ring of bitmaps the new bitmap will be placed. The new bitmap will be created so that the bitmap specified as *`next`* will be after the new bitmap in the ring. If the *`next`* bitmap is the current firstbitmap for the printout, then the new bitmap will become the new firstbitmap. If there are already some bitmaps in the printout and *`next`* is specified to be `.nul` then the next bitmap will be the current firstbitmap for the printout, but the new bitmap will not become a new firstbitmap (i.e. it becomes a 'last' bitmap). |

# addpage()

## Description

Adds a page to the printout.

### Note

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

## Prototype

*`wxprintoutvar`*`.addpage` ( wxprintpagetemplate *`template`*, string *`name`*, wxprintpage *`next`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| template | None | wxprintpagetemplate | The wxprintpagetemplate object that is used to specify the items that will be printed on the page. |
| name | None | string | The name of the page object. |
| next | `.nul` | wxprintpage | This determines where in the ring of pages the new page will be placed. The new page will be created so that the page specified as *`next`* will be after the new page in the ring. If the *`next`* page is |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | the current firstpage for the printout, then the new page will become the new firstpage. If there are already some pages in the printout and *next* is specified to be .nul then the next page will be the current firstpage for the printout, but the new page will not become a new firstpage (i.e. it becomes a 'last' page). |

# addstring()

## Description

Adds a wxprintstring object to the ring of printstrings. An object of this type is added to hold the content for a wxprinttextitem object that is placed in a template. If the content is the same everywhere in the printout, then the string should be added to the printout, if it is the same on each page that uses a specific template, then it should be added to the template, and if it is specific to a given page, then it should be added to the page.

### Note

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

## Prototype

*wxprintoutvar*.addstring ( string *string*, string *name*, wxprintstring *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| string | "" | string | The content for this wxprintstring object. |
| name | .nul | string | The name of the wxprintstring object. This value is used to link the object with the wxprinttextitem that will use it to provide the content for an item on a printed page. |
| next | .nul | wxprintstring | This determines where in the ring of strings the new string will be placed. The new string will be created so that the string specified as *next* will be after the new string in the ring. If the *next* string is the current firststring for the printout, then the new string |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | will become the new firststring. If there are already some strings in the printout and *next* is specified to be .nul then the next string will be the current firststring for the printout, but the new string will not become a new firststring (i.e. it becomes a 'last' string). |

# print()

## Description

Sends the printout to the printer.

## Prototype

*wxprintoutvar*.print ( string *title*, boolean *dialog*, string *dialogdata*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| title | None | string | The title of the printout. This is used on some operating systems or by some drivers for the title of the print job in the printer queue, or as the default file name when printing to file. |
| dialog | .true | boolean | Decides whether or not to show the printer dialog prior to printing. |
| dialogdata | None | string | The parameters used to preset the printer dialog. |
| error | .nul | integer | Specifies an object which is used to output any error code generated during printing of the wxprintout object. If error is not specified or is .nul then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution of the operation then the error code is output into that object. |

# startprintpreview()

## Description

Sends the printout to the print preview window, from where it can also be printed.

### Prototype

*wxprintoutvar*.startprintpreview ( string *title*, string *dialogdata*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| title | None | string | The title of the printout. This is used on some operating systems or by some drivers for the title of the print job in the printer queue, or as the default file name when printing to file. |
| dialogdata | None | string | The parameters used to preset the printer dialog. |
| error | .nul | integer | Specifies an object which is used to output any error code generated during showing or printing of the wxprintout object. If error is not specified or is .nul then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution of the operation then the error code is output into that object. |

# wxprintout!

## Get

If there is a page of the name following the member operator, then the page is returned, otherwise if there is a string of that name, then the string is returned, otherwise if there is a bitmap of that name the bitmap is returned, and if none of these are true an error will occur.

## Set

Attempting to assign a reference or a value to a wxprintout object using the ! operator is not supported.

# wxprintpage

# Description

A wxprintpage object represents a page in a wxprintout. Each page in the printout requires a template that establishes what will be printed and where. The page contains the strings and bitmaps that are specific to it and these are associated with the items on the template. In simple terms, the template contains the layout and the page contains the data. In the case of printing a two or more records that use the same layout each time, only one template will be created for the entire printout and it will be used on each page. In the case of a graphical report, there will very likely be a template for each page. If a programmer chooses to create a

very complex report, then it may have a template for the beginning, a template for many pages representing the content (where each page may be a record and the same layout), and another for the results page.

# Type Tags

wxprintdatacontainer

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| firstbitmap | wxprintbitmap | Specifies the first wxprintbitmap object in the ring of bitmaps that contain the data associated with this page. This is `.nul` if there are no bitmaps associated with the page. |
| firststring | wxprintstring | Specifies the first wxprintstring object in the ring of strings that contain the data associated with this page. This is `.nul` if there are no strings associated with the page. |
| name | string | The name of the wxprintpage object. |
| next | wxprintpage | Specifies the next wxprintpage object in the ring of pages that make up the printout. |
| printout | wxprintout | Specifies the wxprintout object to which this page belongs. |
| template | wxprintpagetemplate | Specifies the wxprintpagetemplate object that is used to output the data from this page. |
| type | type | Specifies the wxprintpage type object. |

# Methods

## addbitmap()

### Description

Adds a wxprintbitmap object to the ring of printbitmaps. An object of this type is added to hold the content for a wxprintbitmapitem object that is placed in a template. If the content is the same everywhere in the printout, then the bitmap should be added to the printout, if it is the same on each page that uses a specific template, then it should be added to the template, and if it is specific to a given page, then it should be added to the page.

### Note

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

## Prototype

*wxprintpagevar*.addbitmap ( wxbitmap *bitmap*, string *name*, wxprintbitmap *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bitmap | .nul | wxbitmap | The content for this wxprint-bitmap object. |
| name | .nul | string | The name of the wxprintbitmap object. This value is used to link the object with the wxprint-bitmapitem that will use it to provide the content for an item on a printed page. |
| next | .nul | wxprintbitmap | This determines where in the ring of bitmaps the new bitmap will be placed. The new bitmap will be created so that the bitmap specified as *next* will be after the new bitmap in the ring. If the *next* bitmap is the current firstbitmap for the page, then the new bitmap will become the new firstbitmap. If there are already some bitmaps in the page and *next* is specified to be .nul then the next bitmap will be the current firstbitmap for the page, but the new bitmap will not become a new firstbitmap (i.e. it becomes a 'last' bitmap). |

# addstring()

## Description

Adds a wxprintstring object to the ring of printstrings. An object of this type is added to hold the content for a wxprinttextitem object that is placed in a template. If the content is used on more than one template, then the string should be added to the printout, if it is the same on each page that uses a specific template, then it should be added to the template, and if it is specific to a given page, then it should be added to the page.

### Note

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case

of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

### Prototype

*wxprintpagevar*.addstring ( string *string*, string *name*, wxprintstring *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| string | "" | string | The content for this wxprintstring object. |
| name | .nul | string | The name of the wxprintstring object. This value is used to link the object with the wxprinttextitem that will use it to provide the content for an item on a printed page. |
| next | .nul | wxprintstring | This determines where in the ring of strings the new string will be placed. The new string will be created so that the string specified as *next* will be after the new string in the ring. If the *next* string is the current firststring for the page, then the new string will become the new firststring. If there are already some strings in the page and *next* is specified to be .nul then the next string will be the current firststring for the page, but the new string will not become a new firststring (i.e. it becomes a 'last' string). |

## setname()

### Description

Sets the name of the page. The wxprintpage object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxprintpagevar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the page.<br>  <br>**Note**<br>The name of any item: page, string, |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring. |

## setnext()

### Description

Sets the position of the page in the ring of pages.

### Prototype

*wxprintpagevar*.setnext ( wxprintpage *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | wxprintpage | Calling the method with the value .nul sets a page to be the last one in the ring (the one before wxprintoutput.firstpage) and passing any other page as the argument puts the target page immediately before the one specified as the parameter. |

# wxprintpage!

## Get

If there is a string of that name, then the string is returned, otherwise if there is a bitmap of that name the bitmap is returned, and if none of these are true an error will occur.

## Set

Attempting to assign a reference or a value to a wxprintpage object using the ! operator is not supported.

# wxprintpagetemplate

## Description

A wxprintpagetemplate object contains the description of where things will be printed, which font, which position, as well as the graphical objects that appear on any page that uses the template in a wxprintout. Each page in the printout requires a template that establishes what will be printed and where. The data for any given item of text or any image can be associated with the printout, the template or the individual page.

## Type Tags

wxgraphiccontainer, wxprintdatacontainer

## Object Value

## wxprintpagetemplate.new()

### Description

Creates a new wxprintpagetemplate object of the specified size and with the specified background color.

### Prototype

*wxprintpagetemplate*.new ( integer *width*, integer *height*, integer *backgroundrgb* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | None | integer | The width of the new page template, in micrometers. |
| height | None | integer | The height of the new page template, in micrometers. |
| backgroundrgb | White | integer | The background color of the new page template. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this |

| Property | Type | Description |
|---|---|---|
| | | property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Shows the background color of the page. |
| firstbitmap | wxprintbitmap | Specifies the first wxprintbitmap object in the ring of bitmaps that contain the data associated with this template. This is `.nul` if there are no bitmaps associated with the template. |
| firstbitmapitem | wxprintbitmapitem | Specifies the first wxprintbitmapitem object in the ring of bitmap items that contain the layout information for an item that is to be printed using this template. This is `.nul` if there are no bitmap items associated with the template. |
| firstgraphic | type(wxgraphic) | Gives the first graphic item on the template, or is `.nul` if there are no graphic items on the template. These include lines, triangles, rectangles, arcs, and ellipses. |
| firststring | wxprintstring | Specifies the first wxprintstring object in the ring of strings that contain the data associated with this template. This is `.nul` if there are no strings associated with the template. |
| firsttextitem | wxprinttextitem | Specifies the first wxprinttextitem object in the ring of text items that contain the layout information for an item that is to be printed using this template. This is `.nul` if there are no text items associated with the template. |
| height | integer | Specifies the height of the printed page in micrometers. |
| type | type | Specifies the wxprintpagetemplate type object. |
| width | integer | Specifies the width of the printed page in micrometers. |

# Methods

## addbitmap()

### Description

Adds a wxprintbitmap object to the ring of printbitmaps. An object of this type is added to hold the content for a wxprintbitmapitem object that is placed in a template. If the content is the same everywhere in the printout, then the bitmap should be added to the printout, if it is the same on each page that uses a specific template, then it should be added to the template, and if it is specific to a given page, then it should be added to the page.

**Note**

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case

of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

### Prototype

*wxprintpagetemplatevar*.addbitmap ( wxbitmap *bitmap*, string *name*, wxprintbitmap *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bitmap | .nul | wxbitmap | The content for this wxprint-bitmap object. |
| name | .nul | string | The name of the wxprintbitmap object. This value is used to link the object with the wxprint-bitmapitem that will use it to provide the content for an item on a printed page. |
| next | .nul | wxprintbitmap | This determines where in the ring of bitmaps the new bitmap will be placed. The new bitmap will be created so that the bitmap specified as *next* will be after the new bitmap in the ring. If the *next* bitmap is the current firstbitmap for the printout, then the new bitmap will become the new first-bitmap. If there are already some bitmaps in the printout and *next* is specified to be .nul then the next bitmap will be the current firstbitmap for the printout, but the new bitmap will not become a new firstbitmap (i.e. it becomes the 'last' bitmap). |

## addbitmapitem()

### Description

Creates a new wxprintbitmapitem object using the given arguments for many of the item's properties.

#### Note

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

### Prototype

*wxprintpagetemplatevar*.addbitmapitem ( integer *left*, integer *top*, integer *width*, integer *height*, string *alignment*, string *scaling*, boolean *undergraphics*, string *contentsource*, string *contentname*, string *name*, wxprintbitmapitem *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | None | integer | The position of the left side of the new bitmap item on the template, in micrometers. The item must be fully contained within the bounds of the template. |
| top | None | integer | The position of the top edge of the new bitmap item on the template, in micrometers. The item must be fully contained within the bounds of the template. |
| width | None | integer | The width of the new bitmap item on the template, in micrometers. The item must be fully contained within the bounds of the template. |
| height | None | integer | The height of the new bitmap item on the template, in micrometers. The item must be fully contained within the bounds of the template. |
| alignment | `""` | string | The alignment of the bitmap within the new bitmap item. The default is `.nul`, which will be left and top aligned. Valid values are: `"left"`, `"right"`, `"top"`, `"bottom"`, `"left,top"`, `"right,bottom"`, `"left,bottom"`, and `"right,top"`. To center something both horizontally and vertically, use the empty string. If a value for a dimension is not supplied, such as with `"left"` by itself, then the unspecified dimension will be centered. |
| scaling | `""` | string | This is the scaling method to use for the bitmap, and can be either: `"hfit"`, `"vfit"`, `"handvfit"`, `"preserveaspect"`, or `""`. |
| undergraphics | `.false` | boolean | Decides whether the bitmap item is in front or behind the wxgraphic items. |
| contentsource | `""` | string | This contains the source of the bitmap item, either "`printout`", "`template`", "`page`", or the empty string (`""`). If the value of this is the empty string then nothing will be printed. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| contentname | "" | string | This is the name of the item that is stored in the content source and which is used to provide the actual content for the item when printed. If the value of this is the empty string, then nothing will be printed. |
| name | .nul | string | The name of the new bitmap item. |
| next | .nul | wxprintbitmapitem | This determines where in the ring of bitmap items the new bitmap item will be placed. The new bitmap item will be created so that the bitmap item specified as *next* will be after the new bitmap item in the ring. If the *next* bitmap item is the current firstbitmapitem for the template, then the new bitmap item will become the new firstbitmapitem. If there are already some bitmap items in the template and *next* is specified to be .nul then the next bitmap item will be the current firstbitmapitem for the template, but the new bitmap item will not become a new firstbitmapitem (i.e. it becomes a 'last' bitmap item). Items are bottom to top in the z-order of the template, so that an item later in the ring covers an item located earlier in the ring. |

# addgraphic()

## Description

Creates a new type(wxgraphic) object of the specified type, using the given arguments for the graphic's properties. Graphics appear on top of text and bitmap items whose undergraphics property is set to .true, and under those where the undergraphics property is set to .false.

### Note

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

## Prototype

*wxprintpagetemplatevar*.addgraphic ( type *type*, point *point1*, point *point2*, point *point3*, point *midpoint*, integer *rgb*, integer *borderrgb*, integer *width*, integer *borderwidth*, boolean *visible*, boolean *bordervisible*, string *name*, type(wxgraphic) *next*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| type | None | type | The type of the new graphic. This must be of a type that is tagged as being a wxgraphic. |
| point1 | None | point | The starting point for the graphic. This can be created inline using: `point.new(10, 10)` for example. All graphic types use this property. |
| point2 | None | point | The second point of the graphic. This can be created inline using: `point.new(50, 100)` for example. |
| point3 | None | point | This is the third vertex point for a triangle. Only the triangle type makes use of this parameter. This can be created inline using: `point.new(80, 60)` for example. |
| midpoint | None | point | This point represents the center of an ellipse or arc. This can be created inline using: `point.new(10, 10)` for example. |
| rgb | White | integer | The fill color of the new wxgraphic object. In the case of the line object, this is the color of the line, in all others the color of the border lines is handled by the borderrgb parameter. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| borderrgb | Black | integer | The color of the lines representing the border of the new wxgraphic object. In the case of the line object, this parameter is invalid, use the rgb parameter instead. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| width | 1 | integer | This is the width in pixels of a line object. The minimum value is |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | 1. To make the line invisible, set its visibility to .false. |
| borderwidth | 1 | integer | This is the width in pixels of the border of a graphic object. The minimum value is 1. If no border is desired, set its visibility to .false. |
| visible | .true | boolean | Whether the content of the new graphic is visible or not. In the case of a line object, this decides if the line is visible, in the case of the other objects, this decides if the object is filled. If this portion is set to invisible, then objects behind it will be visible. To set any fillable object (triangle, rectangle, arc, or ellipse) to be completely invisible, both this and the bordervisible properties must be set to .false. |
| bordervisible | .true | boolean | Whether the border of the new graphic is visible or not. This parameter is invalid for line objects. If this portion is set to invisible, then no border will be shown. If the visible parameter is also set to .false, then the entrie graphic will be invisible. |
| name | .nul | string | The name of the graphic. This can be used together with the member operator (!) to access the graphic on a template via the wxprintpagetemplate object. |
| next | .nul | type(wxgraphic) | This determines where in the ring of graphics the new graphic will be placed. The new graphic will be created so that the graphic specified as next will be after the new graphic in the ring. If the next graphic is the current firstgraphic for the template, then the new graphic will become the new firstgraphic. If there are already some graphics on the template and next is specified to be .nul then the next graphic will be the current firstgraphic for the template, but the new graphic will not become a new first- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | `graphic` (i.e. it becomes the 'last' graphic). |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the `wxgraphic` object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `addgraphic` method returns `.nul`. |

# addstring()

## Description

Adds a wxprintstring object to the ring of printstrings. An object of this type is added to hold the content for a wxprinttextitem object that is placed in a template. If the content is used on more than one template, then the string should be added to the printout, if it is the same on each page that uses a specific template, then it should be added to the template, and if it is specific to a given page, then it should be added to the page.

### Note

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

## Prototype

*`wxprintpagetemplatevar`*`.addstring` ( string *`string`*, string *`name`*, wxprintstring *`next`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| string | `""` | string | The content for this wxprintstring object. |
| name | `.nul` | string | The name of the wxprintstring object. This value is used to link the object with the wxprinttextitem that will use it to provide the content for an item on a printed page. |
| next | `.nul` | wxprintstring | This determines where in the ring of strings the new string will be placed. The new string will be created so that the string specified as *`next`* will be after the new string in the ring. If the *`next`* string is the current firststring for |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | the printout, then the new string will become the new firststring. If there are already some strings in the printout and *next* is specified to be .nul then the next string will be the current firststring for the printout, but the new string will not become a new firststring (i.e. it becomes the 'last' string). |

# addtextitem()

## Description

Creates a new wxprinttextitem object using the given arguments for many of the item's properties.

### Note

The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring.

## Prototype

*wxprintpagetemplatevar*.addtextitem ( integer *left*, integer *top*, integer *width*, integer *height*, boolean *backgroundvisible*, integer *backgroundrgb*, integer *textrgb*, string *alignment*, wxfont *font*, boolean *undergraphics*, boolean *underbitmaps*, string *contentsource*, string *contentname*, string *name*, wxprinttextitem *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | None | integer | The position of the left side of the new text item on the template, in micrometers. The item must be fully contained within the bounds of the template. |
| top | None | integer | The position of the top edge of the new text item on the template, in micrometers. The item must be fully contained within the bounds of the template. |
| width | None | integer | The width of the new text item on the template, in micrometers. The item must be fully contained within the bounds of the template. |
| height | None | integer | The height of the new text item on the template, in micrometers. The item must be fully contained within the bounds of the template. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| backgroundvisible | .true | boolean | Establishes whether the background of the text item is visible or transparent. |
| backgroundrgb | The current background color of the template | integer | The background color of the new text item. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| textrgb | Black | integer | The color of the text in the new text item. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| alignment | `""` | string | The alignment of the text within the new text item. The default is `.nul`, which will be left and top aligned. Valid values are: `"left"`, `"right"`, `"top"`, `"bottom"`, `"left,top"`, `"right,bottom"`, `"left,bottom"`, and `"right,top"`. To center something both horizontally and vertically, use the empty string. If a value for a dimension is not supplied, such as with `"left"` by itself, then the unspecified dimension will be centered. |
| font | `.nul` | wxfont | A valid wxfont object that will be used as the font for the associated text. |
| undergraphics | `.false` | boolean | Decides whether the text item is in front or behind the wxgraphic items. |
| underbitmaps | `.false` | boolean | Decides whether the text item is in front or behind the bitmap items. This parameter determines whether the text item appears above or below the bitmap items that have the same undergraphics setting. Text items that are on top of the graphics are (of course) on top of all text and bitmap items that are under the graphics, and text items that are under the graphics are always under all objects that are on top of the graphics. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| contentsource | `""` | string | This contains the source of the text item, either `"printout"`, `"template"`, `"page"`, `"system"` or the empty string (`""`). If the value of this is the empty string, then nothing will be printed. |
| contentname | `""` | string | This is the name of the item that is stored in the content source and which is used to provide the actual content for the item when printed. If the source is `"system"` then the name must be `"pagenumber"`, which will supply the current page number in the printout. If the value of this is the empty string, then nothing will be printed. |
| name | `.nul` | string | The name of the new text item. |
| next | `.nul` | wxprinttextitem | This determines where in the ring of text items the new text item will be placed. The new text item will be created so that the text item specified as *`next`* will be after the new text item in the ring. If the *`next`* text item is the current firsttextitem for the template, then the new text item will become the new firsttextitem. If there are already some text items in the template and *`next`* is specified to be `.nul` then the next text item will be the current firsttextitem for the template, but the new text item will not become a new firsttextitem (i.e. it becomes a 'last' text item). Items are bottom to top in the z-order of the template, so that an item later in the ring covers an item located earlier in the ring. |

# setbackgroundrgb()

## Description

Sets the background color of the template. The wxprintpagetemplate object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*wxprintpagetemplatevar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the backgroundrgb property | integer | The new background color of the page. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| red | The current value of the backgroundrgb.red property | integer | The red component in the new background color of the page. This must be between 0 and 255 inclusive. |
| green | The current value of the backgroundrgb.green property | integer | The green component in the new background color of the page. This must be between 0 and 255 inclusive. |
| blue | The current value of the backgroundrgb.blue property | integer | The blue component in the new background color of the page. This must be between 0 and 255 inclusive. |

# wxprintpagetemplate!

## Get

If there is a graphic of the name following the member operator, then the graphic is returned, otherwise if there is a string of that name, then the string is returned, otherwise if there is a bitmap of that name the bitmap is returned, otherwise if there is a text item of that name the text item is returned, otherwise if there is a bitmap item of that name then the bitmap item is returned and if none of these are true an error will occur.

## Set

Attempting to assign a reference or a value to a wxprintpagetemplate object using the ! operator is not supported.

# wxprintstring

# Description

A wxprintstring object contains the content that is used by a wxprinttextitem object when a page is output during printing. The string can be associated with either the wxprintout object, the wxprintpagetemplate object, or the wxprintpage object. The choice is dependent on whether the item to be printed is the same throughout the printout, the same on every template, or is specific to a given page.

# Type Tags

None

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| container | type(wxprintdatacontainer) | Specifies the object that contains the ring of which the wxprintstring object is a member. This will be either a printout, a template, or a page. |
| name | string | The name of the wxprintstring object. |
| next | wxprintstring | Specifies the next wxprintstring object in the ring of strings associated either with a printout, template, or page. |
| string | string | The content of the wxprintstring object. |
| type | type | Specifies the wxprintstring type object. |

# Methods

## setname()

### Description

Sets the name of the wxprintstring object. The wxprintstring object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxprintstringvar*.setname ( string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the printstring object. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | **Note**<br><br>The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring. |

## setnext()

### Description

Sets the position of the wxprintstring in the ring of printstring objects.

### Prototype

*wxprintstringvar*.setnext ( wxprintstring *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | wxprintstring | Calling the method with the value `.nul` sets a printstring to be the last one in the ring (the one before container.firststring) and passing any other printstring as the argument puts the target printstring immediately before the one specified as the parameter. |

# wxprinttextitem

# Description

A wxprinttextitem object contains the description of where a specific string will be printed on a page as well as its characteristics: which font, which background color, which text color, horizontal and vertical alignment, etc. These are associated with a specific template which in turn is used by a page in a printout. The string used can come from the printout, the template, the actual page being printed, or be system generated, such as with `"pagenumber"`.

# Type Tags

None

# Object Value

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| alignment | string | Shows the alignment of the text item within the rectangle that represents the boundaries of the item. The alignment is for both horizontal and vertical alignment. The string can be made up of two comma separated string values. The valid values are: `"left"`, `"right"`, `"top"`, `"bottom"`, `"left,top"`, `"right,bottom"`, `"left,bottom"`, and `"right,top"`. If a dimension is not present, such as with `"left"` by itself, then the other dimension will be centered. |
| backgroundrgb | rgb | Shows the background color of the text item. |
| backgroundvisible | boolean | Indicates whether the background of the text item is visible (and will be rendered) or not. |
| contentname | string | Contains the name of the wxprintstring object that contains the actual content of the text item. If this is the empty string `""` then nothing will be printed. |
| contentsource | string | Contains the name of the source for the wxprintstring object, either `"printout"`, `"template"`, `"page"`, `"system"`, or the empty string (`""`), in which case nothing will be printed. |
| font | wxfont | This is a reference to the wxfont object that describes how the text in the text item is formatted. |
| height | integer | Specifies the height of the text item in micrometers. |
| left | integer | Specifies the position of the left edge of the text item in micrometers. |
| name | string | The name of the wxprinttextitem object. |
| next | wxprinttextitem | Specifies the next wxprinttextitem object in the ring of text items that make up the template. |
| template | wxprintpagetemplate | Specifies the wxprintpagetemplate object to which this text item belongs. |

| Property | Type | Description |
|---|---|---|
| textrgb | rgb | Shows the color of the text in the text item. |
| top | integer | Specifies the position of the top edge of the text item in micrometers. |
| type | type | Specifies the wxprinttextitem type object. |
| underbitmaps | boolean | Indicates whether the text item is in front of or behind the bitmap items. This property determines whether the text item appears above or below the bitmap items that have the same undergraphics setting. Text items that are on top of the graphics are (of course) on top of all text and bitmap items that are under the graphics, and text items that are under the graphics are always under all objects that are on top of the graphics. |
| undergraphics | boolean | Indicates whether the text item is considered to be over the graphics items or under them in the z-order. It cannot be over some but under others. Either the text item is over all graphics items, or it is under all of them. |
| width | integer | Specifies the width of the text item in micrometers. |

# Methods

## getcontent()

### Description

This method retrieves the value of the text item from the associated string object. The *`page`* parameter is needed for items whose source is `"system"`, `"page"` or `"printout"` as a text or bitmap item doesn't know what pages or printouts it is on and system items may differ on each page (in fact a template can be used for pages from many different printouts simultaneously). If the text item has a contentsource of `"system"` and a contentname of `"pagenumber"`, then `getcontent()` returns a text representation of the page number from the printout of the page passed as the parameter.

### Prototype

*`wxprinttextitemvar`*`.getcontent` ( wxprintpage *`page`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| page | `.nul` | wxprintpage | The specific page from which to retrieve the text item. |

## setalignment()

### Description

This method assigns both the horizontal and vertical alignment for the text item. The wxprinttextitem object itself is returned, to allow multiple setting methods to be put into one expression. The string can

be made up of two comma separated string values. The valid values are: `"left"`, `"right"`, `"top"`, `"bottom"`, `"left,top"`, `"right,bottom"`, `"left,bottom"`, and `"right,top"`. Providing two values for the same dimension of alignment is an error, such as `"left,right"`. If an element is not supplied, it is assumed to be middle (in either dimension). Supplying an empty string would create an alignment both horizontally and vertically centered.

### Prototype

*wxprinttextitemvar*.setalignment ( string *alignment* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| alignment | The current setting of the alignment property | string | The alignment string for both horizontal and vertical alignment for the text item. |

## setbackgroundrgb()

### Description

Sets the background color of the text item. The wxprinttextitem object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

### Prototype

*wxprinttextitemvar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `backgroundrgb` property | integer | The new background color of the text item. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the text item. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the text item. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the text item. This must be between 0 and 255 inclusive. |

# setbackgroundvisible()

## Description

Sets the visibility of the text item background on or off. The wxprinttextitem object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxprinttextitemvar*.setbackgroundvisible ( boolean *backgroundvisible* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| backgroundvisible | The current value of the backgroundvisible property | boolean | The desired visibility of the background of the text item. |

# setcontent()

## Description

Sets the name of the contentsource and the contentname properties for the text item. The wxprinttextitem object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxprinttextitemvar*.setcontent ( string *contentsource*, string *contentname* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| contentsource | The current value of the contentsource property | string | This must be one of: "printout", "template", "page", "system", or the empty string (""), in which case nothing will be printed. |
| contentname | The current value of the contentname property | string | This is the name of the wxprintstring object that is used to provide the content for this text item when it is printed. If the "system" contentsource is used, then currently the only valid contentname is "pagenumber". This allows the page number to be retrieved for the current page in the printout. If the empty string ("") is assigned to this parameter then nothing will be printed. |

# setfont()

## Description

Sets the font to be used for the text in the text item. The wxprinttextitem object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxprinttextitemvar*.setfont ( wxfont *font* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | The current value of the font property | wxfont | The wxfont object that will be used to format the text. |

# setname()

## Description

Sets the name of the text item. The wxprinttextitem object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxprinttextitemvar*.setname ( string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | The new name for the text item.<br>  <br>**Note**<br>The name of any item: page, string, bitmap, etc. that can be accessed using the member operator must be unique across all rings associated with an object. For example in the case of a printout object, the same name cannot be assigned to a page and also to a string or a bitmap. The same name can also not be given to two items in the same ring. |

# setnext()

## Description

Sets the position and z-order of the text item in the ring of text items. The first item in the ring is the lowest in the z-order.

## Prototype

*wxprinttextitemvar*.setnext ( wxprinttextitem *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | wxprinttextitem | Calling the method with the value .nul sets a text item to be the last one in the ring (the one before wxprintpagetemplate.firsttextitem) and passing any other text item as the argument puts the target text item immediately before the one specified as the parameter. |

# setposition()

## Description

Sets the size and/or position of the text item. The wxprinttextitem object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxprinttextitemvar*.setposition ( integer *left*, integer *top*, integer *width*, integer *height* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the text item on the template. |
| top | The current value of the top property | integer | The new position of the top edge of the text item on the template. |
| width | The current value of the width property | integer | The new width of the text item on the template. |
| height | The current value of the height property | integer | The new height of the text item on the template. |

# settextrgb()

## Description

Sets the color of the text for the text item. The wxprinttextitem object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

## Prototype

*`wxprinttextitemvar`*`.settextrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `textrgb` property | integer | The new color of the text for the text item. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the `textrgb.red` property | integer | The red component in the new text color of the text item. This must be between 0 and 255 inclusive. |
| green | The current value of the `textrgb.green` property | integer | The green component in the new text color of the text item. This must be between 0 and 255 inclusive. |
| blue | The current value of the `text.blue` property | integer | The blue component in the new text color of the text item. This must be between 0 and 255 inclusive. |

# setunders()

## Description

Sets whether the text item appears above or below graphics and bitmaps.

## Prototype

*`wxprinttextitemvar`*`.setunders` ( boolean *`undergraphics`*, boolean *`underbitmaps`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| undergraphics | The current value of the undergraphics property | boolean | This decides whether text items are rendered as being underneath all objects of type wxgraphic on the template. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| underbitmaps | The current value of the underbitmaps property | boolean | Decides whether the text item is in front or behind the bitmap items. This parameter determines whether the text item appears above or below the bitmap items that have the same undergraphics setting. Text items that are on top of the graphics are (of course) on top of all text and bitmap items that are under the graphics, and text items that are under the graphics are always under all objects that are on top of the graphics. |

# wxstatusbar

## Description

A wxstatusbar object provides a panel at the bottom of the window where text messages can be displayed. It can only be associated with one window at a time. It can also exist independently of being hosted within a window.

## Type Tags

None

## Object Value

## wxstatusbar.new()

### Description

Creates a new wxstatusbar object.

### Prototype

*wxstatusbar*.new ( integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the wxstatusbar object. If `error` is not specified or is `.nul` then any error which occurs during object creation will |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `new` methods returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| text | string | This property contains the text that is currently displayed in the status bar. |
| type | type | Specifies the wxtoolbar type object. |
| window | wxwindow | This property holds a reference to the window where the status bar is hosted. |

# Methods

## settext()

### Description

Sets the text in the status bar. The wxstatusbar object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*`wxstatusbarvar`*`.settext` ( string *`text`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | The new text to set in the status bar. |

## setwindow()

### Description

Creates the status bar in the target window.

### Prototype

*wxstatusbarvar*.setwindow ( wxwindow *window* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| window | None | wxwindow | The window that will host the status bar. |

# wxtool

## Description

A wxtool object represents an entry on a tool bar and when selected will normally cause an event to occur in the program. Tools are buttons that have a bitmap and optionally a disabled bitmap.

## Type Tags

None

## Object Value

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| bitmap | wxbitmap | The bitmap that is shown representing the tool on the tool bar. |
| disabledbitmap | wxbitmap | The bitmap that is shown representing the tool on the tool bar if the tool is disabled. |
| enabled | boolean | Specifies whether or not the tool is enabled. |
| name | string | The name of the tool, to allow it to be referenced using the member operator (!). |
| next | wxtool | This property holds a reference to the next tool. All tools for a given tool bar are in a ring. |
| onclick | event | An event that is triggered every time the user clicks on the tool. |

| Property | Type | Description |
|---|---|---|
| toolbar | wxtoolbar | This property holds a reference to the tool bar that contains the tool. |
| tooltip | string | Contains the test that is displayed as a tooltip for the tool. |
| type | type | Specifies the wxtool type object. |

# Methods

## setenabled()

### Description

Sets the enabled state of the tool. The wxtool object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxtoolvar*.setenabled ( boolean *enabled* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enabled | None | boolean | The desired enabled state for the tool. |

## settooltip()

### Description

Sets the text for the tooltip associated with the tool. The wxtool object itself is returned, to allow multiple setting methods to be put into one expression.

### Prototype

*wxtoolvar*.settooltip ( string *tooltip* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tooltip | None | string | The new text to set for the tooltip. |

# wxtoolbar

# Description

A wxtoolbar object provides the tool (icon) bar where the various tools (icons) are located that are associated with a given window. A tool bar object can only be associated with one window at a time. It can also

exist independently of being hosted within a window. The toolbar can also host one or more forms as if they were tools, where each form can contain one or more form controls.

# Type Tags

None

# Object Value

# wxtoolbar.new()

## Description

Creates a new wxtoolbar object. The size of the toolbar tools can be controlled by setting the desired size in pixels in the `new()` method.

## Prototype

*`wxtoolbar`*.`new` ( integer *`toolwidth`*, integer *`toolheight`*, rgb *`backgroundrgb`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| toolwidth | 16 | integer | The width of the tool bitmaps. |
| toolheight | 15 | integer | The height of the tool bitmaps. |
| backgroundrgb | The system default color | rgb | The background color for the toolbar. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the wxtoolbar object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `new` methods returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| backgroundrgb | rgb | Specifies the background color of the tool bar. |
| firsttool | wxtool | This property holds a reference to the first tool attached to the tool bar. |
| toolheight | integer | Specifies the height of the wxtool objects. |
| toolwidth | integer | Specifies the width of the wxtool objects. |
| type | type | Specifies the wxtoolbar type object. |
| window | wxwindow | This property holds a reference to the window where the tool bar is hosted. |

# Methods

## insert()

### Description

Inserts a tool into the tool bar. The wxtoolbar object itself is returned, to allow multiple insertions to be put into one expression.

### Prototype

*wxtoolbarvar*.insert ( wxbitmap *bitmap*, wxbitmap *disabledbitmap*, integer *position*, boolean *enabled*, string *tooltip*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bitmap | None | wxbitmap | The wxbitmap object that will be shown on the tool. |
| disabledbitmap | .nul | wxbitmap | The wxbitmap object that will be shown on the tool if it is disabled. If this is not supplied it will not be evident that the tool is disabled. |
| position | Current entry count plus one | integer | The place where the tool is to be inserted. By default, it will be appended to the end. |
| enabled | .true | boolean | Indicates whether the tool is enabled or disabled. |
| tooltip | .nul | string | The text to be used as the tooltip for the control. |
| name | .nul | string | This is the name of the tool. It is not required, but can be useful if |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | the tool is to be manipulated using the member (!) operator. |

# insertform()

## Description

Inserts a form into the tool bar. The wxtoolbar object itself is returned, to allow multiple insertions to be put into one expression.

## Prototype

*wxtoolbarvar*.insertform ( wxform *form*, integer *position*, string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| form | None | wxform | The wxform object that will be hosted in the tool bar. |
| position | Current entry count plus one | integer | The place where the form is to be inserted. By default, it will be appended to the end. |
| name | .nul | string | This is the name of the form tool. It is not required, but it is the only to access the form from the tool bar object; via the member (!) operator and this name. |

# setbackgroundrgb()

## Description

Sets the background color of the tool bar. The wxtoolbar object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the rgb argument and one or more of the red, green or blue arguments.

## Prototype

*wxtoolbarvar*.setbackgroundrgb ( integer *rgb*, integer *red*, integer *green*, integer *blue* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the backgroundrgb property | integer | The new background color of the tool bar. It is inadvisable to specify any value for this argument that is not the value of an existing rgb object. |
| red | The current value of the | integer | The red component in the new background color of the tool bar. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | `backgroundrgb.red` property | | This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the tool bar. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the tool bar. This must be between 0 and 255 inclusive. |

## setwindow()

### Description

Creates the tool bar in the target window.

### Prototype

*`wxtoolbarvar`*`.setwindow` ( wxwindow *`window`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| window | None | wxwindow | The window that will host the tool bar. |

# wxtoolbar[]

## Get

### Subscripts

A numeric value giving the 1-based index of an item in the list.

### Description

Retrieves the specified item from the wxtoolbar object. The variable that is receiving the object should be declared as type(*), since if not and the tooolbar contains embedded wxform objects, an error will occur when the object is assigned. By checking the type property the type of the object that was returned can be tested. If there is no item at that index position, then `.nul` will be returned. Once the value `.nul` has been returned, there are no more items remaining in the toolbar. This is the only way of reliably retrieving all of the elements of a toolbar, since embedded forms only show up in this way, they are not part of the ring of wxtool objects.

## Set

Attempting to set the value is not supported.

## Set Reference

Attempting to set a reference to an object is not supported.

# wxtoolbar!

## Get

The member operator followed by the name of a form tool returns a reference to the wxform object. If the name provided is not correct (including case-sensitivity), then an error will occur.

## Set

Attempting to assign a reference or a value to a wxtoolbar object using the ! operator is not supported.

# wxwindow

# Description

A wxwindow object is a window on the screen, which the programmer has control over and is used for interacting with the user. It can either be a top-level frame window or a child window. Child windows are primarily useful for creating various panels in a frame window. Child windows cannot have a caption bar, so can't have system menus, min buttons, max buttons or close (visibility) buttons. It is an error to pass values to wxwindow.new() that try to create a child window with a caption bar, a system menu, a min button, a max button, a close button, that is minimized, or that is maximized. Child windows cannot have a menubar, toolbar or statusbar, so can't be passed to the `setwindow()` method of any of these. These are limitations of wxWidgets.

### Note

Child windows are not held in a ring hanging off the parent. If you don't hang onto the window reference yourself then the window vanishes.

# Type Tags

wxcontainer, wxdialogparent, wxwindowparent

# Object Value

# wxwindow.new()

## Description

Creates a new wxwindow object with the required properties.

## Prototype

*wxwindow*.new ( integer *left*, integer *top*, integer *outerwidth*, integer *outerheight*, integer *innerwidth*, integer *innerheight*, boolean *visible*, boolean *minimized*, boolean *maximized*, type(*) *parent*, boolean *captionbar*, string *captiontext*, boolean *menubutton*, boolean *minbutton*, boolean *maxbutton*, boolean *visbutton*, wxbitmap *icon*, boolean *vscrollbar*, boolean *hscrollbar*, string *border*, integer *backgroundrgb*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | None | integer | The position of the left side of the window either relative to its parent window, if any, or on the screen. |
| top | None | integer | The position of the top edge of the window either relative to its parent window, if any, or on the screen. |
| outerwidth | None | integer | The width of the entire window, including borders, titles, and other parts which are not used by window content or child windows. Either `outerwidth` or `innerwidth` must be specified, but it is an error to specify both. |
| outerheight | None | integer | The height of the entire window, including borders, titles, and other parts which are not used by window content or child windows. Either `outerheight` or `innerheight` must be specified, but it is an error to specify both. |
| innerwidth | None | integer | The width of the useable, inner part of the window, not including borders, titles, etc. Either `outerwidth` or `innerwidth` must be specified, but it is an error to specify both. |
| innerheight | None | integer | The height of the useable, inner part of the window, not including borders, titles, etc. Either `outerheight` or `innerheight` must be specified, but it is an error to specify both. |
| visible | `.true` | boolean | Determines whether or not the window will be visible on the screen. |
| minimized | `.false` | boolean | Determines whether or not the window is to be displayed in a minimized state (also known as iconized). It is an error for both `minimized` and `maximized` arguments to be specified as `.true`. If both `minimized` and `maximized` arguments are specified as `.false` then the win- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | dow will be created in a restored or 'normal' state. |
| maximized | `.false` | boolean | Determines whether or not the window is to be displayed in a maximized state, i.e. expanded to cover the whole available display. It is an error for both `minimized` and `maximized` arguments to be specified as `.true`. If both `minimized` and `maximized` arguments are specified as `.false` then the window will be created in a restored or 'normal' state. |
| parent | None | type(*) | This contains the reference to a wxwindow object that is the parent of a child window. It is an error to pass values to wxwindow.new() that try to create a child window with a caption bar, a system menu, a min button, a max button, a close button, that is minimized, or that is maximized. |
| captionbar | `.true` | boolean | Determines whether or not the window is to have a caption bar, which is used to display text and to allow the user to move the window by dragging it using the mouse. If a window does not have a caption bar, then it cannot have a menu button, min button, max button or vis button, so these must not be specified as arguments with the value of `.true` (but they can be passed as `.false` or omitted). |
| captiontext | "" | string | The text which is to appear in any caption bar the window has. |
| menubutton | `.true` | boolean | Determines whether or not the window is to have a menu button, which allows the user to access a menu giving the ability to manage the window, for example to minimize or maximize it. A window must have a caption bar if it to have a menu button, so the `captionbar` argument must not be passed as `.false` (though it can be passed as `.true` or omitted). |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| minbutton | `.true` | boolean | Determines whether or not the window is to have a minimize button, which allows the user to minimize the window. A window must have a caption bar if it to have a min button, so the `captionbar` argument must not be passed as `.false` (though it can be passed as `.true` or omitted). |
| maxbutton | `.true` | boolean | Determines whether or not the window is to have a maximize button, which allows the user to maximize the window. A window must have a caption bar if it to have a max button, so the `captionbar` argument must not be passed as `.false` (though it can be passed as `.true` or omitted). |
| visbutton | `.true` | boolean | Determines whether or not the window is to have a visibilty button, which allows the user to make the window invisible, which is often considered to be 'closing' it. A window must have a caption bar if it to have a vis button, so the `captionbar` argument must not be passed as `.false` (though it can be passed as `.true` or omitted). |
| icon | None | wxbitmap | The wxbitmap to use as the icon for the new window. It is recommended to use an appropriately sized image for this purpose (16 x 16 pixels). |
| vscrollbar | `.false` | boolean | Determines whether the window has a vertical scroll bar to allow the user to scroll the content of the window up or down. If this is `.false` then the window never has a vertical scroll bar. If this is `.true` then the window has a vertical scroll bar if the size of the content exceeds the size of the inner part of the window in the vertical direction, otherwise it does not have a vertical scroll bar. |
| hscrollbar | `.false` | boolean | Determines whether the window has a horizontal scroll bar to allow the user to scroll the content of the window right or left. If this |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | is `.false` then the window never has a horizontal scroll bar. If this is `.true` then the window has a horizontal scroll bar if the size of the content exceeds the size of the inner part of the window in the horizontal direction, otherwise it does not have a horizontal scroll bar. |
| border | "sizeable" | string | Determines what type of border with which the window is created and in association with that, whether the window is sizeable or not. The options are `"simple"`, `"sizeable"`, `"none"`, `"sunken"`, and `"raised"` with `"sizeable"` being the default value for top level windows and `"simple"` the default for child windows. |
| backgroundrgb | The system dependent default window background color | integer | The background color of the new window. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the wxwindow object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `new` methods returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |

| Property | Type | Description |
|---|---|---|
| backgroundrgb | rgb | Shows the color of the inner part of the window when no content is visible. |
| border | string | This setting determines the type of window border that will be used and that will in turn decide whether the window can be resized or not. For a resizeable window use "sizeable", for a window that can't be resized, use the value "simple". |
| captionbar | boolean | Indicates whether or not the window has a caption bar. The caption bar is used to display a title for the window, and is also used to move the window by dragging with the mouse. |
| captiontext | string | Contains the text which is displayed in a window's caption bar. A window which doesn't have a caption bar can still have caption text. This is the text which will be displayed if the window is changed so that it does have a caption bar. |
| content | type(wxcontent) | The current content of the window. |
| hscrollbar | boolean | Indicates whether the window has a horizontal scroll bar to allow the user to scroll the content of the window left or right. If this is set to `.false` then the window never has a horizontal scroll bar. If this is set to `.false` then the window has a horizontal scroll bar if the size of the content exceeds the size of the inner part of the window in the horizontal direction, otherwise it does not have a horizontal scroll bar. |
| icon | wxbitmap | The wxbitmap object that is used as the source of the icon for the window. |
| innerheight | integer | Gives the height of the useable, inner part of the window, not including borders, titles, etc. |
| innerwidth | integer | Gives the width of the useable, inner part of the window, not including borders, titles, etc. |
| left | integer | Gives the x-position of the left hand side of the window on the screen. |
| maxbutton | boolean | Indicates whether or not the window has a button in the caption bar with which the user can maximize the window. Not all combinations of `.true` and `.false` settings for `menubutton`, `minbutton`, `maxbutton` and `visbutton` are possible on all platforms. |
| maximized | boolean | Indicates whether or not the window is currently maximized. A window cannot be both minimized and maximized at the same time. |
| menubar | wxmenubar | A reference to the menu bar object (if there is one) for the window. |
| menubutton | boolean | Indicates whether or not the window has a button in the caption bar which provides the user with access |

| Property | Type | Description |
|---|---|---|
| | | to a menu that allows the window to be managed, for example to be minimized or maximized. Not all combinations of `.true` and `.false` settings for `menubutton`, `minbutton`, `maxbutton` and `visbutton` are possible on all platforms. |
| minbutton | boolean | Indicates whether or not the window has a button in the caption bar with which the user can minimize the window. Not all combinations of `.true` and `.false` settings for `menubutton`, `minbutton`, `maxbutton` and `visbutton` are possible on all platforms. |
| minimized | boolean | Indicates whether or not the window is currently minimized. A window cannot be both minimized and maximized at the same time. |
| onmove | event | An event which is triggered when the position of the window is changed by the user. |
| onsize | event | An event which is triggered when the size of the window is changed by the user. |
| onstatechange | event | An event which is triggered when the user changes the state of the window, for example by minimizing or maximizing the window. |
| onvisibility-change | event | An event which is triggered when the user changes the visibility the window, for example by 'closing' the window by clicking the visibility button in the caption bar. |
| outerheight | integer | Gives the height of the entire window, including borders, titles, and other parts which are not used by window content or child windows. |
| outerwidth | integer | Gives the width of the entire window, including borders, titles, and other parts which are not used by window content or child windows. |
| parent | type(*) | This contains a reference to the parent window of a child window. |
| statusbar | wxstatusbar | A reference to the status bar object (if there is one) for the window. |
| toolbar | wxtoolbar | A reference to the tool bar object (if there is one) for the window. |
| top | integer | Gives the y-position of the top edge of the window on the screen. |
| type | type | Specifies the wxwindow type object. |
| visbutton | boolean | Indicates whether or not the window has a button in the caption bar with which the user can make the window invisible. This is normally thought of as 'closing' the window. Not all combinations of `.true` and `.false` settings for `menubutton`, `minbutton`, `maxbutton` and `visbutton` are possible on all platforms. |

| Property | Type | Description |
|---|---|---|
| visible | boolean | Indicates whether or not the window is currently visible on the screen. |
| vscrollbar | boolean | Indicates whether the window has a vertical scroll bar to allow the user to scroll the content of the window up or down. If this is set to `.false` then the window never has a vertical scroll bar. If this is set to `.false` then the window has a vertical scroll bar if the size of the content exceeds the size of the inner part of the window in the vertical direction, otherwise it does not have a vertical scroll bar. |

# Methods

## getscrollpositions()

### Description

Retrieves the horizontal and vertical scroll position information for the window.

### Prototype

*wxwindowvar*.getscrollpositions ( integer *vposition*, integer *vrange*, integer *hposition*, integer *hrange*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| vposition | `.nul` | integer | Specifies an object to hold the vertical scroll position. This must be a pre-initialized integer object. |
| vrange | `.nul` | integer | Specifies an object to hold the range of the vertical scroll bar. This must be a pre-initialized integer object. |
| hposition | `.nul` | integer | Specifies an object to hold the horizontal scroll position. This must be a pre-initialized integer object. |
| hrange | `.nul` | integer | Specifies an object to hold the range of the horizontal scroll bar. This must be a pre-initialized integer object. |
| error | `.nul` | integer | Specifies an object that is used to output any error code generated during execution of the function. If `error` is not specified or is `.nul` then any error which occurs during execution will halt the program. If an error object is specified and an error occurs dur- |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | ing execution then the error code is output into that object. |

# setbackgroundrgb()

## Description

Sets the background color of the window. The wxwindow object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to specify both the `rgb` argument and one or more of the `red`, `green` or `blue` arguments.

## Prototype

*wxwindowvar*.`setbackgroundrgb` ( integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | The current value of the `backgroundrgb` property | integer | The new background color of the window. It is inadvisable to specify any value for this argument which is not the value of an existing rgb object. |
| red | The current value of the `backgroundrgb.red` property | integer | The red component in the new background color of the window. This must be between 0 and 255 inclusive. |
| green | The current value of the `backgroundrgb.green` property | integer | The green component in the new background color of the window. This must be between 0 and 255 inclusive. |
| blue | The current value of the `backgroundrgb.blue` property | integer | The blue component in the new background color of the window. This must be between 0 and 255 inclusive. |

# setcaptiontext()

## Description

Sets the text in the caption bar, if any, of the window. The wxwindow object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxwindowvar*.`setcaptiontext` ( string *`captiontext`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| captiontext | None | string | The text which is to appear in any caption bar the window has. |

# seticon()

## Description

Sets the icon for the window. This must be a window that has a caption bar, otherwise an error is returned in the *error* parameter. The wxwindow object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxwindowvar*.seticon ( wxbitmap *icon*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| icon | None | wxbitmap | The wxbitmap that should be used as the window icon. This should be a bitmap that is 16x16 pixels normally. |
| error | .nul | integer | Specifies an object that is used to output any error code generated during execution of the function. If error is not specified or is .nul then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object. |

# setposition()

## Description

Sets the size and/or position of a window. The wxwindow object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxwindowvar*.setposition ( integer *left*, integer *top*, integer *outerwidth*, integer *outerheight*, integer *innerwidth*, integer *innerheight* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| left | The current value of the left property | integer | The new position of the left side of the window either relative to its parent window, if any, or on the screen. |
| top | The current value of the top property | integer | The new position of the top edge of the window either relative to its parent window, if any, or on the screen. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| outerwidth | The current value of the `outerwidth` property | integer | The new width of the entire window, including borders, titles, and other parts which are not used by window content or child windows. It is an error to specify both `outerwidth` and `innerwidth` arguments. |
| outerheight | The current value of the `outerheight` property | integer | The new height of the entire window, including borders, titles, and other parts which are not used by window content or child windows. It is an error to specify both `outerheight` and `innerheight` arguments. |
| innerwidth | The current value of the `innerwidth` property | integer | The new width of the useable, inner part of the window, not including borders, titles, etc. It is an error to specify both `outerwidth` and `innerwidth` arguments. |
| innerheight | The current value of the `innerheight` property | integer | The new height of the useable, inner part of the window, not including borders, titles, etc. It is an error to specify both `outerheight` and `innerheight` arguments. |

# setscrollbars()

## Description

Sets the visibility of the scroll bars for the window. The wxwindow object itself is returned, to allow multiple setting methods to be put into one expression.

## Prototype

*wxwindowvar*.`setscrollbars` ( boolean *`vscrollbar`*, boolean *`hscrollbar`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| vscrollbar | The current value of the `vscrollbar` property | boolean | Determines whether or not the vertical scroll bar is visible on the screen. |
| hscrollbar | The current value of the `hscrollbar` property | boolean | Determines whether or not the horizontal scroll bar is visible on the screen. |

# setscrollpositions()

## Description

Sets the horizontal and/or vertical scroll position of the content of the window.

## Prototype

*wxwindowvar*.setscrollpositions ( integer *vposition*, integer *hposition* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| vposition | Current vertical position | integer | Specifies the vertical position to where the content should be moved. |
| hposition | Current horizontal position | integer | Specifies the horizontal position to where the content should be moved. |

# setstate()

## Description

Sets the 'state' of the window. The 'state' is considered to be those things which determine the general accessibility of the window. The wxwindow object itself is returned, to allow multiple setting methods to be put into one expression. It is an error to call this method on a child window with values that will try to minimize or maximize it.

## Prototype

*wxwindowvar*.setstate ( boolean *visible*, boolean *minimized*, boolean *maximized* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| visible | The current value of the `visible` property | boolean | Determines whether or not the window is visible on the screen. |
| minimized | The current value of the `minimized` property | boolean | Determines whether or not the window is to be displayed in a minimized state (also known as iconized). It is an error for both `minimized` and `maximized` arguments to be specified as `.true`. If both `minimized` and `maximized` arguments are specified as `.false` then the window will be restored, or returned to a 'normal' state. |
| maximized | The current value of the `max-` | boolean | Determines whether or not the window is to be displayed in a maximized state, i.e. expanded to |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | `imized` property | | cover the whole available display. It is an error for both minimized and maximized arguments to be specified as `.true`. If both `minimized` and `maximized` arguments are specified as `.false` then the window will be restored, or returned to a 'normal' state. |

# wxbreak()

## Description

Breaks out of all pending calls to the `wxprocess()` function. Code continues execution in that thread on the line following the call to `wxprocess()`.

## Prototype

`wxbreak` ()

## Parameters

None

# wxclipboardgetdata()

## Description

This function retrieves from the clipboard either text, a bitmap, or both (if both are present). Current limitations in wxWidgets prevent the retrieval of non-Unicode text in Unicode applications. This does not present a problem in Windows XP or later (NT and 2000 will require testing), since the operating system automatically puts a Unicode copy up as well. In Windows 9x and Me, this means that text copied from non-Unicode-enabled programs cannot be retrieved using this call. This will hopefully change in the future.

## Prototype

`wxclipboardgetdata` ( string *`text`*, wxbitmap *`bitmap`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | `.nul` | string | The string object that will contain any text retrieved from the clipboard. This must be a pre-initialized string object! |
| bitmap | `.nul` | wxbitmap | The wxbitmap object that will contain any bitmap retrieved from the clipboard. This must be a pre-initialized wxbitmap object! |

# wxclipboardputdata()

## Description

This function puts data onto the clipboard; either text, a bitmap, or both (if both are present). The data will only remain on the clipboard as long as a wxwindow (or wxdialog) is in existence. If the data is copied from the clipboard, then it will remain on the clipboard even if the anchoring window or dialog is subsequently closed.

## Prototype

`wxclipboardputdata` ( string *`text`*, wxbitmap *`bitmap`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | `.nul` | string | The string object that contains the text to be put on the clipboard. |
| bitmap | `.nul` | wxbitmap | The wxbitmap object that contains the bitmap to be put on the clipboard. |

# wxdirectorydialog()

## Description

Displays a dialog (normally an OS common dialog) to allow the selection of a directory name. This can optionally have an new directory button on those OS's that have the new directory button support in their common dialog for selecting directories.

## Prototype

`wxdirectorydialog` ( type(wxdialogparent) *`parent`*, string *`message`*, string *`defaultdirectory`*, string *`style`*, string *`directory`*, string *`result`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | `.nul` | type(wxdialogparent) | Gives the parent of the common dialog, if any. |
| message | `""` | string | The text shown in the title bar of the common dialog. |
| defaultdirectory | `""` | string | The directory in which the dialog first starts. If this is "" then it will start in the current directory. |
| style | `""` | string | This is either the empty string (`""`) or `newdirectorybutton`. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| directory | `.nul` | string | The selected directory will be returned in the string object passed as this parameter. It must be an existing (preinitialized) object. |
| result | `.nul` | string | The result (either `"ok"` or `"cancel"`) will be returned in the string object passed as this parameter. It must be an existing (preinitialized) object. |

# wxfiledialog()

## Description

Displays a dialog (normally an OS common dialog) to allow the selection or the entering of a file name. The same function is used for both files being selected for opening as well as for saving. The behavior of the dialog is dependent on the values passed.

## Prototype

`wxfiledialog` ( type(wxdialogparent) *`parent`*, string *`message`*, string *`defaultdirectory`*, string *`defaultfilename`*, string *`wildcard`*, string *`style`*, string *`filename`*, string *`result`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | `.nul` | type(wxdialogparent) | Gives the parent of the common dialog, if any. |
| message | `""` | string | The text shown in the title bar of the common dialog. |
| defaultdirectory | `""` | string | The directory in which the dialog first starts. If this is "" then where it will start is dependent on the operating system. On Windows 2000, XP and later:<br>1. If the *`defaultfilename`* parameter contains a path, then that path will be the starting directory.<br>2. Otherwise if the *`defaultdirectory`* parameter contains a path, then that path will be the starting directory.<br>3. If neither is specified and the application (in this case the program used to run the SIMPOL program) has been run in |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | the past and used this function before, then the most recently used path will be the starting directory (if much time has passed, then the path may have been discarded).<br><br>4. If the *defaultdirectory* parameter is empty and the current directory contains any files matching any of the types specified in the *wildcard* parameter, then the starting directory is the current directory.<br><br>5. Otherwise the starting directory will be the personal files directory (if defined) of the current user.<br><br>6. Otherwise the starting directory will be the desktop folder of the current user.<br><br>On Windows 98 and Me:<br><br>1. If the *defaultdirectory* parameter contains a path, then that path will be the starting directory.<br><br>2. If the *defaultdirectory* is empty and the *defaultfilename* parameter contains a path, then that path will be the starting directory.<br><br>3. Otherwise if the current directory contains any files matching any of the types specified in the *wildcard* parameter, then the starting directory is the current directory.<br><br>4. Otherwise the starting directory will be the personal files directory of the current user.<br><br>On earlier versions of Windows and Windows NT:<br><br>1. If the *defaultdirectory* parameter contains a path, then |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | that path will be the starting directory.<br><br>2. If the *defaultdirectory* is empty and the *defaultfilename* parameter contains a path, then that path will be the starting directory.<br><br>3. Otherwise the current directory is the starting directory. |
| defaultfilename | "" | string | The default value that will be shown for the file name. |
| wildcard | "" | string | This allows the showing of various categories of files, such as *.sma. The wildcard may be a specification for multiple types of file with a description for each, such as: "BMP and GIF files (*.bmp;*.gif)\|*.bmp;*.gif\|PNG files (*.png)\|*.png". |
| style | open,mustexist | string | This is a comma delimited list of one of either open or save and any of: mustexist,overwriteprompt,multip Commonly open is associated with mustexist and save with overwriteprompt but there are cases that are exceptions. |
| filename | .nul | string | The selected file name(s) will be returned in the string object passed as this parameter. It must be an existing (preinitialized) object. If using the multiple style and multiple names are selected, the fully-qualified path and file names will be returned in a character "{0}" delimited string, i.e. "foo.txt{0}foobar.txt{0}foobar2.txt" where the string "{0}" represents the single character with the value 0. |
| result | .nul | string | The result (either "ok" or "cancel") will be returned in the string object passed as this parameter. It must be an existing (preinitialized) object. |

# wxfontdialog()

## Description

Displays a dialog (normally an OS common dialog) to allow the selection of a font. This can optionally have a font passed in as the default starting point in the list. If the user exits via the OK button, the selected font will be created as a wxfont object and returned.

## Prototype

`wxfontdialog` ( type(wxdialogparent) *`parent`*, string *`message`*, wxfont *`defaultfont`*, string *`result`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | `.nul` | type(wxdialogparent) | Gives the parent of the common dialog, if any. |
| message | `""` | string | The text shown in the title bar of the common dialog. |
| defaultfont | `.nul` | wxfont | The font that is preselected (if available) in the dialog. |
| result | `.nul` | string | The result (either `"ok"` or `"cancel"`) will be returned in the string object passed as this parameter. It must be an existing (preinitialized) object. |

# wxgetscreentextextent()

## Description

This function determines the height and width of the text passed using the font passed. The height will not necessarily be the height of the text but rather the height required for the font. The parameters *`width`* and *`height`* must be pre-initialized integers since the values will be returned in the objects passed.

## Prototype

`wxgetscreentextextent` ( wxfont *`font`*, string *`text`*, integer *`width`*, integer *`height`*, integer *`descent`*, integer *`extleading`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | The font to be used to evaluate the size of the text. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | The text for which a size is being requested. |
| width | None | integer | This will return the width in pixels of the text. This must be a pre-initialized object. |
| height | None | integer | This will return the height in pixels of the text. This must be a pre-initialized object. |
| descent | None | integer | This will return the height in pixels of the descent, which is the dimension from the baseline of the font to the bottom of the descender. This is necessary to calculate the full line height. This must be a pre-initialized object. |
| extleading | None | integer | This will return the height in pixels of the external leading, which is any extra vertical space added to the font by the font designer. This is necessary to calculate the full line height. This must be a pre-initialized object. |

# wxmessagedialog()

## Description

Displays a message box dialog (normally an OS common dialog) to inform the user of something and in some cases to allow the user to make a choice, such as selecting ok, cancel, yes, or no.

## Prototype

`wxmessagedialog` ( type(wxdialogparent) *`parent`*, string *`message`*, string *`captiontext`*, string *`style`*, string *`icon`*, string *`result`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | `.nul` | type(wxdialogparent) | Gives the parent of the common dialog, if any. |
| message | `""` | string | The text shown in the body of the message box dialog. |
| captiontext | `""` | string | The text shown in the title bar of the common dialog. |
| style | `ok` | string | This is one of `"ok"`, `"cancel"`, `"okcancel"`, `"yesno"`, or `"yesno_defaultno"`. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| icon | `"information"` | string | This is one of `"exclaim"`, `"hand"`, `"error"`, `"question"`, or `"information"`. |
| result | `.nul` | string | The result (either `"ok"`, `"cancel"`, `"yes"`, or `"no"`) will be returned in the string object passed as this parameter. It must be an existing (preinitialized) object. |

# wxprintdialog()

## Description

Displays a dialog (normally an OS common dialog) to allow the setting up of the printer. This can optionally have printer settings passed in as the existing configuration. If the user exits via the OK button, the printer settings will be returned in the *`dialogdata`* parameter, which must be a pre-initialized string object.

## Prototype

`wxprintdialog` ( type(wxdialogparent) *`parent`*, string *`dialogdata`*, string *`result`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | `.nul` | type(wxdialogparent) | Gives the parent of the common dialog, if any. |
| dialogdata | `.nul` | string | This will contain the printer settings when the function returns if the user closed the dialog with OK. It can also be used to preset the printer dialog settings before they are shown. It must be an existing (pre-initialized) object. |
| result | `.nul` | string | The result (either `"ok"` or `"cancel"`) will be returned in the string object passed as this parameter. It must be an existing (pre-initialized) object. |

# wxprocess()

## Description

Processes the user's interaction with all wxwindows. `wxprocess()` has to be called regularly for the user interface of an application to be smooth and responsive.

## Prototype

`wxprocess` ( integer *`timeout`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| timeout | `0` | integer | Specifies an amount of time (in microseconds) for which the method will continue to process user interaction, unless the `wxprocess()` method is explicitly exited by calling the `wxbreak()` function. If a value of `.inf` is specified, then user interaction will be processed indefinitely, or until the `wxbreak()` method is called. |

# wxrgbdialog()

## Description

Displays a dialog (normally an OS common dialog) to allow the selection of a color. This can have a color value passed in as the default starting color. If the user exits via the OK button, the selected color will be returned in the *`rgb`* parameter.

## Prototype

`wxrgbdialog` ( type(wxdialogparent) *`parent`*, string *`message`*, integer *`defaultrgb`*, integer *`rgb`*, string *`result`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | `.nul` | type(wxdialogparent) | Gives the parent of the common dialog, if any. |
| message | `""` | string | The text shown in the title bar of the common dialog. |
| defaultrgb | `0` | integer | This supplies the desired default color that should be pre-selected in the dialog. |
| rgb | `.nul` | integer | This will hold the color that was selected (if one was selected). |
| result | `.nul` | string | The result (either `"ok"` or `"cancel"`) will be returned in the string object passed as this parameter. It must be an existing (preinitialized) object. |

# wxsystemfont()

## Description

Retrieves the system default font corresponding to the type requested. Returns either a wxfont object or `.nul`, whereby this would not necessarily be an error. The *`error`* parameter should be checked if an error is suspected.

## Prototype

`wxsystemfont` ( string *`font`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | `"system"` | string | Specifies which default font is desired. This can be any of: `"oem-fixed"`, `"ansi-fixed"`, `"ansi-variable"`, `"system"`, `"default-device"`, or `"default-gui"`. |
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during the execution of the function. If `error` is not specified or is `.nul` then any error which occurs during function execution will halt the program. If an error object is specified and an error occurs during function execution then the error code is output into that object and the methods returns `.nul`. |

# wxsystemvalues()

## Description

Retrieves the system values corresponding to the parameters passed. These are returned in the pre-initialized objects passed to the specified parameters.

## Prototype

`wxsystemvalues` ( integer *`rgb_background`*, integer *`rgb_activeborder`*, integer *`rgb_inactiveborder`*, integer *`rgb_windowframe`*, integer *`rgb_activecaption`*, integer *`rgb_captiontext`*, integer *`rgb_inactivecaption`*, integer *`rgb_inactivecaptiontext`*, integer *`rgb_menu`*, integer *`rgb_menutext`*, integer *`rgb_window`*, integer *`rgb_appworkspace`*, integer *`rgb_windowtext`*, integer *`rgb_scrollbar`*, integer *`rgb_3ddarkshadow`*, integer *`rgb_3dlight`*, integer *`rgb_highlight`*, integer *`rgb_highlighttext`*, integer *`rgb_graytext`*, integer *`rgb_buttonface`*, integer *`rgb_buttonshadow`*, integer *`rgb_buttontext`*, integer *`rgb_buttonhighlight`*, inte-

ger *rgb_infotext*, integer *rgb_infobackground*, integer *metric_networkpresent*, integer *metric_penwindowspresent*, integer *metric_showsounds*, integer *metric_mousebuttons*, integer *metric_swapbuttons*, integer *metric_doubleclick_x*, integer *metric_doubleclick_y*, integer *metric_drag_x*, integer *metric_drag_y*, integer *metric_screen_x*, integer *metric_screen_y*, integer *metric_windowminimum_x*, integer *metric_windowminimum_y*, integer *metric_border_x*, integer *metric_border_y*, integer *metric_framesize_x*, integer *metric_framesize_y*, integer *metric_edge_x*, integer *metric_edge_y*, integer *metric_caption_y*, integer *metric_menu_y*, integer *metric_hscroll_arrow_x*, integer *metric_hscroll_arrow_y*, integer *metric_hscroll_y*, integer *metric_hthumb_x*, integer *metric_vscroll_arrow_x*, integer *metric_vscroll_arrow_y*, integer *metric_vscroll_x*, integer *metric_vthumb_y*, integer *metric_cursor_x*, integer *metric_cursor_y*, integer *metric_icon_x*, integer *metric_icon_y*, integer *metric_iconspacing_x*, integer *metric_iconspacing_y*, integer *metric_smallicon_x*, integer *metric_smallicon_y*, integer *error* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb_background | .nul | integer | Retrieves the desktop color. |
| rgb_activeborder | .nul | integer | Retrieves the active window border color. |
| rgb_inactiveborder | .nul | integer | Retrieves the inactive window border color. |
| rgb_windowframe | .nul | integer | Retrieves the window frame color. |
| rgb_activecaption | .nul | integer | Retrieves the active window caption color. |
| rgb_captiontext | .nul | integer | Retrieves the color for text in caption, size box and scrollbar arrow box. |
| rgb_inactivecaption | .nul | integer | Retrieves the color of the inactive window caption. |
| rgb_inactivecaptiontext | .nul | integer | Retrieves the color of text in inactive window captions. |
| rgb_menu | .nul | integer | Retrieves the menu background color. |
| rgb_menutext | .nul | integer | Retrieves the menu text color. |
| rgb_window | .nul | integer | Retrieves the window background color. |
| rgb_appworkspace | .nul | integer | Retrieves the background color for MDI applications. |
| rgb_windowtext | .nul | integer | Retrieves the color for text in windows. |
| rgb_scrollbar | .nul | integer | Retrieves the color for the scrollbar gray area. |
| rgb_3ddarkshadow | .nul | integer | Retrieves the color of dark shadow for three-dimensional display elements. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb_3dlight | .nul | integer | Retrieves the light color for three-dimensional display elements. |
| rgb_highlight | .nul | integer | Retrieves the color for items selected in a control. |
| rgb_highlighttext | .nul | integer | Retrieves the color for text of items selected in a control. |
| rgb_graytext | .nul | integer | Retrieves the color for gray (disabled) text. |
| rgb_buttonface | .nul | integer | Retrieves the color used for face shading on push buttons. |
| rgb_buttonshadow | .nul | integer | Retrieves the color for edge shading on push buttons. |
| rgb_buttontext | .nul | integer | Retrieves the color of text on push buttons. |
| rgb_buttonhighlight | .nul | integer | Retrieves the highlight color for buttons. |
| rgb_infotext | .nul | integer | Retrieves the text color for tool tip controls. |
| rgb_infobackground | .nul | integer | Retrieves the background color for tool tip controls. |
| metric_networkpresent | .nul | integer | Returns 1 if a network is present, 0 otherwise. |
| metric_penwindowspresent | .nul | integer | Returns 1 if PenWindows is installed, 0 otherwise. |
| metric_showsounds | .nul | integer | Returns Non-zero if the user requires an application to present information visually in situations where it would otherwise present the information only in audible form; zero otherwise. |
| metric_mousebuttons | .nul | integer | Returns the number of buttons on the mouse, or zero if no mouse was installed. |
| metric_swapbuttons | .nul | integer | Returns non-zero if the meanings of the left and right mouse buttons are swapped; zero otherwise. |
| metric_doubleclick_x | .nul | integer | Returns the width in pixels of the rectangle within which two successive mouse clicks must fall to generate a double-click. |
| metric_doubleclick_y | .nul | integer | Returns the height in pixels of the rectangle within which two successive mouse clicks must fall to generate a double-click. |
| metric_drag_x | .nul | integer | Returns the width in pixels of a rectangle centered on a drag point |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | to allow for limited movement of the mouse pointer before a drag operation begins. |
| metric_drag_y | `.nul` | integer | Returns the height in pixels of a rectangle centered on a drag point to allow for limited movement of the mouse pointer before a drag operation begins. |
| metric_screen_x | `.nul` | integer | Returns the width of the screen in pixels. |
| metric_screen_y | `.nul` | integer | Returns the height of the screen in pixels. |
| metric_windowminimum_x | `.nul` | integer | Returns the minimum width of a window in pixels. |
| metric_windowminimum_y | `.nul` | integer | Returns the minimum height of a window in pixels. |
| metric_border_x | `.nul` | integer | Returns the width in pixels of the single border. |
| metric_border_y | `.nul` | integer | Returns the height in pixels of the single border. |
| metric_framesize_x | `.nul` | integer | Returns the width in pixels of the window frame for a window with a `sizeable` border. |
| metric_framesize_y | `.nul` | integer | Returns the height in pixels of the window frame for a window with a `sizeable` border. |
| metric_edge_x | `.nul` | integer | Returns the width of a 3-D border in pixels. |
| metric_edge_y | `.nul` | integer | Returns the height of a 3-D border in pixels. |
| metric_caption_y | `.nul` | integer | Returns the height in pixels of a normal caption area. |
| metric_menu_y | `.nul` | integer | Returns the height in pixels of a single-line menu bar. |
| metric_hscroll_arrow_x | `.nul` | integer | Returns the width in pixels of the arrow bitmap on a horizontal scrollbar. |
| metric_hscroll_arrow_y | `.nul` | integer | Returns the height in pixels of the arrow bitmap on a horizontal scrollbar. |
| metric_hscroll_y | `.nul` | integer | Returns the height in pixels of the horizontal scrollbar. |
| metric_hthumb_x | `.nul` | integer | Returns the width in pixels of the horizontal scrollbar thumb. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| metric_vscroll_arrow_x | .nul | integer | Returns the width in pixels of the arrow bitmap on a vertical scrollbar. |
| metric_vscroll_arrow_y | .nul | integer | Returns the height in pixels of the arrow bitmap on a vertical scrollbar. |
| metric_vscroll_x | .nul | integer | Returns the width in pixels of the vertical scrollbar. |
| metric_vthumb_y | .nul | integer | Returns the height in pixels of the vertical scrollbar thumb. |
| metric_cursor_x | .nul | integer | Returns the width in pixels of the cursor. |
| metric_cursor_y | .nul | integer | Returns the height in pixels of the cursor. |
| metric_icon_x | .nul | integer | Returns the default width in pixels of an icon. |
| metric_icon_y | .nul | integer | Returns the default height in pixels of an icon. |
| metric_iconspacing_x | .nul | integer | Returns the default width of a grid cell for items in large icon view, in pixels. Each item fits into a rectangle of this size when arranged. |
| metric_iconspacing_y | .nul | integer | Returns the default height of a grid cell for items in large icon view, in pixels. Each item fits into a rectangle of this size when arranged. |
| metric_smallicon_x | .nul | integer | Returns the recommended width in pixels of a small icon (in window captions, and small icon view). |
| metric_smallicon_y | .nul | integer | Returns the recommended height in pixels of a small icon (in window captions, and small icon view). |
| error | .nul | integer | Specifies an object which is used to output any error code generated during the execution of the function. If `error` is not specified or is `.nul` then any error which occurs during function execution will halt the program. If an error object is specified and an error occurs during function execution then the error code is output into that object and the methods returns `.nul`. |

# Chapter 12. The Shared Library (SLIB) Component

Provides a set of types for interacting with shared library functions in order to access functionality outside of the scope of SIMPOL.

# sharedlibrary

## Description

A sharedlibrary object represents a shared library (in Windows a DLL) in an operating system. To access functions in a shared library, the `findfunction()` method of the sharedlibrary object is called and the name of the function and the appropriate parameters must be passed. This returns a sharedlibraryfunction object and places it in the ring of functions.

### Note

The use of this functionality will prevent the program from operating successfully on other platforms, unless equivalent functionality is added to the program for each target platform. Whenever possible, avoid using this functionality if the same result can be achieved using libraries provided by SIMPOL.

## Type Tags

None

## Object Value

The value of a sharedlibrary object is undefined and it is an error to attempt to either get or set it.

# sharedlibrary.new()

## Description

Creates a new sharedlibrary object for the specified shared library.

## Prototype

*`sharedlibrary`*.new ( string *`filename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | Specifies the file name (which should be a full path name if the file is not in the path or the current directory). |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the sharedlibrary object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `new` method returns `.nul`. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| filename | string | Provides the file name of the shared library that was opened. |
| firstfunction | sharedlibraryfunction | Contains a reference to the first function that is being made available through the use of the shared library. Only functions that have been retrieved using the `findfunction()` method appear in the ring of functions. For information about the available functions for a given shared library, see the associated library or operating system documentation. The makers of SIMPOL are not able to provide assistance on functionality that is accessed in this way. |
| type | type | Specifies the sharedlibrary type object. |

# Methods

## findfunction()

### Description

Creates a new sharedlibraryfunction object for a specific function in a shared library.

### Prototype

*`sharedlibraryvar`*`.findfunction` ( string *`functionname`*, string *`name`*, string *`returntype`*, string *`parameters`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| functionname | None | string | Specifies the name of the function within the shared library that is to be made available. Currently it is not possible to access functions in a Windows DLL by ordinal, only by name. Either this parameter or else the *`name`* must be provided. If only one is provided, it will be assumed that it is the same for both. |
| name | None | string | Specifies the name to use when accessing the function using the member (!) operator as an argument of the sharedlibrary object. Either this parameter or else the *`functionname`* must be provided. If only one is provided, it will be assumed that it is the same for both. |
| returntype | "" | string | Specifies the data type returned by the function. If this is the empty string, then the function is not expected to a return a value. This parameter must be either "" (if the return is void), or else a valid entry in the form "xyyy" where x is the letter representing the type and yyy is the size. Valid types are:<br>• u\|U — unsigned int<br>• s\|S — signed int<br>• f\|F — float<br>Valid sizes are:<br>• p\|P — pointer, the same size as a pointer for the current machine, but the type has to be a u\|U<br>• 8\|16\|32 — for types s\|S and u\|U on a 32-bit machine<br>• 32\|64 — for f\|F (or the number of bits in a float or a double on the target machine) |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parameters | "" | string | Specifies the data type being passed to the function. If this is the empty string, then the function is expecting any parameters. This parameter must be either "", or else a valid set of parameters in the form "xyyy" where x is the letter representing the type and yyy is the size. Valid types are:<br>• b\|B() — byte<br>• u\|U — unsigned int<br>• s\|S — signed int<br>• f\|F — float<br>• t\|T() — text<br>• p\|P[] — pointer<br>Valid sizes are:<br>• p\|P — pointer, the same size as a pointer for the current machine, but the type has to be a u\|U<br>• 8\|16\|32 — for types s\|S and u\|U on a 32-bit machine<br>• 8\|16 — for type t\|T (8-bit or 16-bit characters)<br>• 32\|64 — for f\|F (or the number of bits in a float or a double on the target machine)<br>In addition, the "t" and "b" parameters also require a value in parentheses, that indicates the size of the buffer being passed. It is also possible to pass a pointer to a structure, including the definition of the structure and having the elements created for the structure. To pass a buffer that can be modified, the parameter must be passed preceded by the "+" character. For full details on how to use shared library functions, see the chapter on them in the Programmer's Guide. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | integer | Specifies an object which is used to output any error code generated during creation of the UTOSdirectoryentry object. If `error` is not specified or is `.nul` then any error which occurs during object creation will halt the program. If an error object is specified and an error occurs during object creation then the error code is output into that object and the `getentry` method returns `.nul`. |

# sharedlibraryfunction

## Description

A sharedlibraryfunction object represents a function in a shared library that has been previously retrieved using the `findfunction()` method. To call the function, it is necessary to use the `call()` method and pass the appropriate parameters (if any) in accordance to the way the function expects them. It is currently not possible to pass a SIMPOL type in the place where a pointer to a structure is expected. Instead, each member of the type must be passed explicitly.

## Type Tags

None

## Object Value

The value of a sharedlibraryfunction object is undefined and it is an error to attempt to set or get it.

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| functionname | string | The name of the function in the shared library. |
| library | sharedlibrary | This contains a reference to the shared library object that contains the function. |
| name | string | The name of the sharedlibraryfunction object. Using this name and the member operator it is possible to call the function from the sharedlibrary object |

| Property | Type | Description |
|---|---|---|
| | | without the need to store a reference to the function. For example, assuming a library object referenced by a variable called foo, with a function named bar: `foo!bar.call()`. |
| next | sharedlibraryfunction | This is a reference to the next function in the ring of functions that have been retrieved for the shared library using the `findfunction()` method. Only functions retrieved in this way appear in the ring. |
| type | type | Specifies the sharedlibraryfunction type object. |

# Methods

## call()

### Description

Calls the shared library function. Depending on the definition of the function, it may have any number of parameters and may or may not have a return value. The use of this functionality can cause unexpected results, including the crash of the program, the operating system, etc. Using these functions goes outside the safeguards provided by the SIMPOL programming language and is completely at the users own risk. As a general rule of thumb, do not allocate resources without giving them back. Do not hog resources (retain control of resources for longer than necessary).

### Prototype

*sharedlibraryfunctionvar*.call( …, … )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| … | None | | Function-dependent. |
| … | None | | |

# Chapter 13. The ODBC Client (ODBC) Component

Provides client access to ODBC data sources.

# odbc1columndescription

## Description

Creates a new odbc1columndescription object. This object is returned from a call to the `describecolumn()` method of the odbc1statement type. It contains the information that describes the column within the statement referenced by the original *`columnnumber`* parameter.

## Type Tags

None

## Object Value

Objects of type odbc1columndescription have no value, and it is an error to try to get or set this value.

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| columnnumber | integer | This is the column number of the column within the statement being described by this object. |
| columnsize | integer | This is the maximum width of the column described by this object. This only applies to appropriate data types. |
| datatype | type | This holds a reference to the type object that represents the SIMPOL data type of the column. |
| decimaldigits | integer | This is the number of digits following the decimal point that the column can support. This only applies if the data type is appropriate. |
| error | integer | If a SIMPOL error occurs, this will hold the SIMPOL error number. |

| Property | Type | Description |
|---|---|---|
| name | string | This contains the name of the column being described. |
| nullable | boolean | This describes whether or not the SQL value NULL can be assigned to this column, and therefore also, whether the SIMPOL value of `.nul` can be assigned. |
| sqltype | string | This contains the name of the SQL data type representing the column. |
| type | type | Specifies the odbc1columndescription type object. |

# odbc1connection

## Description

Creates a new odbc1connection object and establishes a connection to an ODBC datasource.

## Type Tags

None

## Object Value

Objects of type odbc1connection have no value, and it is an error to try to get or set this value.

## odbc1connection.new()

### Description

Creates a new odbc1connection object and attempts to connect in one of three ways:

1. To connect using the first mechanism, datasourcename, username and authentication should be used to give the name of a data source and a username/authentication string (password) to complete the connection. If the connection is successful then connected will be set to .true and a newly connected odbc1connection object is returned, otherwise error wil be filled with error information and .nul is returned.

2. To connect using the second mechanisme, browsestring and browseresult should be specified to be the input and output of an attempt to 'browse connect' to a data source (see the ODBC documentation for SQLBrowseConnect). If the connection is successful and complete, then connected will be set to .true and a newly connected odbc1connection object will be output. If the call is successful but the connection is not complete then connected is set to .false and a new odbc1connection object which is not yet connected is returned. If the call fails then error is filled with error information and .nul is returned.

3. To connect using the third mechanism, driverstring and driverresult should be specified to be the input and output of an attempt to connect using the ODBC 'driver connect' functionality (see the documentation of SQLDriverConnect). If the connection is successful the connected is set to .true and a newly connected odbc1connection object is returned. If the connection fails then error is filled with error information and .nul is returned.

## Prototype

*odbc1connection*.new ( string *datasourcename*, string *username*, string *authentication*, string *browsestring*, string *browseresult*, string *driverstring*, string *driverresult*, boolean *connected*, odbc1error *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datasource-name | "" | string | The data source name to which the connection is being made. |
| username | "" | string | The user name to be used when logging into the connection. |
| authentication | "" | string | The user's password for accessing the data source. |
| browsestring | "" | string | The browse string to be used to attempt a browse connection. |
| browseresult | .nul | string | The browse result string into which the result will be returned from the attempt to connect using the browsestring if it fails to connect through lack of information. This must be a pre-initialized string object! |
| driverstring | "" | string | The driver string to be used to attempt a driver connection. |
| driverresult | .nul | string | The driver result string into which the result will be returned from the attempt to connect using the driverstring if it fails to connect through lack of information. This must be a pre-initialized string object! |
| connected | .nul | boolean | This parameter will be set to .true if the connection attempt is successful, or .false if it is not. |
| error | .nul | odbc1error | Specifies an object that is used to output any error information generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns .nul. |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| type | type | Specifies the odbc1connection type object. |

# Methods

## browseconnect()

### Description

Attempts to connect using odbc1connection object and the supplied information.

### Prototype

*odbc1connectionvar*.browseconnect ( string *browsestring*, string *browseresult*, boolean *connected*, odbc1error *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| browsestring | "" | string | The browse string to be used to attempt a browse connection. |
| browseresult | .nul | string | The browse result string into which the result will be returned from the attempt to connect using the browsestring if it fails to connect through lack of information. This must be a pre-initialized string object! |
| connected | .nul | boolean | This parameter will be set to .true if the connection attempt is successful, or .false if it is not. |
| error | .nul | odbc1error | Specifies an object that is used to output any error information generated during the execution of the method. This parameter must be an object, not an integer value. If *error* is not specified or is .nul then any error that occurs during object creation will halt |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| | | | the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

## createstatement()

### Description

Creates a SQL statement in preparation for executing the statement. If it fails, the error object will be filled in.

### Prototype

*`odbc1connectionvar`*`.createstatement` ( string *`statement`*, odbc1error *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| statement | `.nul` | string | The SQL statement to be prepared for execution. |
| error | `.nul` | odbc1error | Specifies an object that is used to output any error information generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# odbc1error

# Description

Creates a new odbc1error object. This object is used for the output of errors from the various odbc1 objects.

# Type Tags

None

# Object Value

Objects of type odbc1error have no value, and it is an error to try to get or set this value.

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| error | integer | If a SIMPOL error occurs, this will hold the SIMPOL error number. |
| nativeerror | integer | If an error occurs, this will hold the error number (if any) returned from ODBC. |
| sqlerrortext | string | The text that accompanies any SQL error, if any. |
| sqlstate | string | If an error occurs, then this will hold the five character SQL error code from ODBC. |
| type | type | Specifies the odbc1error type object. |

# odbc1statement

## Description

Creates a new odbc1statement object. This object is used for to hold the SQL statement prior to its execution. It's `execute()` method is used to actually execute the prepared statement on the ODBC data source.

## Type Tags

None

## Object Value

Objects of type odbc1statement have no value, and it is an error to try to get or set this value.

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| connection | odbc1connection | This is the ODBC connection with which this statement is associated. |

| Property | Type | Description |
|---|---|---|
| statement | string | This contains the prepared SQL statement that was passed to the `createstatement()` method of the odb1connection object when this object was created. |
| type | type | Specifies the odbc1statement type object. |

# Methods

## describecolumn()

### Description

Retrieves an odbc1columndescription object that describes the column corresponding to the integer index value passed. If an error occurs, the error object will be filled out accordingly, and a .nul object will be returned. The column number is based on the position of the column in the statement.

### Prototype

*odbc1statementvar*.`describecolumn` ( integer *`columnnumber`*, odbc1error *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| columnnumber | `.nul` | integer | The column number for which a description is desired, based on the position of the column in the statement. |
| error | `.nul` | odbc1error | Specifies an object that is used to output any error information generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

## execute()

### Description

Executes a perviously prepared SQL statement. If it fails, the error object will be filled in.

### Prototype

*odbc1statementvar*.`execute` ( array *`parameters`*, odbc1error *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parameters | `.nul` | array | Specifies an array that contains parameters for replacement into the SQL statement. The placeholder character in the SQL statement is the question mark (?). The placeholders will be replaced by the values in the array, beginning with the subscript 1, and continuing in sequential order until the placeholders have all been replaced. If the array does not contain a value for a parameter, then it will receive the value `.nul`. |
| error | `.nul` | odbc1error | Specifies an object that is used to output any error information generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

## fetch()

### Description

Retrieves a row of results for a statement that has been executed. If there are no further results for the statement, then the SIMPOL error value 64 (end of data) will be returned in the appropriate property of the odbc1error object. The results are retrieved into the array passed. It is a good idea to create a new array for each row, since empty values may not be assigned as such to the array and in a loop might not overwrite any previous values, potentially giving faulty results. The column description will tell what the column name is and also the data type in SQL as well as that in SIMPOL.

### Prototype

*`odbc1statementvar`*`.fetch` ( array *`results`*, odbc1error *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| results | `.nul` | array | The array object that will be used to return the row of results from the executed SQL statement. This must be a pre-initialized array object. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | odbc1error | Specifies an object that is used to output any error information generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# getcolumncount()

## Description

Retrieves the number of columns that are returned from the SQL statement that has been executed. Check the error object before using the return value!

## Prototype

*`odbc1statementvar`*`.getcolumncount` ( odbc1error *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | `.nul` | odbc1error | Specifies an object that is used to output any error information generated during the execution of the method. This parameter must be an object, not an integer value. If *`error`* is not specified or is `.nul` then any error that occurs during object creation will halt the program. If an error object is specified and an error occurs during execution then the error code is output into that object and the method returns `.nul`. |

# Chapter 14. The ODBC Client Companion Library

This chapter contains information that is specific to the SIMPOL language companion library `odbc2.sml`. To use these features you must make sure that the ODBC component is added to your project and that the `odbc2.sml` file is added as a library file to your project.

Provides functions to perform basic table creation and data transfer between an ODBC connection and native simpol db1* database objects

# odbc2_addodbcrecord()

## Description

Copies the data from the supplied record to the specified table on an ODBC connection. If the return value is not `.nul` then the function has failed and the string contains information describing the error.

## Prototype

`odbc2_addodbcrecord` ( odbc1connection *`con`*, type(db1table) *`table`*, string *`name`*, array *`fields`*, type(db1record) *`record`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| con | None | odbc1connection | Specifies the ODBC connection on which the destination table is present. |
| table | `.nul` | type(db1table) | Gives an object representing a table which will be used to provide information to identify the table on the ODBC connection to which the new row will be added. If `table` is not supplied, or is `.nul`, then `name` and `fields` must be supplied. If `table` is supplied then it must be the table from which `record` has been selected. |
| name | `.nul` | string | Gives the name of the ODBC table to have the data from `record` inserted. If no name is supplied then the name of `table` is used. |
| fields | `.nul` | array | Gives a number of fields which are used to define which values from `record` are to be transferred to the ODBC table. If `fields` is not supplied then the fields of `table` are used. If `fields` is supplied then `fields[]` must contain the number of fields being supplied, and a consecutive sequence of elements of `fields`, starting at `fields[1]`, must contain references to the fields to use. Each of these fields must belong to the table from which `record` has been selected. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| record | `.nul` | type(db1record) | A record whose data is to be inserted into the destination ODBC table. |

# odbc2_buildodbctable()

## Description

Creates a table on the specified ODBC connection which has a structure matching that of the supplied table or fields, and fills it with data from the table, if supplied. If the return value is not `.nul` then the function has failed and the string contains information describing the error. If the table is successfully created and an error occurs while transfering data then the target table will not necessarily be dropped, and is likely to contain the data which was successfully transferred before the error was raised.

## Prototype

`odbc2_buildodbctable` ( odbc1connection *`con`*, type(db1table) *`table`*, string *`name`*, array *`fields`*, function *`filter`*, type(*) *`filterref`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| con | None | odbc1connection | Specifies the ODBC connection on which the new table will be built. |
| table | `.nul` | type(db1table) | Gives an object representing a table which will be used to provide information for the new ODBC table. If `table` is not supplied, or is `.nul`, then `name` and `fields` must be supplied, and no data will be put into the new table |
| name | `.nul` | string | Gives the name of the new table to be built. If no name is supplied then the name of `table` is used. |
| fields | `.nul` | array | Gives a number of fields which are to define the columns of the new table. If `fields` is not supplied then the fields of `table` are used. If `fields` is supplied then `fields[]` must contain the number of fields being supplied, and a consecutive sequence of elements of `fields`, starting at `fields[1]`, must contain references to the fields to use. Each of these fields must belong to `table`. |
| filter | `.nul` | function | A function which is used to filter record which may be used to create records in the new table. If supplied `filter` must take a record as its first argument, and `filterref` as its second parameter, if `filterref` is supplied and not `.nul`. The return value of `filter` must be a boolean which is `.true` for records which are to be put into the new table, and `.false` if not. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filterref | `.nul` | type(*) | A reference to any object which will be passed to the filter function `filter` function, if supplied, for each record which may be put into the new table. |

# odbc2_buildtablefromodbc()

## Description

Creates a new table, of a db1table tagged type, based on an ODBC statement, and fills it with the output data from that statement. If the return value is not `.nul` then the function has failed and the string contains information describing the error. After the new table is created and has data added to it, there is no attempt to commit the base object; that is the responsibility of the caller.

## Prototype

`odbc2_buildtablefromodbc` ( type(*) *`base`*, odbc1connection *`con`*, string *`statement`*, array *`parameters`*, string *`name`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| base | None | type(*) | Gives a base which supports the creation of new tables using the interface which is common to types which are tagged as db1table. |
| con | `.nul` | odbc1connection | Specifies the ODBC connection to be used for executing the provided statement. |
| statement | `.nul` | string | The ODBC SQL statement whose output columns will be used to define the fields of the new table to be created. The output of this statement is the data used to add new records to the new table. |
| parameters | `.nul` | array | The parameters, if any, to be passed to ODBC when the statement is executed. If `parameters` is provided then the parameter values must be stored in consecutive elements starting with `parameters[1]`. |
| name | `.nul` | string | The name of the new table to create. |

# odbc2_createodbctable()

## Description

Creates an empty table on the specified ODBC connection which has a structure matching that of the supplied table or fields. If the return value is not `.nul` then the function has failed and the string contains information describing the error.

## Prototype

`odbc2_createodbctable` ( odbc1connection *`con`*, type(db1table) *`table`*, string *`name`*, array *`fields`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| con | None | odbc1connection | Specifies the ODBC connection on which the new table will be created. |
| table | `.nul` | type(db1table) | Gives an object representing a table which will be used to provide information for the new ODBC table. If `table` is not supplied, or is `.nul`, then `name` and `fields` must be supplied. |
| name | `.nul` | string | Gives the name of the new table to be created. If no name is supplied then the name of `table` is used. |
| fields | `.nul` | array | Gives a number of fields which are used to define the columns of the new table. If `fields` is not supplied then the fields of `table` are used. If `fields` is supplied then `fields[]` must contain the number of fields being supplied, and a consecutive sequence of elements of `fields`, starting at `fields[1]`, must contain references to the fields to use. |

# odbc2_createtablefromodbc()

## Description

Creates a new table, of a db1table tagged type, based on an ODBC statement. If the return value is not `.nul` then the function has failed and the string contains information describing the error. After the new table is created, there is no attempt to commit the base object; that is the responsibility of the caller.

## Prototype

`odbc2_createtablefromodbc` ( type(*) *`base`*, odbc1connection *`con`*, string *`statement`*, array *`parameters`*, string *`name`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| base | None | type(*) | Gives a base which supports the creation of new tables using the interface which is common to types which are tagged as db1table. |
| con | `.nul` | odbc1connection | Specifies the ODBC connection to be used for executing the provided statement. |
| statement | `.nul` | string | The ODBC SQL statement whose output columns will be used to define the fields of the new table to be created. |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parameters | `.nul` | array | The parameters, if any, to be passed to ODBC when the statement is executed. If `parameters` is provided then the parameter values must be stored in consecutive elements starting with `parameters[1]`. |
| name | `.nul` | string | The name of the new table to create. |

# odbc2_fetchandsaverecords()

## Description

Adds the output data of an ODBC SQL statement to a specified table. If the return value is not `.nul` then the function has failed and the string contains information describing the error. After the data is added to the table then there is no attempt to commit the base object; that is the responsibility of the caller. If the function fails at some point then no attempt is made to rollback or commit the records added to the table up to the point of the failure.

## Prototype

`odbc2_fetchandsaverecords` ( type(db1table) *`table`*, odbc1connection *`con`*, string *`statement`*, array *`parameters`*, array *`fields`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | The table to which new records will be added. |
| con | `.nul` | odbc1connection | Specifies the ODBC connection to be used for executing the provided statement. |
| statement | `.nul` | string | The ODBC SQL statement whose output columns will be used to create records which are added to `table`. |
| parameters | `.nul` | array | The parameters, if any, to be passed to ODBC when the statement is executed. If `parameters` is provided then the parameter values must be stored in consecutive elements starting with `parameters[1]`. |
| fields | `.nul` | array | Gives a number of fields that are used to define which values will be set in the newly created records. If `fields` is not supplied then all the fields of `table` are used. If `fields` is supplied then `fields[]` must contain the number of fields being supplied, and a consecutive sequence of elements of `fields`, starting at `fields[1]`, must contain references to the fields to use. Each of these fields must belong to `table`. |

# Chapter 15. The Language Utilities (UTIL) Component

This component provides a common location for utility types and functions that are important but not worth including in the core language component.

# dlist

## Description

The dlist type consists of a doubly-linked list of dnode objects. Instead of being a ring, this provides a first and a last property (with the obvious meanings). It also provides a method of retrieving the count of nodes, and the nodes at the beginning and the end of a list refer to `.nul` in their properties for the appropriate direction (either next or prev).

## Type Tags

None

## Object Value

Objects of type dlist have no value, and it is an error to try to get or set this value.

# dlist.new()

### Description

Creates a new empty dlist object.

### Prototype

*`dlist`*`.new` ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of |

| Property | Type | Description |
|---|---|---|
| | | being marked with the resolve keyword, so that object resolution can continue down this property. |
| first | dnode | This is the first node in the list. If there are no elements in the list, then this will be `.nul`. |
| last | dnode | This is the last node in the list. If there are no elements in the list, then this will be `.nul`. |
| type | type | Specifies the dlist type object. |

# Methods

## clear()

### Description

This clears the nodes from a list and makes any nodes that are still referred to anywhere into a complete ring. This means that the nodes still exist, they simply aren't part of the list any more, unless none of the nodes are referred to by any other variable, in which case they will no longer exist.

### Prototype

*`dlistvar`*`.clear` ()

### Parameters

None

## count()

### Description

Returns the number of nodes in the list.

### Prototype

*`dlistvar`*`.count` ()

### Parameters

None

## getnode()

### Description

Returns the node from the list corresponding to the index value passed as the parameter. The first node has the index value `1` and the last node will have an index value equal to that returned by the `count()` method.

### Prototype

*`dlistvar`*`.getnode` ( integer *`index`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| index | None | integer | Specifies the index value of the node to retrieve. |

## insertfirst()

### Description

Inserts the node passed as the first node in the list. If the node is part of a ring consisting of more than one node, then the ring will be broken following the node passed and all entries will be inserted prior to that node which will then have the former first node of the list (if any) as its next node and it will become the prev node of the former first node. If, for example, there is a rings of nodes: `a, b, c (, a, b, …)` and a list `x` containing: `d, e, f`, and then the call: `x.insertfirst(b)` is made, then the result in `x` will be: `c, a, b, d, e, f`. It is an error to pass a node to this method if that node is already in a list.

### Prototype

*`dlistvar`*`.insertfirst` ( dnode *`node`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| node | None | dnode | Specifies the node that should be inserted into the beginning of the list. |

## insertlast()

### Description

Inserts the node passed as the last node in the list. If the node is part of a ring consisting of more than one node, then the ring will be broken following the node passed and all entries will be inserted prior to that node, the first of which will then have the former last node of the list (if any) as its prev node and it will become the next node of the former last node. If, for example, there is a rings of nodes: `a, b, c (, a, b, …)` and a list `x` containing: `d, e, f`, and then the call: `x.insertlast(b)` is made, then the result in `x` will be: `d, e, f, c, a, b`. It is an error to pass a node to this method if that node is already in a list.

### Prototype

*`dlistvar`*`.insertlast` ( dnode *`node`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| node | None | dnode | Specifies the node that should be inserted at the end of the list. |

# dnode

## Description

This type is used in the implementation of a doubly-linked list and by itself provides a doubly-linked ring. The nodes can be traversed in both directions from the head of the list to the tail, and back. The most useful characteristic of this component is that it can be added to an existing type to provide list functionality to the type. The way to do that, is to assign the containing object to the __ when creating the dnode object. This allows access back from the object included in a larger type to the containing type.

## Type Tags

None

## Object Value

Objects of type dnode have no value, and it is an error to try to get or set this value.

## dnode.new()

### Description

Creates a new dnode object.

### Prototype

*dnode*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. |
| __ | type(*) | This property is provided for use by the user to attach any object of any type to the type in which this property is provided. It has the additional feature of being marked with the resolve keyword, so that object resolution can continue down this property. |
| next | dnode | This holds a reference to either the next dnode in the list, or else to `.nul`. |
| prev | dnode | This holds a reference to either the previous dnode in the list, or else to `.nul`. |
| type | type | Specifies the dnode type object. |

# Methods

## insertafter()

### Description

Inserts the calling node after the node passed as the parameter in the target ring or list. If the calling node is part of a ring of multiple nodes then the whole ring is added with the ring being broken between the calling node and the node pointed to by its next. If, for example, there are two rings of nodes: `a, b, c (, a, b, …)` and `d, e, f (, d, e, …)`. If the call: `b.insertafter(e)` is made, then the result will be: `d, e, c, a, b, f (, d, e, c, …)`.

### Prototype

*dnodevar*`.insertafter` ( dnode *node* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| node | None | dnode | Specifies the node after which the calling node (and the remainder of the nodes that form part of the ring or list) should be placed. |

## insertbefore()

### Description

Inserts the calling node before the node passed as the parameter in the target ring or list. If the calling node is part of a ring of multiple nodes then the whole ring is added with the ring being broken between the calling node and the node pointed to by its prev. If, for example, there are two rings of nodes: `a, b, c (, a, b, …)` and `d, e, f (, d, e, …)`. If the call: `b.insertbefore(e)` is made, then the result will be: `d, c, a, b, e, f (, d, c, a, …)`.

### Prototype

*dnodevar*`.insertbefore` ( dnode *node* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| node | None | dnode | Specifies the node before which the calling node (and the remainder of the nodes that form part of the ring or list) should be placed. |

## remove()

### Description

This removes one or more nodes from a chain of nodes. If no parameter is passed, it removes only the calling node. If another node in the same chain is passed, then all nodes between the node passed as the parameter and the calling node will be removed into a separate ring.

## Prototype

*dnodevar*.remove ( dnode *removestart* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| removestart | .nul | dnode | This is the starting node to define the chain of nodes to be removed into a new ring. If this is not passed, then only the calling node will be removed. |

# Part III. SIMPOL-Language Libraries

The libraries in this section are written in the SIMPOL programming language and are found in the `lib` directory. Many of them are provided in source as projects, which can be found in the `projects\libs` directory. These components are added to a project in the IDE by selecting the Project → Settings menu item and in the second tab, adding the libraries required. Once added to the project, the information about the types and functions will appear in the Project section of the IDE in the Type View area.

# Table of Contents

# Chapter 16. ABS

# ABS()

## Description

Returns the absolute value of the number passed (negative values will have the sign removed).

## Prototype

ABS ( anyvalue *n* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| n | None | anyvalue | |

# Chapter 17. appframework

The appframework library incorporates the databaseforms, formlib, uisyshelp, filsyslib, db1util, http-clientlib, and others together with its own application and appwindow types to make it fast and easy to create data-aware applications.

# application

## Description

## Type Tags

application

## Object Value

Objects of type application have no value, and it is an error to try to get or set this value.

# application.new()

## Description

## Prototype

*application*.new ( application *me*, string *appiconfile*, string *iconimagetype*, string *inifilename*, string *apptitle* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | application | |
| appiconfile | None | string | |
| iconimagetype | None | string | |
| inifilename | None | string | |
| apptitle | None | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| SBLlocale | localeinfoold | |
| _ | type(*) | |
| __ | type(*) | |
| adddatasource | function | |

| Property | Type | Description |
|---|---|---|
| closedata-source | function | |
| datasources | dring | |
| datasource-unused | function | |
| deffont | wxfont | |
| displayformats | tdisplayformats | |
| exit | function | |
| finddatasrc | function | |
| inifilename | string | |
| locale | localeinfo | |
| onexitrequest | event | |
| opendatasource | function | |
| ostype | integer | |
| ppcs | ppcstype1 | |
| run | function | |
| running | boolean | |
| systeminfo | sysinfo | |
| title | string | |
| type | type | |
| windowicon | wxbitmap | |
| windows | dring | |

# Methods

## adddatasource()

### Description

### Prototype

*`applicationvar`*`.adddatasource` ( application *`me`*, type *`sourcetype`*, string *`source`*, type(*) *`datasource`*, string *`username`*, string *`password`*, integer *`retry`*, integer *`timeout`*, integer *`code-page`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | application | |
| sourcetype | None | type | |
| source | None | string | |
| datasource | None | type(*) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| username | None | string | |
| password | None | string | |
| retry | None | integer | |
| timeout | None | integer | |
| codepage | None | integer | |

# closedatasource()

## Description

## Prototype

*applicationvar*.closedatasource ( application *me*, datasourceinfo *dinfo* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | application | |
| dinfo | None | datasourceinfo | |

# datasourceunused()

## Description

## Prototype

*applicationvar*.datasourceunused ( application *me*, datasourceinfo *src* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | application | |
| src | None | datasourceinfo | |

# exit()

## Description

## Prototype

*applicationvar*.exit ( application *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | application | |

# finddatasrc()

## Description

## Prototype

*applicationvar*.finddatasrc ( application *me*, string *sourcename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | application | |
| sourcename | None | string | |

# newwindow()

## Description

## Prototype

*applicationvar*.newwindow ( application *me*, integer *left*, integer *top*, integer *width*, integer *height*, string *caption*, wxmenubar *mb*, wxtoolbar *tb*, boolean *visible*, boolean *usestatusbar*, wxbitmap *icon*, boolean *horizontalscroll*, boolean *verticalscroll*, boolean *visbutton*, boolean *menubutton*, boolean *minbutton*, boolean *maxbutton*, string *border*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | application | |
| left | 0 | integer | |
| top | 0 | integer | |
| width | 200 | integer | |
| height | 200 | integer | |
| caption | None | string | |
| mb | None | wxmenubar | |
| tb | None | wxtoolbar | |
| visible | .true | boolean | |
| usestatusbar | .true | boolean | |
| icon | None | wxbitmap | |
| horizontalscroll | .false | boolean | |
| verticalscroll | .false | boolean | |
| visbutton | .true | boolean | |
| menubutton | .true | boolean | |
| minbutton | .true | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| maxbutton | `.true` | boolean | |
| border | `sizeable` | string | |
| error | None | integer | |

## opendatasource()

### Description

### Prototype

*`applicationvar`*.`opendatasource` ( application *`me`*, string *`sourcetype`*, string *`source`*, type(appwindow) *`appw`*, string *`username`*, string *`password`*, integer *`retry`*, integer *`timeout`*, integer *`codepage`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | application | |
| sourcetype | None | string | |
| source | None | string | |
| appw | None | type(appwindow) | |
| username | None | string | |
| password | None | string | |
| retry | None | integer | |
| timeout | None | integer | |
| codepage | None | integer | |
| error | None | integer | |

## run()

### Description

### Prototype

*`applicationvar`*.`run` ( application *`me`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | application | |

# appwindow

## Description

# Type Tags

appwindow

# Object Value

Objects of type appwindow have no value, and it is an error to try to get or set this value.

# appwindow.new()

## Description

## Prototype

*appwindow*.new ( appwindow *me*, type(application) *app*, boolean *visible*, integer *left*, integer *top*, integer *width*, integer *height*, wxmenubar *mb*, wxtoolbar *tb*, wxstatusbar *sb*, wxbitmap *icon*, type *windowtype*, boolean *horizontalscroll*, boolean *verticalscroll*, boolean *visbutton*, boolean *menubutton*, boolean *minbutton*, boolean *maxbutton*, type(wxdialogparent) *parent*, string *border*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| app | None | type(application) | |
| visible | .true | boolean | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| mb | None | wxmenubar | |
| tb | None | wxtoolbar | |
| sb | None | wxstatusbar | |
| icon | None | wxbitmap | |
| windowtype | None | type | |
| horizontalscroll | .false | boolean | |
| verticalscroll | .false | boolean | |
| visbutton | .true | boolean | |
| menubutton | .true | boolean | |
| minbutton | .true | boolean | |
| maxbutton | .true | boolean | |
| parent | None | type(wxdialogparent) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| border | `sizeable` | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addformdata-sources | function | |
| addtable | function | |
| addtables | function | |
| alt_tb | wxtoolbar | |
| app | type(application) | |
| appnode | dlistnode | |
| close | function | |
| closeall | function | |
| closeform | function | |
| closetable | function | |
| closetablenode | function | |
| createformdi-rect | function | |
| currentpath | string | |
| currenttable | tableinfo | |
| dataview | tdataview | |
| disablewin-dowresize | boolean | |
| fastselection | boolean | |
| filter | string | |
| filteractive | boolean | |
| filterview | tdataview | |
| findtable | function | |
| firsteditable-control | type(dataform1control) | |
| form | type(dataform1) | |
| getcurrentpath | function | |
| gettablesarray | function | |
| innerwindow | wxwindow | |
| isresizing | boolean | |
| lastheight | integer | |

| Property | Type | Description |
|---|---|---|
| lastinternaluniquekey | string | |
| lastselkeyvalue | anyvalue | |
| lastwidth | integer | |
| mb | wxmenubar | |
| ondeleterecord | event | |
| onmanagemenu | event | |
| onmanagetoolbar | event | |
| onopenform | event | |
| opendatatable | function | |
| openformdirect | function | |
| quickreport | quickreport1 | |
| recordview | trecordview | |
| report | type(*) | |
| resizewindowtoform | function | |
| sb | wxstatusbar | |
| setcurrentindex | function | |
| setcurrentpath | function | |
| setcurrenttable | function | |
| showtablepathincaption | boolean | |
| tables | dring | |
| tableview | ttableview | |
| tb | wxtoolbar | |
| type | type | |
| viewmode | string | |
| w | type(wxcontainer) | |
| windowlist | array | |

# Methods

## addformdatasources()

### Description

### Prototype

*appwindowvar*.addformdatasources ( appwindow *me*, dataform1 *f* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| f | None | dataform1 | |

## addtable()

### Description

### Prototype

*appwindowvar*.addtable ( appwindow *me*, type(db1table) *table*, datasourceinfo *source*, boolean *createview* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| table | None | type(db1table) | |
| source | None | datasourceinfo | |
| createview | .true | boolean | |

## addtables()

### Description

### Prototype

*appwindowvar*.addtables ( appwindow *me*, dataform1 *f* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| f | None | dataform1 | |

## close()

### Description

### Prototype

*appwindowvar*.close ( appwindow *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |

# closeall()

## Description

## Prototype

*appwindowvar*.closeall ( appwindow *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |

# closeform()

## Description

## Prototype

*appwindowvar*.closeform ( appwindow *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |

# closetable()

## Description

## Prototype

*appwindowvar*.closetable ( appwindow *me*, string *tablename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| tablename | None | string | |

# closetablenode()

## Description

## Prototype

*appwindowvar*.closetablenode ( appwindow *me*, dlistnode *tablenode* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| tablenode | None | dlistnode | |

## createformdirect()

### Description

### Prototype

*appwindowvar*.createformdirect ( appwindow *me*, string *functionname*, string *appmsgtitle* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| functionname | None | string | |
| appmsgtitle | `Create Form Error` | string | |

## findtable()

### Description

### Prototype

*appwindowvar*.findtable ( appwindow *me*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| tablename | None | string | |

## getcurrentpath()

### Description

### Prototype

*appwindowvar*.getcurrentpath ( appwindow *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |

# gettablesarray()

## Description

## Prototype

*appwindowvar*.gettablesarray ( appwindow *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |

# opendatatable()

## Description

## Prototype

*appwindowvar*.opendatatable ( appwindow *me*, datasourceinfo *src*, string *tablename*, string *username*, string *password*, integer *retry*, integer *timeout*, boolean *createview*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| src | None | datasourceinfo | |
| tablename | None | string | |
| username | None | string | |
| password | None | string | |
| retry | 1000000 | integer | |
| timeout | 5000000 | integer | |
| createview | .true | boolean | |
| error | None | integer | |

# openformdirect()

## Description

## Prototype

*appwindowvar*.openformdirect ( appwindow *me*, string *filename*, string *appmsgtitle*, type(db1record) *r* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| filename | None | string | |
| appmsgtitle | `Open Form Error` | string | |
| r | None | type(db1record) | |

## resizewindowtoform()

### Description

### Prototype

*appwindowvar*.`resizewindowtoform` ( appwindow *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |

## setcurrentindex()

### Description

### Prototype

*appwindowvar*.`setcurrentindex` ( appwindow *me*, string *indexname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| indexname | None | string | |

## setcurrentpath()

### Description

### Prototype

*appwindowvar*.`setcurrentpath` ( appwindow *me*, string *filename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| filename | None | string | |

### setcurrenttable()

#### Description

#### Prototype

*appwindowvar*.setcurrenttable ( appwindow *me*, tableinfo *tinfo*, boolean *createview*, boolean *skipformcheck* )

#### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | appwindow | |
| tinfo | None | tableinfo | |
| createview | .true | boolean | |
| skipformcheck | .false | boolean | |

# localeinfoold

## Description

## Type Tags

None

## Object Value

Objects of type localeinfoold have no value, and it is an error to try to get or set this value.

## localeinfoold.new()

### Description

### Prototype

*localeinfoold*.new ( localeinfoold *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | localeinfoold | |

## Properties

| Property | Type | Description |
|---|---|---|
| datelocale | SBLlocaledateinfo | |

| Property | Type | Description |
|---|---|---|
| numlocale | SBLNumSettings | |
| type | type | |

# sysinfo

## Description

## Type Tags

None

## Object Value

Objects of type sysinfo have no value, and it is an error to try to get or set this value.

## sysinfo.new()

### Description

### Prototype

*sysinfo*.new ( sysinfo *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sysinfo | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| defaultfont | wxfont | |
| displayheight | integer | |
| displayusable-height | integer | |
| displayus-ablewidth | integer | |
| displaywidth | integer | |
| hscroll-barheight | integer | |

| Property | Type | Description |
|---|---|---|
| maxinner-height | integer | |
| maxinnerwidth | integer | |
| systemcolors | syscolors | |
| type | type | |
| vscrollbar-width | integer | |
| windowsver | windowsversion | |

# tableinfo

## Description

## Type Tags

None

## Object Value

Objects of type tableinfo have no value, and it is an error to try to get or set this value.

## tableinfo.new()

### Description

### Prototype

*tableinfo*.new ( tableinfo *me*, type(appwindow) *appw*, datasourceinfo *source*, type(db1table) *table*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tableinfo | |
| appw | None | type(appwindow) | |
| source | None | datasourceinfo | |
| table | None | type(db1table) | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | |
| __tbinfo | tbinfo | |
| appwin | type(appwindow) | |
| appwinnode | dlistnode | |
| currentidx | type(db1index) | |
| record | type(db1record) | |
| type | type | |

# tdisplayformats

## Description

## Type Tags

None

## Object Value

Objects of type tdisplayformats have no value, and it is an error to try to get or set this value.

## tdisplayformats.new()

### Description

### Prototype

*`tdisplayformats`*.new ( tdisplayformats *`me`*, string *`defboolean`*, string *`definteger`*, string *`defnumber`*, string *`defdate`*, string *`deftime`*, string *`defdatetime`*, localeinfo *`locale`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tdisplayformats | |
| defboolean | `T\|F` | string | |
| definteger | `.` | string | |
| defnumber | `999999.00` | string | |
| defdate | `yyyy.0m.0d` | string | |
| deftime | `hh:mm:ss` | string | |
| defdatetime | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |
| locale | None | localeinfo | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| defboolean | string | |
| defdate | string | |
| defdatetime | string | |
| definteger | string | |
| defnumber | string | |
| deftime | string | |
| type | type | |

# __fillindexlist()

## Description

## Prototype

`__fillindexlist` ( wxformcombo *`cb`*, type(db1table) *`table`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| cb | None | wxformcombo | |
| table | None | type(db1table) | |

# __findformcontrolintoolbar()

## Description

## Prototype

`__findformcontrolintoolbar` ( wxtoolbar *`tb`*, string *`controlname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tb | None | wxtoolbar | |
| controlname | None | string | |

# __formcontrolexistsintoolbar()

## Description

## Prototype

`__formcontrolexistsintoolbar` ( wxtoolbar *`tb`*, string *`controlname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tb | None | wxtoolbar | |
| controlname | None | string | |

# checkneedsave()

## Description

## Prototype

`checkneedsave` ( type(appwindow) *`appw`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| appw | None | type(appwindow) | |

# clearstatusbar()

## Description

## Prototype

`clearstatusbar` ( wxstatusbar *`sb`*, integer *`delay`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sb | None | wxstatusbar | |
| delay | `2000000` | integer | |

# closewindow()

## Description

## Prototype

`closewindow` ( wxwindow *`me`*, type(application) *`app`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxwindow | |
| app | None | type(application) | |

# defer()

## Description

## Prototype

`defer` ( function *`func`*, type(*) *`mevar`*, type(*) *`ref`*, wxform *`form`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| func | None | function | |
| mevar | None | type(*) | |
| ref | None | type(*) | |
| form | None | wxform | |

# deferprocessing()

## Description

## Prototype

`deferprocessing` ( function *`func`*, type(*) *`mevar`*, type(*) *`ref`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| func | None | function | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| mevar | None | type(*) | |
| ref | None | type(*) | |

# deleterecord()

## Description

## Prototype

`deleterecord` ( type(*) *`me`*, type(application) *`app`*, boolean *`dodefer`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | type(*) | |
| app | None | type(application) | |
| dodefer | `.true` | boolean | |

# doformview()

## Description

## Prototype

`doformview` ( type(appwindow) *`appw`*, boolean *`dorefresh`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| appw | None | type(appwindow) | |
| dorefresh | None | boolean | |

# doselrec()

## Description

## Prototype

`doselrec` ( string *`op`*, appwindow *`appw`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| op | None | string | |
| appw | None | appwindow | |

# duplicaterecord()

## Description

## Prototype

`duplicaterecord` ( type(*) *`me`*, type(application) *`app`*, boolean *`dodefer`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | type(*) | |
| app | None | type(application) | |
| dodefer | `.true` | boolean | |

# fieldselection()

## Description

## Prototype

`fieldselection` ( wxmenuitem *`me`*, type(application) *`app`*, boolean *`dodefer`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxmenuitem | |
| app | None | type(application) | |
| dodefer | `.true` | boolean | |

# findfirstfocusablecontrol()

## Description

## Prototype

`findfirstfocusablecontrol` ( dataform1 *`form`*, integer *`pagenum`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| form | None | dataform1 | |
| pagenum | `1` | integer | |

# findmenuinmenubar()

## Description

## Prototype

`findmenuinmenubar` ( wxmenubar *`mb`*, string *`menuname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| mb | None | wxmenubar | |
| menuname | None | string | |

# formview()

## Description

## Prototype

`formview` ( wxmenuitem *`me`*, type(application) *`app`*, boolean *`dorefresh`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxmenuitem | |
| app | None | type(application) | |
| dorefresh | `.true` | boolean | |

# getappwindowfromwindow()

## Description

## Prototype

`getappwindowfromwindow` ( type(wxcontainer) *`w`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| w | None | type(wxcontainer) | |

# getemptyprompt()

## Description

## Prototype

`getemptyprompt` ( type(db1field) *`field`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | type(db1field) | |

# getmenuitemwindow()

## Description

## Prototype

`getmenuitemwindow` ( wxmenuitem *`mi`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| mi | None | wxmenuitem | |

# gettableformatstrings()

## Description

## Prototype

`gettableformatstrings` ( type(db1table) *`table`*, type(appwindow) *`appw`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| appw | None | type(appwindow) | |

# gettablesarray()

## Description

## Prototype

`gettablesarray` ( type(appwindow) *`me`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | type(appwindow) | |

# lookup()

## Description

## Prototype

`lookup` ( type(db1index) *`idx`*, anyvalue *`v`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| idx | None | type(db1index) | |
| v | None | anyvalue | |
| error | None | integer | |

# modifyrecord()

## Description

## Prototype

`modifyrecord` ( type(*) *`me`*, type(application) *`app`*, boolean *`dodefer`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | type(*) | |
| app | None | type(application) | |
| dodefer | `.false` | boolean | |

# newrecord()

## Description

## Prototype

`newrecord` ( type(*) *`me`*, type(application) *`app`*, boolean *`dodefer`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | type(*) | |
| app | None | type(application) | |
| dodefer | `.true` | boolean | |

# removedialogfromlist()

## Description

## Prototype

`removedialogfromlist` ( wxdialog *`me`*, appwindow *`appw`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxdialog | |
| appw | None | appwindow | |

# saverecord()

## Description

## Prototype

`saverecord` ( type(*) *`me`*, type(application) *`app`*, boolean *`dodefer`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | type(*) | |
| app | None | type(application) | |
| dodefer | `.true` | boolean | |

# selectff_rw()

## Description

## Prototype

`selectff_rw` ( appwindow *`appw`*, string *`op`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| appw | None | appwindow | |
| op | None | string | |
| error | None | integer | |

# selectrecord()

## Description

## Prototype

`selectrecord` ( appwindow *`appw`*, string *`operation`*, boolean *`silent`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| appw | None | appwindow | |
| operation | None | string | |
| silent | `.false` | boolean | |
| error | None | integer | |

# selrec()

## Description

## Prototype

`selrec` ( wxtool *`me`*, appwindow *`appw`*, boolean *`dodefer`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxtool | |
| appw | None | appwindow | |
| dodefer | `.true` | boolean | |

# showrecordview()

## Description

## Prototype

`showrecordview` ( type(appwindow) *`appw`*, type(application) *`app`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| appw | None | type(appwindow) | |
| app | None | type(application) | |

# showtableview()

## Description

## Prototype

`showtableview` ( type(appwindow) *`appw`*, type(application) *`app`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| appw | None | type(appwindow) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| app | None | type(application) | |

# windresize()

## Description

## Prototype

`windresize` ( wxwindow *`w`*, type(appwindow) *`appw`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| w | None | wxwindow | |
| appw | None | type(appwindow) | |

# writedataview()

## Description

## Prototype

`writedataview` ( tdataview *`dataview`*, string *`inifilename`*, string *`section`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dataview | None | tdataview | |
| inifilename | None | string | |
| section | `DataView` | string | |

# Chapter 18. boolstr

This library contains the implementation of the formatted output and conversion to string for boolean and datetime objects.

# boolstr()

## Description

Takes a boolean value and converts it to a string using the format string provided. Examples of valid format strings are: T|F and Y|N.

## Prototype

`boolstr` ( boolean *`b`*, string *`format`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| b | None | boolean | |
| format | None | string | |

# datetimestr()

## Description

Formats a datetime and returns it as a string. It supports any of these format strings: "yyyy-mm-dd hh:mm:ss.nnnnnn", "yyyy-mm-dd-hh.mm.ss.nnnnnn", or "yyyy-mm-ddThh:mm:ss.nnn".

## Prototype

`datetimestr` ( datetime *`dt`*, string *`format`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | datetime | |
| format | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |

# string2datetime()

## Description

Converts a string in one of the supported datetime formats and returns it as a datetime object. It supports any of these format strings: "yyyy-mm-dd hh:mm:ss.nnnnnn", "yyyy-mm-dd-hh.mm.ss.nnnnnn", or "yyyy-mm-ddThh:mm:ss.nnn".

# Prototype

`string2datetime` ( string *`s`*, string *`format`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| format | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |

# Chapter 19. bzip2

# bzip2info

## Description

## Type Tags

None

## Object Value

Objects of type bzip2info have no value, and it is an error to try to get or set this value.

# bzip2info.new()

## Description

## Prototype

*bzip2info*.new ( bzip2info *me*, string *bz2libdll*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | bzip2info | |
| bz2libdll | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| bz2libdll | string | |
| currfile | string | |
| isextracting | boolean | |
| jobcurrent | integer | |
| jobtotal | integer | |
| stop | boolean | |
| type | type | |
| usercanceled | boolean | |

| Property | Type | Description |
|---|---|---|
| userreference | type(*) | |
| userupdate | function | |

# createarchive()

## Description

## Prototype

createarchive ( bzip2info *bz2info*, array *filenames*, string *outfile*, boolean *compresstofiles*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bz2info | None | bzip2info | |
| filenames | None | array | |
| outfile | None | string | |
| compresstofiles | .false | boolean | |
| error | None | integer | |

# extractarchive()

## Description

## Prototype

extractarchive ( bzip2info *bz2info*, string *archivename*, string *outtargetdir*, boolean *all*, array *filenames*, boolean *extractcompressedfiles*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bz2info | None | bzip2info | |
| archivename | None | string | |
| outtargetdir | None | string | |
| all | .true | boolean | |
| filenames | None | array | |
| extractcompressedfiles | .false | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# packfile()

## Description

## Prototype

`packfile` ( bzip2info *`bz2info`*, string *`filename`*, string *`outfilename`*, integer *`packsize`*, fsfileoutputstream *`fpo`*, integer *`filesize`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bz2info | None | bzip2info | |
| filename | None | string | |
| outfilename | None | string | |
| packsize | None | integer | |
| fpo | None | fsfileoutputstream | |
| filesize | None | integer | |
| error | None | integer | |

# unpackfile()

## Description

## Prototype

`unpackfile` ( bzip2info *`bz2info`*, string *`filename`*, string *`outfilename`*, fsfileinputstream *`fpi`*, integer *`filesize`*, integer *`filepos`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bz2info | None | bzip2info | |
| filename | None | string | |
| outfilename | None | string | |
| fpi | None | fsfileinputstream | |
| filesize | None | integer | |
| filepos | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# Chapter 20. calceval

## evalnode

### Description

## Type Tags

None

## Object Value

Objects of type evalnode have no value, and it is an error to try to get or set this value.

## evalnode.new()

### Description

### Prototype

*evalnode*.new ( evalnode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | evalnode | |

## Properties

| Property | Type | Description |
|---|---|---|
| argtype | string | |
| escape | string | |
| hasdate | boolean | |
| hasnot | boolean | |
| hastime | boolean | |
| isnumeric | boolean | |
| precedence | integer | |
| setvalues | function | |
| type | type | |
| value | type(*) | |

# Methods

## setvalues()

### Description

### Prototype

*evalnodevar*.setvalues ( evalnode *me*, string *argtype*, integer *precedence*, type(*) *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | evalnode | |
| argtype | None | string | |
| precedence | None | integer | |
| value | None | type(*) | |

# __ce_getfield()

# Description

# Prototype

__ce_getfield ( type(db1table) *table*, string *sFieldname* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| sFieldname | None | string | |

# __lex()

# Description

# Prototype

__lex ( string *s*, SBLNumSettings *numlocale* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| numlocale | None | SBLNumSettings | |

# __rt_getnextvalidtokens()

## Description

## Prototype

__rt_getnextvalidtokens ( array *statement*, array *db1tables*, type *datatype* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| statement | None | array | |
| db1tables | None | array | |
| datatype | None | type | |

# __rt_prep()

## Description

## Prototype

__rt_prep ( array *statement*, array *db1tables* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| statement | None | array | |
| db1tables | None | array | |

# calceval()

## Description

## Prototype

calceval ( string *s*, SBLNumSettings *numlocale*, type(db1record) *r*, integer *error* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| numlocale | None | SBLNumSettings | |
| r | None | type(db1record) | |
| error | None | integer | |

# Chapter 21. codepageslib

This library provides a set of codepage conversion functions. It can be extended to cover codepages not currently supported.

## convert8bitcharval()

### Description

### Prototype

`convert8bitcharval` ( integer *`charval`*, string *`codepage`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| charval | None | integer | |
| codepage | None | string | |

## convert8bitsupported()

### Description

### Prototype

`convert8bitsupported` ( string *`codepage`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| codepage | None | string | |

## convertfrom8bitblob()

### Description

### Prototype

`convertfrom8bitblob` ( blob *`b`*, string *`codepage`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| b | None | blob | |
| codepage | None | string | |

# convertfrom8bitstring()

## Description

## Prototype

`convertfrom8bitstring` ( string *`s`*, string *`codepage`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| codepage | None | string | |

# convertto8bitblob()

## Description

## Prototype

`convertto8bitblob` ( string *`s`*, string *`codepage`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| codepage | None | string | |

# convertto8bitstring()

## Description

## Prototype

`convertto8bitstring` ( string *`s`*, string *`codepage`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| codepage | None | string | |

# Chapter 22. colorpalette

## colorpalette

### Description

## Type Tags

None

## Object Value

Objects of type colorpalette have no value, and it is an error to try to get or set this value.

## colorpalette.new()

### Description

### Prototype

*colorpalette*.new ( colorpalette *me*, string *name*, integer *maxcolors* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |
| name | None | string | |
| maxcolors | 342 | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _private | colorpaletteprivate | |
| add | function | |
| buildstrindex | function | |
| findcolorindex | function | |
| findstrindex | function | |
| getcolorblob | function | |
| getcolorrgbvals | function | |
| getcolorstring | function | |
| getcolorvalue | function | |
| getcount | function | |

| Property | Type | Description |
|---|---|---|
| maxcolors | integer | |
| name | string | |
| type | type | |

# Methods

## add()

### Description

### Prototype

*colorpalettevar*.add ( colorpalette *me*, blob *blobrgb*, integer *rgbvalue*, integer *red*, integer *green*, integer *blue*, string *strindex*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |
| blobrgb | None | blob | |
| rgbvalue | None | integer | |
| red | 0 | integer | |
| green | 0 | integer | |
| blue | 0 | integer | |
| strindex | None | string | |
| error | None | integer | |

## buildstrindex()

### Description

### Prototype

*colorpalettevar*.buildstrindex ( colorpalette *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |

## findcolorindex()

### Description

### Prototype

*colorpalettevar*.findcolorindex ( colorpalette *me*, blob *blobrgb*, integer *rgbvalue*, integer *red*, integer *green*, integer *blue*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |
| blobrgb | None | blob | |
| rgbvalue | None | integer | |
| red | None | integer | |
| green | None | integer | |
| blue | None | integer | |
| error | None | integer | |

## findstrindex()

### Description

### Prototype

*colorpalettevar*.findstrindex ( colorpalette *me*, integer *colorindex*, blob *blobrgb*, integer *rgbvalue*, integer *red*, integer *green*, integer *blue*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |
| colorindex | None | integer | |
| blobrgb | None | blob | |
| rgbvalue | None | integer | |
| red | None | integer | |
| green | None | integer | |
| blue | None | integer | |
| error | None | integer | |

## getcolorblob()

### Description

### Prototype

*colorpalettevar*.getcolorblob ( colorpalette *me*, integer *colorindex*, string *strindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |
| colorindex | `1` | integer | |
| strindex | None | string | |

## getcolorrgbvals()

### Description

### Prototype

*`colorpalettevar`*.`getcolorrgbvals` ( colorpalette *`me`*, integer *`colorindex`*, string *`strindex`*, integer *`red`*, integer *`green`*, integer *`blue`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |
| colorindex | None | integer | |
| strindex | None | string | |
| red | `0` | integer | |
| green | `0` | integer | |
| blue | `0` | integer | |

## getcolorstring()

### Description

### Prototype

*`colorpalettevar`*.`getcolorstring` ( colorpalette *`me`*, integer *`colorindex`*, string *`strindex`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |
| colorindex | None | integer | |
| strindex | None | string | |

## getcolorvalue()

### Description

### Prototype

*colorpalettevar*.getcolorvalue ( colorpalette *me*, integer *colorindex*, string *strindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |
| colorindex | None | integer | |
| strindex | None | string | |

## getcount()

### Description

### Prototype

*colorpalettevar*.getcount ( colorpalette *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | colorpalette | |

# Chapter 23. commonreportgui

## commonaggregateinfo

### Description

### Type Tags

None

### Object Value

Objects of type commonaggregateinfo have no value, and it is an error to try to get or set this value.

## commonaggregateinfo.new()

### Description

### Prototype

*commonaggregateinfo*.new ( commonaggregateinfo *me*, integer *typeid*, string *colname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commonaggregateinfo | |
| typeid | None | integer | |
| colname | None | string | |

### Properties

| Property | Type | Description |
|---|---|---|
| colname | string | |
| type | type | |
| typeid | integer | |

## commonfilterinfo

### Description

### Type Tags

None

# Object Value

Objects of type commonfilterinfo have no value, and it is an error to try to get or set this value.

# commonfilterinfo.new()

## Description

## Prototype

*commonfilterinfo*.new ( commonfilterinfo *me*, wxform *f*, type(dataform1) *form*, type(report1) *report*, string *errmsgtitle*, string *filter*, function *okhandler*, SBLlocaledateinfo *defdatelocale*, string *defdateformat*, update1 *update*, wxform *mainform* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commonfilterinfo | |
| f | None | wxform | |
| form | None | type(dataform1) | |
| report | None | type(report1) | |
| errmsgtitle | `Common Report GUI Error` | string | |
| filter | None | string | |
| okhandler | None | function | |
| defdatelocale | None | SBLlocaledateinfo | |
| defdateformat | `yyyy-0m-0d` | string | |
| update | None | update1 | |
| mainform | None | wxform | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| canceled | boolean | |
| defdateformat | string | |
| defdatelocale | SBLlocaledateinfo | |
| editmode | boolean | |
| errmsgtitle | string | |
| f | wxform | |

| Property | Type | Description |
|---|---|---|
| filter | string | |
| findtable | function | |
| form | type(dataform1) | |
| iscalculation | boolean | |
| ismodify | boolean | |
| isupdate | boolean | |
| mainform | wxform | |
| onassign-whereclause | event | |
| onok | event | |
| parse | function | |
| report | type(report1) | |
| statestack | stack | |
| tablename | string | |
| tables | array | |
| tokenneeded | integer | |
| type | type | |
| update | update1 | |

# Methods

## findtable()

### Description

### Prototype

*commonfilterinfovar*.findtable ( commonfilterinfo *me*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commonfilterinfo | |
| tablename | None | string | |

## parse()

### Description

### Prototype

*commonfilterinfovar*.parse ( commonfilterinfo *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commonfilterinfo | |

# commongroupinfo

## Description

## Type Tags

None

## Object Value

Objects of type commongroupinfo have no value, and it is an error to try to get or set this value.

## commongroupinfo.new()

### Description

### Prototype

*commongroupinfo*.new ( commongroupinfo *me*, dring *parent*, string *colname*, boolean *docount* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commongroupinfo | |
| parent | None | dring | |
| colname | None | string | |
| docount | `.false` | boolean | |

## Properties

| Property | Type | Description |
|---|---|---|
| addagg | function | |
| agginfo | array | |
| colname | string | |
| docount | boolean | |
| findaggbycol-name | function | |

| Property | Type | Description |
|---|---|---|
| findaggbycol-nameandtype | function | |
| node | dlistnode | |
| removeagg | function | |
| type | type | |

# Methods

## addagg()

### Description

### Prototype

*commongroupinfovar*.addagg ( commongroupinfo *me*, integer *typeid*, string *colname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commongroupinfo | |
| typeid | None | integer | |
| colname | None | string | |

## findaggbycolname()

### Description

### Prototype

*commongroupinfovar*.findaggbycolname ( commongroupinfo *me*, string *colname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commongroupinfo | |
| colname | None | string | |

## findaggbycolnameandtype()

### Description

### Prototype

*commongroupinfovar*.findaggbycolnameandtype ( commongroupinfo *me*, string *colname*, integer *typeid* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commongroupinfo | |
| colname | None | string | |
| typeid | None | integer | |

## removeagg()

### Description

### Prototype

*commongroupinfovar*.removeagg ( commongroupinfo *me*, string *colname*, integer *typeid* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commongroupinfo | |
| colname | None | string | |
| typeid | None | integer | |

# commonorderinfo

# Description

# Type Tags

None

# Object Value

Objects of type commonorderinfo have no value, and it is an error to try to get or set this value.

# commonorderinfo.new()

## Description

## Prototype

*commonorderinfo*.new ( commonorderinfo *me*, report1 *report*, string *errmsgtitle*, string *order*, function *okhandler*, type(*) *ref* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | commonorderinfo | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| errmsgtitle | `Common Report GUI Error` | string | |
| order | None | string | |
| okhandler | None | function | |
| ref | None | type(*) | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| canceled | boolean | |
| errmsgtitle | string | |
| onok | event | |
| order | string | |
| ref | type(*) | |
| report | report1 | |
| type | type | |

# linkmanager

## Description

## Type Tags

None

## Object Value

Objects of type linkmanager have no value, and it is an error to try to get or set this value.

## linkmanager.new()

### Description

### Prototype

*`linkmanager`*.new ( linkmanager *`me`*, array *`linkitems`*, type(*) *`linkbase`*, boolean *`allowlinktolink`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | linkmanager | |
| linkitems | None | array | |
| linkbase | None | type(*) | |
| allowlinktolink | `.true` | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| allowlinktolink | boolean | |
| changed | boolean | |
| colors | syscolors | |
| detailblockim-age | wxbitmap | |
| dlg | wxfont | |
| dlgbold | wxfont | |
| frame | wxwindow | |
| gridimage | wxbitmap | |
| isselected | boolean | |
| linkbase | type(*) | |
| main | wxwindow | |
| mainform | linkform | |
| onclose | event | |
| panel | wxwindow | |
| panelform | linkpanelform | |
| running | boolean | |
| selecteditem | linkhalf | |
| state | string | |
| tableimage | wxbitmap | |
| type | type | |

# __rep_flt_cancel_oc()

# Description

# Prototype

`__rep_flt_cancel_oc` ( wxformbutton *`me`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformbutton | |

# __rep_flt_clear_oc()

## Description

## Prototype

__rep_flt_clear_oc ( wxformbutton *me*, commonfilterinfo *finfo* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformbutton | |
| finfo | None | commonfilterinfo | |

# __rep_flt_cols()

## Description

## Prototype

__rep_flt_cols ( wxformcombo *me*, commonfilterinfo *finfo* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformcombo | |
| finfo | None | commonfilterinfo | |

# __rep_flt_columns_osc()

## Description

## Prototype

__rep_flt_columns_osc ( wxformlist *me*, commonfilterinfo *finfo* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformlist | |
| finfo | None | commonfilterinfo | |

# __rep_flt_filter_edit()

## Description

## Prototype

`__rep_flt_filter_edit` ( wxformedittext *`me`*, commonfilterinfo *`finfo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformedittext | |
| finfo | None | commonfilterinfo | |

# __rep_flt_filter_edit_oc()

## Description

## Prototype

`__rep_flt_filter_edit_oc` ( wxformedittext *`me`*, commonfilterinfo *`finfo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformedittext | |
| finfo | None | commonfilterinfo | |

# __rep_flt_join()

## Description

## Prototype

`__rep_flt_join` ( wxformbutton *`me`*, commonfilterinfo *`finfo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformbutton | |
| finfo | None | commonfilterinfo | |

# __rep_flt_ok_oc()

## Description

## Prototype

`__rep_flt_ok_oc` ( wxformbutton *`me`*, commonfilterinfo *`finfo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformbutton | |
| finfo | None | commonfilterinfo | |

# __rep_flt_op()

## Description

## Prototype

`__rep_flt_op` ( wxformbutton *`me`*, commonfilterinfo *`finfo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformbutton | |
| finfo | None | commonfilterinfo | |

# __rep_flt_val()

## Description

## Prototype

`__rep_flt_val` ( wxformbutton *`me`*, commonfilterinfo *`finfo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformbutton | |
| finfo | None | commonfilterinfo | |

# __rep_movedown()

## Description

## Prototype

`__rep_movedown` ( wxformlist *`lb`*, integer *`basepos`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lb | None | wxformlist | |
| basepos | `1` | integer | |

# __rep_movelast()

## Description

## Prototype

`__rep_movelast` ( wxformlist *`lb`*, integer *`basepos`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lb | None | wxformlist | |
| basepos | `1` | integer | |

# __rep_movetop()

## Description

## Prototype

`__rep_movetop` ( wxformlist *`lb`*, integer *`basepos`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lb | None | wxformlist | |
| basepos | `1` | integer | |

# __rep_moveup()

## Description

## Prototype

`__rep_moveup` ( wxformlist *`lb`*, integer *`basepos`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lb | None | wxformlist | |
| basepos | `1` | integer | |

# dofilter()

## Description

## Prototype

`dofilter` ( report1 *`report`*, type(wxdialogparent) *`w`*, array *`tables`*, string *`filter`*, string *`newclause`*, function *`okhandler`*, boolean *`allowtablechange`*, boolean *`showjoinbutton`*, string *`deftablename`*, boolean *`iscalculation`*, string *`asname`*, boolean *`ismodify`*, wxform *`mainform`*, string *`errmsgtitle`*, type(*) *`ref`*, type(dataform1) *`form`*, string *`caption`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| w | None | type(wxdialogparent) | |
| tables | None | array | |
| filter | None | string | |
| newclause | None | string | |
| okhandler | None | function | |
| allowtablechange | `.true` | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| showjoinbutton | `.true` | boolean | |
| deftablename | None | string | |
| iscalculation | `.false` | boolean | |
| asname | None | string | |
| ismodify | `.false` | boolean | |
| mainform | None | wxform | |
| errmsgtitle | `Common Report GUI Error` | string | |
| ref | None | type(*) | |
| form | None | type(dataform1) | |
| caption | `Superbase NG Quick Report Filter Definition` | string | |

# doreportorder()

## Description

## Prototype

`doreportorder` ( report1 *`report`*, type(wxdialogparent) *`w`*, string *`order`*, function *`okhandler`*, string *`errmsgtitle`*, type(*) *`ref`*, string *`caption`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| w | None | type(wxdialogparent) | |
| order | None | string | |
| okhandler | None | function | |
| errmsgtitle | `Common Report GUI Error` | string | |
| ref | None | type(*) | |
| caption | `Superbase NG Quick Report Sort Order Definition` | string | |

# qrfilterfrm()

## Description

## Prototype

`qrfilterfrm` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# uparrowoutputrow()

## Description

## Prototype

`uparrowoutputrow` ( integer *`spaces`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| spaces | None | integer | |

# Chapter 24. conflib

This library is the configuration management library. It is used for storing and retrieving configuration information from various formats. Currently the INI file format is supported, with functions for reading and writing values to the file. These are cross-platform, not specific to the Windows platform.

# getallprofilestrings()

## Description

## Prototype

`getallprofilestrings` ( string *`sSection`*, string *`sInifile`*, string *`sEOLchar`*, string *`sInifilecontent`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSection | None | string | |
| sInifile | None | string | |
| sEOLchar | None | string | |
| sInifilecontent | None | string | |

# getprivateprofilestring()

## Description

## Prototype

`getprivateprofilestring` ( string *`sSection`*, string *`sEntry`*, string *`sDefault`*, string *`sReturnval`*, string *`sInifile`*, string *`sEOLchar`*, string *`sInifilecontent`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSection | None | string | |
| sEntry | None | string | |
| sDefault | None | string | |
| sReturnval | None | string | |
| sInifile | None | string | |
| sEOLchar | None | string | |
| sInifilecontent | None | string | |

# openinifile()

## Description

## Prototype

`openinifile` ( string *`sInifile`*, string *`sEOLchar`*, boolean *`usesBOM`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sInifile | None | string | |
| sEOLchar | None | string | |
| usesBOM | None | boolean | |

# writeprivateprofilestring()

## Description

## Prototype

`writeprivateprofilestring` ( string *`sSection`*, string *`sEntry`*, string *`sValue`*, string *`sInifile`*, string *`sEOLchar`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSection | None | string | |
| sEntry | None | string | |
| sValue | None | string | |
| sInifile | None | string | |
| sEOLchar | None | string | |

# writeprivateprofilestrings()

## Description

## Prototype

`writeprivateprofilestrings` ( array *`items`*, string *`sInifile`*, string *`sEOLchar`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| items | None | array | |
| sInifile | None | string | |
| sEOLchar | None | string | |

# Chapter 25. consolelib

## tConsole

### Description

### Type Tags

None

### Object Value

Objects of type tConsole have no value, and it is an error to try to get or set this value.

## tConsole.new()

### Description

### Prototype

*tConsole*.new( tConsole *me*, integer *left*, integer *top*, integer *width*, integer *height*, string *caption*, integer *backcolor*, integer *textcolor*, function *onclose*, type(*) *oncloseref*, string *fontname*, integer *fontsize*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tConsole | |
| left | `1` | integer | |
| top | `1` | integer | |
| width | `600` | integer | |
| height | `400` | integer | |
| caption | `Console` | string | |
| backcolor | `0` | integer | |
| textcolor | `16777215` | integer | |
| onclose | None | function | |
| oncloseref | None | type(*) | |
| fontname | `Lucida Console` | string | |
| fontsize | `9` | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _private | tConsolePrivate | |
| caption | string | |
| clear | function | |
| close | function | |
| framewindow | wxwindow | |
| inputform | wxform | |
| inputtext | wxformedittext | |
| inputwindow | wxwindow | |
| onclose | event | |
| outputform | wxform | |
| outputtext | wxformedittext | |
| outputwindow | wxwindow | |
| read | function | |
| type | type | |
| write | function | |

# Methods

## clear()

### Description

### Prototype

*tConsolevar*.clear ( tConsole *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tConsole | |

## close()

### Description

### Prototype

*tConsolevar*.close ( tConsole *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tConsole | |

# read()

## Description

## Prototype

*tConsolevar*.read ( tConsole *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tConsole | |

# write()

## Description

## Prototype

*tConsolevar*.write ( tConsole *me*, string *text* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tConsole | |
| text | None | string | |

# Chapter 26. databaseforms

The databaseforms library is the implementation of data-aware display and print forms for SIMPOL. The dataform1 family of objects are a wrapper around the wxform* objects and the printform1 family of objects are a wrapper around the print architecture supplied by the WXWN component. The databaseforms library provides a multi-page, data-aware layer that includes the data-aware bitmap, data-aware scrollbar, data-aware grid, and detail block implentations, among others. It also provides the grouping mechanism for option buttons. This library is key to efficiently creating database-aware GUIs using SIMPOL. One significant difference between dataform1* and the native wxform* types is the use of drings in dataform1 for collections of objects. To traverse a dring use the dring.`getfirst()` to retrieve the first object, and then the `getnext()` method of the appropriate node property of the object to get to the next.

# dataform1

## Description

## Type Tags

dataform1linkcontainer, dataform1

## Object Value

Objects of type dataform1 have no value, and it is an error to try to get or set this value.

## dataform1.new()

### Description

### Prototype

*`dataform1`*.new ( dataform1 *`me`*, integer *`defpagewidth`*, integer *`defpageheight`*, integer *`defpagebackcolor`*, string *`defbooleanformat`*, string *`defintegerformat`*, string *`defnumberformat`*, string *`defdateformat`*, string *`deftimeformat`*, string *`defdatetimeformat`*, SBLocaledateinfo *`defdatelocale`*, SBLNumSettings *`defnumericlocale`*, wxfont *`deffont`*, integer *`designdpi`*, string *`currentworkingdirectory`*, boolean *`loading`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| defpagewidth | `50` | integer | |
| defpageheight | `50` | integer | |
| defpagebackcolor | `16777215` | integer | |
| defbooleanformat | `T\|F` | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defintegerformat | `.` | string | |
| defnumberformat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimeformat | None | string | |
| defdatelocale | None | SBLlocaledateinfo | |
| defnumericlocale | None | SBLNumSettings | |
| deffont | None | wxfont | |
| designdpi | `96` | integer | |
| currentworkingdirectory | None | string | |
| loading | `.false` | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | dataform1private | |
| addbitmap | function | |
| addcontrol | function | |
| adddatasource | function | |
| adddetailblock | function | |
| addgraphic | function | |
| addlink | function | |
| addoptiongroup | function | |
| addpage | function | |
| addsiblinglink | function | |
| addtable | function | |
| assignfilterobject | function | |
| autolocking | boolean | |
| bitmaps | dring | |
| blank | function | |

| Property | Type | Description |
|---|---|---|
| check-dirtyrecords | function | |
| clearlinks | function | |
| clearsib-linglinks | function | |
| container | type(wxcontainer) | |
| controls | dring | |
| currentdpi | integer | |
| currentpage | dataform1page | |
| currentwork-ingdirectory | string | |
| datasources | dring | |
| defbooleanfor-mat | string | |
| defdateformat | string | |
| defdatelocale | SBLlocaledateinfo | |
| defdatetimefor-mat | string | |
| deffont | wxfont | |
| defintegerfor-mat | string | |
| defnumberfor-mat | string | |
| defnumericlo-cale | SBLNumSettings | |
| defpageback-color | integer | |
| defpageheight | integer | |
| defpagewidth | integer | |
| deftimeformat | string | |
| deleterecord | function | |
| designdpi | integer | |
| designmode | boolean | |
| detailblocks | dring | |
| dirty | boolean | |
| discardrecord | function | |
| dpiadjfactor | number | |
| duplicaterecord | function | |
| filename | string | |
| filllists | function | |

| Property | Type | Description |
|---|---|---|
| filter | dataform1filter | |
| findbitmap-source | function | |
| findcontrol | function | |
| finddatasource | function | |
| findgraphic | function | |
| findnextfocus-ablecontrol | function | |
| findsiblinglink | function | |
| findtable | function | |
| fonts | array | |
| getfieldand-table | function | |
| getfont | function | |
| getlinke-drecord | function | |
| graphics | dring | |
| lastusedrecord | dataform1record | |
| links | dring | |
| loading | boolean | |
| lock | function | |
| locked | boolean | |
| masterrecord | dataform1record | |
| mastertable | dataform1table | |
| name | string | |
| nameinuse | function | |
| newrecord | function | |
| obgroups | dring | |
| onchangere-cord | event | |
| ondelete | event | |
| ondiscard | event | |
| onkey | event | |
| onkeylostfocus | event | |
| onnewrecord | event | |
| onsave | event | |
| onselect | event | |
| pages | dring | |
| preventfocus | boolean | |

| Property | Type | Description |
|---|---|---|
| preventfocus-mode | boolean | |
| refresh | function | |
| saverecord | function | |
| selectcurrent | function | |
| selectfirst | function | |
| selectkey | function | |
| selectlast | function | |
| selectnext | function | |
| selectprevious | function | |
| setcontainer | function | |
| setdirtystate | function | |
| setfilter | function | |
| setkeyfocus | function | |
| setlastuse-drecord | function | |
| setmaster-record | function | |
| setmastertable | function | |
| showpage | function | |
| siblinglinks | dring | |
| tables | dring | |
| type | type | |
| unlock | function | |
| valid | boolean | |

# Methods

## !()

### Description

### Prototype

*dataform1var*.! ( dataform1 *me*, string *controlname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| controlname | None | string | |

# addbitmap()

## Description

## Prototype

*dataform1var*.addbitmap ( dataform1 *me*, string *filename*, string *format*, blob *rgb*, integer *width*, integer *height*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| filename | None | string | |
| format | xpm | string | |
| rgb | None | blob | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# addcontrol()

## Description

## Prototype

*dataform1var*.addcontrol ( dataform1 *me*, type *controltype*, integer *left*, integer *top*, integer *width*, integer *height*, string *text*, boolean *enabled*, boolean *visible*, wxbitmap *bitmap*, string *scaling*, wxbitmap *selectedbitmap*, wxbitmap *disabledbitmap*, wxbitmap *focusbitmap*, integer *backgroundrgb*, integer *textrgb*, integer *rgb*, string *edittype*, string *selectiontype*, integer *rowcount*, integer *colcount*, integer *rowheight*, integer *colwidth*, boolean *rowheightdraggable*, boolean *colwidthdraggable*, integer *rowlabelwidth*, integer *collabelheight*, string *rowlabelalignment*, string *collabelalignment*, string *alignment*, string *editstyle*, string *orientation*, integer *range*, integer *position*, integer *pagesize*, integer *thumbsize*, wxfont *font*, wxfont *labelfont*, string *tooltip*, integer *onmousemask*, string *name*, type(dataform1control) *next*, dataform1page *page*, anyvalue *valueon*, anyvalue *valueoff*, type(db1field) *field*, dataform1table *table*, string *displayformat*, dataform1optiongroup *obgroup*, boolean *suppressfill*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| controltype | None | type | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | |
| enabled | `.true` | boolean | |
| visible | `.true` | boolean | |
| bitmap | None | wxbitmap | |
| scaling | None | string | |
| selectedbitmap | None | wxbitmap | |
| disabledbitmap | None | wxbitmap | |
| focusbitmap | None | wxbitmap | |
| backgroundrgb | None | integer | |
| textrgb | `0` | integer | |
| rgb | None | integer | |
| edittype | `droplist` | string | |
| selectiontype | `single` | string | |
| rowcount | `1` | integer | |
| colcount | `1` | integer | |
| rowheight | `20` | integer | |
| colwidth | `80` | integer | |
| rowheightdrag-gable | `.true` | boolean | |
| colwidthdrag-gable | `.true` | boolean | |
| rowlabelwidth | `80` | integer | |
| collabelheight | `20` | integer | |
| rowlabelalign-ment | `right` | string | |
| collabelalign-ment | `left,top` | string | |
| alignment | `left,top` | string | |
| editstyle | None | string | |
| orientation | None | string | |
| range | `1` | integer | |
| position | `0` | integer | |
| pagesize | `1` | integer | |
| thumbsize | `1` | integer | |
| font | None | wxfont | |
| labelfont | None | wxfont | |
| tooltip | None | string | |
| onmousemask | `0` | integer | |
| name | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(dataform1control) | |
| page | None | dataform1page | |
| valueon | None | anyvalue | |
| valueoff | None | anyvalue | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| obgroup | None | dataform1optiongroup | |
| suppressfill | `.false` | boolean | |
| error | None | integer | |

# adddatasource()

## Description

## Prototype

*dataform1var*.`adddatasource` ( dataform1 *me*, type(*) *datasource*, string *source*, string *username*, string *password*, integer *codepage*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| datasource | None | type(*) | |
| source | None | string | |
| username | None | string | |
| password | None | string | |
| codepage | None | integer | |
| error | None | integer | |

# adddetailblock()

## Description

## Prototype

*dataform1var*.`adddetailblock` ( dataform1 *me*, dataform1page *page*, array *controls*, integer *rows*, integer *rowoffset*, integer *columns*, integer *columnoffset*, string *scrollbar*, integer *scrollbaroffset*, boolean *tabacross*, string *name*, boolean *readonly*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| page | None | dataform1page | |
| controls | None | array | |
| rows | `1` | integer | |
| rowoffset | `30` | integer | |
| columns | `1` | integer | |
| columnoffset | `200` | integer | |
| scrollbar | `right` | string | |
| scrollbaroffset | `15` | integer | |
| tabacross | `.true` | boolean | |
| name | None | string | |
| readonly | `.true` | boolean | |
| error | None | integer | |

# addgraphic()

## Description

## Prototype

*`dataform1var`*.`addgraphic` ( dataform1 *`me`*, type *`graphictype`*, point *`point1`*, point *`point2`*, point *`point3`*, point *`midpoint`*, integer *`rgb`*, integer *`borderrgb`*, integer *`width`*, integer *`borderwidth`*, boolean *`visible`*, boolean *`bordervisible`*, string *`name`*, type(dataform1graphic) *`next`*, dataform1page *`page`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| graphictype | None | type | |
| point1 | None | point | |
| point2 | None | point | |
| point3 | None | point | |
| midpoint | None | point | |
| rgb | None | integer | |
| borderrgb | None | integer | |
| width | None | integer | |
| borderwidth | None | integer | |
| visible | None | boolean | |
| bordervisible | None | boolean | |
| name | None | string | |
| next | None | type(dataform1graphic) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| page | None | dataform1page | |
| error | None | integer | |

# addlink()

## Description

## Prototype

*dataform1var*.addlink ( dataform1 *me*, type(db1field) *srcfield*, dataform1table *srctable*, type(db1field) *destfield*, dataform1table *desttable*, type(dataform1linkcontainer) *container*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| container | None | type(dataform1linkcontainer) | |
| error | None | integer | |

# addoptiongroup()

## Description

## Prototype

*dataform1var*.addoptiongroup ( dataform1 *me*, string *name*, type(db1field) *field*, dataform1table *table*, string *displayformat*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| name | None | string | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| error | None | integer | |

# addpage()

## Description

## Prototype

*dataform1var*.addpage ( dataform1 *me*, integer *width*, integer *height*, integer *backgroundrgb*, dataform1page *after*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| width | None | integer | |
| height | None | integer | |
| backgroundrgb | None | integer | |
| after | None | dataform1page | |
| error | None | integer | |

# addsiblinglink()

## Description

## Prototype

*dataform1var*.addsiblinglink ( dataform1 *me*, type(db1field) *srcfield*, dataform1table *srctable*, type(db1field) *destfield*, dataform1table *desttable*, type(dataform1linkcontainer) *container*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| container | None | type(dataform1linkcontainer) | |
| error | None | integer | |

# addtable()

## Description

### Prototype

*dataform1var*.addtable ( dataform1 *me*, type(db1table) *table*, dataform1datasource *source*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| table | None | type(db1table) | |
| source | None | dataform1datasource | |
| error | None | integer | |

## assignfilterobject()

### Description

### Prototype

*dataform1var*.assignfilterobject ( dataform1 *me*, dataform1filter *dffilter* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| dffilter | None | dataform1filter | |

## blank()

### Description

### Prototype

*dataform1var*.blank ( dataform1 *me*, boolean *datacontrolsonly* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| datacontrolsonly | .true | boolean | |

## checkdirtyrecords()

### Description

### Prototype

*dataform1var*.checkdirtyrecords ( dataform1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |

## clearlinks()

### Description

### Prototype

*dataform1var*.clearlinks ( dataform1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |

## clearsiblinglinks()

### Description

### Prototype

*dataform1var*.clearsiblinglinks ( dataform1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |

## deleterecord()

### Description

### Prototype

*dataform1var*.deleterecord ( dataform1 *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# discardrecord()

## Description

## Prototype

*dataform1var*.discardrecord ( dataform1 *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| error | None | integer | |

# duplicaterecord()

## Description

## Prototype

*dataform1var*.duplicaterecord ( dataform1 *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| error | None | integer | |

# filllists()

## Description

## Prototype

*dataform1var*.filllists ( dataform1 *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |

# findbitmapsource()

## Description

## Prototype

*dataform1var*.findbitmapsource ( dataform1 *me*, string *sourcename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| sourcename | None | string | |

# findcontrol()

## Description

## Prototype

*dataform1var*.findcontrol ( dataform1 *me*, string *controlname* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| controlname | None | string | |

# finddatasource()

## Description

## Prototype

*dataform1var*.finddatasource ( dataform1 *me*, string *sourcename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| sourcename | None | string | |

# findgraphic()

## Description

### Prototype

*dataform1var*.findgraphic ( dataform1 *me*, string *graphicname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| graphicname | None | string | |

## findsiblinglink()

### Description

### Prototype

*dataform1var*.findsiblinglink ( dataform1 *me*, dataform1table *desttable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| desttable | None | dataform1table | |

## findtable()

### Description

### Prototype

*dataform1var*.findtable ( dataform1 *me*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| tablename | None | string | |

## getfieldandtable()

### Description

### Prototype

*dataform1var*.getfieldandtable ( dataform1 *me*, string *fieldname*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| fieldname | None | string | |
| tablename | None | string | |

## getfont()

### Description

### Prototype

*dataform1var*.getfont ( dataform1 *me*, string *facename*, integer *size*, string *style*, string *weight*, string *decoration* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| facename | None | string | |
| size | None | integer | |
| style | None | string | |
| weight | None | string | |
| decoration | None | string | |

## getlinkedrecord()

### Description

### Prototype

*dataform1var*.getlinkedrecord ( dataform1 *me*, dataform1table *desttable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| desttable | None | dataform1table | |

## lock()

### Description

### Prototype

*dataform1var*.lock ( dataform1 *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| error | None | integer | |

## nameinuse()

### Description

### Prototype

*dataform1var*.nameinuse ( dataform1 *me*, string *controlname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| controlname | None | string | |

## newrecord()

### Description

### Prototype

*dataform1var*.newrecord ( dataform1 *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| error | None | integer | |

## refresh()

### Description

### Prototype

*dataform1var*.refresh ( dataform1 *me*, boolean *clearfocus* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| clearfocus | `.true` | boolean | |

## saverecord()

### Description

### Prototype

*`dataform1var`*`.saverecord` ( dataform1 *`me`*, boolean *`lock`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| lock | `.false` | boolean | |
| error | None | integer | |

## selectcurrent()

### Description

### Prototype

*`dataform1var`*`.selectcurrent` ( dataform1 *`me`*, type(db1index) *`index`*, boolean *`lock`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| index | None | type(db1index) | |
| lock | `.false` | boolean | |
| error | None | integer | |

## selectfirst()

### Description

### Prototype

*`dataform1var`*`.selectfirst` ( dataform1 *`me`*, boolean *`lock`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| lock | `.false` | boolean | |
| error | None | integer | |

## selectkey()

### Description

### Prototype

*`dataform1var`*.`selectkey` ( dataform1 *`me`*, anyvalue *`value`*, type(db1index) *`index`*, boolean *`lock`*, boolean *`found`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| value | None | anyvalue | |
| index | None | type(db1index) | |
| lock | `.false` | boolean | |
| found | None | boolean | |
| error | None | integer | |

## selectlast()

### Description

### Prototype

*`dataform1var`*.`selectlast` ( dataform1 *`me`*, boolean *`lock`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| lock | `.false` | boolean | |
| error | None | integer | |

## selectnext()

### Description

### Prototype

*dataform1var*.selectnext ( dataform1 *me*, boolean *lock*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| lock | .false | boolean | |
| error | None | integer | |

## selectprevious()

### Description

### Prototype

*dataform1var*.selectprevious ( dataform1 *me*, boolean *lock*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| lock | .false | boolean | |
| error | None | integer | |

## setcontainer()

### Description

### Prototype

*dataform1var*.setcontainer ( dataform1 *me*, type(wxcontainer) *container* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| container | None | type(wxcontainer) | |

## setdirtystate()

### Description

### Prototype

*dataform1var*.setdirtystate ( dataform1 *me*, boolean *dirty* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| dirty | .true | boolean | |

## setfilter()

### Description

### Prototype

*dataform1var*.setfilter ( dataform1 *me*, string *filter*, string *errtext*, integer *errindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| filter | None | string | |
| errtext | None | string | |
| errindex | None | integer | |

## setkeyfocus()

### Description

### Prototype

*dataform1var*.setkeyfocus ( dataform1 *me*, function *onkey*, type(*) *onkeyreference*, function *onlostfocus*, type(*) *onlostfocusreference* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| onkey | None | function | |
| onkeyreference | None | type(*) | |
| onlostfocus | None | function | |
| onlostfocusreference | None | type(*) | |

# setlastusedrecord()

## Description

## Prototype

*dataform1var*.setlastusedrecord ( dataform1 *me*, dataform1record *record* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| record | None | dataform1record | |

# setmasterrecord()

## Description

## Prototype

*dataform1var*.setmasterrecord ( dataform1 *me*, type(db1record) *r*, boolean *dorefresh*, boolean *retaindetailblockpositions*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| r | None | type(db1record) | |
| dorefresh | .true | boolean | |
| retaindetail-blockpositions | .false | boolean | |
| error | None | integer | |

# setmastertable()

## Description

## Prototype

*dataform1var*.setmastertable ( dataform1 *me*, dataform1table *table* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | dataform1table | |

## showpage()

### Description

### Prototype

*dataform1var*.showpage ( dataform1 *me*, integer *pagenum*, boolean *clearfocus* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |
| pagenum | None | integer | |
| clearfocus | .true | boolean | |

## unlock()

### Description

### Prototype

*dataform1var*.unlock ( dataform1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1 | |

# dataform1arc

# Description

# Type Tags

dataform1graphic

# Object Value

Objects of type dataform1arc have no value, and it is an error to try to get or set this value.

# dataform1arc.new()

## Description

## Prototype

*dataform1arc*.new ( dataform1arc *me*, dataform1page *page*, wxgraphicarc *graphic*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1arc | |
| page | None | dataform1page | |
| graphic | None | wxgraphicarc | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| borderrgbid | integer | |
| form | dataform1 | |
| formnode | dlistnode | |
| getarcquadrant | function | |
| getboundingrectangle | function | |
| graphic | wxgraphicarc | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| rgbid | integer | |
| setname | function | |
| setposition | function | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## getarcquadrant()

### Description

### Prototype

*dataform1arcvar*.getarcquadrant ( dataform1arc *me*, type(point) *p* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1arc | |
| p | None | type(point) | |

## getboundingrectangle()

### Description

### Prototype

*dataform1arcvar*.getboundingrectangle ( dataform1arc *me*, point *point1*, point *point2* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1arc | |
| point1 | None | point | |
| point2 | None | point | |

## remove()

### Description

### Prototype

*dataform1arcvar*.remove ( dataform1arc *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1arc | |

## setname()

### Description

### Prototype

*dataform1arcvar*.setname ( dataform1arc *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1arc | |
| name | None | string | |
| error | None | integer | |

## setposition()

### Description

### Prototype

*dataform1arcvar*.setposition ( dataform1arc *me*, point *point1*, point *point2*, point *point3*, point *midpoint*, integer *width*, integer *borderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1arc | |
| point1 | None | point | |
| point2 | None | point | |
| point3 | None | point | |
| midpoint | None | point | |
| width | None | integer | |
| borderwidth | None | integer | |
| error | None | integer | |

# dataform1bitmap

# Description

# Type Tags

dataform1control

# Object Value

Objects of type dataform1bitmap have no value, and it is an error to try to get or set this value.

# dataform1bitmap.new()

## Description

## Prototype

*dataform1bitmap*.new ( dataform1bitmap *me*, dataform1page *page*, wxformbitmap *control*, type(db1field) *field*, dataform1table *table*, string *displayformat*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmap | |
| page | None | dataform1page | |
| control | None | wxformbitmap | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | wxformbitmap | |
| controlsource | dataform1controlsource | |
| empty | wxbitmap | |
| emptycolor | integer | |
| form | dataform1 | |
| formnode | dlistnode | |
| missing | wxbitmap | |
| missingcolor | integer | |
| missinglinecol-or | integer | |
| onmouse | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setbitmaps | function | |
| setcolors | function | |
| setempty | function | |
| setmissing | function | |
| setname | function | |

| Property | Type | Description |
|---|---|---|
| setnext | function | |
| setposition | function | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## remove()

### Description

### Prototype

*dataform1bitmapvar*.remove ( dataform1bitmap *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmap | |

## setbitmaps()

### Description

### Prototype

*dataform1bitmapvar*.setbitmaps ( dataform1bitmap *me*, wxbitmap *emptybmp*, wxbitmap *missingbmp* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmap | |
| emptybmp | None | wxbitmap | |
| missingbmp | None | wxbitmap | |

## setcolors()

### Description

### Prototype

*dataform1bitmapvar*.setcolors ( dataform1bitmap *me*, integer *emptycolor*, integer *missingcolor*, integer *missinglinecolor* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmap | |
| emptycolor | None | integer | |
| missingcolor | None | integer | |
| missinglinecol-or | None | integer | |

## setempty()

### Description

### Prototype

*dataform1bitmapvar*.setempty ( dataform1bitmap *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmap | |
| error | None | integer | |

## setmissing()

### Description

### Prototype

*dataform1bitmapvar*.setmissing ( dataform1bitmap *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmap | |
| error | None | integer | |

## setname()

### Description

### Prototype

*dataform1bitmapvar*.setname ( dataform1bitmap *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmap | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1bitmapvar*.setnext ( dataform1bitmap *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmap | |
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1bitmapvar*.setposition ( dataform1bitmap *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmap | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# dataform1bitmapbutton

## Description

# Type Tags

dataform1control

# Object Value

Objects of type dataform1bitmapbutton have no value, and it is an error to try to get or set this value.

# dataform1bitmapbutton.new()

## Description

## Prototype

*dataform1bitmapbutton*.new ( dataform1bitmapbutton *me*, dataform1page *page*, wxformbitmapbutton *control*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmapbutton | |
| page | None | dataform1page | |
| control | None | wxformbitmapbutton | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | wxformbitmapbutton | |
| displayformat | string | |
| form | dataform1 | |
| formnode | dlistnode | |
| onclick | event | |
| ongotfocus | event | |
| onlostfocus | event | |
| onmouse | event | |
| page | dataform1page | |
| pagenode | dlistnode | |

| Property | Type | Description |
|---|---|---|
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## remove()

### Description

### Prototype

*dataform1bitmapbuttonvar*.remove ( dataform1bitmapbutton *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmapbutton | |

## setname()

### Description

### Prototype

*dataform1bitmapbuttonvar*.setname ( dataform1bitmapbutton *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmapbutton | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1bitmapbuttonvar*.setnext ( dataform1bitmapbutton *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmapbutton | |
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1bitmapbuttonvar*.setposition ( dataform1bitmapbutton *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmapbutton | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# dataform1bitmapsource

# Description

# Type Tags

None

# Object Value

Objects of type dataform1bitmapsource have no value, and it is an error to try to get or set this value.

## dataform1bitmapsource.new()

### Description

## Prototype

*dataform1bitmapsource*.new ( dataform1bitmapsource *me*, type(dataform1) *form*, dring *parent*, string *filename*, string *format*, blob *rgb*, integer *width*, integer *height*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1bitmapsource | |
| form | None | type(dataform1) | |
| parent | None | dring | |
| filename | None | string | |
| format | xpm | string | |
| rgb | None | blob | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| bitmap | wxbitmap | |
| filename | string | |
| form | type(dataform1) | |
| format | string | |
| formnode | dlistnode | |
| type | type | |

# dataform1button

## Description

## Type Tags

dataform1control

## Object Value

Objects of type dataform1button have no value, and it is an error to try to get or set this value.

# dataform1button.new()

## Description

## Prototype

*dataform1button*.new ( dataform1button *me*, dataform1page *page*, wxformbutton *control*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1button | |
| page | None | dataform1page | |
| control | None | wxformbutton | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrgbid | integer | |
| control | wxformbutton | |
| form | dataform1 | |
| formnode | dlistnode | |
| onclick | event | |
| ongotfocus | event | |
| onlostfocus | event | |
| onmouse | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| textrgbid | integer | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## remove()

### Description

### Prototype

*dataform1buttonvar*.remove ( dataform1button *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1button | |

## setname()

### Description

### Prototype

*dataform1buttonvar*.setname ( dataform1button *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1button | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1buttonvar*.setnext ( dataform1button *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1button | |
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1buttonvar*.setposition ( dataform1button *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1button | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# dataform1checkbox

## Description

## Type Tags

dataform1control

## Object Value

Objects of type dataform1checkbox have no value, and it is an error to try to get or set this value.

## dataform1checkbox.new()

### Description

### Prototype

*dataform1checkbox*.new ( dataform1checkbox *me*, dataform1page *page*, wxformcheckbox *control*, type(db1field) *field*, dataform1table *table*, anyvalue *valueon*, anyvalue *valueoff*, string *displayformat*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1checkbox | |
| page | None | dataform1page | |
| control | None | wxformcheckbox | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | type(db1field) | |
| table | None | dataform1table | |
| valueon | None | anyvalue | |
| valueoff | None | anyvalue | |
| displayformat | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | wxformcheckbox | |
| controlsource | dataform1controlsource | |
| form | dataform1 | |
| formnode | dlistnode | |
| onchange | event | |
| ongotfocus | event | |
| onlostfocus | event | |
| onmouse | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| textrgbid | integer | |
| type | type | |
| usesystemcolor | boolean | |
| valueoff | anyvalue | |
| valueon | anyvalue | |

# Methods

## remove()

### Description

### Prototype

*dataform1checkboxvar*.remove ( dataform1checkbox *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1checkbox | |

## setname()

### Description

### Prototype

*dataform1checkboxvar*.setname ( dataform1checkbox *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1checkbox | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1checkboxvar*.setnext ( dataform1checkbox *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1checkbox | |
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1checkboxvar*.setposition ( dataform1checkbox *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1checkbox | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# dataform1combo

## Description

## Type Tags

dataform1control

## Object Value

Objects of type dataform1combo have no value, and it is an error to try to get or set this value.

## dataform1combo.new()

### Description

### Prototype

*dataform1combo*.new ( dataform1combo *me*, dataform1page *page*, wxformcombo *control*, type(db1field) *field*, dataform1table *table*, string *displayformat*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1combo | |
| page | None | dataform1page | |
| control | None | wxformcombo | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | wxformcombo | |
| controlsource | dataform1controlsource | |
| fill | function | |
| form | dataform1 | |
| formnode | dlistnode | |
| listsource | type(*) | |
| listsourcetable | dataform1table | |
| listsourcetype | string | |
| onfill | event | |
| ongotfocus | event | |
| onlostfocus | event | |
| onmouse | event | |
| onselection-change | event | |
| order | type(db1field) | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| settext | function | |
| sorted | boolean | |
| textrgbid | integer | |
| type | type | |
| usesystemcolor | boolean | |
| values | array | |
| valuesource | type(*) | |

# Methods

## fill()

### Description

### Prototype

*dataform1combovar*.fill ( dataform1combo *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1combo | |

## remove()

### Description

### Prototype

*dataform1combovar*.remove ( dataform1combo *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1combo | |

## setname()

### Description

### Prototype

*dataform1combovar*.setname ( dataform1combo *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1combo | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1combovar*.setnext ( dataform1combo *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1combo | |
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1combovar*.setposition ( dataform1combo *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1combo | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

## settext()

### Description

### Prototype

*dataform1combovar*.settext ( dataform1combo *me*, string *text* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1combo | |
| text | None | string | |

# dataform1control

# Description

# Type Tags

dataform1control

# Object Value

Objects of type dataform1control have no value, and it is an error to try to get or set this value.

# dataform1control.new()

## Description

## Prototype

*dataform1control*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | type(wxformcontrol) | |
| controlsource | dataform1controlsource | |
| form | dataform1 | |
| formnode | dlistnode | |
| onmouse | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| settext | function | |
| textrgbid | integer | |
| type | type | |
| usesystemcolor | boolean | |

# dataform1controlsource

## Description

## Type Tags

None

## Object Value

Objects of type dataform1controlsource have no value, and it is an error to try to get or set this value.

## dataform1controlsource.new()

### Description

### Prototype

*dataform1controlsource*.new ( dataform1controlsource *me*, type(dataform1linkcontainer) *container*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1controlsource | |
| container | None | type(dataform1linkcontainer) | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| assignlink | function | |
| container | type(dataform1linkcontainer) | |
| dbsection | detailblocksection | |
| dbsectionindex | integer | |
| detailblock | dataform1detailblock | |
| detailblockrow | integer | |
| displayformat | string | |
| field | type(db1field) | |
| link | dataform1link | |
| origcontrol | type(dataform1control) | |
| table | dataform1table | |
| type | type | |

# Methods

## assignlink()

### Description

### Prototype

*dataform1controlsourcevar*.assignlink ( dataform1controlsource *me*, dataform1link *link*, type(dataform1linkcontainer) *container*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1controlsource | |
| link | None | dataform1link | |
| container | None | type(dataform1linkcontainer) | |
| error | None | integer | |

# dataform1datagrid

# Description

# Type Tags

dataform1datagrid, dataform1control, dataform1linkcontainer

# Object Value

Objects of type dataform1datagrid have no value, and it is an error to try to get or set this value.

# dataform1datagrid.new()

## Description

## Prototype

*dataform1datagrid*.new ( dataform1datagrid *me*, dataform1page *page*, wxformgrid *control*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| page | None | dataform1page | |
| control | None | wxformgrid | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | dataform1gridprivate | |
| addcolumn | function | |
| addsiblinglink | function | |
| clearsiblinglinks | function | |
| columns | dring | |
| control | wxformgrid | |
| findlink | function | |
| form | dataform1 | |
| formnode | dlistnode | |
| getrowdata | function | |
| init | function | |
| lastrowcount | integer | |
| masterlink | dataform1link | |
| mastertable | dataform1table | |
| oncellchange | event | |
| oncellleftclick | event | |
| oncellleftdblclick | event | |
| oncellrightclick | event | |
| oncellrightdblclick | event | |
| oncellselect | event | |
| oncolwidthchange | event | |
| ongotfocus | event | |
| onlabelclick | event | |
| onlabelleftdblclick | event | |
| onlabelrightclick | event | |
| onlabelrightdblclick | event | |

| Property | Type | Description |
|---|---|---|
| onlostfocus | event | |
| onrecorddelete | event | |
| onrecordnew | event | |
| onrecordsave | event | |
| on-rowheightchange | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| popupmenu | wxmenu | |
| refresh | function | |
| remove | function | |
| setlink | function | |
| setmastertable | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| siblinglinks | dring | |
| type | type | |

# Methods

## addcolumn()

### Description

### Prototype

*dataform1datagridvar*.addcolumn ( dataform1datagrid *me*, type(db1field) *field*, dataform1table *table*, string *displayformat*, string *label*, string *cellalignment*, integer *width*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| label | None | string | |
| cellalignment | None | string | |
| width | None | integer | |
| error | None | integer | |

# addsiblinglink()

## Description

## Prototype

*dataform1datagridvar*.addsiblinglink ( dataform1datagrid *me*, type(db1field) *srcfield*, dataform1table *srctable*, type(db1field) *destfield*, dataform1table *desttable*, type(dataform1linkcontainer) *container*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| container | None | type(dataform1linkcontainer) | |
| error | None | integer | |

# clearsiblinglinks()

## Description

## Prototype

*dataform1datagridvar*.clearsiblinglinks ( dataform1datagrid *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |

# findlink()

## Description

## Prototype

*dataform1datagridvar*.findlink ( dataform1datagrid *me*, dataform1table *table* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | dataform1table | |

# getrowdata()

## Description

## Prototype

*dataform1datagridvar*.getrowdata ( dataform1datagrid *me*, integer *row*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| row | None | integer | |
| error | None | integer | |

# init()

## Description

## Prototype

*dataform1datagridvar*.init ( dataform1datagrid *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| error | None | integer | |

# refresh()

## Description

## Prototype

*dataform1datagridvar*.refresh ( dataform1datagrid *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |

# remove()

## Description

## Prototype

*dataform1datagridvar*.remove ( dataform1datagrid *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |

# setlink()

## Description

## Prototype

*dataform1datagridvar*.setlink ( dataform1datagrid *me*, type(db1field) *srcfield*, dataform1table *srctable*, type(db1field) *destfield*, dataform1table *desttable*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| error | None | integer | |

# setmastertable()

## Description

## Prototype

*dataform1datagridvar*.setmastertable ( dataform1datagrid *me*, dataform1table *mastertable* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| mastertable | None | dataform1table | |

# setname()

## Description

## Prototype

*dataform1datagridvar*.setname ( dataform1datagrid *me*, string *name*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| name | None | string | |
| error | None | integer | |

# setnext()

## Description

## Prototype

*dataform1datagridvar*.setnext ( dataform1datagrid *me*, type(dataform1control) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| next | None | type(dataform1control) | |

# setposition()

## Description

## Prototype

*dataform1datagridvar*.setposition ( dataform1datagrid *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagrid | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| height | None | integer | |
| error | None | integer | |

# dataform1datagridcolumn

## Description

## Type Tags

None

## Object Value

Objects of type dataform1datagridcolumn have no value, and it is an error to try to get or set this value.

## dataform1datagridcolumn.new()

### Description

### Prototype

*dataform1datagridcolumn*.new ( dataform1datagridcolumn *me*, dataform1datagrid *parent*, string *displayformat*, string *label*, string *cellalignment*, type(db1field) *field*, dataform1table *table*, integer *width*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datagridcolumn | |
| parent | None | dataform1datagrid | |
| displayformat | None | string | |
| label | None | string | |
| cellalignment | None | string | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| width | None | integer | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| cellalignment | string | |

| Property | Type | Description |
|---|---|---|
| columnid | integer | |
| controlsource | dataform1controlsource | |
| grid | dataform1datagrid | |
| gridnode | dlistnode | |
| label | string | |
| type | type | |
| width | integer | |

# dataform1datasource

## Description

## Type Tags

None

## Object Value

Objects of type dataform1datasource have no value, and it is an error to try to get or set this value.

## dataform1datasource.new()

### Description

### Prototype

*dataform1datasource*.new ( dataform1datasource *me*, type(dataform1) *form*, type(*) *datasource*, string *source*, string *username*, string *password*, integer *codepage*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datasource | |
| form | None | type(dataform1) | |
| datasource | None | type(*) | |
| source | None | string | |
| username | None | string | |
| password | None | string | |
| codepage | None | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| codepage | integer | |
| datasource | type(*) | |
| form | type(dataform1) | |
| formnode | dlistnode | |
| getdatasource-info | function | |
| opentable | function | |
| password | string | |
| source | string | |
| sourcetype | string | |
| tables | dring | |
| type | type | |
| username | string | |

# Methods

## getdatasourceinfo()

### Description

### Prototype

*dataform1datasourcevar*.getdatasourceinfo ( dataform1datasource *me*, dring *parent* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datasource | |
| parent | None | dring | |

## opentable()

### Description

### Prototype

*dataform1datasourcevar*.opentable ( dataform1datasource *me*, string *tablename*, string *password*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1datasource | |
| tablename | None | string | |
| password | None | string | |
| error | None | integer | |

# dataform1detailblock

## Description

## Type Tags

dataform1linkcontainer

## Object Value

Objects of type dataform1detailblock have no value, and it is an error to try to get or set this value.

## dataform1detailblock.new()

### Description

### Prototype

*`dataform1detailblock`*.new ( dataform1detailblock *`me`*, dataform1page *`page`*, array *`controls`*, integer *`rows`*, integer *`rowoffset`*, integer *`columns`*, integer *`columnoffset`*, string *`scrollbar`*, integer *`scrollbaroffset`*, boolean *`tabacross`*, string *`name`*, boolean *`readonly`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| page | None | dataform1page | |
| controls | None | array | |
| rows | `1` | integer | |
| rowoffset | `30` | integer | |
| columns | `1` | integer | |
| columnoffset | `200` | integer | |
| scrollbar | `right` | string | |
| scrollbaroffset | `15` | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tabacross | `.true` | boolean | |
| name | None | string | |
| readonly | `.true` | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| __movedtb_cols | function | |
| __movedtb_rows | function | |
| _private | detailblockprivate | |
| addcontrol | function | |
| addrowdata | function | |
| addsiblinglink | function | |
| blank | function | |
| childlinks | dring | |
| clearsib-linglinks | function | |
| columnoffset | integer | |
| columns | integer | |
| controls | array | |
| currentscroll-position | function | |
| enablerow | function | |
| findlink | function | |
| fixtaborder | function | |
| form | dataform1 | |
| formnode | dlistnode | |
| getcontrolrow-column | function | |
| getrdata | function | |
| getrootcontrol | function | |
| getrootcontrols | function | |
| getrowdata | function | |
| masterlink | dataform1link | |
| mastertable | dataform1table | |
| name | string | |

| Property | Type | Description |
|---|---|---|
| page | dataform1page | |
| parent | dataform1detailblock | |
| recordcount | function | |
| remove | function | |
| removecontrol | function | |
| removerowdata | function | |
| rowoffset | integer | |
| rows | integer | |
| runquery | function | |
| scrollbar | string | |
| scrollbaroffset | integer | |
| selectfirst | function | |
| selectlast | function | |
| selectnext | function | |
| selectnextpage | function | |
| selectprevious | function | |
| selectprevious-page | function | |
| setlink | function | |
| setmastertable | function | |
| setname | function | |
| setparams | function | |
| setrdata | function | |
| setrowdata | function | |
| setsortparams | function | |
| siblinglinks | dring | |
| sort | boolean | |
| sortascending | boolean | |
| sortfield | type(db1field) | |
| tabacross | boolean | |
| type | type | |
| usequery | boolean | |
| whereclause | string | |

# Methods

## __movedtb_cols()

### Description

### Prototype

*dataform1detailblockvar*.__movedtb_cols ( dataform1detailblock *me*, integer *rows*, integer *rowoffset*, integer *columns*, integer *columnoffset*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| rows | None | integer | |
| rowoffset | None | integer | |
| columns | None | integer | |
| columnoffset | None | integer | |
| error | None | integer | |

## __movedtb_rows()

### Description

### Prototype

*dataform1detailblockvar*.__movedtb_rows ( dataform1detailblock *me*, integer *rows*, integer *rowoffset*, integer *columns*, integer *columnoffset*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| rows | None | integer | |
| rowoffset | None | integer | |
| columns | None | integer | |
| columnoffset | None | integer | |
| error | None | integer | |

## addcontrol()

### Description

### Prototype

*dataform1detailblockvar*.addcontrol ( dataform1detailblock *me*, type(dataform1control) *control*, boolean *fixtaborder*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| control | None | type(dataform1control) | |
| fixtaborder | `.true` | boolean | |
| error | None | integer | |

# addrowdata()

## Description

## Prototype

*`dataform1detailblockvar`*.`addrowdata` ( dataform1detailblock *`me`*, dataform1recordset *`rset`*, integer *`regionid`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| rset | None | dataform1recordset | |
| regionid | `1` | integer | |
| error | None | integer | |

# addsiblinglink()

## Description

## Prototype

*`dataform1detailblockvar`*.`addsiblinglink` ( dataform1detailblock *`me`*, type(db1field) *`srcfield`*, dataform1table *`srctable`*, type(db1field) *`destfield`*, dataform1table *`desttable`*, type(dataform1linkcontainer) *`container`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| container | None | type(dataform1linkcontainer) | |
| error | None | integer | |

# blank()

## Description

### Prototype

*dataform1detailblockvar*.blank ( dataform1detailblock *me*, boolean *datacontrolsonly* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| datacontrolsonly | .true | boolean | |

## clearsiblinglinks()

### Description

### Prototype

*dataform1detailblockvar*.clearsiblinglinks ( dataform1detailblock *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |

## currentscrollposition()

### Description

### Prototype

*dataform1detailblockvar*.currentscrollposition ( dataform1detailblock *me*, integer *regionnumber*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| regionnumber | 1 | integer | |
| error | None | integer | |

## enablerow()

### Description

### Prototype

*dataform1detailblockvar*.enablerow ( dataform1detailblock *me*, integer *row*, boolean *enable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| row | None | integer | |
| enable | None | boolean | |

## findlink()

### Description

### Prototype

*dataform1detailblockvar*.findlink ( dataform1detailblock *me*, dataform1table *table* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| table | None | dataform1table | |

## fixtaborder()

### Description

### Prototype

*dataform1detailblockvar*.fixtaborder ( dataform1detailblock *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |

## getcontrolrowcolumn()

### Description

### Prototype

*dataform1detailblockvar*.getcontrolrowcolumn ( dataform1detailblock *me*, type(dataform1control) *control*, integer *row*, integer *column* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| control | None | type(dataform1control) | |
| row | None | integer | |
| column | None | integer | |

# getrdata()

## Description

## Prototype

*dataform1detailblockvar*.getrdata ( dataform1detailblock *me*, integer *row*, integer *regionid*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| row | None | integer | |
| regionid | 1 | integer | |
| error | None | integer | |

# getrootcontrol()

## Description

## Prototype

*dataform1detailblockvar*.getrootcontrol ( dataform1detailblock *me*, type(dataform1control) *control*, integer *index* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| control | None | type(dataform1control) | |
| index | None | integer | |

# getrootcontrols()

## Description

## Prototype

*dataform1detailblockvar*.getrootcontrols ( dataform1detailblock *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |

## getrowdata()

### Description

### Prototype

*dataform1detailblockvar*.getrowdata ( dataform1detailblock *me*, integer *row*, integer *regionid*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| row | None | integer | |
| regionid | 1 | integer | |
| error | None | integer | |

## recordcount()

### Description

### Prototype

*dataform1detailblockvar*.recordcount ( dataform1detailblock *me*, integer *regionnumber*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| regionnumber | 1 | integer | |
| error | None | integer | |

## remove()

### Description

### Prototype

*dataform1detailblockvar*.remove ( dataform1detailblock *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |

## removecontrol()

### Description

### Prototype

*dataform1detailblockvar*.removecontrol ( dataform1detailblock *me*, type(dataform1control) *control*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| control | None | type(dataform1control) | |
| error | None | integer | |

## removerowdata()

### Description

### Prototype

*dataform1detailblockvar*.removerowdata ( dataform1detailblock *me*, integer *row*, integer *regionid*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| row | None | integer | |
| regionid | 1 | integer | |
| error | None | integer | |

## runquery()

### Description

### Prototype

*dataform1detailblockvar*.runquery ( dataform1detailblock *me*, string *errtext*, integer *errindex*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| errtext | None | string | |
| errindex | None | integer | |
| error | None | integer | |

## selectfirst()

### Description

### Prototype

*dataform1detailblockvar*.selectfirst ( dataform1detailblock *me*, integer *regionnumber*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| regionnumber | 1 | integer | |
| error | None | integer | |

## selectlast()

### Description

### Prototype

*dataform1detailblockvar*.selectlast ( dataform1detailblock *me*, integer *regionnumber*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| regionnumber | 1 | integer | |
| error | None | integer | |

## selectnext()

### Description

### Prototype

*dataform1detailblockvar*.selectnext ( dataform1detailblock *me*, integer *regionnumber*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| regionnumber | 1 | integer | |
| error | None | integer | |

## selectnextpage()

### Description

### Prototype

*dataform1detailblockvar*.selectnextpage ( dataform1detailblock *me*, integer *regionnumber*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| regionnumber | 1 | integer | |
| error | None | integer | |

## selectprevious()

### Description

### Prototype

*dataform1detailblockvar*.selectprevious ( dataform1detailblock *me*, integer *regionnumber*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| regionnumber | 1 | integer | |
| error | None | integer | |

## selectpreviouspage()

### Description

### Prototype

*dataform1detailblockvar*.selectpreviouspage ( dataform1detailblock *me*, integer *regionnumber*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| regionnumber | 1 | integer | |
| error | None | integer | |

## setlink()

### Description

### Prototype

*dataform1detailblockvar*.setlink ( dataform1detailblock *me*, type(db1field) *srcfield*, dataform1table *srctable*, type(db1field) *destfield*, dataform1table *desttable*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| error | None | integer | |

## setmastertable()

### Description

### Prototype

*dataform1detailblockvar*.setmastertable ( dataform1detailblock *me*, dataform1table *mastertable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| mastertable | None | dataform1table | |

# setname()

## Description

## Prototype

*dataform1detailblockvar*.setname ( dataform1detailblock *me*, string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| name | None | string | |

# setparams()

## Description

## Prototype

*dataform1detailblockvar*.setparams ( dataform1detailblock *me*, integer *rows*, integer *rowoffset*, integer *columns*, integer *columnoffset*, string *scrollbar*, integer *scrollbaroffset*, boolean *tabacross*, boolean *usequery*, string *whereclause*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| rows | None | integer | |
| rowoffset | None | integer | |
| columns | None | integer | |
| columnoffset | None | integer | |
| scrollbar | None | string | |
| scrollbaroffset | None | integer | |
| tabacross | None | boolean | |
| usequery | None | boolean | |
| whereclause | None | string | |
| error | None | integer | |

# setrdata()

## Description

### Prototype

*dataform1detailblockvar*.setrdata ( dataform1detailblock *me*, integer *row*, dataform1recordset *rset*, integer *regionid*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| row | None | integer | |
| rset | None | dataform1recordset | |
| regionid | 1 | integer | |
| error | None | integer | |

## setrowdata()

### Description

### Prototype

*dataform1detailblockvar*.setrowdata ( dataform1detailblock *me*, integer *row*, dataform1recordset *rset*, integer *regionid*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |
| row | None | integer | |
| rset | None | dataform1recordset | |
| regionid | 1 | integer | |
| error | None | integer | |

## setsortparams()

### Description

### Prototype

*dataform1detailblockvar*.setsortparams ( dataform1detailblock *me*, boolean *sort*, type(db1field) *sortfield*, boolean *sortascending* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1detailblock | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sort | None | boolean | |
| sortfield | None | type(db1field) | |
| sortascending | None | boolean | |

# dataform1edittext

## Description

## Type Tags

dataform1control

## Object Value

Objects of type dataform1edittext have no value, and it is an error to try to get or set this value.

## dataform1edittext.new()

### Description

### Prototype

*dataform1edittext*.new ( dataform1edittext *me*, dataform1page *page*, wxformedittext *control*, type(db1field) *field*, dataform1table *table*, string *displayformat*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1edittext | |
| page | None | dataform1page | |
| control | None | wxformedittext | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |

| Property | Type | Description |
|---|---|---|
| __dlistinfo | df1droplistinfo | |
| backgroundrg-bid | integer | |
| control | wxformedittext | |
| controlsource | dataform1controlsource | |
| enabledroplist | function | |
| form | dataform1 | |
| formnode | dlistnode | |
| onchange | event | |
| ongotfocus | event | |
| onlostfocus | event | |
| onmouse | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| sethiddencon-trols | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| settext | function | |
| textrgbid | integer | |
| type | type | |
| usedroplist | boolean | |
| usesystemcolor | boolean | |

# Methods

## enabledroplist()

### Description

### Prototype

*`dataform1edittextvar`*.`enabledroplist` ( dataform1edittext *`me`*, boolean *`enable`*, type(db1index) *`index`*, integer *`activationcharcount`*, integer *`listheight`*, integer *`maxsearchentries`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1edittext | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| enable | `.false` | boolean | |
| index | None | type(db1index) | |
| activationchar-count | None | integer | |
| listheight | None | integer | |
| maxsearchen-tries | None | integer | |
| error | None | integer | |

# remove()

## Description

## Prototype

*dataform1edittextvar*.`remove` ( dataform1edittext *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1edittext | |

# sethiddencontrols()

## Description

## Prototype

*dataform1edittextvar*.`sethiddencontrols` ( dataform1edittext *me*, array *hiddencon-trols* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1edittext | |
| hiddencontrols | None | array | |

# setname()

## Description

## Prototype

*dataform1edittextvar*.`setname` ( dataform1edittext *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1edittext | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1edittextvar*.setnext ( dataform1edittext *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1edittext | |
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1edittextvar*.setposition ( dataform1edittext *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1edittext | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

## settext()

### Description

### Prototype

*dataform1edittextvar*.settext ( dataform1edittext *me*, string *text* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1edittext | |
| text | None | string | |

# dataform1ellipse

## Description

## Type Tags

dataform1graphic

## Object Value

Objects of type dataform1ellipse have no value, and it is an error to try to get or set this value.

## dataform1ellipse.new()

### Description

### Prototype

*dataform1ellipse*.new ( dataform1ellipse *me*, dataform1page *page*, wxgraphicellipse *graphic*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1ellipse | |
| page | None | dataform1page | |
| graphic | None | wxgraphicellipse | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | |
| borderrgbid | integer | |
| form | dataform1 | |
| formnode | dlistnode | |
| getboundin-grectangle | function | |
| graphic | wxgraphicellipse | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| rgbid | integer | |
| setname | function | |
| setposition | function | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## getboundingrectangle()

### Description

### Prototype

*dataform1ellipsevar*.getboundingrectangle ( dataform1ellipse *me*, point *point1*, point *point2* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1ellipse | |
| point1 | None | point | |
| point2 | None | point | |

## remove()

### Description

### Prototype

*dataform1ellipsevar*.remove ( dataform1ellipse *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1ellipse | |

## setname()

### Description

### Prototype

*dataform1ellipsevar*.setname ( dataform1ellipse *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1ellipse | |
| name | None | string | |
| error | None | integer | |

## setposition()

### Description

### Prototype

*dataform1ellipsevar*.setposition ( dataform1ellipse *me*, point *point1*, point *point2*, point *point3*, point *midpoint*, integer *width*, integer *borderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1ellipse | |
| point1 | None | point | |
| point2 | None | point | |
| point3 | None | point | |
| midpoint | None | point | |
| width | None | integer | |
| borderwidth | None | integer | |
| error | None | integer | |

# dataform1gauge

## Description

# Type Tags

dataform1control

# Object Value

Objects of type dataform1gauge have no value, and it is an error to try to get or set this value.

# dataform1gauge.new()

## Description

## Prototype

*dataform1gauge*.new ( dataform1gauge *me*, dataform1page *page*, wxformgauge *control*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1gauge | |
| page | None | dataform1page | |
| control | None | wxformgauge | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | wxformgauge | |
| form | dataform1 | |
| formnode | dlistnode | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## remove()

### Description

### Prototype

*dataform1gaugevar*.remove ( dataform1gauge *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1gauge | |

## setname()

### Description

### Prototype

*dataform1gaugevar*.setname ( dataform1gauge *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1gauge | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1gaugevar*.setnext ( dataform1gauge *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1gauge | |
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1gaugevar*.setposition ( dataform1gauge *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1gauge | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# dataform1graphic

## Description

## Type Tags

dataform1graphic

## Object Value

Objects of type dataform1graphic have no value, and it is an error to try to get or set this value.

## dataform1graphic.new()

### Description

### Prototype

*dataform1graphic*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |

| Property | Type | Description |
|---|---|---|
| borderrgbid | integer | |
| form | dataform1 | |
| formnode | dlistnode | |
| graphic | type(wxgraphic) | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| rgbid | integer | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| setrgb | function | |
| setvisible | function | |
| type | type | |
| usesystemcolor | boolean | |

# dataform1grid

## Description

## Type Tags

dataform1control

## Object Value

Objects of type dataform1grid have no value, and it is an error to try to get or set this value.

## dataform1grid.new()

### Description

### Prototype

*dataform1grid*.new ( dataform1grid *me*, dataform1page *page*, wxformgrid *control*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1grid | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| page | None | dataform1page | |
| control | None | wxformgrid | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | dataform1gridprivate | |
| control | wxformgrid | |
| form | dataform1 | |
| formnode | dlistnode | |
| oncellchange | event | |
| oncellleftclick | event | |
| oncellleftd-blclick | event | |
| oncell-rightclick | event | |
| oncellrightd-blclick | event | |
| oncellselect | event | |
| oncol-widthchange | event | |
| ongotfocus | event | |
| onlabelclick | event | |
| onlabelleftd-blclick | event | |
| onlabel-rightclick | event | |
| onlabelrightd-blclick | event | |
| onlostfocus | event | |
| on-rowheightchange | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |

| Property | Type | Description |
|---|---|---|
| type | type | |

# Methods

## remove()

### Description

### Prototype

*dataform1gridvar*.remove ( dataform1grid *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1grid | |

## setname()

### Description

### Prototype

*dataform1gridvar*.setname ( dataform1grid *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1grid | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1gridvar*.setnext ( dataform1grid *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1grid | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1gridvar*.setposition ( dataform1grid *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1grid | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# dataform1line

## Description

## Type Tags

dataform1graphic

## Object Value

Objects of type dataform1line have no value, and it is an error to try to get or set this value.

## dataform1line.new()

### Description

### Prototype

*dataform1line*.new ( dataform1line *me*, dataform1page *page*, wxgraphicline *graphic*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1line | |
| page | None | dataform1page | |
| graphic | None | wxgraphicline | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | dataform1 | |
| formnode | dlistnode | |
| graphic | wxgraphicline | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| rgbid | integer | |
| setname | function | |
| setposition | function | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## remove()

### Description

### Prototype

*`dataform1linevar`*`.remove` ( dataform1line *`me`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1line | |

## setname()

### Description

### Prototype

*dataform1linevar*.setname ( dataform1line *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1line | |
| name | None | string | |
| error | None | integer | |

## setposition()

### Description

### Prototype

*dataform1linevar*.setposition ( dataform1line *me*, point *point1*, point *point2*, point *point3*, point *midpoint*, integer *width*, integer *borderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1line | |
| point1 | None | point | |
| point2 | None | point | |
| point3 | None | point | |
| midpoint | None | point | |
| width | None | integer | |
| borderwidth | None | integer | |
| error | None | integer | |

# dataform1link

# Description

# Type Tags

dataform1linkcontainer

# Object Value

Objects of type dataform1link have no value, and it is an error to try to get or set this value.

# dataform1link.new()

## Description

## Prototype

*dataform1link*.new ( dataform1link *me*, type(db1field) *srcfield*, dataform1table *srctable*, type(db1field) *destfield*, dataform1table *desttable*, type(dataform1linkcontainer) *container*, dring *parentdring*, dataform1link *parentlink*, type(dataform1linkcontainer) *parentcontainer*, boolean *onetoone*, boolean *dovalidation*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| container | None | type(dataform1linkcontainer) | |
| parentdring | None | dring | |
| parentlink | None | dataform1link | |
| parentcontainer | None | type(dataform1linkcontainer) | |
| onetoone | .false | boolean | |
| dovalidation | .true | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | dblock | |
| addchild | function | |
| addrecord | function | |
| addsibling | function | |
| clearrecords | function | |
| container | type(dataform1linkcontainer) | |
| currentposition | integer | |
| delrecord | function | |
| destfield | type(db1field) | |
| desttable | dataform1table | |

| Property | Type | Description |
|---|---|---|
| getrecord | function | |
| isdirty | function | |
| links | dring | |
| maxrecords | integer | |
| node | dlistnode | |
| onetoone | boolean | |
| parentcontainer | type(dataform1linkcontainer) | |
| readrecords | function | |
| remove | function | |
| setlink | function | |
| setparent | function | |
| setrecord | function | |
| siblinglinks | dring | |
| srcfield | type(db1field) | |
| srctable | dataform1table | |
| totalrecords | integer | |
| type | type | |
| unlockrecords | function | |
| valid | boolean | |

# Methods

## addchild()

### Description

### Prototype

*`dataform1linkvar`*`.addchild` ( dataform1link *`me`*, type(db1field) *`srcfield`*, dataform1table *`srctable`*, type(db1field) *`destfield`*, dataform1table *`desttable`*, type(dataform1linkcontainer) *`container`*, type(dataform1linkcontainer) *`parentcontainer`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| container | None | type(dataform1linkcontainer) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parentcontainer | None | type(dataform1linkcontainer) | |
| error | None | integer | |

# addrecord()

## Description

## Prototype

*dataform1linkvar*.addrecord ( dataform1link *me*, integer *region*, type(db1record) *record*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| region | None | integer | |
| record | None | type(db1record) | |
| error | None | integer | |

# addsibling()

## Description

## Prototype

*dataform1linkvar*.addsibling ( dataform1link *me*, type(db1field) *srcfield*, dataform1table *srctable*, type(db1field) *destfield*, dataform1table *desttable*, type(dataform1linkcontainer) *container*, type(dataform1linkcontainer) *parentcontainer*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| container | None | type(dataform1linkcontainer) | |
| parentcontainer | None | type(dataform1linkcontainer) | |
| error | None | integer | |

# clearrecords()

## Description

### Prototype

*dataform1linkvar*.clearrecords ( dataform1link *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |

## delrecord()

### Description

### Prototype

*dataform1linkvar*.delrecord ( dataform1link *me*, integer *region*, integer *row*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| region | None | integer | |
| row | None | integer | |
| error | None | integer | |

## getrecord()

### Description

### Prototype

*dataform1linkvar*.getrecord ( dataform1link *me*, integer *region*, integer *row*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| region | None | integer | |
| row | None | integer | |
| error | None | integer | |

## isdirty()

### Description

### Prototype

*dataform1linkvar*.isdirty ( dataform1link *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |

## readrecords()

### Description

### Prototype

*dataform1linkvar*.readrecords ( dataform1link *me*, boolean *lock*, boolean *startatlastread*, integer *count*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| lock | .false | boolean | |
| startatlastread | .false | boolean | |
| count | .inf | integer | |
| error | None | integer | |

## remove()

### Description

### Prototype

*dataform1linkvar*.remove ( dataform1link *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |

## setlink()

### Description

### Prototype

*dataform1linkvar*.setlink ( dataform1link *me*, type(db1field) *srcfield*, dataform1table *srctable*, type(db1field) *destfield*, dataform1table *desttable*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| error | None | integer | |

## setparent()

### Description

### Prototype

*dataform1linkvar*.setparent ( dataform1link *me*, dring *parentdring*, type(dataform1linkcontainer) *parentcontainer*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| parentdring | None | dring | |
| parentcontainer | None | type(dataform1linkcontainer) | |
| error | None | integer | |

## setrecord()

### Description

### Prototype

*dataform1linkvar*.setrecord ( dataform1link *me*, integer *region*, integer *row*, type(db1record) *record*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |
| region | None | integer | |
| row | None | integer | |
| record | None | type(db1record) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

## unlockrecords()

### Description

### Prototype

*dataform1linkvar*.unlockrecords ( dataform1link *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1link | |

# dataform1list

# Description

# Type Tags

dataform1control

# Object Value

Objects of type dataform1list have no value, and it is an error to try to get or set this value.

# dataform1list.new()

## Description

## Prototype

*dataform1list*.new ( dataform1list *me*, dataform1page *page*, wxformlist *control*, type(db1field) *field*, dataform1table *table*, string *displayformat*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1list | |
| page | None | dataform1page | |
| control | None | wxformlist | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | wxformlist | |
| controlsource | dataform1controlsource | |
| fill | function | |
| form | dataform1 | |
| formnode | dlistnode | |
| listsource | type(*) | |
| listsourcetable | dataform1table | |
| listsourcetype | string | |
| ondoubleclick | event | |
| onfill | event | |
| ongotfocus | event | |
| onlostfocus | event | |
| onmouse | event | |
| onselection-change | event | |
| order | type(db1field) | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| sorted | boolean | |
| textrgbid | integer | |
| type | type | |
| usesystemcolor | boolean | |
| values | array | |

| Property | Type | Description |
|---|---|---|
| valuesource | type(*) | |

# Methods

## fill()

### Description

### Prototype

*dataform1listvar*.fill ( dataform1list *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1list | |

## remove()

### Description

### Prototype

*dataform1listvar*.remove ( dataform1list *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1list | |

## setname()

### Description

### Prototype

*dataform1listvar*.setname ( dataform1list *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1list | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1listvar*.setnext ( dataform1list *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1list | |
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1listvar*.setposition ( dataform1list *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1list | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# dataform1option

# Description

# Type Tags

dataform1control

# Object Value

Objects of type dataform1option have no value, and it is an error to try to get or set this value.

# dataform1option.new()

## Description

## Prototype

*dataform1option*.new ( dataform1option *me*, dataform1page *page*, wxformoption *control*, dataform1optiongroup *parent*, anyvalue *value*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1option | |
| page | None | dataform1page | |
| control | None | wxformoption | |
| parent | None | dataform1optiongroup | |
| value | None | anyvalue | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | wxformoption | |
| form | dataform1 | |
| formnode | dlistnode | |
| groupnode | dlistnode | |
| onchange | event | |
| ongotfocus | event | |
| onlostfocus | event | |
| onmouse | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| parent | dataform1optiongroup | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| textrgbid | integer | |

| Property | Type | Description |
|---|---|---|
| type | type | |
| usesystemcolor | boolean | |
| value | anyvalue | |

# Methods

## remove()

### Description

### Prototype

*dataform1optionvar*.remove ( dataform1option *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1option | |

## setname()

### Description

### Prototype

*dataform1optionvar*.setname ( dataform1option *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1option | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1optionvar*.setnext ( dataform1option *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1option | |
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1optionvar*.setposition ( dataform1option *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1option | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# dataform1optiongroup

# Description

# Type Tags

None

# Object Value

Objects of type dataform1optiongroup have no value, and it is an error to try to get or set this value.

# dataform1optiongroup.new()

## Description

## Prototype

*dataform1optiongroup*.new ( dataform1optiongroup *me*, dataform1 *form*, string *name*, type(db1field) *field*, dataform1table *table*, string *displayformat*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1optiongroup | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| form | None | dataform1 | |
| name | None | string | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addmember | function | |
| controlsource | dataform1controlsource | |
| form | dataform1 | |
| formnode | dlistnode | |
| members | dring | |
| name | string | |
| remove | function | |
| select | function | |
| type | type | |

# Methods

## addmember()

### Description

### Prototype

*dataform1optiongroupvar*.addmember ( dataform1optiongroup *me*, wxformoption *ob*, dataform1page *page*, anyvalue *value*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1optiongroup | |
| ob | None | wxformoption | |
| page | None | dataform1page | |
| value | None | anyvalue | |
| error | None | integer | |

## remove()

### Description

### Prototype

*dataform1optiongroupvar*.remove ( dataform1optiongroup *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1optiongroup | |

## select()

### Description

### Prototype

*dataform1optiongroupvar*.select ( dataform1optiongroup *me*, dataform1option *member* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1optiongroup | |
| member | None | dataform1option | |

# dataform1page

# Description

# Type Tags

dataform1page

# Object Value

Objects of type dataform1page have no value, and it is an error to try to get or set this value.

# dataform1page.new()

## Description

## Prototype

*dataform1page*.new ( dataform1page *me*, dataform1 *form*, integer *width*, integer *height*, integer *backgroundrgb*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1page | |
| form | None | dataform1 | |
| width | `50` | integer | |
| height | `50` | integer | |
| backgroundrgb | `16777215` | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addcontrol | function | |
| addgraphic | function | |
| backgroundrg-bid | integer | |
| changename | function | |
| controls | dring | |
| form | dataform1 | |
| formnode | dlistnode | |
| graphics | dring | |
| name | string | |
| pagenum | integer | |
| remove | function | |
| resize | function | |
| setactive | function | |
| type | type | |
| usesystemcolor | boolean | |
| wxformpage | wxform | |

# Methods

## addcontrol()

### Description

### Prototype

*`dataform1pagevar`*`.addcontrol(` dataform1page *`me`*, type *`controltype`*, integer *`left`*, integer *`top`*, integer *`width`*, integer *`height`*, string *`text`*, boolean *`enabled`*, boolean *`visible`*, wxbitmap

*bitmap*, string *scaling*, wxbitmap *selectedbitmap*, wxbitmap *disabledbitmap*, wxbitmap *focusbitmap*, integer *backgroundrgb*, integer *textrgb*, integer *rgb*, string *edittype*, string *selectiontype*, integer *rowcount*, integer *colcount*, integer *rowheight*, integer *colwidth*, boolean *rowheightdraggable*, boolean *colwidthdraggable*, integer *rowlabelwidth*, integer *collabelheight*, string *rowlabelalignment*, string *collabelalignment*, string *alignment*, string *editstyle*, string *orientation*, integer *range*, integer *position*, integer *pagesize*, integer *thumbsize*, wxfont *font*, wxfont *labelfont*, string *tooltip*, integer *onmousemask*, string *name*, type(dataform1control) *next*, anyvalue *valueon*, anyvalue *valueoff*, type(db1field) *field*, dataform1table *table*, string *displayformat*, dataform1optiongroup *obgroup*, boolean *suppressfill*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1page | |
| controltype | None | type | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| text | None | string | |
| enabled | `.true` | boolean | |
| visible | `.true` | boolean | |
| bitmap | None | wxbitmap | |
| scaling | None | string | |
| selectedbitmap | None | wxbitmap | |
| disabledbitmap | None | wxbitmap | |
| focusbitmap | None | wxbitmap | |
| backgroundrgb | None | integer | |
| textrgb | `0` | integer | |
| rgb | None | integer | |
| edittype | `droplist` | string | |
| selectiontype | `single` | string | |
| rowcount | `1` | integer | |
| colcount | `1` | integer | |
| rowheight | `20` | integer | |
| colwidth | `80` | integer | |
| rowheightdraggable | `.true` | boolean | |
| colwidthdraggable | `.true` | boolean | |
| rowlabelwidth | `80` | integer | |
| collabelheight | `20` | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rowlabelalign-ment | `left,top` | string | |
| collabelalign-ment | `left,top` | string | |
| alignment | `left,top` | string | |
| editstyle | None | string | |
| orientation | None | string | |
| range | `1` | integer | |
| position | `0` | integer | |
| pagesize | `1` | integer | |
| thumbsize | `1` | integer | |
| font | None | wxfont | |
| labelfont | None | wxfont | |
| tooltip | None | string | |
| onmousemask | `0` | integer | |
| name | None | string | |
| next | None | type(dataform1control) | |
| valueon | None | anyvalue | |
| valueoff | None | anyvalue | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| obgroup | None | dataform1optiongroup | |
| suppressfill | `.false` | boolean | |
| error | None | integer | |

# addgraphic()

## Description

## Prototype

*`dataform1pagevar`*`.addgraphic` ( dataform1page *`me`*, type *`graphictype`*, point *`point1`*, point *`point2`*, point *`point3`*, point *`midpoint`*, integer *`rgb`*, integer *`borderrgb`*, integer *`width`*, integer *`borderwidth`*, boolean *`visible`*, boolean *`bordervisible`*, string *`name`*, type(wxgraphic) *`next`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1page | |
| graphictype | None | type | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| point1 | None | point | |
| point2 | None | point | |
| point3 | None | point | |
| midpoint | None | point | |
| rgb | None | integer | |
| borderrgb | None | integer | |
| width | None | integer | |
| borderwidth | None | integer | |
| visible | None | boolean | |
| bordervisible | None | boolean | |
| name | None | string | |
| next | None | type(wxgraphic) | |
| error | None | integer | |

# changename()

## Description

## Prototype

*dataform1pagevar*.changename ( dataform1page *me*, string *name*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1page | |
| name | None | string | |
| error | None | integer | |

# remove()

## Description

## Prototype

*dataform1pagevar*.remove ( dataform1page *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1page | |

## resize()

### Description

### Prototype

*dataform1pagevar*.resize ( dataform1page *me*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1page | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

## setactive()

### Description

### Prototype

*dataform1pagevar*.setactive ( dataform1page *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1page | |

# dataform1record

# Description

# Type Tags

db1record

# Object Value

Objects of type dataform1record have no value, and it is an error to try to get or set this value.

## dataform1record.new()

### Description

## Prototype

*dataform1record*.new ( dataform1record *me*, type(db1record) *record*, dataform1table *table*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1record | |
| record | None | type(db1record) | |
| table | None | dataform1table | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| delete | function | |
| isdirty | boolean | |
| lock | function | |
| ondeleterecord | event | |
| onsaverecord | event | |
| record | type(db1record) | |
| save | function | |
| setrecord | function | |
| table | dataform1table | |
| type | type | |
| unlock | function | |

# Methods

## delete()

### Description

### Prototype

*dataform1recordvar*.delete ( dataform1record *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1record | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# lock()

## Description

## Prototype

*dataform1recordvar*.lock ( dataform1record *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1record | |
| error | None | integer | |

# save()

## Description

## Prototype

*dataform1recordvar*.save ( dataform1record *me*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1record | |
| lock | .false | boolean | |
| error | None | integer | |

# setrecord()

## Description

## Prototype

*dataform1recordvar*.setrecord ( dataform1record *me*, type(db1record) *record*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1record | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| record | None | type(db1record) | |
| error | None | integer | |

## unlock()

### Description

### Prototype

*dataform1recordvar*.unlock ( dataform1record *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1record | |

# dataform1rectangle

# Description

# Type Tags

dataform1graphic

# Object Value

Objects of type dataform1rectangle have no value, and it is an error to try to get or set this value.

# dataform1rectangle.new()

## Description

## Prototype

*dataform1rectangle*.new ( dataform1rectangle *me*, dataform1page *page*, wxgraphicrectangle *graphic*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1rectangle | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| page | None | dataform1page | |
| graphic | None | wxgraphicrectangle | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| borderrgbid | integer | |
| form | dataform1 | |
| formnode | dlistnode | |
| graphic | wxgraphicrectangle | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| rgbid | integer | |
| setname | function | |
| setposition | function | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## remove()

### Description

### Prototype

*dataform1rectanglevar*.remove ( dataform1rectangle *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1rectangle | |

## setname()

### Description

### Prototype

*dataform1rectanglevar*.setname ( dataform1rectangle *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1rectangle | |
| name | None | string | |
| error | None | integer | |

## setposition()

### Description

### Prototype

*dataform1rectanglevar*.setposition ( dataform1rectangle *me*, point *point1*, point *point2*, point *point3*, point *midpoint*, integer *width*, integer *borderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1rectangle | |
| point1 | None | point | |
| point2 | None | point | |
| point3 | None | point | |
| midpoint | None | point | |
| width | None | integer | |
| borderwidth | None | integer | |
| error | None | integer | |

# dataform1scrollbar

# Description

# Type Tags

dataform1control

# Object Value

Objects of type dataform1scrollbar have no value, and it is an error to try to get or set this value.

# dataform1scrollbar.new()

## Description

## Prototype

*dataform1scrollbar*.new ( dataform1scrollbar *me*, dataform1page *page*, wxformscrollbar *control*, type(db1field) *field*, dataform1table *table*, string *displayformat*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1scrollbar | |
| page | None | dataform1page | |
| control | None | wxformscrollbar | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | wxformscrollbar | |
| controlsource | dataform1controlsource | |
| form | dataform1 | |
| formnode | dlistnode | |
| ongotfocus | event | |
| onlostfocus | event | |
| onmouse | event | |
| onscroll | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |

| Property | Type | Description |
|---|---|---|
| type | type | |
| usesystemcolor | boolean | |

# Methods

## remove()

### Description

### Prototype

*dataform1scrollbarvar*.remove ( dataform1scrollbar *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1scrollbar | |

## setname()

### Description

### Prototype

*dataform1scrollbarvar*.setname ( dataform1scrollbar *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1scrollbar | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*dataform1scrollbarvar*.setnext ( dataform1scrollbar *me*, type(dataform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1scrollbar | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(dataform1control) | |

## setposition()

### Description

### Prototype

*dataform1scrollbarvar*.setposition ( dataform1scrollbar *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1scrollbar | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# dataform1table

# Description

# Type Tags

None

# Object Value

Objects of type dataform1table have no value, and it is an error to try to get or set this value.

# dataform1table.new()

## Description

## Prototype

*dataform1table*.new ( dataform1table *me*, type(db1table) *table*, type(dataform1) *form*, dring *parent*, dataform1datasource *datasource*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1table | |
| table | None | type(db1table) | |
| form | None | type(dataform1) | |
| parent | None | dring | |
| datasource | None | dataform1datasource | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| currindex | type(db1index) | |
| datasource | dataform1datasource | |
| datasourcenode | dlistnode | |
| fieldinfo | array | |
| form | type(dataform1) | |
| getfieldinfo | function | |
| gettablename | function | |
| gettbinfo | function | |
| newfieldinfo | array | |
| newrecord | function | |
| ondeleterecord | event | |
| onnewrecord | event | |
| onsaverecord | event | |
| parentnode | dlistnode | |
| table | type(db1table) | |
| type | type | |

# Methods

## getfieldinfo()

### Description

### Prototype

*`dataform1tablevar`*`.getfieldinfo` ( dataform1table *`me`*, string *`fieldname`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1table | |
| fieldname | None | string | |

## gettablename()

### Description

### Prototype

*dataform1tablevar*.gettablename ( dataform1table *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1table | |

## gettbinfo()

### Description

### Prototype

*dataform1tablevar*.gettbinfo ( dataform1table *me*, datasourceinfo *dsinfo* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1table | |
| dsinfo | None | datasourceinfo | |

## newrecord()

### Description

### Prototype

*dataform1tablevar*.newrecord ( dataform1table *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1table | |

# dataform1text

## Description

## Type Tags

dataform1control

## Object Value

Objects of type dataform1text have no value, and it is an error to try to get or set this value.

## dataform1text.new()

### Description

### Prototype

*dataform1text*.new ( dataform1text *me*, dataform1page *page*, wxformtext *control*, type(db1field) *field*, dataform1table *table*, string *displayformat*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1text | |
| page | None | dataform1page | |
| control | None | wxformtext | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrg-bid | integer | |
| control | wxformtext | |
| controlsource | dataform1controlsource | |

| Property | Type | Description |
|---|---|---|
| form | dataform1 | |
| formnode | dlistnode | |
| onmouse | event | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| settext | function | |
| textrgbid | integer | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## remove()

### Description

### Prototype

*dataform1textvar*.remove ( dataform1text *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1text | |

## setname()

### Description

### Prototype

*dataform1textvar*.setname ( dataform1text *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1text | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | |
| error | None | integer | |

# setnext()

## Description

## Prototype

*dataform1textvar*.setnext ( dataform1text *me*, type(dataform1control) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1text | |
| next | None | type(dataform1control) | |

# setposition()

## Description

## Prototype

*dataform1textvar*.setposition ( dataform1text *me*, integer *left*, integer *top*, integer *width*, integer *height*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1text | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

# settext()

## Description

## Prototype

*dataform1textvar*.settext ( dataform1text *me*, string *text* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1text | |
| text | None | string | |

# dataform1triangle

## Description

## Type Tags

dataform1graphic

## Object Value

Objects of type dataform1triangle have no value, and it is an error to try to get or set this value.

## dataform1triangle.new()

### Description

### Prototype

*dataform1triangle*.new ( dataform1triangle *me*, dataform1page *page*, wxgraphictriangle *graphic*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1triangle | |
| page | None | dataform1page | |
| graphic | None | wxgraphictriangle | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| borderrgbid | integer | |

| Property | Type | Description |
|---|---|---|
| form | dataform1 | |
| formnode | dlistnode | |
| graphic | wxgraphictriangle | |
| page | dataform1page | |
| pagenode | dlistnode | |
| remove | function | |
| rgbid | integer | |
| setname | function | |
| setposition | function | |
| type | type | |
| usesystemcolor | boolean | |

# Methods

## remove()

### Description

### Prototype

*dataform1trianglevar*.remove ( dataform1triangle *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1triangle | |

## setname()

### Description

### Prototype

*dataform1trianglevar*.setname ( dataform1triangle *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1triangle | |
| name | None | string | |
| error | None | integer | |

## setposition()

### Description

### Prototype

*dataform1trianglevar*.setposition ( dataform1triangle *me*, point *point1*, point *point2*, point *point3*, point *midpoint*, integer *width*, integer *borderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dataform1triangle | |
| point1 | None | point | |
| point2 | None | point | |
| point3 | None | point | |
| midpoint | None | point | |
| width | None | integer | |
| borderwidth | None | integer | |
| error | None | integer | |

# fdevent

## Description

## Type Tags

None

## Object Value

Objects of type fdevent have no value, and it is an error to try to get or set this value.

## fdevent.new()

### Description

### Prototype

*fdevent*.new ( fdevent *me*, type(*) *origref*, string *handler* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fdevent | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| origref | None | type(*) | |
| handler | None | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| handler | string | |
| reference | type(*) | |
| type | type | |

# pageresizeinfo

## Description

## Type Tags

None

## Object Value

Objects of type pageresizeinfo have no value, and it is an error to try to get or set this value.

## pageresizeinfo.new()

### Description

### Prototype

*pageresizeinfo*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| problemcontrols | array | |
| problemgraphics | array | |

| Property | Type | Description |
|---|---|---|
| type | type | |

# printform1

## Description

## Type Tags

printform1, dataform1, dataform1linkcontainer

## Object Value

Objects of type printform1 have no value, and it is an error to try to get or set this value.

## printform1.new()

### Description

### Prototype

*printform1*.new ( printform1 *me*, integer *defpagewidth*, integer *defpageheight*, integer *defpagebackcolor*, string *defbooleanformat*, string *defintegerformat*, string *defnumberformat*, string *defdateformat*, string *deftimeformat*, string *defdatetimeformat*, SBLlocaledateinfo *defdatelocale*, SBLNumSettings *defnumericlocale*, wxfont *deffont*, boolean *createdisplayform*, string *printpreviewtitle*, string *currentworkingdirectory*, boolean *loading*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| defpagewidth | 210000 | integer | |
| defpageheight | 297000 | integer | |
| defpagebackcolor | 16777215 | integer | |
| defbooleanformat | T\|F | string | |
| defintegerformat | . | string | |
| defnumberformat | 999999.00 | string | |
| defdateformat | yyyy.0m.0d | string | |
| deftimeformat | hh:mm:ss | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defdatetimeformat | None | string | |
| defdatelocale | None | SBLlocaledateinfo | |
| defnumericlocale | None | SBLNumSettings | |
| deffont | None | wxfont | |
| createdisplayform | `.false` | boolean | |
| printpreviewtitle | `Superbase NG Print Form` | string | |
| currentworkingdirectory | None | string | |
| loading | `.false` | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | printform1private | |
| addbitmap | function | |
| addcontrol | function | |
| adddatasource | function | |
| addgraphic | function | |
| addpage | function | |
| addsiblinglink | function | |
| addtable | function | |
| assignfilterobject | function | |
| bitmaps | dring | |
| blank | function | |
| builddisplayform | function | |
| centeroverdisplay | boolean | |
| clearsiblinglinks | function | |
| container | type(wxcontainer) | |
| controls | dring | |

| Property | Type | Description |
|---|---|---|
| createdisplay-form | boolean | |
| currentpage | printform1page | |
| currentwork-ingdirectory | string | |
| datasources | dring | |
| defbooleanfor-mat | string | |
| defdateformat | string | |
| defdatelocale | SBLlocaledateinfo | |
| defdatetimefor-mat | string | |
| deffont | wxfont | |
| defintegerfor-mat | string | |
| defnumberfor-mat | string | |
| defnumericlo-cale | SBLNumSettings | |
| defpageback-color | integer | |
| defpageheight | integer | |
| defpagewidth | integer | |
| deftimeformat | string | |
| designmode | boolean | |
| dirty | boolean | |
| filename | string | |
| filter | dataform1filter | |
| findbitmap-source | function | |
| findcontrol | function | |
| finddatasource | function | |
| findgraphic | function | |
| findsiblinglink | function | |
| findtable | function | |
| fonts | array | |
| getfieldand-table | function | |
| getfont | function | |
| graphics | dring | |
| lastusedrecord | dataform1record | |

| Property | Type | Description |
|---|---|---|
| loading | boolean | |
| lock | function | |
| locked | boolean | |
| masterrecord | dataform1record | |
| mastertable | dataform1table | |
| name | string | |
| nameinuse | function | |
| onsave | event | |
| onselect | event | |
| pages | dring | |
| print | function | |
| printpreviewtitle | string | |
| refresh | function | |
| removedisplayform | function | |
| saverecord | function | |
| selectcurrent | function | |
| selectfirst | function | |
| selectkey | function | |
| selectlast | function | |
| selectnext | function | |
| selectprevious | function | |
| setcontainer | function | |
| setdirtystate | function | |
| setfilter | function | |
| setlastusedrecord | function | |
| setmasterrecord | function | |
| setmastertable | function | |
| showpage | function | |
| siblinglinks | dring | |
| startat100percent | boolean | |
| tables | dring | |
| type | type | |
| unlock | function | |
| valid | boolean | |

# Methods

## !()

### Description

### Prototype

*printform1var*.! ( printform1 *me*, string *controlname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| controlname | None | string | |

## addbitmap()

### Description

### Prototype

*printform1var*.addbitmap ( printform1 *me*, string *filename*, string *format*, blob *rgb*, integer *width*, integer *height*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| filename | None | string | |
| format | xpm | string | |
| rgb | None | blob | |
| width | None | integer | |
| height | None | integer | |
| error | None | integer | |

## addcontrol()

### Description

### Prototype

*printform1var*.addcontrol ( printform1 *me*, type *controltype*, integer *printleft*, integer *printtop*, integer *printwidth*, integer *printheight*, string *text*, boolean *visible*, wxbitmap *bitmap*, string *scaling*, integer *backgroundrgb*, integer *textrgb*, string *printalignment*, wxfont *font*, string *printname*, boolean *backgroundvisible*, boolean *undergraphics*, boolean *underbitmaps*, type(printform1control) *next*, printform1page *page*, type(db1field) *field*, dataform1table *table*, string *displayformat*, boolean *skipbitmapcheck*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| controltype | None | type | |
| printleft | None | integer | |
| printtop | None | integer | |
| printwidth | None | integer | |
| printheight | None | integer | |
| text | None | string | |
| visible | `.true` | boolean | |
| bitmap | None | wxbitmap | |
| scaling | None | string | |
| backgroundrgb | None | integer | |
| textrgb | `0` | integer | |
| printalignment | None | string | |
| font | None | wxfont | |
| printname | None | string | |
| backgroundvis-ible | `.true` | boolean | |
| undergraphics | `.false` | boolean | |
| underbitmaps | `.false` | boolean | |
| next | None | type(printform1control) | |
| page | None | printform1page | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| skip-bitmapcheck | `.false` | boolean | |
| error | None | integer | |

## adddatasource()

### Description

### Prototype

*`printform1var`*`.adddatasource` ( printform1 *`me`*, type(*) *`datasource`*, string *`source`*, string *`username`*, string *`password`*, integer *`codepage`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datasource | None | type(*) | |
| source | None | string | |
| username | None | string | |
| password | None | string | |
| codepage | None | integer | |
| error | None | integer | |

# addgraphic()

## Description

## Prototype

*printform1var*.addgraphic ( printform1 *me*, type *graphictype*, point *printpoint1*, point *printpoint2*, point *printpoint3*, point *printmidpoint*, integer *rgb*, integer *borderrgb*, integer *width*, integer *borderwidth*, boolean *visible*, boolean *bordervisible*, string *printname*, type(printform1graphic) *next*, printform1page *page*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| graphictype | None | type | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printpoint3 | None | point | |
| printmidpoint | None | point | |
| rgb | None | integer | |
| borderrgb | None | integer | |
| width | None | integer | |
| borderwidth | None | integer | |
| visible | None | boolean | |
| bordervisible | None | boolean | |
| printname | None | string | |
| next | None | type(printform1graphic) | |
| page | None | printform1page | |
| error | None | integer | |

# addpage()

## Description

### Prototype

*printform1var*.addpage ( printform1 *me*, integer *width*, integer *height*, integer *backgroundrgb*, printform1page *after*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| width | None | integer | |
| height | None | integer | |
| backgroundrgb | None | integer | |
| after | None | printform1page | |
| name | None | string | |
| error | None | integer | |

## addsiblinglink()

### Description

### Prototype

*printform1var*.addsiblinglink ( printform1 *me*, type(db1field) *srcfield*, dataform1table *srctable*, type(db1field) *destfield*, dataform1table *desttable*, type(dataform1linkcontainer) *container*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| srcfield | None | type(db1field) | |
| srctable | None | dataform1table | |
| destfield | None | type(db1field) | |
| desttable | None | dataform1table | |
| container | None | type(dataform1linkcontainer) | |
| error | None | integer | |

## addtable()

### Description

### Prototype

*printform1var*.addtable ( printform1 *me*, type(db1table) *table*, dataform1datasource *source*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| table | None | type(db1table) | |
| source | None | dataform1datasource | |
| error | None | integer | |

## assignfilterobject()

### Description

### Prototype

*printform1var*.assignfilterobject ( printform1 *me*, dataform1filter *dffilter* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| dffilter | None | dataform1filter | |

## blank()

### Description

### Prototype

*printform1var*.blank ( printform1 *me*, boolean *datacontrolsonly* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| datacontrolsonly | .true | boolean | |

## builddisplayform()

### Description

### Prototype

*printform1var*.builddisplayform ( printform1 *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| error | None | integer | |

## clearsiblinglinks()

### Description

### Prototype

*printform1var*.clearsiblinglinks ( printform1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |

## findbitmapsource()

### Description

### Prototype

*printform1var*.findbitmapsource ( printform1 *me*, string *sourcename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| sourcename | None | string | |

## findcontrol()

### Description

### Prototype

*printform1var*.findcontrol ( printform1 *me*, string *controlname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| controlname | None | string | |

# finddatasource()

## Description

## Prototype

*printform1var*.finddatasource ( printform1 *me*, string *sourcename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| sourcename | None | string | |

# findgraphic()

## Description

## Prototype

*printform1var*.findgraphic ( printform1 *me*, string *graphicname* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| graphicname | None | string | |

# findsiblinglink()

## Description

## Prototype

*printform1var*.findsiblinglink ( printform1 *me*, dataform1table *desttable* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| desttable | None | dataform1table | |

# findtable()

## Description

## Prototype

*printform1var*.findtable ( printform1 *me*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| tablename | None | string | |

## getfieldandtable()

### Description

### Prototype

*printform1var*.getfieldandtable ( printform1 *me*, string *fieldname*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| fieldname | None | string | |
| tablename | None | string | |

## getfont()

### Description

### Prototype

*printform1var*.getfont ( printform1 *me*, string *facename*, integer *size*, string *style*, string *weight*, string *decoration* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| facename | None | string | |
| size | None | integer | |
| style | None | string | |
| weight | None | string | |
| decoration | None | string | |

## lock()

### Description

### Prototype

*printform1var*.lock ( printform1 *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| error | None | integer | |

## nameinuse()

### Description

### Prototype

*printform1var*.nameinuse ( printform1 *me*, string *controlname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| controlname | None | string | |

## print()

### Description

### Prototype

*printform1var*.print ( printform1 *me*, boolean *showprintpreview*, string *dialogdata*, wxprintout *printout*, integer *formno*, boolean *closeprintout*, boolean *skipcontent*, pagesetupinfo *psinfo*, boolean *showprinterdialog*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| showprintpreview | .true | boolean | |
| dialogdata | None | string | |
| printout | None | wxprintout | |
| formno | 1 | integer | |
| closeprintout | .true | boolean | |
| skipcontent | .false | boolean | |
| psinfo | None | pagesetupinfo | |
| showprinterdialog | .false | boolean | |
| error | None | integer | |

# refresh()

## Description

## Prototype

*printform1var*.refresh ( printform1 *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |

# removedisplayform()

## Description

## Prototype

*printform1var*.removedisplayform ( printform1 *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |

# saverecord()

## Description

## Prototype

*printform1var*.saverecord ( printform1 *me*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| lock | .false | boolean | |
| error | None | integer | |

# selectcurrent()

## Description

## Prototype

*printform1var*.selectcurrent ( printform1 *me*, type(db1index) *index*, boolean *lock*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| index | None | type(db1index) | |
| lock | `.false` | boolean | |
| error | None | integer | |

## selectfirst()

### Description

### Prototype

*printform1var*.`selectfirst` ( printform1 *me*, boolean *lock*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| lock | `.false` | boolean | |
| error | None | integer | |

## selectkey()

### Description

### Prototype

*printform1var*.`selectkey` ( printform1 *me*, anyvalue *value*, type(db1index) *index*, boolean *lock*, boolean *found*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| value | None | anyvalue | |
| index | None | type(db1index) | |
| lock | `.false` | boolean | |
| found | None | boolean | |
| error | None | integer | |

## selectlast()

### Description

### Prototype

*printform1var*.selectlast ( printform1 *me*, boolean *lock*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| lock | .false | boolean | |
| error | None | integer | |

## selectnext()

### Description

### Prototype

*printform1var*.selectnext ( printform1 *me*, boolean *lock*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| lock | .false | boolean | |
| error | None | integer | |

## selectprevious()

### Description

### Prototype

*printform1var*.selectprevious ( printform1 *me*, boolean *lock*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| lock | .false | boolean | |
| error | None | integer | |

## setcontainer()

### Description

### Prototype

*printform1var*.setcontainer ( printform1 *me*, type(wxcontainer) *container* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| container | None | type(wxcontainer) | |

## setdirtystate()

### Description

### Prototype

*printform1var*.`setdirtystate` ( printform1 *me*, boolean *dirty* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| dirty | `.true` | boolean | |

## setfilter()

### Description

### Prototype

*printform1var*.`setfilter` ( printform1 *me*, string *filter*, string *errtext*, integer *errindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| filter | None | string | |
| errtext | None | string | |
| errindex | None | integer | |

## setlastusedrecord()

### Description

### Prototype

*printform1var*.`setlastusedrecord` ( printform1 *me*, dataform1record *record* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| record | None | dataform1record | |

# setmasterrecord()

## Description

## Prototype

*printform1var*.setmasterrecord ( printform1 *me*, type(db1record) *r*, boolean *dorefresh*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| r | None | type(db1record) | |
| dorefresh | .true | boolean | |
| error | None | integer | |

# setmastertable()

## Description

## Prototype

*printform1var*.setmastertable ( printform1 *me*, dataform1table *table* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| table | None | dataform1table | |

# showpage()

## Description

## Prototype

*printform1var*.showpage ( printform1 *me*, integer *pagenum* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |
| pagenum | None | integer | |

## unlock()

### Description

### Prototype

*printform1var*.unlock ( printform1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1 | |

# printform1arc

# Description

# Type Tags

printform1graphic, dataform1graphic

# Object Value

Objects of type printform1arc have no value, and it is an error to try to get or set this value.

# printform1arc.new()

## Description

## Prototype

*printform1arc*.new ( printform1arc *me*, printform1page *page*, wxgraphicarc *graphic*, point *printpoint1*, point *printpoint2*, point *printmidpoint*, string *printname*, integer *printrgb*, boolean *printvisible*, integer *printborderrgb*, boolean *printbordervisible*, integer *printborderwidth*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |
| page | None | printform1page | |
| graphic | None | wxgraphicarc | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| printmidpoint | None | point | |
| printname | None | string | |
| printrgb | `16777215` | integer | |
| printvisible | `.true` | boolean | |
| printborderrgb | `0` | integer | |
| printbordervisible | `.true` | boolean | |
| printborderwidth | `100` | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | printform1 | |
| formnode | dlistnode | |
| getarcquadrant | function | |
| getboundingrectangle | function | |
| graphic | wxgraphicarc | |
| page | printform1page | |
| pagenode | dlistnode | |
| printable | boolean | |
| printborderrgb | integer | |
| printbordervisible | boolean | |
| printborderwidth | integer | |
| printmidpoint | point | |
| printname | string | |
| printpoint1 | point | |
| printpoint2 | point | |
| printrgb | integer | |
| printvisible | boolean | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |

| Property | Type | Description |
|---|---|---|
| setprintable | function | |
| setrgb | function | |
| setvisible | function | |
| type | type | |

# Methods

## getarcquadrant()

### Description

### Prototype

*printform1arcvar*.getarcquadrant ( printform1arc *me*, type(point) *p* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |
| p | None | type(point) | |

## getboundingrectangle()

### Description

### Prototype

*printform1arcvar*.getboundingrectangle ( printform1arc *me*, point *point1*, point *point2*, boolean *printcoords* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |
| point1 | None | point | |
| point2 | None | point | |
| printcoords | .true | boolean | |

## remove()

### Description

### Prototype

*printform1arcvar*.remove ( printform1arc *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |

## setname()

### Description

### Prototype

*printform1arcvar*.setname ( printform1arc *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*printform1arcvar*.setnext ( printform1arc *me*, type(printform1graphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |
| next | None | type(printform1graphic) | |

## setposition()

### Description

### Prototype

*printform1arcvar*.setposition ( printform1arc *me*, point *printpoint1*, point *printpoint2*, point *printmidpoint*, integer *printborderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printmidpoint | None | point | |
| printborder-width | None | integer | |
| error | None | integer | |

# setprintable()

## Description

## Prototype

*printform1arcvar*.setprintable ( printform1arc *me*, boolean *printable* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |
| printable | .true | boolean | |

# setrgb()

## Description

## Prototype

*printform1arcvar*.setrgb ( printform1arc *me*, integer *rgb*, integer *borderrgb* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |
| rgb | None | integer | |
| borderrgb | None | integer | |

# setvisible()

## Description

## Prototype

*printform1arcvar*.setvisible ( printform1arc *me*, boolean *visible*, boolean *bordervisible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1arc | |
| visible | None | boolean | |
| bordervisible | None | boolean | |

# printform1bitmap

## Description

## Type Tags

printform1control, dataform1control

## Object Value

Objects of type printform1bitmap have no value, and it is an error to try to get or set this value.

## printform1bitmap.new()

### Description

### Prototype

*printform1bitmap*.new ( printform1bitmap *me*, printform1page *page*, wxformbitmap *control*, integer *printleft*, integer *printtop*, integer *printwidth*, integer *printheight*, wxbitmap *bitmap*, string *printscaling*, boolean *printvisible*, string *printname*, type(db1field) *field*, dataform1table *table*, string *displayformat*, boolean *undergraphics*, string *print-alignment*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| page | None | printform1page | |
| control | None | wxformbitmap | |
| printleft | None | integer | |
| printtop | None | integer | |
| printwidth | None | integer | |
| printheight | None | integer | |
| bitmap | None | wxbitmap | |
| printscaling | None | string | |
| printvisible | .true | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| printname | None | string | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| undergraphics | `.false` | boolean | |
| printalignment | `left,top` | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| bitmap | wxbitmap | |
| control | wxformbitmap | |
| controlsource | dataform1controlsource | |
| empty | wxbitmap | |
| emptycolor | integer | |
| form | printform1 | |
| formnode | dlistnode | |
| missing | wxbitmap | |
| missingcolor | integer | |
| missinglinecol-or | integer | |
| page | printform1page | |
| pagenode | dlistnode | |
| printable | boolean | |
| printalignment | string | |
| printheight | integer | |
| printleft | integer | |
| printname | string | |
| printscaling | string | |
| printtop | integer | |
| printvisible | boolean | |
| printwidth | integer | |
| remove | function | |
| setalignment | function | |
| setcolors | function | |
| setempty | function | |

| Property | Type | Description |
|---|---|---|
| setmissing | function | |
| setname | function | |
| setnext | function | |
| setoptions | function | |
| setposition | function | |
| setprintable | function | |
| setscaling | function | |
| setvisible | function | |
| type | type | |
| undergraphics | boolean | |

# Methods

## remove()

### Description

### Prototype

*printform1bitmapvar*.remove ( printform1bitmap *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |

## setalignment()

### Description

### Prototype

*printform1bitmapvar*.setalignment ( printform1bitmap *me*, string *alignment* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| alignment | None | string | |

## setcolors()

### Description

### Prototype

*printform1bitmapvar*.setcolors ( printform1bitmap *me*, integer *emptycolor*, integer *missingcolor*, integer *missinglinecolor* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| emptycolor | None | integer | |
| missingcolor | None | integer | |
| missinglinecolor | None | integer | |

## setempty()

### Description

### Prototype

*printform1bitmapvar*.setempty ( printform1bitmap *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| error | None | integer | |

## setmissing()

### Description

### Prototype

*printform1bitmapvar*.setmissing ( printform1bitmap *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| error | None | integer | |

## setname()

### Description

### Prototype

*printform1bitmapvar*.setname ( printform1bitmap *me*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| name | None | string | |

## setnext()

### Description

### Prototype

*printform1bitmapvar*.setnext ( printform1bitmap *me*, type(printform1control) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| next | None | type(printform1control) | |

## setoptions()

### Description

### Prototype

*printform1bitmapvar*.setoptions ( printform1bitmap *me*, boolean *backgroundvisible*, boolean *undergraphics*, boolean *underbitmaps* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| backgroundvis-ible | None | boolean | |
| undergraphics | None | boolean | |
| underbitmaps | None | boolean | |

## setposition()

### Description

### Prototype

*printform1bitmapvar*.setposition ( printform1bitmap *me*, integer *printleft*, integer *printtop*, integer *printwidth*, integer *printheight*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| printleft | None | integer | |
| printtop | None | integer | |
| printwidth | None | integer | |
| printheight | None | integer | |
| error | None | integer | |

## setprintable()

### Description

### Prototype

*printform1bitmapvar*.`setprintable` ( printform1bitmap *me*, boolean *printable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| printable | `.true` | boolean | |

## setscaling()

### Description

### Prototype

*printform1bitmapvar*.`setscaling` ( printform1bitmap *me*, string *scaling* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| scaling | None | string | |

## setvisible()

### Description

### Prototype

*printform1bitmapvar*.`setvisible` ( printform1bitmap *me*, boolean *visible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1bitmap | |
| visible | None | boolean | |

# printform1control

## Description

## Type Tags

printform1control

## Object Value

Objects of type printform1control have no value, and it is an error to try to get or set this value.

## printform1control.new()

### Description

### Prototype

*printform1control*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundrgb | integer | |
| control | type(wxformcontrol) | |
| controlsource | dataform1controlsource | |
| form | printform1 | |
| formnode | dlistnode | |
| page | printform1page | |
| pagenode | dlistnode | |
| printable | boolean | |

| Property | Type | Description |
|---|---|---|
| printheight | integer | |
| printleft | integer | |
| printname | string | |
| printtop | integer | |
| printwidth | integer | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| setprintable | function | |
| type | type | |

# printform1ellipse

## Description

## Type Tags

printform1graphic, dataform1graphic

## Object Value

Objects of type printform1ellipse have no value, and it is an error to try to get or set this value.

## printform1ellipse.new()

### Description

### Prototype

*printform1ellipse*.new ( printform1ellipse *me*, printform1page *page*, wxgraphicellipse *graphic*, point *printpoint1*, point *printpoint2*, point *printmidpoint*, string *printname*, integer *printrgb*, boolean *printvisible*, integer *printborderrgb*, boolean *printbordervisible*, integer *printborderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1ellipse | |
| page | None | printform1page | |
| graphic | None | wxgraphicellipse | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printmidpoint | None | point | |
| printname | None | string | |
| printrgb | `16777215` | integer | |
| printvisible | `.true` | boolean | |
| printborderrgb | `0` | integer | |
| printbordervis-ible | `.true` | boolean | |
| printborder-width | `100` | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | printform1 | |
| formnode | dlistnode | |
| getboundin-grectangle | function | |
| graphic | wxgraphicellipse | |
| page | printform1page | |
| pagenode | dlistnode | |
| printable | boolean | |
| printborderrgb | integer | |
| printbordervis-ible | boolean | |
| printborder-width | integer | |
| printmidpoint | point | |
| printname | string | |
| printpoint1 | point | |
| printpoint2 | point | |
| printrgb | integer | |
| printvisible | boolean | |
| remove | function | |
| setname | function | |
| setnext | function | |

| Property | Type | Description |
|---|---|---|
| setposition | function | |
| setprintable | function | |
| setrgb | function | |
| setvisible | function | |
| type | type | |

# Methods

## getboundingrectangle()

### Description

### Prototype

*printform1ellipsevar*.getboundingrectangle ( printform1ellipse *me*, point *point1*, point *point2* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1ellipse | |
| point1 | None | point | |
| point2 | None | point | |

## remove()

### Description

### Prototype

*printform1ellipsevar*.remove ( printform1ellipse *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1ellipse | |

## setname()

### Description

### Prototype

*printform1ellipsevar*.setname ( printform1ellipse *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1ellipse | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*printform1ellipsevar*.setnext ( printform1ellipse *me*, type(printform1graphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1ellipse | |
| next | None | type(printform1graphic) | |

## setposition()

### Description

### Prototype

*printform1ellipsevar*.setposition ( printform1ellipse *me*, point *printpoint1*, point *printpoint2*, point *printmidpoint*, integer *printborderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1ellipse | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printmidpoint | None | point | |
| printborder-width | None | integer | |
| error | None | integer | |

## setprintable()

### Description

### Prototype

*printform1ellipsevar*.setprintable ( printform1ellipse *me*, boolean *printable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1ellipse | |
| printable | .true | boolean | |

## setrgb()

### Description

### Prototype

*printform1ellipsevar*.setrgb ( printform1ellipse *me*, integer *rgb*, integer *borderrgb* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1ellipse | |
| rgb | None | integer | |
| borderrgb | None | integer | |

## setvisible()

### Description

### Prototype

*printform1ellipsevar*.setvisible ( printform1ellipse *me*, boolean *visible*, boolean *bordervisible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1ellipse | |
| visible | None | boolean | |
| bordervisible | None | boolean | |

# printform1graphic

## Description

# Type Tags

printform1graphic

# Object Value

Objects of type printform1graphic have no value, and it is an error to try to get or set this value.

# printform1graphic.new()

## Description

## Prototype

*printform1graphic*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | printform1 | |
| formnode | dlistnode | |
| graphic | type(wxgraphic) | |
| page | printform1page | |
| pagenode | dlistnode | |
| printable | boolean | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| setprintable | function | |
| setrgb | function | |
| setvisible | function | |
| type | type | |

# printform1line

## Description

# Type Tags

printform1graphic, dataform1graphic

# Object Value

Objects of type printform1line have no value, and it is an error to try to get or set this value.

# printform1line.new()

## Description

## Prototype

*printform1line*.new ( printform1line *me*, printform1page *page*, wxgraphicline *graphic*, point *printpoint1*, point *printpoint2*, string *printname*, boolean *printvisible*, integer *printrgb*, integer *printwidth*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1line | |
| page | None | printform1page | |
| graphic | None | wxgraphicline | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printname | None | string | |
| printvisible | `.true` | boolean | |
| printrgb | `0` | integer | |
| printwidth | `100` | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | printform1 | |
| formnode | dlistnode | |
| graphic | wxgraphicline | |
| page | printform1page | |
| pagenode | dlistnode | |
| printable | boolean | |

| Property | Type | Description |
|---|---|---|
| printname | string | |
| printpoint1 | point | |
| printpoint2 | point | |
| printrgb | integer | |
| printvisible | boolean | |
| printwidth | integer | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| setprintable | function | |
| setrgb | function | |
| setvisible | function | |
| type | type | |

# Methods

## remove()

### Description

### Prototype

*printform1linevar*.remove ( printform1line *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1line | |

## setname()

### Description

### Prototype

*printform1linevar*.setname ( printform1line *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1line | |
| name | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# setnext()

## Description

## Prototype

*printform1linevar*.setnext ( printform1line *me*, type(printform1graphic) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1line | |
| next | None | type(printform1graphic) | |

# setposition()

## Description

## Prototype

*printform1linevar*.setposition ( printform1line *me*, point *printpoint1*, point *printpoint2*, integer *printwidth*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1line | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printwidth | None | integer | |
| error | None | integer | |

# setprintable()

## Description

## Prototype

*printform1linevar*.setprintable ( printform1line *me*, boolean *printable* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1line | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| printable | `.true` | boolean | |

## setrgb()

### Description

### Prototype

*printform1linevar*.`setrgb` ( printform1line *me*, integer *rgb*, integer *borderrgb* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1line | |
| rgb | None | integer | |
| borderrgb | None | integer | |

## setvisible()

### Description

### Prototype

*printform1linevar*.`setvisible` ( printform1line *me*, boolean *visible*, boolean *bordervisible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1line | |
| visible | None | boolean | |
| bordervisible | None | boolean | |

# printform1page

# Description

# Type Tags

dataform1page

# Object Value

Objects of type printform1page have no value, and it is an error to try to get or set this value.

# printform1page.new()

## Description

## Prototype

*printform1page*.new ( printform1page *me*, printform1 *form*, integer *printwidth*, integer *printheight*, integer *backgroundrgb*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1page | |
| form | None | printform1 | |
| printwidth | 210000 | integer | |
| printheight | 297000 | integer | |
| backgroundrgb | 16777215 | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addcontrol | function | |
| addgraphic | function | |
| builddisplay-form | function | |
| changename | function | |
| controls | dring | |
| form | printform1 | |
| formnode | dlistnode | |
| graphics | dring | |
| name | string | |
| pagenum | integer | |
| print | function | |
| printback-groundrgb | integer | |
| printheight | integer | |
| printwidth | integer | |
| remove | function | |
| resize | function | |

| Property | Type | Description |
|---|---|---|
| setactive | function | |
| type | type | |
| wxformpage | wxform | |

# Methods

## addcontrol()

### Description

### Prototype

*printform1pagevar*.addcontrol ( printform1page *me*, type *controltype*, integer *printleft*, integer *printtop*, integer *printwidth*, integer *printheight*, string *printtext*, boolean *visible*, wxbitmap *bitmap*, string *scaling*, integer *backgroundrgb*, integer *textrgb*, string *printalignment*, wxfont *font*, string *printname*, boolean *backgroundvisible*, boolean *undergraphics*, boolean *underbitmaps*, type(printform1control) *next*, type(db1field) *field*, dataform1table *table*, string *displayformat*, boolean *skipbitmapcheck*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1page | |
| controltype | None | type | |
| printleft | None | integer | |
| printtop | None | integer | |
| printwidth | None | integer | |
| printheight | None | integer | |
| printtext | None | string | |
| visible | .true | boolean | |
| bitmap | None | wxbitmap | |
| scaling | None | string | |
| backgroundrgb | None | integer | |
| textrgb | 0 | integer | |
| printalignment | None | string | |
| font | None | wxfont | |
| printname | None | string | |
| backgroundvisible | .true | boolean | |
| undergraphics | .false | boolean | |
| underbitmaps | .false | boolean | |
| next | None | type(printform1control) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| skip-bitmapcheck | `.false` | boolean | |
| error | None | integer | |

# addgraphic()

## Description

## Prototype

*`printform1pagevar`*.`addgraphic` ( printform1page *`me`*, type *`graphictype`*, point *`printpoint1`*, point *`printpoint2`*, point *`printpoint3`*, point *`printmidpoint`*, integer *`printrgb`*, integer *`printborderrgb`*, integer *`printwidth`*, integer *`printborderwidth`*, boolean *`printvisible`*, boolean *`printbordervisible`*, string *`printname`*, type(printform1graphic) *`next`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1page | |
| graphictype | None | type | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printpoint3 | None | point | |
| printmidpoint | None | point | |
| printrgb | None | integer | |
| printborderrgb | None | integer | |
| printwidth | None | integer | |
| printborder-width | None | integer | |
| printvisible | None | boolean | |
| printbordervis-ible | None | boolean | |
| printname | None | string | |
| next | None | type(printform1graphic) | |
| error | None | integer | |

# builddisplayform()

## Description

### Prototype

*printform1pagevar*.builddisplayform ( printform1page *me*, boolean *override*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1page | |
| override | .false | boolean | |
| error | None | integer | |

## changename()

### Description

### Prototype

*printform1pagevar*.changename ( printform1page *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1page | |
| name | None | string | |
| error | None | integer | |

## print()

### Description

### Prototype

*printform1pagevar*.print ( printform1page *me*, wxprintout *po*, integer *formno*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1page | |
| po | None | wxprintout | |
| formno | None | integer | |
| error | None | integer | |

## remove()

### Description

### Prototype

*printform1pagevar*.remove ( printform1page *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1page | |

## resize()

### Description

### Prototype

*printform1pagevar*.resize ( printform1page *me*, integer *printwidth*, integer *printheight*, boolean *testonly*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1page | |
| printwidth | None | integer | |
| printheight | None | integer | |
| testonly | .false | boolean | |
| error | None | integer | |

## setactive()

### Description

### Prototype

*printform1pagevar*.setactive ( printform1page *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1page | |

# printform1rectangle

# Description

# Type Tags

printform1graphic, dataform1graphic

# Object Value

Objects of type printform1rectangle have no value, and it is an error to try to get or set this value.

# printform1rectangle.new()

## Description

## Prototype

*printform1rectangle*.new ( printform1rectangle *me*, printform1page *page*, wxgraphicrectangle *graphic*, point *printpoint1*, point *printpoint2*, string *printname*, integer *printrgb*, boolean *printvisible*, integer *printborderrgb*, boolean *printbordervisible*, integer *printborderwidth*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1rectangle | |
| page | None | printform1page | |
| graphic | None | wxgraphicrectangle | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printname | None | string | |
| printrgb | `16777215` | integer | |
| printvisible | `.true` | boolean | |
| printborderrgb | `0` | integer | |
| printbordervisible | `.true` | boolean | |
| printborderwidth | `100` | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | printform1 | |
| formnode | dlistnode | |
| graphic | wxgraphicrectangle | |
| page | printform1page | |
| pagenode | dlistnode | |

| Property | Type | Description |
|---|---|---|
| printable | boolean | |
| printborderrgb | integer | |
| printbordervis-ible | boolean | |
| printborder-width | integer | |
| printname | string | |
| printpoint1 | point | |
| printpoint2 | point | |
| printrgb | integer | |
| printvisible | boolean | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| setprintable | function | |
| setrgb | function | |
| setvisible | function | |
| type | type | |

# Methods

## remove()

### Description

### Prototype

*printform1rectanglevar*.remove ( printform1rectangle *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1rectangle | |

## setname()

### Description

### Prototype

*printform1rectanglevar*.setname ( printform1rectangle *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1rectangle | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*printform1rectanglevar*.setnext ( printform1rectangle *me*, type(printform1graphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1rectangle | |
| next | None | type(printform1graphic) | |

## setposition()

### Description

### Prototype

*printform1rectanglevar*.setposition ( printform1rectangle *me*, point *printpoint1*, point *printpoint2*, integer *printborderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1rectangle | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printborder-width | None | integer | |
| error | None | integer | |

## setprintable()

### Description

### Prototype

*printform1rectanglevar*.setprintable ( printform1rectangle *me*, boolean *printable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1rectangle | |
| printable | `.true` | boolean | |

## setrgb()

### Description

### Prototype

*`printform1rectanglevar`*`.setrgb` ( printform1rectangle *`me`*, integer *`rgb`*, integer *`borderrgb`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1rectangle | |
| rgb | None | integer | |
| borderrgb | None | integer | |

## setvisible()

### Description

### Prototype

*`printform1rectanglevar`*`.setvisible` ( printform1rectangle *`me`*, boolean *`visible`*, boolean *`bordervisible`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1rectangle | |
| visible | None | boolean | |
| bordervisible | None | boolean | |

# printform1text

# Description

# Type Tags

printform1control, dataform1control

# Object Value

Objects of type printform1text have no value, and it is an error to try to get or set this value.

# printform1text.new()

## Description

## Prototype

*printform1text*.new ( printform1text *me*, printform1page *page*, wxformtext *control*, integer *printleft*, integer *printtop*, integer *printwidth*, integer *printheight*, string *printtext*, integer *printbackgroundrgb*, integer *printtextrgb*, string *printalignment*, boolean *printvisible*, string *printname*, wxfont *printfont*, boolean *undergraphics*, boolean *underbitmaps*, boolean *backgroundvisible*, type(db1field) *field*, dataform1table *table*, string *displayformat*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| page | None | printform1page | |
| control | None | wxformtext | |
| printleft | None | integer | |
| printtop | None | integer | |
| printwidth | None | integer | |
| printheight | None | integer | |
| printtext | None | string | |
| printbackgroundrgb | 16777215 | integer | |
| printtextrgb | 0 | integer | |
| printalignment | None | string | |
| printvisible | .true | boolean | |
| printname | None | string | |
| printfont | None | wxfont | |
| undergraphics | .false | boolean | |
| underbitmaps | .false | boolean | |
| backgroundvisible | None | boolean | |
| field | None | type(db1field) | |
| table | None | dataform1table | |
| displayformat | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| backgroundvis-ible | boolean | |
| control | wxformtext | |
| controlsource | dataform1controlsource | |
| form | printform1 | |
| formnode | dlistnode | |
| page | printform1page | |
| pagenode | dlistnode | |
| printable | boolean | |
| printalignment | string | |
| printback-groundrgb | integer | |
| printfont | wxfont | |
| printheight | integer | |
| printleft | integer | |
| printname | string | |
| printtext | string | |
| printtextrgb | integer | |
| printtop | integer | |
| printvisible | boolean | |
| printwidth | integer | |
| remove | function | |
| setalignment | function | |
| setback-groundrgb | function | |
| setfont | function | |
| setname | function | |
| setnext | function | |
| setoptions | function | |
| setposition | function | |
| setprintable | function | |
| settext | function | |
| settextrgb | function | |
| setvisible | function | |
| type | type | |

| Property | Type | Description |
|---|---|---|
| underbitmaps | boolean | |
| undergraphics | boolean | |

# Methods

## remove()

### Description

### Prototype

*printform1textvar*.remove ( printform1text *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |

## setalignment()

### Description

### Prototype

*printform1textvar*.setalignment ( printform1text *me*, string *alignment* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| alignment | None | string | |

## setbackgroundrgb()

### Description

### Prototype

*printform1textvar*.setbackgroundrgb ( printform1text *me*, integer *rgb*, integer *red*, integer *green*, integer *blue* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| rgb | None | integer | |
| red | None | integer | |
| green | None | integer | |
| blue | None | integer | |

# setfont()

## Description

## Prototype

*printform1textvar*.setfont ( printform1text *me*, wxfont *font* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| font | None | wxfont | |

# setname()

## Description

## Prototype

*printform1textvar*.setname ( printform1text *me*, string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| name | None | string | |

# setnext()

## Description

## Prototype

*printform1textvar*.setnext ( printform1text *me*, type(printform1control) *next* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(printform1control) | |

# setoptions()

## Description

## Prototype

*printform1textvar*.setoptions ( printform1text *me*, boolean *backgroundvisible*, boolean *undergraphics*, boolean *underbitmaps* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| backgroundvis-ible | None | boolean | |
| undergraphics | None | boolean | |
| underbitmaps | None | boolean | |

# setposition()

## Description

## Prototype

*printform1textvar*.setposition ( printform1text *me*, integer *printleft*, integer *print-top*, integer *printwidth*, integer *printheight*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| printleft | None | integer | |
| printtop | None | integer | |
| printwidth | None | integer | |
| printheight | None | integer | |
| error | None | integer | |

# setprintable()

## Description

## Prototype

*printform1textvar*.setprintable ( printform1text *me*, boolean *printable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| printable | `.true` | boolean | |

## settext()

### Description

### Prototype

*`printform1textvar`*`.settext` ( printform1text *`me`*, string *`text`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| text | None | string | |

## settextrgb()

### Description

### Prototype

*`printform1textvar`*`.settextrgb` ( printform1text *`me`*, integer *`rgb`*, integer *`red`*, integer *`green`*, integer *`blue`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| rgb | None | integer | |
| red | None | integer | |
| green | None | integer | |
| blue | None | integer | |

## setvisible()

### Description

### Prototype

*`printform1textvar`*`.setvisible` ( printform1text *`me`*, boolean *`visible`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1text | |
| visible | None | boolean | |

# printform1triangle

## Description

## Type Tags

printform1graphic, dataform1graphic

## Object Value

Objects of type printform1triangle have no value, and it is an error to try to get or set this value.

## printform1triangle.new()

### Description

### Prototype

*`printform1triangle`*.new ( printform1triangle *`me`*, printform1page *`page`*, wxgraphictriangle *`graphic`*, point *`printpoint1`*, point *`printpoint2`*, point *`printpoint3`*, string *`printname`*, integer *`printrgb`*, boolean *`printvisible`*, integer *`printborderrgb`*, boolean *`printbordervisible`*, integer *`printborderwidth`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1triangle | |
| page | None | printform1page | |
| graphic | None | wxgraphictriangle | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printpoint3 | None | point | |
| printname | None | string | |
| printrgb | `16777215` | integer | |
| printvisible | `.true` | boolean | |
| printborderrgb | `0` | integer | |
| printbordervisible | `.true` | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| printborder-width | `100` | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | printform1 | |
| formnode | dlistnode | |
| graphic | wxgraphictriangle | |
| page | printform1page | |
| pagenode | dlistnode | |
| printable | boolean | |
| printborderrgb | integer | |
| printbordervis-ible | boolean | |
| printborder-width | integer | |
| printname | string | |
| printpoint1 | point | |
| printpoint2 | point | |
| printpoint3 | point | |
| printrgb | integer | |
| printvisible | boolean | |
| remove | function | |
| setname | function | |
| setnext | function | |
| setposition | function | |
| setprintable | function | |
| setrgb | function | |
| setvisible | function | |
| type | type | |

# Methods

## remove()

### Description

### Prototype

*printform1trianglevar*.remove ( printform1triangle *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1triangle | |

## setname()

### Description

### Prototype

*printform1trianglevar*.setname ( printform1triangle *me*, string *name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1triangle | |
| name | None | string | |
| error | None | integer | |

## setnext()

### Description

### Prototype

*printform1trianglevar*.setnext ( printform1triangle *me*, type(printform1graphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1triangle | |
| next | None | type(printform1graphic) | |

## setposition()

### Description

### Prototype

*printform1trianglevar*.setposition ( printform1triangle *me*, point *printpoint1*, point *printpoint2*, point *printpoint3*, integer *printborderwidth*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1triangle | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printpoint3 | None | point | |
| printborder-width | None | integer | |
| error | None | integer | |

## setprintable()

### Description

### Prototype

*printform1trianglevar*.setprintable ( printform1triangle *me*, boolean *printable* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1triangle | |
| printable | .true | boolean | |

## setrgb()

### Description

### Prototype

*printform1trianglevar*.setrgb ( printform1triangle *me*, integer *rgb*, integer *borderrgb* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1triangle | |
| rgb | None | integer | |
| borderrgb | None | integer | |

## setvisible()

### Description

### Prototype

*printform1trianglevar*.setvisible ( printform1triangle *me*, boolean *visible*, boolean *bordervisible* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printform1triangle | |
| visible | None | boolean | |
| bordervisible | None | boolean | |

# createblankbmp()

## Description

## Prototype

createblankbmp ( integer *width*, integer *height*, integer *color*, boolean *missing*, integer *linecolor*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | None | integer | |
| height | None | integer | |
| color | 10526880 | integer | |
| missing | .false | boolean | |
| linecolor | 0 | integer | |
| error | None | integer | |

# findnextfocusablecontrol()

## Description

## Prototype

findnextfocusablecontrol ( type(wxformcontrol) *c* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| c | None | type(wxformcontrol) | |

# getarcboundingrectangle()

## Description

## Prototype

`getarcboundingrectangle` ( type(point) *`a_point1`*, type(point) *`a_point2`*, type(point) *`a_midpoint`*, point *`point1`*, point *`point2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a_point1 | None | type(point) | |
| a_point2 | None | type(point) | |
| a_midpoint | None | type(point) | |
| point1 | None | point | |
| point2 | None | point | |

# getarcquadrant()

## Description

## Prototype

`getarcquadrant` ( type(point) *`midpoint`*, type(point) *`p`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| midpoint | None | type(point) | |
| p | None | type(point) | |

# getbitmaptype()

## Description

## Prototype

`getbitmaptype` ( string *`filename`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |

# getellipseboundingrectangle()

## Description

## Prototype

`getellipseboundingrectangle` ( type(point) *`e_point1`*, type(point) *`e_point2`*, type(point) *`e_midpoint`*, point *`point1`*, point *`point2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| e_point1 | None | type(point) | |
| e_point2 | None | type(point) | |
| e_midpoint | None | type(point) | |
| point1 | None | point | |
| point2 | None | point | |

# isvaliddbcontrol()

## Description

## Prototype

`isvaliddbcontrol` ( type(dataform1control) *`control`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| control | None | type(dataform1control) | |
| error | None | integer | |

# renderprintform1page()

## Description

## Prototype

`renderprintform1page` ( wxprintout *`po`*, printform1page *`p`*, integer *`formno`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| po | None | wxprintout | |
| p | None | printform1page | |
| formno | None | integer | |
| error | None | integer | |

# retrievebitmap()

## Description

## Prototype

`retrievebitmap` ( string *`filename`*, string *`format`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| format | None | string | |
| error | None | integer | |

# Chapter 27. datetimelib

## datefromdatetime()

### Description

### Prototype

`datefromdatetime` ( datetime *`dt`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | datetime | |

## datetimefromdateandtime()

### Description

### Prototype

`datetimefromdateandtime` ( date *`d`*, time *`t`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| d | None | date | |
| t | None | time | |

## easter()

### Description

### Prototype

`easter` ( integer *`year`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| year | None | integer | |

# isleapyear()

## Description

## Prototype

`isleapyear` ( integer *`year`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| year | None | integer | |

# timefromdatetime()

## Description

## Prototype

`timefromdatetime` ( datetime *`dt`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | datetime | |

# Chapter 28. db1util

The db1util library is the primary collection of helpful functions for working with databases in SIMPOL. The db1util name comes from the type tag names that are associated with the single-user SBME component and the multi-user PPCS component. These functions are key components used throughout the data-aware environment and are also used in numerous support libraries.

# db1fieldinfo

## Description

## Type Tags

None

## Object Value

Objects of type db1fieldinfo have no value, and it is an error to try to get or set this value.

# db1fieldinfo.new()

### Description

### Prototype

*db1fieldinfo*.new( db1fieldinfo *me*, type *datatype*, string *name*, string *displayformat*, string *help*, string *comment*, string *sharename*, string *sharetype*, boolean *shareable*, string *default*, string *calculation*, string *validation*, boolean *required*, boolean *read_only*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | db1fieldinfo | |
| datatype | None | type | |
| name | None | string | |
| displayformat | None | string | |
| help | None | string | |
| comment | None | string | |
| sharename | None | string | |
| sharetype | None | string | |
| shareable | None | boolean | |
| default | None | string | |
| calculation | None | string | |
| validation | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| required | `.false` | boolean | |
| read_only | `.false` | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| calculation | string | |
| comment | string | |
| datatype | type | |
| default | string | |
| displayformat | string | |
| help | string | |
| name | string | |
| read_only | boolean | |
| required | boolean | |
| shareable | boolean | |
| sharename | string | |
| sharetype | string | |
| type | type | |
| validation | string | |

# tdataview

# Description

# Type Tags

None

# Object Value

Objects of type tdataview have no value, and it is an error to try to get or set this value.

# tdataview.new()

## Description

## Prototype

*`tdataview`*.new ( tdataview *`me`*, type(db1table) *`table`*, array *`fieldlist`*, dring *`parent`*, string *`name`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tdataview | |
| table | None | type(db1table) | |
| fieldlist | None | array | |
| parent | None | dring | |
| name | `untitled` | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| fieldlist | array | |
| name | string | |
| node | dlistnode | |
| setfieldlist | function | |
| setname | function | |
| setparent | function | |
| table | type(db1table) | |
| type | type | |

# Methods

## setfieldlist()

### Description

### Prototype

*`tdataviewvar`*`.setfieldlist` ( tdataview *`me`*, array *`fieldlist`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tdataview | |
| fieldlist | None | array | |
| error | None | integer | |

## setname()

### Description

### Prototype

*tdataviewvar*.setname ( tdataview *me*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tdataview | |
| name | None | string | |

## setparent()

### Description

### Prototype

*tdataviewvar*.setparent ( tdataview *me*, dring *parent* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tdataview | |
| parent | None | dring | |

# FCASE()

## Description

## Prototype

FCASE ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# TCASE()

## Description

## Prototype

TCASE ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# addsysfield()

## Description

## Prototype

addsysfield ( type(db1table) *fieldstable*, integer *tableid*, string *fieldname*, string *fieldtype*, boolean *indexed*, string *help*, string *comment*, string *default*, string *calculation*, string *validation*, boolean *required*, boolean *read_only*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| fieldstable | None | type(db1table) | |
| tableid | None | integer | |
| fieldname | None | string | |
| fieldtype | None | string | |
| indexed | .false | boolean | |
| help | None | string | |
| comment | None | string | |
| default | None | string | |
| calculation | None | string | |
| validation | None | string | |
| required | .false | boolean | |
| read_only | .false | boolean | |
| error | None | integer | |

# addsysfieldext()

## Description

## Prototype

addsysfieldext ( type(db1table) *fieldsexttable*, integer *tableid*, integer *fieldid*, boolean *shareable*, string *sharename*, string *sharetype*, string *displayformat*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| fieldsexttable | None | type(db1table) | |
| tableid | None | integer | |
| fieldid | None | integer | |
| shareable | `.true` | boolean | |
| sharename | None | string | |
| sharetype | None | string | |
| displayformat | None | string | |
| error | None | integer | |

# addsystable()

## Description

## Prototype

`addsystable` ( type(db1base) *`sbmfile`*, string *`tablename`*, integer *`initialserialnumber`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| tablename | None | string | |
| initialserial-number | `0` | integer | |
| error | None | integer | |

# analyzerecorddata()

## Description

## Prototype

`analyzerecorddata` ( type(db1record) *`r`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |

# blobtotext()

## Description

## Prototype

`blobtotext` ( blob *b* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| b | None | blob | |

# checkandupdatetabledefs()

## Description

## Prototype

`checkandupdatetabledefs` ( sbme1 *`sbmfile`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | sbme1 | |
| error | None | integer | |

# compareeqdb1recs()

## Description

This function compares the first record to the second and return either `.true` or `.false` depending on whether they are equal or not. If either a field data type is not the same or the values are not equivalent, then the result will be `.false`.

## Prototype

`compareeqdb1recs` ( type(db1record) *`r1`*, type(db1record) *`r2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r1 | None | type(db1record) | |
| r2 | None | type(db1record) | |

# copy_db1rec_to_db1rec()

## Description

## Prototype

`copy_db1rec_to_db1rec` ( type(db1record) *`fr`*, type(db1record) *`tr`*, integer *`iErrnum`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| fr | None | type(db1record) | |
| tr | None | type(db1record) | |
| iErrnum | None | integer | |

# copy_db1rec_to_db1table()

## Description

## Prototype

`copy_db1rec_to_db1table` ( type(db1record) *`fr`*, type(db1table) *`tt`*, string *`sErrlog`*, integer *`iErrnum`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| fr | None | type(db1record) | |
| tt | None | type(db1table) | |
| sErrlog | `errlog.txt` | string | |
| iErrnum | None | integer | |

# copyextendedinfo()

## Description

## Prototype

`copyextendedinfo` ( type(db1base) *`sbmfilefrom`*, type(db1base) *`sbmfileto`*, string *`table-name`*, string *`tablenameout`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfilefrom | None | type(db1base) | |
| sbmfileto | None | type(db1base) | |
| tablename | None | string | |
| tablenameout | None | string | |
| error | None | integer | |

# create_sbme1table_from_db1table()

## Description

This function examines the file definition of the db1table passed and creates a new table with the same definition in the sbme1 file that was passed. Only characteristics that are appropriate for sbme1table will be taken across (ie. no field formats, just data types). Returns `0` if successful, otherwise one of several possible error messages.

## Prototype

`create_sbme1table_from_db1table` ( type(db1table) *`dbSrc`*, type(db1base) *`sbmFile`*, string *`sNewTableName`*, boolean *`skipindexes`*, boolean *`createextendedinfo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dbSrc | None | type(db1table) | |
| sbmFile | None | type(db1base) | |
| sNewTable-Name | None | string | |
| skipindexes | `.false` | boolean | |
| createex-tendedinfo | `.true` | boolean | |

# createdefaultsystemtableentries()

## Description

## Prototype

`createdefaultsystemtableentries` ( type(db1base) *`sbmfile`*, type(db1table) *`table`*, string *`tablename`*, type(db1table) *`tTables`*, type(db1table) *`tFields`*, type(db1table) *`tFieldsext`*, string *`errtext`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| table | None | type(db1table) | |
| tablename | None | string | |
| tTables | None | type(db1table) | |
| tFields | None | type(db1table) | |
| tFieldsext | None | type(db1table) | |
| errtext | None | string | |
| error | None | integer | |

# createextendedtableinfo()

## Description

## Prototype

`createextendedtableinfo` ( type(db1base) *`sbmfile`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| error | None | integer | |

# createsysfields()

## Description

## Prototype

`createsysfields` ( type(db1base) *`sbmfile`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| error | None | integer | |

# createsysfieldsext()

## Description

## Prototype

`createsysfieldsext` ( type(db1base) *`sbmfile`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| error | None | integer | |

# createsystables()

## Description

## Prototype

`createsystables` ( type(db1base) *`sbmfile`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| error | None | integer | |

# createsystablesysteminexistingsbm()

## Description

## Prototype

`createsystablesysteminexistingsbm` ( type(db1base) *`sbmfile`*, integer *`error`*, string *`errtext`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| error | None | integer | |
| errtext | None | string | |

# deleteallrecordsfromtable()

## Description

## Prototype

`deleteallrecordsfromtable` ( type(db1table) *`table`*, boolean *`optimize`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| optimize | `.false` | boolean | |
| error | None | integer | |

# fieldval2string()

## Description

## Prototype

`fieldval2string` ( type(db1field) *`fld`*, type(db1record) *`r`*, SBLlocaledateinfo *`ldiLocale`*, SBLNumSettings *`ns`*, string *`sFormat`*, boolean *`shownull`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| fld | None | type(db1field) | |
| r | None | type(db1record) | |
| ldiLocale | None | SBLlocaledateinfo | |
| ns | None | SBLNumSettings | |
| sFormat | None | string | |
| shownull | `.false` | boolean | |

# getdefaultdisplayformat()

## Description

## Prototype

`getdefaultdisplayformat` ( type *`simpoldatatype`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| simpoldatatype | None | type | |

# getdefaultformat()

## Description

This function returns the default format for the data type that is passed.

## Prototype

`getdefaultformat` ( type *`datatype`*, string *`defboolean`*, string *`definteger`*, string *`defnumber`*, string *`defdate`*, string *`deftime`*, string *`defdatetime`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datatype | None | type | |
| defboolean | `T\|F` | string | |
| definteger | `.` | string | |
| defnumber | `999999.00` | string | |
| defdate | `yyyy.mm.0d` | string | |
| deftime | `hh:mm:ss` | string | |
| defdatetime | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |

# getfield()

## Description

If the string passed is the name of a field in the table passed, it returns a reference to the field object, otherwise it returns .nul (case-sensitive).

## Prototype

`getfield` ( type(db1table) *`table`*, string *`sFieldname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| sFieldname | None | string | |

# getfieldextid()

## Description

Given a *.sbm file, a field name, and a table ID – see gettableid() – this function will return the fieldext ID in the system tables for that table. If the field name is not present it will return `.nul` and an error code in the *`error`* parameter.

## Prototype

`getfieldextid` ( type(db1base) *`sbmfile`*, integer *`tableid`*, string *`fieldname`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| tableid | None | integer | |
| fieldname | None | string | |
| error | None | integer | |

# getfieldid()

## Description

Given a *.sbm file, a field name, and a table ID – see gettableid() – this function will return the field ID in the system tables for that table. If the field name is not present it will return `.nul` and an error code in the *`error`* parameter.

## Prototype

`getfieldid` ( type(db1base) *`sbmfile`*, integer *`tableid`*, string *`fieldname`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| tableid | None | integer | |
| fieldname | None | string | |
| error | None | integer | |

# getfieldinfoarray()

## Description

This is a legacy function for retrieving an array of field info information. It retrieves an array that has the field names from a table in index order beginning with 1, and for each name has a field reference like

fieldinfo["myfield"] and the display format for the field in fieldinfo["myfield", 1] with the total number of fields stored in teh array nul position []. Additional information is stored in the positions 2, 3, 4, 5, and 6, as follows: the help string, comment, field name, if the field is considered shareable, and its share name.

## Prototype

`getfieldinfoarray` ( type(db1table) *`table`*, string *`defboolean`*, string *`definteger`*, string *`defnumber`*, string *`defdate`*, string *`deftime`*, string *`defdatetime`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| defboolean | `T\|F` | string | |
| definteger | `.` | string | |
| defnumber | `.` | string | |
| defdate | `yyyy0m0d` | string | |
| deftime | `hh:mm:ss` | string | |
| defdatetime | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |

# getfieldscount()

## Description

## Prototype

`getfieldscount` ( type(db1base) *`sbmfile`*, integer *`tableid`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| tableid | None | integer | |
| error | None | integer | |

# getfieldsextcount()

## Description

## Prototype

`getfieldsextcount` ( type(db1base) *`sbmfile`*, integer *`tableid`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| tableid | None | integer | |
| error | None | integer | |

# getindex()

## Description

If the field name passed is a field in the table passed, and it is indexed, then this function will return the index reference, otherwise it will return `.nul` (case-sensitive).

## Prototype

`getindex` ( type(db1table) *`table`*, string *`sFieldname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| sFieldname | None | string | |

# getnewfieldinfoarray()

## Description

## Prototype

`getnewfieldinfoarray` ( type(db1table) *`table`*, string *`defboolean`*, string *`definteger`*, string *`defnumber`*, string *`defdate`*, string *`deftime`*, string *`defdatetime`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| defboolean | `T\|F` | string | |
| definteger | `.` | string | |
| defnumber | `.` | string | |
| defdate | `yyyy0m0d` | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| deftime | `hh:mm:ss` | string | |
| defdatetime | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |

# getsysserial()

## Description

## Prototype

`getsysserial` ( type(db1base) ***`sbmfile`***, string ***`tablename`***, integer ***`error`*** )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| tablename | None | string | |
| error | None | integer | |

# getsystemtable()

## Description

## Prototype

`getsystemtable` ( type(db1base) ***`sbmfile`***, string ***`systablename`***, integer ***`error`*** )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| systablename | None | string | |
| error | None | integer | |

# gettableid()

## Description

Given an sbm file and a table name, this function will return the table ID in the system tables for that table. If the table is not present it will return .nul and an error code in the error parameter.

## Prototype

`gettableid` ( type(db1base) *`sbmfile`*, string *`tablename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| tablename | None | string | |
| error | None | integer | |

# gettablename()

## Description

This function was implemented at a time when the ppcstype1file type only had a filename property. Now it has a tablename property as well. It will still return the correct table name for a ppcstype1file, sbme1table, or vola1table.

## Prototype

`gettablename` ( type(db1table) *`table`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |

# gettableparent()

## Description

This returns the parent of the table passed. If it is of type sbme1table, it returns the sbme1 object, if it is of type ppcstype1file, it returns the ppcstype1 object, and if it is of type vola1table, it returns the vola1base object.

## Prototype

`gettableparent` ( type(db1table) *`table`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |

# getuniqueindex()

## Description

Returns the first unique index found in the table passed. If none of the indexes are unique, it returns `.nul`.

## Prototype

`getuniqueindex` ( type(db1table) *`table`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |

# getuniquenindex()

## Description

Returns the first unique numeric index found in the table passed. If none of the indexes are unique and numeric, it returns `.nul`.

## Prototype

`getuniquenindex` ( type(db1table) *`table`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |

# isdb1field()

## Description

This checks that the field passed is either of type ppcstype1field, sbme1field, or vola1field.

## Prototype

`isdb1field` ( type(*) *`field`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| field | None | type(*) | |

# isdb1table()

## Description

This checks that the table passed is either of type ppcstype1file, sbme1table, or vola1table.

## Prototype

`isdb1table` ( type(*) *`table`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(*) | |

# isfield()

## Description

This checks the field name against the table object passed and returns `.true` if it is the name of a field (case-sensitive) or `.false` if it can't find it.

## Prototype

`isfield` ( type(db1table) *`table`*, string *`sFieldname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| sFieldname | None | string | |

# isindex()

## Description

Returns `.true` if the name passed is a field in the table passed and if the field is indexed. Otherwise it returns `.false`.

## Prototype

`isindex` ( type(db1table) *`table`*, string *`sFieldname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFieldname | None | string | |

# islocked()

## Description

## Prototype

`islocked` ( type(db1record) *`r`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |

# lockrecord()

## Description

## Prototype

`lockrecord` ( type(db1record) *`r`*, string *`locktype`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |
| locktype | `exclusive` | string | |
| error | None | integer | |

# outputdb1record()

## Description

## Prototype

`outputdb1record` ( type(db1record) *`r`*, SBLlocaledateinfo *`ldiLocale`*, SBLNumSettings *`ns`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| ldiLocale | None | SBLlocaledateinfo | |
| ns | None | SBLNumSettings | |

# paddedhex()

## Description

## Prototype

`paddedhex` ( integer *`i`*, integer *`places`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| i | None | integer | |
| places | None | integer | |

# removetablefromsystemtables()

## Description

For the table name passed if it is found it removes the table and all field and fieldext entries from the system information in a *.sbm file.

## Prototype

`removetablefromsystemtables` ( type(db1base) *`sbmfile`*, string *`tablename`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| tablename | None | string | |

# storerecord()

## Description

## Prototype

`storerecord` ( type(db1record) *`r`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |
| error | None | integer | |

# string2fieldval()

## Description

## Prototype

`string2fieldval` ( string *`s`*, type(db1field) *`fld`*, SBLlocaledateinfo *`ldiLocale`*, SBLNumSettings *`ns`*, string *`sFormat`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| fld | None | type(db1field) | |
| ldiLocale | None | SBLlocaledateinfo | |
| ns | None | SBLNumSettings | |
| sFormat | None | string | |
| error | None | integer | |

# string2val()

## Description

## Prototype

`string2val` ( string *`s`*, type *`datatype`*, SBLlocaledateinfo *`ldiLocale`*, SBLNumSettings *`ns`*, string *`sFormat`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| datatype | None | type | |
| ldiLocale | None | SBLlocaledateinfo | |
| ns | None | SBLNumSettings | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFormat | None | string | |
| error | None | integer | |

# tableexists()

## Description

## Prototype

`tableexists` ( type(db1base) *`sbmfile`*, string *`tablename`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbmfile | None | type(db1base) | |
| tablename | None | string | |

# unlockrecord()

## Description

## Prototype

`unlockrecord` ( type(db1record) *`r`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |
| error | None | integer | |

# updateextfieldinfo()

## Description

## Prototype

`updateextfieldinfo` ( type(db1table) *`table`*, string *`oldfieldname`*, string *`newfieldname`*, string *`fieldtype`*, boolean *`indexed`*, string *`help`*, string *`comment`*, boolean *`shareable`*, string *`sharename`*, string *`sharetype`*, string *`displayformat`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| oldfieldname | None | string | |
| newfieldname | None | string | |
| fieldtype | None | string | |
| indexed | None | boolean | |
| help | None | string | |
| comment | None | string | |
| shareable | None | boolean | |
| sharename | None | string | |
| sharetype | None | string | |
| displayformat | None | string | |
| error | None | integer | |

# updatesysfield()

## Description

## Prototype

`updatesysfield` ( type(db1record) *`r`*, integer *`tableid`*, integer *`fieldid`*, string *`fieldname`*, string *`fieldtype`*, boolean *`indexed`*, string *`help`*, string *`comment`*, string *`default`*, string *`calculation`*, string *`validation`*, boolean *`required`*, boolean *`read_only`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |
| tableid | None | integer | |
| fieldid | None | integer | |
| fieldname | None | string | |
| fieldtype | None | string | |
| indexed | `.false` | boolean | |
| help | None | string | |
| comment | None | string | |
| default | None | string | |
| calculation | None | string | |
| validation | None | string | |
| required | `.false` | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| read_only | `.false` | boolean | |
| error | None | integer | |

# updatesysfieldext()

## Description

## Prototype

`updatesysfieldext` ( type(db1record) *`r`*, integer *`tableid`*, integer *`fieldid`*, boolean *`shareable`*, string *`sharename`*, string *`sharetype`*, string *`displayformat`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |
| tableid | None | integer | |
| fieldid | None | integer | |
| shareable | `.true` | boolean | |
| sharename | None | string | |
| sharetype | None | string | |
| displayformat | None | string | |
| error | None | integer | |

# val2string()

## Description

## Prototype

`val2string` ( type *`datatype`*, anyvalue *`value`*, SBLlocaledateinfo *`ldiLocale`*, SBLNumSettings *`ns`*, string *`sFormat`*, boolean *`shownull`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datatype | None | type | |
| value | None | anyvalue | |
| ldiLocale | None | SBLlocaledateinfo | |
| ns | None | SBLNumSettings | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFormat | None | string | |
| shownull | None | boolean | |

# Chapter 29. dbconverter

This library implements the various import/export converters and provides a method for adding new ones.

# __tdisplayformats

## Description

## Type Tags

None

## Object Value

Objects of type __tdisplayformats have no value, and it is an error to try to get or set this value.

# __tdisplayformats.new()

### Description

### Prototype

__*tdisplayformats*.new ( __tdisplayformats *me*, string *defboolean*, string *definteger*, string *defnumber*, string *defdate*, string *deftime*, string *defdatetime* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | __tdisplayformats | |
| defboolean | `T\|F` | string | |
| definteger | `.` | string | |
| defnumber | `999999.00` | string | |
| defdate | `yyyy.0m.0d` | string | |
| deftime | `hh:mm:ss` | string | |
| defdatetime | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| defboolean | string | |

| Property | Type | Description |
|---|---|---|
| defdate | string | |
| defdatetime | string | |
| definteger | string | |
| defnumber | string | |
| deftime | string | |
| type | type | |

# dbASDConverter

## Description

## Type Tags

None

## Object Value

Objects of type dbASDConverter have no value, and it is an error to try to get or set this value.

## dbASDConverter.new()

### Description

### Prototype

*dbASDConverter*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| charsize | integer | |
| datastartrow | integer | |
| fielddelimiter | string | |
| fieldnamesrow | integer | |
| fieldseparator | string | |
| lbo | boolean | |
| recordseparator | string | |
| type | type | |

# dbASDExport

## Description

## Type Tags

dbconverter, dbexportconverter

## Object Value

Objects of type dbASDExport have no value, and it is an error to try to get or set this value.

## dbASDExport.new()

### Description

### Prototype

*dbASDExport*.new ( dbASDExport *me*, string *fielddelimiter*, string *fieldseparator*, string *recordseparator*, integer *fieldnamesrow*, integer *datastartrow* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDExport | |
| fielddelimiter | None | string | |
| fieldseparator | , | string | |
| recordseparator | | string | |
| fieldnamesrow | 0 | integer | |
| datastartrow | 1 | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| append | boolean | |
| createtable | function | |
| expconverter | dbexportconverter | |
| export | function | |
| fileref | fsfileoutputstream | |
| headerwritten | boolean | |
| opentarget | function | |
| props | dbASDConverter | |

| Property | Type | Description |
|---|---|---|
| rowsout | integer | |
| type | type | |
| writeheader | function | |
| writerecord | function | |

# Methods

## createtable()

### Description

### Prototype

*dbASDExportvar*.createtable ( dbASDExport *me*, dbconvtable *table* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDExport | |
| table | None | dbconvtable | |

## export()

### Description

### Prototype

*dbASDExportvar*.export ( dbASDExport *me*, type(dbimportconverter) *imp*, integer *error*, integer *reccount* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDExport | |
| imp | None | type(dbimportconverter) | |
| error | None | integer | |
| reccount | .inf | integer | |

## opentarget()

### Description

### Prototype

*dbASDExportvar*.opentarget ( dbASDExport *me*, string *filename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDExport | |
| filename | None | string | |

## writeheader()

### Description

### Prototype

*dbASDExportvar*.writeheader ( dbASDExport *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDExport | |

## writerecord()

### Description

### Prototype

*dbASDExportvar*.writerecord ( dbASDExport *me*, dbconvrecord *r* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDExport | |
| r | None | dbconvrecord | |

# dbASDImport

# Description

# Type Tags

dbconverter, dbimportconverter

# Object Value

Objects of type dbASDImport have no value, and it is an error to try to get or set this value.

# dbASDImport.new()

## Description

## Prototype

*dbASDImport*.new ( dbASDImport *me*, string *fielddelimiter*, string *fieldseparator*, string *recordseparator*, integer *fieldnamesrow*, integer *datastartrow* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDImport | |
| fielddelimiter | None | string | |
| fieldseparator | , | string | |
| recordseparator | | string | |
| fieldnamesrow | 0 | integer | |
| datastartrow | 1 | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| buffer | string | |
| bufpos | integer | |
| fileref | fsfileinputstream | |
| impconverter | dbimportconverter | |
| opensource | function | |
| props | dbASDConverter | |
| readrecord | function | |
| recordcount | function | |
| reopensource | function | |
| rowsread | integer | |
| type | type | |

# Methods

## opensource()

### Description

### Prototype

*dbASDImportvar*.opensource ( dbASDImport *me*, string *filename*, string *codepage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDImport | |
| filename | None | string | |
| codepage | None | string | |

## readrecord()

### Description

### Prototype

*dbASDImportvar*.readrecord ( dbASDImport *me*, integer *error*, boolean *rawmode* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDImport | |
| error | None | integer | |
| rawmode | .false | boolean | |

## recordcount()

### Description

### Prototype

*dbASDImportvar*.recordcount ( dbASDImport *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDImport | |

## reopensource()

### Description

### Prototype

*dbASDImportvar*.reopensource ( dbASDImport *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbASDImport | |

# dbCSVConverter

## Description

## Type Tags

None

## Object Value

Objects of type dbCSVConverter have no value, and it is an error to try to get or set this value.

## dbCSVConverter.new()

### Description

### Prototype

*dbCSVConverter*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| charsize | integer | |
| datastartrow | integer | |
| fielddelimiter | string | |
| fieldnamesrow | integer | |
| fieldseparator | string | |
| fileencoding | integer | |
| lbo | boolean | |
| recordseparator | string | |
| setfileencoding | function | |
| type | type | |
| useBOM | boolean | |

## Methods

### setfileencoding()

#### Description

### Prototype

*dbCSVConvertervar*.setfileencoding ( dbCSVConverter *me*, integer *fileencoding*, boolean *useBOM* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVConverter | |
| fileencoding | None | integer | |
| useBOM | .false | boolean | |

# dbCSVExport

## Description

## Type Tags

dbconverter, dbexportconverter

## Object Value

Objects of type dbCSVExport have no value, and it is an error to try to get or set this value.

## dbCSVExport.new()

### Description

### Prototype

*dbCSVExport*.new ( dbCSVExport *me*, string *fieldseparator*, string *recordseparator*, integer *fieldnamesrow*, integer *datastartrow* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVExport | |
| fieldseparator | , | string | |
| recordseparator | | string | |
| fieldnamesrow | 1 | integer | |
| datastartrow | 2 | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| append | boolean | |

| Property | Type | Description |
|---|---|---|
| createtable | function | |
| expconverter | dbexportconverter | |
| export | function | |
| fileref | fsfileoutputstream | |
| headerwritten | boolean | |
| opentarget | function | |
| props | dbCSVConverter | |
| rowsout | integer | |
| type | type | |
| writeheader | function | |
| writerecord | function | |

# Methods

## createtable()

### Description

### Prototype

*dbCSVExportvar*.createtable ( dbCSVExport *me*, dbconvtable *table* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVExport | |
| table | None | dbconvtable | |

## export()

### Description

### Prototype

*dbCSVExportvar*.export ( dbCSVExport *me*, type(dbimportconverter) *imp*, integer *error*, integer *reccount* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVExport | |
| imp | None | type(dbimportconverter) | |
| error | None | integer | |
| reccount | .inf | integer | |

## opentarget()

### Description

### Prototype

*dbCSVExportvar*.opentarget ( dbCSVExport *me*, string *filename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVExport | |
| filename | None | string | |

## writeheader()

### Description

### Prototype

*dbCSVExportvar*.writeheader ( dbCSVExport *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVExport | |

## writerecord()

### Description

### Prototype

*dbCSVExportvar*.writerecord ( dbCSVExport *me*, dbconvrecord *r* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVExport | |
| r | None | dbconvrecord | |

# dbCSVImport

# Description

# Type Tags

dbconverter, dbimportconverter

# Object Value

Objects of type dbCSVImport have no value, and it is an error to try to get or set this value.

# dbCSVImport.new()

## Description

## Prototype

*dbCSVImport*.new ( dbCSVImport *me*, string *fieldseparator*, string *recordseparator*, integer *fieldnamesrow*, integer *datastartrow* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVImport | |
| fieldseparator | , | string | |
| recordseparator | | string | |
| fieldnamesrow | 0 | integer | |
| datastartrow | 1 | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| buffer | string | |
| bufpos | integer | |
| fileref | fsfileinputstream | |
| impconverter | dbimportconverter | |
| opensource | function | |
| props | dbCSVConverter | |
| readrecord | function | |
| recordcount | function | |
| reopensource | function | |
| rowsread | integer | |
| type | type | |

# Methods

## opensource()

### Description

### Prototype

*dbCSVImportvar*.opensource ( dbCSVImport *me*, string *filename*, string *codepage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVImport | |
| filename | None | string | |
| codepage | None | string | |

## readrecord()

### Description

### Prototype

*dbCSVImportvar*.readrecord ( dbCSVImport *me*, integer *error*, boolean *rawmode* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVImport | |
| error | None | integer | |
| rawmode | .false | boolean | |

## recordcount()

### Description

### Prototype

*dbCSVImportvar*.recordcount ( dbCSVImport *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVImport | |

## reopensource()

### Description

### Prototype

*dbCSVImportvar*.reopensource ( dbCSVImport *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbCSVImport | |

# dbPPCS1Export

## Description

## Type Tags

dbconverter, dbexportconverter

## Object Value

Objects of type dbPPCS1Export have no value, and it is an error to try to get or set this value.

## dbPPCS1Export.new()

### Description

### Prototype

*dbPPCS1Export*.new ( dbPPCS1Export *me*, ppcstype1file *table* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Export | |
| table | None | ppcstype1file | |

## Properties

| Property | Type | Description |
|---|---|---|
| createtable | function | |
| currentview | tdataview | |
| expconverter | dbexportconverter | |
| export | function | |
| findrecord | function | |
| firstuniqueidx-field | ppcstype1field | |
| handle_ask | function | |

| Property | Type | Description |
|---|---|---|
| handleuniqueindexconflict | integer | |
| merge | boolean | |
| opentarget | function | |
| ppcs | ppcstype1 | |
| ppcstable | ppcstype1file | |
| setdataview | function | |
| settargettable | function | |
| type | type | |
| uniqueindexes | array | |
| writerecord | function | |

# Methods

## createtable()

### Description

### Prototype

*dbPPCS1Exportvar*.createtable ( dbPPCS1Export *me*, ppcstype1file *table*, tdataview *dataview* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Export | |
| table | None | ppcstype1file | |
| dataview | None | tdataview | |

## export()

### Description

### Prototype

*dbPPCS1Exportvar*.export ( dbPPCS1Export *me*, type(dbimportconverter) *imp*, integer *error*, integer *reccount* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Export | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| imp | None | type(dbimportconverter) | |
| error | None | integer | |
| reccount | `.inf` | integer | |

# opentarget()

## Description

## Prototype

*dbPPCS1Exportvar*.opentarget ( dbPPCS1Export *me*, string *datasourcename*, string *tablename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Export | |
| datasource-name | None | string | |
| tablename | None | string | |

# setdataview()

## Description

## Prototype

*dbPPCS1Exportvar*.setdataview ( dbPPCS1Export *me*, array *fieldlist*, string *name*, tdataview *dataview*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Export | |
| fieldlist | None | array | |
| name | None | string | |
| dataview | None | tdataview | |
| error | None | integer | |

# settargettable()

## Description

## Prototype

*dbPPCS1Exportvar*.settargettable ( dbPPCS1Export *me*, ppcstype1file *table* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Export | |
| table | None | ppcstype1file | |

## writerecord()

### Description

### Prototype

*dbPPCS1Exportvar*.writerecord ( dbPPCS1Export *me*, dbconvrecord *r* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Export | |
| r | None | dbconvrecord | |

# dbPPCS1Import

## Description

## Type Tags

dbconverter, dbimportconverter

## Object Value

Objects of type dbPPCS1Import have no value, and it is an error to try to get or set this value.

## dbPPCS1Import.new()

### Description

### Prototype

*dbPPCS1Import*.new ( dbPPCS1Import *me*, integer *localport*, integer *retry*, integer *timeout* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Import | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| localport | `.nul` | integer | |
| retry | `1000000` | integer | |
| timeout | `5000000` | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| currentrecord | ppcstype1record | |
| currentview | tdataview | |
| impconverter | dbimportconverter | |
| index | ppcstype1index | |
| opensource | function | |
| ppcs | ppcstype1 | |
| ppcstable | ppcstype1file | |
| readrecord | function | |
| recordcount | function | |
| retry | integer | |
| setdataview | function | |
| setindex | function | |
| setsource | function | |
| timeout | integer | |
| type | type | |

# Methods

## opensource()

### Description

### Prototype

*`dbPPCS1Importvar`*.`opensource` ( dbPPCS1Import *`me`*, string *`address`*, integer *`codepage`*, string *`tablename`*, string *`password`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Import | |
| address | None | string | |
| codepage | `850` | integer | |
| tablename | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| password | None | string | |

# readrecord()

## Description

## Prototype

*dbPPCS1Importvar*.readrecord ( dbPPCS1Import *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Import | |
| error | None | integer | |

# recordcount()

## Description

## Prototype

*dbPPCS1Importvar*.recordcount ( dbPPCS1Import *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Import | |

# setdataview()

## Description

## Prototype

*dbPPCS1Importvar*.setdataview ( dbPPCS1Import *me*, array *fieldlist*, string *name*, tdataview *dataview*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Import | |
| fieldlist | None | array | |
| name | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dataview | None | tdataview | |
| error | None | integer | |

## setindex()

### Description

### Prototype

*dbPPCS1Importvar*.setindex ( dbPPCS1Import *me*, string *indexname*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Import | |
| indexname | None | string | |
| error | None | integer | |

## setsource()

### Description

### Prototype

*dbPPCS1Importvar*.setsource ( dbPPCS1Import *me*, ppcstype1file *table* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbPPCS1Import | |
| table | None | ppcstype1file | |

# dbSBMEExport

# Description

# Type Tags

dbconverter, dbexportconverter

# Object Value

Objects of type dbSBMEExport have no value, and it is an error to try to get or set this value.

# dbSBMEExport.new()

## Description

## Prototype

*dbSBMEExport*.new ( dbSBMEExport *me*, string *openmode*, boolean *merge*, integer *commitinterval*, boolean *createsystables* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEExport | |
| openmode | OC | string | |
| merge | .false | boolean | |
| commitinterval | 1000 | integer | |
| createsystables | .true | boolean | |

# Properties

| Property | Type | Description |
|---|---|---|
| commitinterval | integer | |
| createsbmetable | function | |
| createsystables | boolean | |
| createtable | function | |
| currentview | tdataview | |
| expconverter | dbexportconverter | |
| export | function | |
| findrecord | function | |
| handle_ask | function | |
| handleuniqueindexconflict | integer | |
| merge | boolean | |
| opentarget | function | |
| recidfield | sbme1field | |
| sbme | sbme1 | |
| sbmeopenmode | string | |
| sbmetable | sbme1table | |
| setdataview | function | |

| Property | Type | Description |
|---|---|---|
| settargettable | function | |
| sysfields | sbme1table | |
| sysfieldsext | sbme1table | |
| systables | sbme1table | |
| type | type | |
| uniqueindexes | array | |
| writerecord | function | |

# Methods

## createsbmetable()

### Description

### Prototype

*dbSBMEExportvar*.createsbmetable ( dbSBMEExport *me*, dbconvtable *table* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEExport | |
| table | None | dbconvtable | |

## createtable()

### Description

### Prototype

*dbSBMEExportvar*.createtable ( dbSBMEExport *me*, sbme1table *table*, tdataview *dataview* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEExport | |
| table | None | sbme1table | |
| dataview | None | tdataview | |

## export()

### Description

### Prototype

*dbSBMEExportvar*.export ( dbSBMEExport *me*, type(dbimportconverter) *imp*, integer *error*, integer *reccount* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEExport | |
| imp | None | type(dbimportconverter) | |
| error | None | integer | |
| reccount | .inf | integer | |

## opentarget()

### Description

### Prototype

*dbSBMEExportvar*.opentarget ( dbSBMEExport *me*, string *filename*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEExport | |
| filename | None | string | |
| tablename | None | string | |

## setdataview()

### Description

### Prototype

*dbSBMEExportvar*.setdataview ( dbSBMEExport *me*, array *fieldlist*, string *name*, tdataview *dataview*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEExport | |
| fieldlist | None | array | |
| name | None | string | |
| dataview | None | tdataview | |
| error | None | integer | |

## settargettable()

### Description

### Prototype

*dbSBMEExportvar*.settargettable ( dbSBMEExport *me*, sbme1table *table* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEExport | |
| table | None | sbme1table | |

## writerecord()

### Description

### Prototype

*dbSBMEExportvar*.writerecord ( dbSBMEExport *me*, dbconvrecord *r* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEExport | |
| r | None | dbconvrecord | |

# dbSBMEImport

# Description

# Type Tags

dbconverter, dbimportconverter

# Object Value

Objects of type dbSBMEImport have no value, and it is an error to try to get or set this value.

# dbSBMEImport.new()

## Description

## Prototype

*dbSBMEImport*.new ( dbSBMEImport *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEImport | |

# Properties

| Property | Type | Description |
|---|---|---|
| currentrecord | sbme1record | |
| currentview | tdataview | |
| displayformats | __tdisplayformats | |
| impconverter | dbimportconverter | |
| index | sbme1index | |
| opensource | function | |
| readrecord | function | |
| recordcount | function | |
| sbme | sbme1 | |
| sbmetable | sbme1table | |
| setdataview | function | |
| setindex | function | |
| setsource | function | |
| type | type | |

# Methods

## opensource()

### Description

### Prototype

*dbSBMEImportvar*.opensource ( dbSBMEImport *me*, string *filename*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEImport | |
| filename | None | string | |
| tablename | None | string | |

# readrecord()

## Description

## Prototype

*dbSBMEImportvar*.readrecord ( dbSBMEImport *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEImport | |
| error | None | integer | |

# recordcount()

## Description

## Prototype

*dbSBMEImportvar*.recordcount ( dbSBMEImport *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEImport | |

# setdataview()

## Description

## Prototype

*dbSBMEImportvar*.setdataview ( dbSBMEImport *me*, array *fieldlist*, string *name*, tdataview *dataview*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEImport | |
| fieldlist | None | array | |
| name | None | string | |
| dataview | None | tdataview | |
| error | None | integer | |

# setindex()

## Description

### Prototype

*dbSBMEImportvar*.setindex ( dbSBMEImport *me*, string *indexname*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEImport | |
| indexname | None | string | |
| error | None | integer | |

## setsource()

### Description

### Prototype

*dbSBMEImportvar*.setsource ( dbSBMEImport *me*, sbme1table *table* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbSBMEImport | |
| table | None | sbme1table | |

# dbXMLConverter

# Description

# Type Tags

None

# Object Value

Objects of type dbXMLConverter have no value, and it is an error to try to get or set this value.

# dbXMLConverter.new()

## Description

## Prototype

*dbXMLConverter*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| charsize | integer | |
| lbo | boolean | |
| recordtag | string | |
| roottag | string | |
| tagnames | array | |
| type | type | |
| useutf8 | boolean | |

# dbXMLExport

## Description

## Type Tags

dbconverter, dbexportconverter

## Object Value

Objects of type dbXMLExport have no value, and it is an error to try to get or set this value.

## dbXMLExport.new()

### Description

### Prototype

*dbXMLExport*.new ( dbXMLExport *me*, string *roottag*, string *recordtag*, boolean *outputdeclaration*, boolean *useutf8*, boolean *outputbyteordermark*, boolean *uselbo* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLExport | |
| roottag | None | string | |
| recordtag | `row` | string | |
| outputdeclaration | `.true` | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| useutf8 | `.true` | boolean | |
| outputbyteor-dermark | `.false` | boolean | |
| uselbo | `.true` | boolean | |

# Properties

| Property | Type | Description |
|---|---|---|
| append | boolean | |
| createtable | function | |
| declaration | string | |
| expconverter | dbexportconverter | |
| export | function | |
| fileref | fsfileoutputstream | |
| headerwritten | boolean | |
| opentarget | function | |
| outputbyteor-dermark | boolean | |
| outputdeclara-tion | boolean | |
| props | dbXMLConverter | |
| rowsout | integer | |
| type | type | |
| uselbo | boolean | |
| writeheader | function | |
| writerecord | function | |

# Methods

## createtable()

### Description

### Prototype

*`dbXMLExportvar`*`.createtable` ( dbXMLExport *`me`*, dbconvtable *`table`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLExport | |
| table | None | dbconvtable | |

# export()

## Description

## Prototype

*dbXMLExportvar*.export ( dbXMLExport *me*, type(dbimportconverter) *imp*, integer *error*, integer *reccount* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLExport | |
| imp | None | type(dbimportconverter) | |
| error | None | integer | |
| reccount | .inf | integer | |

# opentarget()

## Description

## Prototype

*dbXMLExportvar*.opentarget ( dbXMLExport *me*, string *filename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLExport | |
| filename | None | string | |

# writeheader()

## Description

## Prototype

*dbXMLExportvar*.writeheader ( dbXMLExport *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLExport | |

# writerecord()

## Description

### Prototype

*dbXMLExportvar*.writerecord ( dbXMLExport *me*, dbconvrecord *r* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLExport | |
| r | None | dbconvrecord | |

# dbXMLImport

## Description

## Type Tags

dbconverter, dbimportconverter

## Object Value

Objects of type dbXMLImport have no value, and it is an error to try to get or set this value.

## dbXMLImport.new()

### Description

### Prototype

*dbXMLImport*.new ( dbXMLImport *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLImport | |

## Properties

| Property | Type | Description |
|---|---|---|
| buffer | string | |
| bufpos | integer | |
| fileref | fsfileinputstream | |
| impconverter | dbimportconverter | |
| opensource | function | |
| props | dbXMLConverter | |

| Property | Type | Description |
|---|---|---|
| readrecord | function | |
| recordcount | function | |
| reopensource | function | |
| rowsread | integer | |
| type | type | |

# Methods

## opensource()

### Description

### Prototype

*dbXMLImportvar*.opensource ( dbXMLImport *me*, string *filename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLImport | |
| filename | None | string | |

## readrecord()

### Description

### Prototype

*dbXMLImportvar*.readrecord ( dbXMLImport *me*, integer *error*, boolean *rawmode* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLImport | |
| error | None | integer | |
| rawmode | .false | boolean | |

## recordcount()

### Description

### Prototype

*dbXMLImportvar*.recordcount ( dbXMLImport *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbXMLImport | |

# dbconverter

## Description

## Type Tags

dbconverter

## Object Value

Objects of type dbconverter have no value, and it is an error to try to get or set this value.

## dbconverter.new()

### Description

### Prototype

*dbconverter*.new ( dbconverter *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconverter | |

## Properties

| Property | Type | Description |
|---|---|---|
| booleanfalse | string | |
| booleantrue | string | |
| datasource-name | string | |
| datelocale | SBLlocaledateinfo | |
| dbconvversion | string | |
| logerrors | boolean | |
| logfilename | string | |
| name | string | |
| notifyfrequen-cy | integer | |

| Property | Type | Description |
|---|---|---|
| numericlocale | SBLNumSettings | |
| onnotify | event | |
| table | dbconvtable | |
| type | type | |
| version | string | |
| view | dbconvtable | |

# dbconverterinfo

## Description

## Type Tags

None

## Object Value

Objects of type dbconverterinfo have no value, and it is an error to try to get or set this value.

## dbconverterinfo.new()

### Description

### Prototype

*dbconverterinfo*.new ( dbconverterinfo *me*, string *name*, boolean *import*, type *typeref* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconverterinfo | |
| name | None | string | |
| import | None | boolean | |
| typeref | None | type | |

## Properties

| Property | Type | Description |
|---|---|---|
| import | boolean | |
| name | string | |
| type | type | |
| typeref | type | |

# dbconvfield

## Description

## Type Tags

None

## Object Value

Objects of type dbconvfield have no value, and it is an error to try to get or set this value.

## dbconvfield.new()

### Description

### Prototype

*dbconvfield*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| calculation | string | |
| codepage | string | |
| comment | string | |
| datatype | type | |
| default | string | |
| fieldname | string | |
| format | string | |
| help | string | |
| index | integer | |
| indextype | string | |
| maxlen | integer | |
| next | dbconvfield | |
| read_only | boolean | |
| required | boolean | |
| shareable | boolean | |

| Property | Type | Description |
|---|---|---|
| sharename | string | |
| sharetype | string | |
| table | dbconvtable | |
| type | type | |
| user | type(*) | |
| validation | string | |

# dbconvrecord

## Description

## Type Tags

None

## Object Value

Objects of type dbconvrecord have no value, and it is an error to try to get or set this value.

## dbconvrecord.new()

### Description

### Prototype

*dbconvrecord*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| copyrecord | function | |
| data | array | |
| get | function | |
| getstring | function | |
| put | function | |
| putstring | function | |
| table | dbconvtable | |
| type | type | |

# Methods

## copyrecord()

### Description

### Prototype

*dbconvrecordvar*.copyrecord ( dbconvrecord *me*, dbconvrecord *r* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconvrecord | |
| r | None | dbconvrecord | |

## get()

### Description

### Prototype

*dbconvrecordvar*.get ( dbconvrecord *me*, dbconvfield *fld*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconvrecord | |
| fld | None | dbconvfield | |
| error | None | integer | |

## getstring()

### Description

### Prototype

*dbconvrecordvar*.getstring ( dbconvrecord *me*, dbconvfield *fld*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconvrecord | |
| fld | None | dbconvfield | |
| error | None | integer | |

## put()

### Description

### Prototype

*dbconvrecordvar*.put ( dbconvrecord *me*, dbconvfield *fld*, type(*) *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconvrecord | |
| fld | None | dbconvfield | |
| value | None | type(*) | |

## putstring()

### Description

### Prototype

*dbconvrecordvar*.putstring ( dbconvrecord *me*, dbconvfield *fld*, string *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconvrecord | |
| fld | None | dbconvfield | |
| value | None | string | |

# dbconvtable

# Description

# Type Tags

None

# Object Value

Objects of type dbconvtable have no value, and it is an error to try to get or set this value.

# dbconvtable.new()

## Description

## Prototype

*dbconvtable*.new ( dbconvtable *me*, type(dbconverter) *converter* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconvtable | |
| converter | None | type(dbconverter) | |

# Properties

| Property | Type | Description |
|---|---|---|
| addfield | function | |
| codepage | string | |
| converter | type(dbconverter) | |
| datasource-name | string | |
| fieldcount | integer | |
| firstfield | dbconvfield | |
| name | string | |
| newrecord | function | |
| type | type | |

# Methods

## addfield()

### Description

### Prototype

*dbconvtablevar*.addfield ( dbconvtable *me*, type *datatype*, string *fieldname*, string *format*, string *codepage*, integer *maxlen*, type(*) *user*, string *help*, string *comment*, string *calculation*, string *default*, string *validation*, boolean *read_only*, boolean *required*, boolean *shareable*, string *sharename*, string *sharetype* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconvtable | |
| datatype | None | type | |
| fieldname | None | string | |
| format | None | string | |
| codepage | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| maxlen | None | integer | |
| user | None | type(*) | |
| help | None | string | |
| comment | None | string | |
| calculation | None | string | |
| default | None | string | |
| validation | None | string | |
| read_only | None | boolean | |
| required | None | boolean | |
| shareable | None | boolean | |
| sharename | None | string | |
| sharetype | None | string | |

## newrecord()

### Description

### Prototype

*dbconvtablevar*.newrecord ( dbconvtable *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbconvtable | |

# dbexportconverter

# Description

# Type Tags

dbconverter, dbexportconverter

# Object Value

Objects of type dbexportconverter have no value, and it is an error to try to get or set this value.

# dbexportconverter.new()

## Description

## Prototype

*dbexportconverter*.new ( dbexportconverter *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbexportconverter | |

# Properties

| Property | Type | Description |
|---|---|---|
| cvtr | dbconverter | |
| export | function | |
| opentarget | function | |
| type | type | |
| writerecord | function | |

# Methods

## export()

### Description

### Prototype

*dbexportconvertervar*.export ( dbexportconverter *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbexportconverter | |

## opentarget()

### Description

### Prototype

*dbexportconvertervar*.opentarget ( dbexportconverter *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbexportconverter | |

## writerecord()

### Description

### Prototype

*dbexportconvertervar*.writerecord ( dbexportconverter *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbexportconverter | |

# dbimportconverter

# Description

# Type Tags

dbconverter, dbimportconverter

# Object Value

Objects of type dbimportconverter have no value, and it is an error to try to get or set this value.

# dbimportconverter.new()

## Description

## Prototype

*dbimportconverter*.new ( dbimportconverter *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbimportconverter | |

# Properties

| Property | Type | Description |
|---|---|---|
| bytesprocessed | integer | |
| cvtr | dbconverter | |
| filesize | integer | |

| Property | Type | Description |
|---|---|---|
| import | function | |
| opendatasource | function | |
| readrecord | function | |
| type | type | |
| usereccount | boolean | |

# Methods

## import()

### Description

### Prototype

*dbimportconvertervar*.`import` ( dbimportconverter *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbimportconverter | |

## opendatasource()

### Description

### Prototype

*dbimportconvertervar*.`opendatasource` ( dbimportconverter *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbimportconverter | |

## readrecord()

### Description

### Prototype

*dbimportconvertervar*.`readrecord` ( dbimportconverter *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dbimportconverter | |

# convert8bitchar()

## Description

## Prototype

`convert8bitchar` ( string *`s`*, string *`codepage`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| codepage | None | string | |

# enumdbconverter()

## Description

## Prototype

`enumdbconverter` ()

## Parameters

None

# Chapter 30. displayformat

## getbooleanformat()

### Description

### Prototype

`getbooleanformat` ( type(wxdialogparent) *`parent`*, string *`currformat`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| currformat | None | string | |
| error | None | integer | |

## getdateformat()

### Description

### Prototype

`getdateformat` ( type(wxdialogparent) *`parent`*, string *`currformat`*, localeinfo *`locale`*, SBLlocaledateinfo *`SBLLocale`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| currformat | `yyyymm0d` | string | |
| locale | None | localeinfo | |
| SBLLocale | None | SBLlocaledateinfo | |
| error | None | integer | |

## getdatetimeformat()

### Description

### Prototype

`getdatetimeformat` ( type(wxdialogparent) *`parent`*, string *`currformat`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| currformat | None | string | |
| error | None | integer | |

# getnumformat()

## Description

This function displays the dialog to allow the user to select the number format. The result is returned as a string.

## Prototype

`getnumformat` ( type(wxdialogparent) *`parent`*, string *`currformat`*, boolean *`isinteger`*, locale-info *`locale`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| currformat | None | string | |
| isinteger | `.false` | boolean | |
| locale | None | localeinfo | |
| error | None | integer | |

# getsharetypefromdatatype()

## Description

## Prototype

`getsharetypefromdatatype` ( type *`datatype`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datatype | None | type | |

# getstringformat()

## Description

## Prototype

`getstringformat` ( type(wxdialogparent) *`parent`*, string *`currformat`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| currformat | None | string | |
| error | None | integer | |

# gettimeformat()

## Description

## Prototype

`gettimeformat` ( type(wxdialogparent) *`parent`*, string *`currformat`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| currformat | None | string | |
| error | None | integer | |

# validatedisplayformat()

## Description

## Prototype

`validatedisplayformat` ( string *`sbldisplayformat`*, string *`sharetype`*, type *`datatype`*, localeinfo *`locale`*, displayformatinfo *`dispinfo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbldisplayfor-mat | None | string | |
| sharetype | None | string | |
| datatype | None | type | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| locale | None | localeinfo | |
| dispinfo | None | displayformatinfo | |

# Chapter 31. drilldown

## __ts_filter

### Description

## Type Tags

None

## Object Value

Objects of type __ts_filter have no value, and it is an error to try to get or set this value.

## __ts_filter.new()

### Description

### Prototype

__*ts_filter*.new ( __ts_filter *me*, tablesearchinfo *tsinfo*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | __ts_filter | |
| tsinfo | None | tablesearchinfo | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| checkfilter | function | |
| currpos | integer | |
| filter | sqlq1 | |
| filterapplied | boolean | |
| rset | array | |
| selectcurrent | function | |
| selectfirst | function | |
| selectlast | function | |
| selectnext | function | |

| Property | Type | Description |
|---|---|---|
| selectprevious | function | |
| setfilter | function | |
| tsinfo | tablesearchinfo | |
| type | type | |

# Methods

## checkfilter()

### Description

### Prototype

*__ts_filtervar*.checkfilter ( __ts_filter *me*, string *errortext*, integer *errorindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | __ts_filter | |
| errortext | None | string | |
| errorindex | None | integer | |

## selectcurrent()

### Description

### Prototype

*__ts_filtervar*.selectcurrent ( __ts_filter *me*, string *errortext*, integer *errorindex*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | __ts_filter | |
| errortext | None | string | |
| errorindex | None | integer | |
| error | None | integer | |

## selectfirst()

### Description

### Prototype

*__ts_filtervar*.selectfirst ( __ts_filter *me*, string *errortext*, integer *errorindex*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | __ts_filter | |
| errortext | None | string | |
| errorindex | None | integer | |
| error | None | integer | |

## selectlast()

### Description

### Prototype

*__ts_filtervar*.selectlast ( __ts_filter *me*, string *errortext*, integer *errorindex*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | __ts_filter | |
| errortext | None | string | |
| errorindex | None | integer | |
| error | None | integer | |

## selectnext()

### Description

### Prototype

*__ts_filtervar*.selectnext ( __ts_filter *me*, string *errortext*, integer *errorindex*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | __ts_filter | |
| errortext | None | string | |
| errorindex | None | integer | |
| error | None | integer | |

## selectprevious()

### Description

### Prototype

__*ts_filtervar*.selectprevious ( __ts_filter *me*, string *errortext*, integer *errorindex*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | __ts_filter | |
| errortext | None | string | |
| errorindex | None | integer | |
| error | None | integer | |

## setfilter()

### Description

### Prototype

__*ts_filtervar*.setfilter ( __ts_filter *me*, string *filter*, string *errortext*, integer *errorindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | __ts_filter | |
| filter | None | string | |
| errortext | None | string | |
| errorindex | None | integer | |

# drilldown()

# Description

# Prototype

drilldown ( type(wxdialogparent) *parent*, integer *width*, integer *height*, type(db1index) *searchidx*, integer *maxentries*, integer *minactivationchars*, string *dlgtitle*, string *searchcap*, string *OKtext*, string *canceltext*, string *readytext*, string *workingtext*, array *dispflds*, array *colwidths*, string *defboolean*, string *definteger*, string *defnumber*, string *defdate*, string *deftime*, string *defdatetime*, SBLlocaledateinfo *datelocale*, SBLNumSettings *numlocale*, boolean *forceupper*, boolean *forcelower*, boolean *caseinsensitive*, string *startvalue*, function *setfilterhandler*, type(*) *reference*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| width | None | integer | |
| height | None | integer | |
| searchidx | None | type(db1index) | |
| maxentries | None | integer | |
| minactivation-chars | `2` | integer | |
| dlgtitle | None | string | |
| searchcap | None | string | |
| OKtext | None | string | |
| canceltext | None | string | |
| readytext | None | string | |
| workingtext | None | string | |
| dispflds | None | array | |
| colwidths | None | array | |
| defboolean | `T\|F` | string | |
| definteger | `.` | string | |
| defnumber | `999999.00` | string | |
| defdate | `yyyy.0m.0d` | string | |
| deftime | `hh:mm:ss` | string | |
| defdatetime | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| forceupper | `.false` | boolean | |
| forcelower | `.false` | boolean | |
| caseinsensitive | `.false` | boolean | |
| startvalue | None | string | |
| setfilterhandler | None | function | |
| reference | None | type(*) | |
| error | None | integer | |

# tablesearch()

## Description

This function provides a pick list functionality implemented using a dialog and a grid control. Once the user selects a row from the grid and clicks on OK, then the *`valuefield`* of the selected record will be

returned to the caller as the function return value. It also includes an optional *whereclause* parameter, though it may only be applied to the database table associated with the *valuefield* parameter. The *dispfields* contains the field names for the fields that should be dislayed and are listed by name separated by semi-colons. The column widths can be specified in the *colwidths* parameter as a semi-colon-separated list. They will be applied in the order received, just like the display fields. To make it easier to establish the best column widths, set the *showcolwidths* parameter to `.true` and the column widths will be displayed. This is very useful during development.

# Prototype

`tablesearch` ( type(wxdialogparent) *parent*, integer *width*, integer *height*, type(db1field) *valuefield*, type(db1index) *orderidx*, string *whereclause*, string *dlgtitle*, string *searchcap*, string *OKtext*, string *canceltext*, string *readytext*, string *workingtext*, string *dispflds*, string *colwidths*, string *defboolean*, string *definteger*, string *defnumber*, string *defdate*, string *deftime*, string *defdatetime*, SBLlocaledateinfo *datelocale*, SBLNumSettings *numlocale*, boolean *showcolwidths*, boolean *allowcolwidthdrag*, integer *error* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| width | None | integer | |
| height | None | integer | |
| valuefield | None | type(db1field) | |
| orderidx | None | type(db1index) | |
| whereclause | None | string | |
| dlgtitle | None | string | |
| searchcap | None | string | |
| OKtext | None | string | |
| canceltext | None | string | |
| readytext | `Ready.` | string | |
| workingtext | `Working...` | string | |
| dispflds | None | string | |
| colwidths | None | string | |
| defboolean | `T\|F` | string | |
| definteger | `.` | string | |
| defnumber | `999999.00` | string | |
| defdate | `yyyy.0m.0d` | string | |
| deftime | `hh:mm:ss` | string | |
| defdatetime | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| showcolwidths | `.false` | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| allowcolwidth-drag | `.true` | boolean | |
| error | None | integer | |

# Chapter 32. dxflib

## convertdxf()

### Description

### Prototype

`convertdxf` ( string *`dllhelperfile`*, string *`dxffilename`*, string *`bmpfilename`*, integer *`width`*, integer *`height`*, string *`errtext`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dllhelperfile | None | string | |
| dxffilename | None | string | |
| bmpfilename | None | string | |
| width | None | integer | |
| height | None | integer | |
| errtext | None | string | |
| error | None | integer | |

## createblankbmp()

### Description

This creates a blank bitmap as a blob in the *`width`* and *`height`* requested with the *`color`* specified (or the default color if not specified). If the *`width`* or *`height`* is not supplied, then an error will be returned in the *`error`* parameter and the function will return `.nul`.

### Prototype

`createblankbmp` ( integer *`width`*, integer *`height`*, integer *`color`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | None | integer | |
| height | None | integer | |
| color | `10526880` | integer | |
| error | None | integer | |

# Chapter 33. errormsgs_en

This library contains all the error messages associated with the error values from SIMPOL and Superbase, as can be found from the errors.sma include file.

# geterrormessage()

## Description

## Prototype

`geterrormessage` ( integer *`errornumber`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| errornumber | `0` | integer | |

# Chapter 34. fastset

This library implements the fastset object. It uses red-black binary trees to provide a set implementation that can be used in place of the internal set type if the functionality required (string sorting of complex objects) is required.

# fastset

## Description

## Type Tags

None

## Object Value

Objects of type fastset have no value, and it is an error to try to get or set this value.

# fastset.new()

### Description

### Prototype

*fastset*.new ( fastset *me*, function *compare* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| compare | None | function | |

## Properties

| Property | Type | Description |
|---|---|---|
| _private | fastsetprivate | |
| add | function | |
| addelement | function | |
| clear | function | |
| compare | function | |
| count | integer | |
| delete | function | |
| deleteelement | function | |
| difference | function | |

| Property | Type | Description |
|---|---|---|
| distinct | boolean | |
| find | function | |
| findelement-bykey | function | |
| getcount | function | |
| getfirst | function | |
| getlast | function | |
| getmaxdepth | function | |
| insert | function | |
| intersect | function | |
| setdistinct | function | |
| totalcount | integer | |
| type | type | |
| unite | function | |

# Methods

## add()

### Description

### Prototype

*fastsetvar*.add ( fastset *me*, anyvalue *key*, type(*) *data* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| key | None | anyvalue | |
| data | None | type(*) | |

## addelement()

### Description

### Prototype

*fastsetvar*.addelement ( fastset *me*, anyvalue *key*, type(*) *data* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| key | None | anyvalue | |
| data | None | type(*) | |

## clear()

### Description

### Prototype

*fastsetvar*.clear ( fastset *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |

## delete()

### Description

### Prototype

*fastsetvar*.delete ( fastset *me*, fastsetnode *node* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| node | None | fastsetnode | |

## deleteelement()

### Description

### Prototype

*fastsetvar*.deleteelement ( fastset *me*, fastsetnode *node* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| node | None | fastsetnode | |

## difference()

### Description

### Prototype

*fastsetvar*.difference ( fastset *me*, fastset *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| t | None | fastset | |

## find()

### Description

### Prototype

*fastsetvar*.find ( fastset *me*, anyvalue *key* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| key | None | anyvalue | |

## findelementbykey()

### Description

### Prototype

*fastsetvar*.findelementbykey ( fastset *me*, anyvalue *key* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| key | None | anyvalue | |

## getcount()

### Description

### Prototype

*fastsetvar*.getcount ( fastset *me*, anyvalue *key* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| key | None | anyvalue | |

# getfirst()

## Description

## Prototype

*fastsetvar*.getfirst ( fastset *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |

# getlast()

## Description

## Prototype

*fastsetvar*.getlast ( fastset *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |

# getmaxdepth()

## Description

## Prototype

*fastsetvar*.getmaxdepth ( fastset *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |

# insert()

## Description

## Prototype

*fastsetvar*.insert ( fastset *me*, anyvalue *key*, type(*) *data* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| key | None | anyvalue | |
| data | None | type(*) | |

## intersect()

### Description

### Prototype

*fastsetvar*.intersect ( fastset *me*, fastset *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| t | None | fastset | |

## setdistinct()

### Description

### Prototype

*fastsetvar*.setdistinct ( fastset *me*, boolean *distinct* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| distinct | None | boolean | |

## unite()

### Description

### Prototype

*fastsetvar*.unite ( fastset *me*, fastset *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastset | |
| t | None | fastset | |

# fastsetnode

## Description

## Type Tags

None

## Object Value

Objects of type fastsetnode have no value, and it is an error to try to get or set this value.

## fastsetnode.new()

### Description

### Prototype

*fastsetnode*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _private | fastsetnodeprivate | |
| getdata | function | |
| getkey | function | |
| getnext | function | |
| getprev | function | |
| type | type | |

## Methods

### getdata()

#### Description

#### Prototype

*fastsetnodevar*.getdata ( fastsetnode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastsetnode | |

## getkey()

### Description

### Prototype

*fastsetnodevar*.getkey ( fastsetnode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastsetnode | |

## getnext()

### Description

### Prototype

*fastsetnodevar*.getnext ( fastsetnode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastsetnode | |

## getprev()

### Description

### Prototype

*fastsetnodevar*.getprev ( fastsetnode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | fastsetnode | |

# Chapter 35. filesyslib

This library is provided to add functionality in a cross-platform way for working with file systems. It provides functions like getcurrentdirectory(), getdirectorysepchar(), filenameparse(), fileexists(), deletefile(), and others.

# copyfile()

## Description

## Prototype

`copyfile` ( string *`fromfile`*, string *`tofile`*, boolean *`replace`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| fromfile | None | string | |
| tofile | None | string | |
| replace | `.false` | boolean | |
| error | None | integer | |

# createdirectory()

## Description

## Prototype

`createdirectory` ( string *`path`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| path | None | string | |
| error | None | integer | |

# createpath()

## Description

## Prototype

`createpath` ( string *`path`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| path | None | string | |
| error | None | integer | |

# deletefile()

## Description

This function will delete the file name passed from the storage media. If it fails it will return an error in the *`error`* parameter.

## Prototype

`deletefile` ( string *`filename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| error | None | integer | |

# deletefileold()

## Description

## Prototype

`deletefileold` ( string *`filename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| error | None | integer | |

# fileexists()

## Description

This returns `.true` if it can find the file name on the storage media, otherwise it returns `.false`. Case-sensitive on case-sensitive file systems.

## Prototype

`fileexists` ( string *`filename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| error | None | integer | |

# fileexists_old()

## Description

## Prototype

`fileexists_old` ( string *`filename`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |

# filenameparse()

## Description

This parses the file name passed in *`sSrc`* and splits it into the path (including trailing slash), the root file name, and the file extension (including the preceding dot separator).

## Prototype

`filenameparse` ( string *`sSrc`*, string *`sPath`*, string *`sRoot`*, string *`sExt`*, string *`sSep`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSrc | None | string | |
| sPath | None | string | |
| sRoot | None | string | |
| sExt | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSep | None | string | |

# getcurrentdirectory()

## Description

Returns the current working directory. There is currently no facility in SIMPOL to change the current working directory.

## Prototype

```
getcurrentdirectory ()
```

## Parameters

None

# getdefaultnewline()

## Description

## Prototype

```
getdefaultnewline ()
```

## Parameters

None

# getdirectorysepchar()

## Description

This returns the correct directory separator character for the current file system. Typically this is the back-slash on Windows and the forward slash on Linux, Unix, and OS-X.

## Prototype

```
getdirectorysepchar ()
```

## Parameters

None

# getenvironmentvariable()

## Description

## Prototype

`getenvironmentvariable` ( string *`name`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | |

# getpublicdatadir()

## Description

## Prototype

`getpublicdatadir` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# gettempfilename()

## Description

Returns a viable temporary file name in the appropriate location on the current operating system. It uses the prefix as the beginning of the file name. The prefix should be three characters in length. On Windows it uses the operating system function for this feature.

## Prototype

`gettempfilename` ( string *`path`*, string *`prefix`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| path | None | string | |
| prefix | None | string | |
| error | None | integer | |

# gettemppath()

## Description

This function returns the appropriate temporary storage (TEMP) directory for the operating system currently running and the user currently logged in. In practice this is the user-specific TEMP directory in Windows and the `/tmp` directory on Linux.

## Prototype

`gettemppath` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# getu32frominteger()

## Description

## Prototype

`getu32frominteger` ( integer *`v`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| v | None | integer | |

# getuserhomedir()

## Description

## Prototype

`getuserhomedir` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# getwindowssysdir()

## Description

## Prototype

`getwindowssysdir` ( integer *`pathid`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| pathid | None | integer | |
| error | None | integer | |

# issamefilename()

## Description

Checks if two file names are the same. If they are identical, returns `.true` immediately. Otherwise, it creates two UTOSdirectoryentry objects, one for each and if their file names are the same, then returns `.true` otherwise it returns `.false`.

## Prototype

`issamefilename` ( string *`filename1`*, string *`filename2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename1 | None | string | |
| filename2 | None | string | |

# iswindows_os()

## Description

## Prototype

`iswindows_os` ()

## Parameters

None

# notrailingdirsep()

## Description

## Prototype

`notrailingdirsep` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# setcurrentdirectory()

## Description

## Prototype

`setcurrentdirectory` ( string *`path`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| path | None | string | |

# trailingdirsep()

## Description

## Prototype

`trailingdirsep` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# Chapter 36. filtergui

## filterGUIInfo

### Description

### Type Tags

None

### Object Value

Objects of type filterGUIInfo have no value, and it is an error to try to get or set this value.

## filterGUIInfo.new()

### Description

### Prototype

*filterGUIInfo*.new ( filterGUIInfo *me*, tdataview *dataview*, wxformgrid *grid*, wxdialog *d*, type(db1field) *sortfield*, type(db1field) *internalrecid*, string *filter*, function *onselectthandler*, type(*) *onselectreference*, function *onclosehandler*, type(*) *onclosereference*, SBLlocaledateinfo *datelocale*, SBLNumSettings *numlocale*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | filterGUIInfo | |
| dataview | None | tdataview | |
| grid | None | wxformgrid | |
| d | None | wxdialog | |
| sortfield | None | type(db1field) | |
| internalrecid | None | type(db1field) | |
| filter | None | string | |
| onselectthandler | None | function | |
| onselectreference | None | type(*) | |
| onclosehandler | None | function | |
| onclosereference | None | type(*) | |
| datelocale | None | SBLlocaledateinfo | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| numlocale | None | SBLNumSettings | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| closed | boolean | |
| dataview | tdataview | |
| datelocale | SBLlocaledateinfo | |
| dialog | wxdialog | |
| filter | string | |
| grid | wxformgrid | |
| internalrecid | type(db1field) | |
| lastselected | string | |
| lastsortval | string | |
| mutex | lock1 | |
| numlocale | SBLNumSettings | |
| onclose | event | |
| onselect | event | |
| results | AVLTree | |
| sortfield | type(db1field) | |
| type | type | |
| updategrid | function | |

# Methods

## updategrid()

### Description

### Prototype

*filterGUIInfovar*.updategrid ( filterGUIInfo *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | filterGUIInfo | |

# filterGUI()

# Description

# Prototype

`filterGUI` ( tdataview *`dataview`*, array *`colwidths`*, string *`filter`*, type(db1field) *`sortfield`*, type(wxdialogparent) *`parent`*, function *`onselecthandler`*, type(*) *`onselectreference`*, function *`onclosehandler`*, type(*) *`onclosereference`*, string *`defboolean`*, string *`definteger`*, string *`defnumber`*, string *`defdate`*, string *`deftime`*, string *`defdatetime`*, SBLlocaledateinfo *`datelocale`*, SBLNumSettings *`numlocale`*, integer *`left`*, integer *`top`*, integer *`width`*, integer *`height`*, string *`sortbytext`*, string *`closetext`*, string *`filtertext`*, string *`caption`*, boolean *`alloworderchange`*, wxfont *`font`*, integer *`error`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dataview | None | tdataview | |
| colwidths | None | array | |
| filter | None | string | |
| sortfield | None | type(db1field) | |
| parent | None | type(wxdialogparent) | |
| onselec-thandler | None | function | |
| onselectrefer-ence | None | type(*) | |
| onclosehandler | None | function | |
| oncloserefer-ence | None | type(*) | |
| defboolean | `T\|F` | string | |
| definteger | `.` | string | |
| defnumber | `.` | string | |
| defdate | `yyyy0m0d` | string | |
| deftime | `hh:mm:ss` | string | |
| defdatetime | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| left | `40` | integer | |
| top | `40` | integer | |
| width | `350` | integer | |
| height | `500` | integer | |
| sortbytext | `Sort by` | string | |
| closetext | `Close` | string | |
| filtertext | `Filter` | string | |
| caption | `Filter re-sults` | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| alloworder-change | `.false` | boolean | |
| font | None | wxfont | |
| error | None | integer | |

# Chapter 37. formlib

The formlib library includes the databaseforms library and provides the functionality for loading and saving dataform1 forms, as well as saving as source code.

# __formlib_getsource()

## Description

## Prototype

`__formlib_getsource` ( type(dataform1) *`form`*, string *`sourcetype`*, string *`sourcename`*, string *`sourcedirectory`*, string *`username`*, integer *`codepage`*, ppcstype1 *`ppcs`*, string *`replacementsourcetype`*, string *`replacementsource`*, string *`tablename`*, type(wxdialogparent) *`parentwindow`*, array *`messages`*, string *`msgtitle`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| form | None | type(dataform1) | |
| sourcetype | None | string | |
| sourcename | None | string | |
| sourcedirectory | None | string | |
| username | None | string | |
| codepage | None | integer | |
| ppcs | None | ppcstype1 | |
| replacementsourcetype | None | string | |
| replacementsource | None | string | |
| tablename | None | string | |
| parentwindow | None | type(wxdialogparent) | |
| messages | None | array | |
| msgtitle | `Form open error` | string | |

# convertwxformtodataform1()

## Description

## Prototype

`convertwxformtodataform1` ( wxform *`form`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| form | None | wxform | |

# dataform1mergeforms()

## Description

## Prototype

`dataform1mergeforms` ( dataform1 *`mainform`*, dataform1 *`newformcontent`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| mainform | None | dataform1 | |
| newformcon-tent | None | dataform1 | |
| error | None | integer | |

# datasourceinuse()

## Description

## Prototype

`datasourceinuse` ( dataform1datasource *`src`*, dataform1 *`f`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| src | None | dataform1datasource | |
| f | None | dataform1 | |

# datasourceinuse_ca()

## Description

Checks to see if the data source passed is in use in the dataform1 object passed. A further restriction is that the control must also be present in the array of controls that was passed. Returns either `.true` or `.false`.

## Prototype

`datasourceinuse_ca` ( dataform1datasource *`src`*, dataform1 *`f`*, array *`controls`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| src | None | dataform1datasource | |
| f | None | dataform1 | |
| controls | None | array | |

# datasourceinusepf()

## Description

## Prototype

`datasourceinusepf` ( dataform1datasource *`src`*, printform1 *`f`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| src | None | dataform1datasource | |
| f | None | printform1 | |

# datasourceinusepf_ca()

## Description

## Prototype

`datasourceinusepf_ca` ( dataform1datasource *`src`*, printform1 *`f`*, array *`controls`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| src | None | dataform1datasource | |
| f | None | printform1 | |
| controls | None | array | |

# getlastcontrolid()

## Description

## Prototype

`getlastcontrolid` ( dataform1 *`f`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | dataform1 | |

# getlastprintcontrolid()

## Description

## Prototype

`getlastprintcontrolid` ( printform1 *`f`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | printform1 | |

# getnewcontrolname()

## Description

## Prototype

`getnewcontrolname` ( type(*) *`c`*, integer *`i`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| c | None | type(*) | |
| i | None | integer | |

# getsyscolornames()

## Description

## Prototype

`getsyscolornames` ()

## Parameters

None

# getsystemcoloridfromstring()

## Description

## Prototype

`getsystemcoloridfromstring` ( string *`colorname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| colorname | None | string | |

# opendataform1()

## Description

## Prototype

`opendataform1` ( string *`filename`*, integer *`defpageheight`*, integer *`defpagewidth`*, integer *`defpagebackcolor`*, string *`defbooleanformat`*, string *`defintegerformat`*, string *`defnumberformat`*, string *`defdateformat`*, string *`deftimeformat`*, string *`defdatetimeformat`*, SBLlocaledateinfo *`datelocale`*, SBLNumSettings *`numericlocale`*, dring *`datasources`*, array *`tables`*, ppcstype1 *`ppcs`*, integer *`error`*, string *`errortext`*, type(wxcontainer) *`window`*, type(wxdialogparent) *`parentwindow`*, array *`messages`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| defpageheight | `50` | integer | |
| defpagewidth | `50` | integer | |
| defpageback-color | `16777215` | integer | |
| defbooleanfor-mat | `T\|F` | string | |
| defintegerfor-mat | `.` | string | |
| defnumberfor-mat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimefor-mat | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datelocale | None | SBLlocaledateinfo | |
| numericlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| error | None | integer | |
| errortext | None | string | |
| window | None | type(wxcontainer) | |
| parentwindow | None | type(wxdialogparent) | |
| messages | None | array | |

# opendataform1fromstring()

## Description

## Prototype

`opendataform1fromstring` ( string *`formcontent`*, integer *`defpageheight`*, integer *`defpagewidth`*, integer *`defpagebackcolor`*, string *`defbooleanformat`*, string *`defintegerformat`*, string *`defnumberformat`*, string *`defdateformat`*, string *`deftimeformat`*, string *`defdatetimeformat`*, SBLlocaledateinfo *`datelocale`*, SBLNumSettings *`numericlocale`*, dring *`datasources`*, array *`tables`*, ppcstype1 *`ppcs`*, integer *`error`*, string *`errortext`*, wxwindow *`window`*, type(wxdialogparent) *`parentwindow`*, array *`messages`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| formcontent | None | string | |
| defpageheight | `50` | integer | |
| defpagewidth | `50` | integer | |
| defpageback-color | `16777215` | integer | |
| defbooleanfor-mat | `T\|F` | string | |
| defintegerfor-mat | `.` | string | |
| defnumberfor-mat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimefor-mat | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datelocale | None | SBLlocaledateinfo | |
| numericlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| error | None | integer | |
| errortext | None | string | |
| window | None | wxwindow | |
| parentwindow | None | type(wxdialogparent) | |
| messages | None | array | |

# openprintform1()

## Description

## Prototype

`openprintform1` ( string *`filename`*, integer *`defpagewidth`*, integer *`defpageheight`*, integer *`defpagebackcolor`*, string *`defbooleanformat`*, string *`defintegerformat`*, string *`defnumberformat`*, string *`defdateformat`*, string *`deftimeformat`*, string *`defdatetimeformat`*, SBLlocaledateinfo *`datelocale`*, SBLNumSettings *`numericlocale`*, dring *`datasources`*, array *`tables`*, ppcstype1 *`ppcs`*, boolean *`createdisplayform`*, string *`printpreviewtitle`*, integer *`error`*, string *`errortext`*, wxwindow *`window`*, type(wxdialogparent) *`parentwindow`*, array *`messages`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| defpagewidth | `210000` | integer | |
| defpageheight | `297000` | integer | |
| defpagebackcolor | `16777215` | integer | |
| defbooleanformat | `T\|F` | string | |
| defintegerformat | `.` | string | |
| defnumberformat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defdatetimeformat | None | string | |
| datelocale | None | SBLlocaledateinfo | |
| numericlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| createdisplayform | `.false` | boolean | |
| printpreviewtitle | None | string | |
| error | None | integer | |
| errortext | None | string | |
| window | None | wxwindow | |
| parentwindow | None | type(wxdialogparent) | |
| messages | None | array | |

# openprintform1fromstring()

## Description

## Prototype

`openprintform1fromstring` ( string *`formcontent`*, integer *`defpagewidth`*, integer *`defpageheight`*, integer *`defpagebackcolor`*, string *`defbooleanformat`*, string *`definteger format`*, string *`defnumberformat`*, string *`defdateformat`*, string *`deftimeformat`*, string *`defdatetimeformat`*, SBLlocaledateinfo *`datelocale`*, SBLNumSettings *`numericlocale`*, dring *`datasources`*, array *`tables`*, ppcstype1 *`ppcs`*, boolean *`createdisplayform`*, string *`printpreviewtitle`*, integer *`error`*, string *`errortext`*, wxwindow *`window`*, type(wxdialogparent) *`parentwindow`*, array *`messages`*, boolean *`dovalidation`*, string *`sourcedirectory`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| formcontent | None | string | |
| defpagewidth | `210000` | integer | |
| defpageheight | `297000` | integer | |
| defpagebackcolor | `16777215` | integer | |
| defbooleanformat | `T\|F` | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defintegerformat | `.` | string | |
| defnumberformat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimeformat | None | string | |
| datelocale | None | SBLlocaledateinfo | |
| numericlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| createdisplayform | `.false` | boolean | |
| printpreviewtitle | None | string | |
| error | None | integer | |
| errortext | None | string | |
| window | None | wxwindow | |
| parentwindow | None | type(wxdialogparent) | |
| messages | None | array | |
| dovalidation | `.true` | boolean | |
| sourcedirectory | None | string | |

# outputprintformcontent()

## Description

## Prototype

`outputprintformcontent` ( printform1 *`f`*, integer *`tabsize`*, integer *`tablevel`*, string *`origfilename`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | printform1 | |
| tabsize | `2` | integer | |
| tablevel | `0` | integer | |
| origfilename | None | string | |

# printform1mergeforms()

## Description

## Prototype

`printform1mergeforms` ( printform1 *`mainform`*, printform1 *`newformcontent`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| mainform | None | printform1 | |
| newformcon-tent | None | printform1 | |
| error | None | integer | |

# savedataform1()

## Description

## Prototype

`savedataform1` ( dataform1 *`f`*, string *`filename`*, integer *`tabsize`*, string *`origfilename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | dataform1 | |
| filename | None | string | |
| tabsize | 2 | integer | |
| origfilename | None | string | |
| error | None | integer | |

# savedataform1ctrlsasstring()

## Description

This function is used to format the form controls as text for placing on the clipboard and is used as part of the copy functionality from the Form Designer.

## Prototype

savedataform1ctrlsasstring ( dataform1 *f*, array *controls*, integer *tabsize*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | dataform1 | |
| controls | None | array | |
| tabsize | 2 | integer | |
| error | None | integer | |

# savedf1program()

## Description

## Prototype

savedf1program ( dataform1 *f*, string *filename*, string *funcname*, integer *tabsize*, boolean *append*, boolean *strippaths*, boolean *commentuisyshdr*, integer *charsize*, boolean *lbo*, boolean *appframeworkstyle* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | dataform1 | |
| filename | None | string | |
| funcname | foo | string | |
| tabsize | 2 | integer | |
| append | .false | boolean | |
| strippaths | .false | boolean | |
| commentu-isyshdr | .false | boolean | |
| charsize | 1 | integer | |
| lbo | .true | boolean | |
| appframework-style | .true | boolean | |

# saveprintform1()

## Description

## Prototype

`saveprintform1` ( printform1 *`f`*, string *`filename`*, integer *`tabsize`*, string *`origfilename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | printform1 | |
| filename | None | string | |
| tabsize | 2 | integer | |
| origfilename | None | string | |
| error | None | integer | |

# saveprintform1ctrlsasstring()

## Description

## Prototype

`saveprintform1ctrlsasstring` ( printform1 *`f`*, array *`controls`*, integer *`tabsize`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | printform1 | |
| controls | None | array | |
| tabsize | 2 | integer | |
| error | None | integer | |

# savewxformprogram()

## Description

## Prototype

`savewxformprogram` ( wxform *`f`*, string *`filename`*, string *`funcname`*, integer *`tabsize`*, boolean *`append`*, boolean *`strippaths`*, boolean *`commentuisyshdr`*, integer *`charsize`*, boolean *`lbo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | wxform | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| funcname | `foo` | string | |
| tabsize | `2` | integer | |
| append | `.false` | boolean | |
| strippaths | `.false` | boolean | |
| commentu-isyshdr | `.false` | boolean | |
| charsize | `1` | integer | |
| lbo | `.true` | boolean | |

# tableinuse()

## Description

## Prototype

`tableinuse` ( dataform1table *`table`*, dataform1 *`f`*, boolean *`excludesubunits`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | dataform1table | |
| f | None | dataform1 | |
| excludesub-units | `.false` | boolean | |

# tableinuse_ca()

## Description

Checks to see if the dataform1table object passed is in use in the dataform1 object passed. A further restriction is that the table must also be used in one of the controls present in the array of controls that was passed. If *`excludesubunits`* is `.false` it will also check the columns of a dataform1datagrid if present. Returns either `.true` or `.false`.

## Prototype

`tableinuse_ca` ( dataform1table *`table`*, dataform1 *`f`*, array *`controls`*, boolean *`excludesub-units`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | dataform1table | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | dataform1 | |
| controls | None | array | |
| excludesub-units | `.false` | boolean | |

# tableinusepf()

## Description

## Prototype

`tableinusepf` ( dataform1table *`table`*, printform1 *`f`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | dataform1table | |
| f | None | printform1 | |

# tableinusepf_ca()

## Description

## Prototype

`tableinusepf_ca` ( dataform1table *`table`*, printform1 *`f`*, array *`controls`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | dataform1table | |
| f | None | printform1 | |
| controls | None | array | |

# Chapter 38. gaugelib

The gaugelib library includes two gauge dialog implementations, one of which allows multiple gauges to be shown in the same dialog and managed separately. These dialogs can be used to provide a progress meter as feedback during routines.

# gaugedialog

## Description

## Type Tags

None

## Object Value

Objects of type gaugedialog have no value, and it is an error to try to get or set this value.

# gaugedialog.new()

### Description

### Prototype

*gaugedialog*.new ( gaugedialog *me*, type(wxdialogparent) *parent*, integer *left*, integer *top*, integer *width*, integer *height*, integer *range*, string *title*, boolean *showcancelbutton*, boolean *knowntotal*, boolean *usenativegauge*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | gaugedialog | |
| parent | None | type(wxdialogparent) | |
| left | 1 | integer | |
| top | 1 | integer | |
| width | 350 | integer | |
| height | 150 | integer | |
| range | None | integer | |
| title | None | string | |
| showcancel-button | .true | boolean | |
| knowntotal | .true | boolean | |
| usenativegauge | .true | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| close | function | |
| currentposition | integer | |
| d | wxdialog | |
| f | wxform | |
| forwards | boolean | |
| height | integer | |
| knowntotal | boolean | |
| left | integer | |
| parent | type(wxdialogparent) | |
| range | integer | |
| setgaugeposition | function | |
| setrange | function | |
| settitle | function | |
| setupforunknownpos | function | |
| show | function | |
| showcancelbutton | boolean | |
| solid | boolean | |
| title | string | |
| top | integer | |
| type | type | |
| usenativegauge | boolean | |
| usercanceled | boolean | |
| width | integer | |

# Methods

## close()

### Description

### Prototype

*gaugedialogvar*.close ( gaugedialog *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | gaugedialog | |

## setgaugeposition()

### Description

### Prototype

*gaugedialogvar*.setgaugeposition ( gaugedialog *me*, integer *position* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | gaugedialog | |
| position | None | integer | |

## setrange()

### Description

### Prototype

*gaugedialogvar*.setrange ( gaugedialog *me*, integer *range* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | gaugedialog | |
| range | None | integer | |

## settitle()

### Description

### Prototype

*gaugedialogvar*.settitle ( gaugedialog *me*, string *text* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | gaugedialog | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | |

## setupforunknownpos()

### Description

### Prototype

*gaugedialogvar*.setupforunknownpos ( gaugedialog *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | gaugedialog | |

## show()

### Description

### Prototype

*gaugedialogvar*.show ( gaugedialog *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | gaugedialog | |

# mgauge

# Description

# Type Tags

None

# Object Value

Objects of type mgauge have no value, and it is an error to try to get or set this value.

# mgauge.new()

## Description

### Prototype

*mgauge*.new ( mgauge *me*, integer *id*, integer *range*, boolean *solid*, boolean *usenativegauge* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | mgauge | |
| id | None | integer | |
| range | 1 | integer | |
| solid | .true | boolean | |
| usenativegauge | .true | boolean | |

## Properties

| Property | Type | Description |
|---|---|---|
| caption | wxformtext | |
| currentposition | integer | |
| id | integer | |
| nativegauge | wxformgauge | |
| next | mgauge | |
| progressback | wxgraphicrectangle | |
| progressbar | wxgraphicrectangle | |
| range | integer | |
| solid | boolean | |
| type | type | |
| usenativegauge | boolean | |

# multigaugedialog

## Description

## Type Tags

None

## Object Value

Objects of type multigaugedialog have no value, and it is an error to try to get or set this value.

## multigaugedialog.new()

### Description

## Prototype

*multigaugedialog*.new ( multigaugedialog *me*, type(wxdialogparent) *parent*, integer *gaugecount*, integer *left*, integer *top*, integer *width*, integer *height*, string *title*, boolean *showcancelbutton*, boolean *solid*, boolean *usenativegauge*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | multigaugedialog | |
| parent | None | type(wxdialogparent) | |
| gaugecount | 1 | integer | |
| left | .inf | integer | |
| top | .inf | integer | |
| width | 350 | integer | |
| height | 150 | integer | |
| title | None | string | |
| showcancel-button | .true | boolean | |
| solid | .true | boolean | |
| usenativegauge | .true | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| close | function | |
| d | wxdialog | |
| f | wxform | |
| firstgauge | mgauge | |
| gaugecount | integer | |
| height | integer | |
| left | integer | |
| parent | type(wxdialogparent) | |
| setcaption | function | |
| setgaugeposi-tion | function | |
| setrange | function | |
| settitle | function | |
| show | function | |
| showcancel-button | boolean | |

| Property | Type | Description |
|---|---|---|
| title | string | |
| top | integer | |
| type | type | |
| usenativegauge | boolean | |
| usercanceled | boolean | |
| width | integer | |

# Methods

## close()

### Description

### Prototype

*multigaugedialogvar*.close ( multigaugedialog *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | multigaugedialog | |

## setcaption()

### Description

### Prototype

*multigaugedialogvar*.setcaption ( multigaugedialog *me*, string *text*, integer *gaugenumber* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | multigaugedialog | |
| text | None | string | |
| gaugenumber | 1 | integer | |

## setgaugeposition()

### Description

### Prototype

*multigaugedialogvar*.setgaugeposition ( multigaugedialog *me*, integer *position*, integer *gaugenumber* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | multigaugedialog | |
| position | None | integer | |
| gaugenumber | 1 | integer | |

## setrange()

### Description

### Prototype

*multigaugedialogvar*.setrange ( multigaugedialog *me*, integer *range*, integer *gaugenumber* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | multigaugedialog | |
| range | None | integer | |
| gaugenumber | 1 | integer | |

## settitle()

### Description

### Prototype

*multigaugedialogvar*.settitle ( multigaugedialog *me*, string *text* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | multigaugedialog | |
| text | None | string | |

## show()

### Description

### Prototype

*multigaugedialogvar*.show ( multigaugedialog *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | multigaugedialog | |

# Chapter 39. graphicreportlib

## graphicreport1

### Description

### Type Tags

dataform1, printform1, report1

### Object Value

Objects of type graphicreport1 have no value, and it is an error to try to get or set this value.

## graphicreport1.new()

### Description

### Prototype

*graphicreport1*.new ( graphicreport1 *me*, integer *paperwidth*, integer *paperheight*, integer *outputtarget*, string *title*, string *defbooleanformat*, string *defintegerformat*, string *defnumberformat*, string *defdateformat*, string *deftimeformat*, string *defdatetimeformat*, SBLlocaledateinfo *datelocale*, SBLNumSettings *numlocale*, wxfont *defaultfont*, boolean *createdisplayform*, string *errmsgtitle*, type(wxdialogparent) *dialogparent*, boolean *loading*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| paperwidth | 210000 | integer | |
| paperheight | 297000 | integer | |
| outputtarget | 1 | integer | |
| title | None | string | |
| defbooleanformat | T\|F | string | |
| defintegerformat | . | string | |
| defnumberformat | 999999.00 | string | |
| defdateformat | yyyy.0m.0d | string | |
| deftimeformat | hh:mm:ss | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defdatetimeformat | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| defaultfont | None | wxfont | |
| createdisplayform | `.false` | boolean | |
| errmsgtitle | `Graphic Report Error` | string | |
| dialogparent | None | type(wxdialogparent) | |
| loading | `.false` | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addaggregate | function | |
| adddatasource | function | |
| addgroup | function | |
| addtable | function | |
| allowuserdefinedfilter | boolean | |
| allowuserdefinedorder | boolean | |
| centeroverdisplay | boolean | |
| currentoutput | printform1 | |
| currpage | wxprintpage | |
| currpagenumber | integer | |
| currrownumber | integer | |
| currtemplate | wxprintpagetemplate | |
| currtopofpage | integer | |
| datasources | dring | |
| defbooleanformat | string | |
| defdateformat | string | |
| defdatetimeformat | string | |

| Property | Type | Description |
|---|---|---|
| defintegerfor-mat | string | |
| defnumberfor-mat | string | |
| deftimeformat | string | |
| dialogdata | string | |
| dialogparent | type(wxdialogparent) | |
| dirty | boolean | |
| dpix | integer | |
| dpiy | integer | |
| errmsgtitle | string | |
| filename | string | |
| finddatasource | function | |
| findtable | function | |
| gauge | gaugedialog | |
| getaggregate-datatype | function | |
| gettbinfotable-sarray | function | |
| hasband | function | |
| hidespecified-filter | boolean | |
| hidespeci-fiedorder | boolean | |
| isportrait | boolean | |
| lastreportedpa-genumber | integer | |
| loading | boolean | |
| outputtarget | integer | |
| pagefooter-height | integer | |
| pageheader-height | integer | |
| paperheight | integer | |
| paperwidth | integer | |
| ppcs | ppcstype1 | |
| printout | wxprintout | |
| promptfordes-tination | boolean | |
| report | report1 | |
| reportform | graphicreport1form | |

| Property | Type | Description |
|---|---|---|
| run | function | |
| setpapersize | function | |
| showprinter-dialog | boolean | |
| startat100percent | boolean | |
| suppresspagefooter_lastpage | boolean | |
| suppresspagefooter_page1 | boolean | |
| suppresspageheader_page1 | boolean | |
| tables | dring | |
| title | string | |
| type | type | |
| usegauge | boolean | |
| valid | boolean | |

# Methods

## addaggregate()

### Description

### Prototype

*graphicreport1var*.addaggregate ( graphicreport1 *me*, report1group *group*, integer *aggregatetype*, integer *colno*, type *datatype*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| group | None | report1group | |
| aggregatetype | None | integer | |
| colno | None | integer | |
| datatype | None | type | |
| error | None | integer | |

## adddatasource()

### Description

### Prototype

*graphicreport1var*.adddatasource ( graphicreport1 *me*, type(*) *datasource*, string *source*, string *username*, string *password*, integer *codepage*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| datasource | None | type(*) | |
| source | None | string | |
| username | None | string | |
| password | None | string | |
| codepage | None | integer | |
| error | None | integer | |

## addgroup()

### Description

### Prototype

*graphicreport1var*.addgroup ( graphicreport1 *me*, string *name*, integer *colno*, type *datatype*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| name | None | string | |
| colno | None | integer | |
| datatype | None | type | |
| error | None | integer | |

## addtable()

### Description

### Prototype

*graphicreport1var*.addtable ( graphicreport1 *me*, type(db1table) *table*, dataform1datasource *source*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| table | None | type(db1table) | |
| source | None | dataform1datasource | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# finddatasource()

## Description

## Prototype

*graphicreport1var*.finddatasource ( graphicreport1 *me*, string *sourcename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| sourcename | None | string | |

# findtable()

## Description

## Prototype

*graphicreport1var*.findtable ( graphicreport1 *me*, string *tablename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| tablename | None | string | |

# getaggregatedatatype()

## Description

## Prototype

*graphicreport1var*.getaggregatedatatype ( graphicreport1 *me*, integer *colno*, integer *aggregatetype* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| colno | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggregatetype | None | integer | |

# gettbinfotablesarray()

## Description

## Prototype

*graphicreport1var*.gettbinfotablesarray ( graphicreport1 *me*, boolean *inuseonly* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| inuseonly | .false | boolean | |

# hasband()

## Description

## Prototype

*graphicreport1var*.hasband ( graphicreport1 *me*, string *bandname*, string *groupname* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| bandname | None | string | |
| groupname | None | string | |

# run()

## Description

## Prototype

*graphicreport1var*.run ( graphicreport1 *me*, string *errmsg*, integer *erridx*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| errmsg | None | string | |
| erridx | None | integer | |
| error | None | integer | |

## setpapersize()

### Description

### Prototype

*graphicreport1var*.setpapersize ( graphicreport1 *me*, integer *paperwidth*, integer *paperheight*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1 | |
| paperwidth | None | integer | |
| paperheight | None | integer | |
| error | None | integer | |

# graphicreport1arc

## Description

## Type Tags

graphicreport1graphic, dataform1graphic, printform1graphic

## Object Value

Objects of type graphicreport1arc have no value, and it is an error to try to get or set this value.

## graphicreport1arc.new()

### Description

### Prototype

*graphicreport1arc*.new ( graphicreport1arc *me*, graphicreport1formpage *page*, printform1arc *graphic*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1arc | |
| page | None | graphicreport1formpage | |
| graphic | None | printform1arc | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | graphicreport1form | |
| formnode | dlistnode | |
| graphic | printform1arc | |
| page | graphicreport1formpage | |
| pagenode | dlistnode | |
| remove | function | |
| setnext | function | |
| type | type | |

# Methods

## remove()

### Description

### Prototype

*graphicreport1arcvar*.remove ( graphicreport1arc *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1arc | |

## setnext()

### Description

### Prototype

*graphicreport1arcvar*.setnext ( graphicreport1arc *me*, type(graphicreport1graphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1arc | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| next | None | type(graphicreport1graphic) | |

# graphicreport1ellipse

## Description

## Type Tags

graphicreport1graphic, dataform1graphic, printform1graphic

## Object Value

Objects of type graphicreport1ellipse have no value, and it is an error to try to get or set this value.

## graphicreport1ellipse.new()

### Description

### Prototype

*graphicreport1ellipse*.new ( graphicreport1ellipse *me*, graphicreport1formpage *page*, printform1ellipse *graphic*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1ellipse | |
| page | None | graphicreport1formpage | |
| graphic | None | printform1ellipse | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | graphicreport1form | |
| formnode | dlistnode | |
| graphic | printform1ellipse | |
| page | graphicreport1formpage | |
| pagenode | dlistnode | |
| remove | function | |

| Property | Type | Description |
|---|---|---|
| setnext | function | |
| type | type | |

# Methods

## remove()

### Description

### Prototype

*graphicreport1ellipsevar*.remove ( graphicreport1ellipse *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1ellipse | |

## setnext()

### Description

### Prototype

*graphicreport1ellipsevar*.setnext ( graphicreport1ellipse *me*, type(graphicreport1graphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1ellipse | |
| next | None | type(graphicreport1graphic) | |

# graphicreport1form

# Description

# Type Tags

dataform1, printform1, dataform1linkcontainer

# Object Value

Objects of type graphicreport1form have no value, and it is an error to try to get or set this value.

# graphicreport1form.new()

## Description

## Prototype

*graphicreport1form*.new ( graphicreport1form *me*, graphicreport1 *graphicreport*, integer *defpagebackcolor*, wxfont *deffont*, boolean *createdisplayform*, boolean *loading*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| graphicreport | None | graphicreport1 | |
| defpageback-color | None | integer | |
| deffont | None | wxfont | |
| createdisplay-form | .false | boolean | |
| loading | .false | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addcontrol | function | |
| addfontinfo | function | |
| addgraphic | function | |
| addpage | function | |
| controlcounter | integer | |
| controls | dring | |
| currentpage | graphicreport1formpage | |
| findcontrol | function | |
| findfontinfo | function | |
| findgraphic | function | |
| findpage | function | |
| fontinfo | array | |
| fontresizek-ludgevalue | number | |

| Property | Type | Description |
|---|---|---|
| gdi | GDI | |
| getprinttextex-tent | function | |
| getwrapheight | function | |
| getwrapheight2 | function | |
| graphicreport | graphicreport1 | |
| graphics | dring | |
| pages | dring | |
| printform | printform1 | |
| type | type | |
| usewrapheight2 | boolean | |
| winspool | WINSPOOL | |
| wrapcharcoun-tkludgevalue | number | |
| wrap-kludgevalue | number | |

# Methods

## !()

### Description

### Prototype

*graphicreport1formvar*.! ( graphicreport1form *me*, string *controlname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| controlname | None | string | |

## addcontrol()

### Description

### Prototype

*graphicreport1formvar*.addcontrol ( graphicreport1form *me*, type *controltype*, integer *printleft*, integer *printtop*, integer *printwidth*, integer *printheight*, string *text*, boolean *visible*, wxbitmap *bitmap*, string *scaling*, integer *backgroundrgb*, integer *textrgb*, string *printalignment*, wxfont *font*, string *printname*, boolean *backgroundvisible*, boolean

*undergraphics*, boolean *underbitmaps*, boolean *wrap*, type(graphicreportform1control) *next*, graphicreport1formpage *page*, integer *colno*, integer *aggregatetype*, string *displayformat*, boolean *skipbitmapcheck*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| controltype | None | type | |
| printleft | None | integer | |
| printtop | None | integer | |
| printwidth | None | integer | |
| printheight | None | integer | |
| text | None | string | |
| visible | `.true` | boolean | |
| bitmap | None | wxbitmap | |
| scaling | None | string | |
| backgroundrgb | `16777215` | integer | |
| textrgb | `0` | integer | |
| printalignment | `left,top` | string | |
| font | None | wxfont | |
| printname | None | string | |
| backgroundvisible | `.false` | boolean | |
| undergraphics | `.false` | boolean | |
| underbitmaps | `.false` | boolean | |
| wrap | `.false` | boolean | |
| next | None | type(graphicreportform1control) | |
| page | None | graphicreport1formpage | |
| colno | None | integer | |
| aggregatetype | None | integer | |
| displayformat | None | string | |
| skipbitmapcheck | `.false` | boolean | |
| error | None | integer | |

## addfontinfo()

### Description

### Prototype

*graphicreport1formvar*.`addfontinfo` ( graphicreport1form *me*, wxfont *font*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| font | None | wxfont | |
| error | None | integer | |

## addgraphic()

### Description

### Prototype

*graphicreport1formvar*.addgraphic ( graphicreport1form *me*, type *graphictype*, point *printpoint1*, point *printpoint2*, point *printpoint3*, point *printmidpoint*, integer *rgb*, integer *borderrgb*, integer *width*, integer *borderwidth*, boolean *visible*, boolean *bordervisible*, string *printname*, type(graphicreportform1control) *next*, graphicreport1formpage *page*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| graphictype | None | type | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printpoint3 | None | point | |
| printmidpoint | None | point | |
| rgb | None | integer | |
| borderrgb | None | integer | |
| width | None | integer | |
| borderwidth | None | integer | |
| visible | .true | boolean | |
| bordervisible | .true | boolean | |
| printname | None | string | |
| next | None | type(graphicreportform1control) | |
| page | None | graphicreport1formpage | |
| error | None | integer | |

## addpage()

### Description

### Prototype

*graphicreport1formvar*.addpage ( graphicreport1form *me*, integer *width*, integer *height*, integer *backgroundrgb*, printform1page *after*, string *name*, report1group *group*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| width | None | integer | |
| height | None | integer | |
| backgroundrgb | None | integer | |
| after | None | printform1page | |
| name | None | string | |
| group | None | report1group | |
| error | None | integer | |

## findcontrol()

### Description

### Prototype

*graphicreport1formvar*.findcontrol ( graphicreport1form *me*, string *controlname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| controlname | None | string | |

## findfontinfo()

### Description

### Prototype

*graphicreport1formvar*.findfontinfo ( graphicreport1form *me*, wxfont *font*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| font | None | wxfont | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# findgraphic()

## Description

## Prototype

*graphicreport1formvar*.findgraphic ( graphicreport1form *me*, string *graphicname* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| graphicname | None | string | |

# findpage()

## Description

## Prototype

*graphicreport1formvar*.findpage ( graphicreport1form *me*, string *pagename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| pagename | None | string | |

# getprinttextextent()

## Description

## Prototype

*graphicreport1formvar*.getprinttextextent ( graphicreport1form *me*, wxfont *font*, string *s*, integer *width*, integer *height*, integer *descent*, integer *extleading*, integer *maxwidth*, integer *charcount* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| font | None | wxfont | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| width | None | integer | |
| height | None | integer | |
| descent | None | integer | |
| extleading | None | integer | |
| maxwidth | None | integer | |
| charcount | None | integer | |

# getwrapheight()

## Description

## Prototype

*graphicreport1formvar*.getwrapheight ( graphicreport1form *me*, wxfont *font*, string *s*, integer *width*, integer *origheight* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| font | None | wxfont | |
| s | None | string | |
| width | None | integer | |
| origheight | None | integer | |

# getwrapheight2()

## Description

## Prototype

*graphicreport1formvar*.getwrapheight2 ( graphicreport1form *me*, wxfont *font*, string *s*, integer *width*, integer *origheight* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1form | |
| font | None | wxfont | |
| s | None | string | |
| width | None | integer | |
| origheight | None | integer | |

# graphicreport1formbitmap

## Description

## Type Tags

graphicreport1formcontrol, dataform1control

## Object Value

Objects of type graphicreport1formbitmap have no value, and it is an error to try to get or set this value.

## graphicreport1formbitmap.new()

### Description

### Prototype

*graphicreport1formbitmap*.new ( graphicreport1formbitmap *me*, graphicreport1formpage *page*, printform1bitmap *control*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formbitmap | |
| page | None | graphicreport1formpage | |
| control | None | printform1bitmap | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| colno | integer | |
| control | printform1bitmap | |
| form | graphicreport1form | |
| formnode | dlistnode | |
| page | graphicreport1formpage | |
| pagenode | dlistnode | |
| remove | function | |
| setnext | function | |

| Property | Type | Description |
|---|---|---|
| type | type | |

# Methods

## remove()

### Description

### Prototype

*graphicreport1formbitmapvar*.remove ( graphicreport1formbitmap *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formbitmap | |

## setnext()

### Description

### Prototype

*graphicreport1formbitmapvar*.setnext ( graphicreport1formbitmap *me*, type(graphicreport1formcontrol) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formbitmap | |
| next | None | type(graphicreport1formcontrol) | |

# graphicreport1formcontrol

# Description

# Type Tags

graphicreport1formcontrol

# Object Value

Objects of type graphicreport1formcontrol have no value, and it is an error to try to get or set this value.

# graphicreport1formcontrol.new()

## Description

## Prototype

*graphicreport1formcontrol*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| colno | integer | |
| control | type(printform1control) | |
| form | graphicreport1form | |
| formnode | dlistnode | |
| page | graphicreport1formpage | |
| pagenode | dlistnode | |
| remove | function | |
| setnext | function | |
| type | type | |

# graphicreport1formpage

## Description

## Type Tags

printform1page, dataform1page

## Object Value

Objects of type graphicreport1formpage have no value, and it is an error to try to get or set this value.

# graphicreport1formpage.new()

## Description

## Prototype

*graphicreport1formpage*.new ( graphicreport1formpage *me*, graphicreport1form *graphicreportform*, printform1page *page*, report1group *group*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formpage | |
| graphicreport-form | None | graphicreport1form | |
| page | None | printform1page | |
| group | None | report1group | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addcontrol | function | |
| addgraphic | function | |
| controls | dring | |
| formnode | dlistnode | |
| graphicreport-form | graphicreport1form | |
| graphics | dring | |
| group | report1group | |
| onoutput | event | |
| page | printform1page | |
| remove | function | |
| type | type | |

# Methods

## addcontrol()

### Description

### Prototype

*graphicreport1formpagevar*.addcontrol ( graphicreport1formpage *me*, type *controltype*, integer *printleft*, integer *printtop*, integer *printwidth*, integer *printheight*, string *text*, boolean *visible*, wxbitmap *bitmap*, string *scaling*, integer *backgroundrgb*, integer *textrgb*, string *printalignment*, wxfont *font*, string *printname*,

boolean *backgroundvisible*, boolean *undergraphics*, boolean *underbitmaps*, boolean *wrap*, type(graphicreportform1control) *next*, integer *colno*, integer *aggregatetype*, string *displayformat*, boolean *skipbitmapcheck*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formpage | |
| controltype | None | type | |
| printleft | None | integer | |
| printtop | None | integer | |
| printwidth | None | integer | |
| printheight | None | integer | |
| text | None | string | |
| visible | .true | boolean | |
| bitmap | None | wxbitmap | |
| scaling | None | string | |
| backgroundrgb | None | integer | |
| textrgb | 0 | integer | |
| printalignment | None | string | |
| font | None | wxfont | |
| printname | None | string | |
| backgroundvisible | .true | boolean | |
| undergraphics | .false | boolean | |
| underbitmaps | .false | boolean | |
| wrap | .false | boolean | |
| next | None | type(graphicreportform1control) | |
| colno | None | integer | |
| aggregatetype | None | integer | |
| displayformat | None | string | |
| skipbitmapcheck | .false | boolean | |
| error | None | integer | |

## addgraphic()

### Description

### Prototype

*graphicreport1formpagevar*.addgraphic ( graphicreport1formpage *me*, type *graphictype*, point *printpoint1*, point *printpoint2*, point *printpoint3*, point *printmidpoint*,

integer *rgb*, integer *borderrgb*, integer *width*, integer *borderwidth*, boolean *visible*, boolean *bordervisible*, string *printname*, type(graphicreportform1control) *next*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formpage | |
| graphictype | None | type | |
| printpoint1 | None | point | |
| printpoint2 | None | point | |
| printpoint3 | None | point | |
| printmidpoint | None | point | |
| rgb | None | integer | |
| borderrgb | None | integer | |
| width | None | integer | |
| borderwidth | None | integer | |
| visible | `.true` | boolean | |
| bordervisible | `.true` | boolean | |
| printname | None | string | |
| next | None | type(graphicreportform1control) | |
| error | None | integer | |

## remove()

### Description

### Prototype

*graphicreport1formpagevar*.remove ( graphicreport1formpage *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formpage | |

# graphicreport1formtext

# Description

# Type Tags

graphicreport1formcontrol, dataform1control

# Object Value

Objects of type graphicreport1formtext have no value, and it is an error to try to get or set this value.

# graphicreport1formtext.new()

## Description

## Prototype

*graphicreport1formtext*.new ( graphicreport1formtext *me*, graphicreport1formpage *page*, printform1text *control*, boolean *wrap*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formtext | |
| page | None | graphicreport1formpage | |
| control | None | printform1text | |
| wrap | .false | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| aggregatetype | integer | |
| calculation | string | |
| colno | integer | |
| control | printform1text | |
| displayformat | string | |
| form | graphicreport1form | |
| formnode | dlistnode | |
| origfont | wxfont | |
| page | graphicreport1formpage | |
| pagenode | dlistnode | |
| remove | function | |
| setcalculation | function | |
| setfont | function | |
| setnext | function | |
| type | type | |

| Property | Type | Description |
|---|---|---|
| wrap | boolean | |

# Methods

## remove()

### Description

### Prototype

*graphicreport1formtextvar*.remove ( graphicreport1formtext *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formtext | |

## setcalculation()

### Description

### Prototype

*graphicreport1formtextvar*.setcalculation ( graphicreport1formtext *me*, string *calculation* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formtext | |
| calculation | None | string | |

## setfont()

### Description

### Prototype

*graphicreport1formtextvar*.setfont ( graphicreport1formtext *me*, wxfont *font* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formtext | |
| font | None | wxfont | |

### setnext()

#### Description

#### Prototype

*graphicreport1formtextvar*.setnext ( graphicreport1formtext *me*, type(graphicreport1formcontrol) *next* )

#### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1formtext | |
| next | None | type(graphicreport1formcontrol) | |

# graphicreport1graphic

## Description

## Type Tags

graphicreport1graphic

## Object Value

Objects of type graphicreport1graphic have no value, and it is an error to try to get or set this value.

## graphicreport1graphic.new()

### Description

### Prototype

*graphicreport1graphic*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | graphicreport1form | |
| formnode | dlistnode | |

| Property | Type | Description |
|---|---|---|
| graphic | type(printform1graphic) | |
| page | graphicreport1formpage | |
| pagenode | dlistnode | |
| remove | function | |
| setnext | function | |
| type | type | |

# graphicreport1line

## Description

## Type Tags

graphicreport1graphic, dataform1graphic, printform1graphic

## Object Value

Objects of type graphicreport1line have no value, and it is an error to try to get or set this value.

## graphicreport1line.new()

### Description

### Prototype

*graphicreport1line*.new ( graphicreport1line *me*, graphicreport1formpage *page*, printform1line *graphic*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1line | |
| page | None | graphicreport1formpage | |
| graphic | None | printform1line | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | graphicreport1form | |

| Property | Type | Description |
|---|---|---|
| formnode | dlistnode | |
| graphic | printform1line | |
| page | graphicreport1formpage | |
| pagenode | dlistnode | |
| remove | function | |
| setnext | function | |
| type | type | |

# Methods

## remove()

### Description

### Prototype

*graphicreport1linevar*.remove ( graphicreport1line *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1line | |

## setnext()

### Description

### Prototype

*graphicreport1linevar*.setnext ( graphicreport1line *me*, type(graphicreport1graphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1line | |
| next | None | type(graphicreport1graphic) | |

# graphicreport1rectangle

# Description

# Type Tags

graphicreport1graphic, dataform1graphic, printform1graphic

# Object Value

Objects of type graphicreport1rectangle have no value, and it is an error to try to get or set this value.

# graphicreport1rectangle.new()

## Description

## Prototype

*graphicreport1rectangle*.new ( graphicreport1rectangle *me*, graphicreport1formpage *page*, printform1rectangle *graphic*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1rectangle | |
| page | None | graphicreport1formpage | |
| graphic | None | printform1rectangle | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | graphicreport1form | |
| formnode | dlistnode | |
| graphic | printform1rectangle | |
| page | graphicreport1formpage | |
| pagenode | dlistnode | |
| remove | function | |
| setnext | function | |
| type | type | |

# Methods

## remove()

### Description

### Prototype

*graphicreport1rectanglevar*.remove ( graphicreport1rectangle *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1rectangle | |

## setnext()

### Description

### Prototype

*graphicreport1rectanglevar*.setnext ( graphicreport1rectangle *me*, type(graphicreport1graphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1rectangle | |
| next | None | type(graphicreport1graphic) | |

# graphicreport1triangle

## Description

## Type Tags

graphicreport1graphic, dataform1graphic, printform1graphic

## Object Value

Objects of type graphicreport1triangle have no value, and it is an error to try to get or set this value.

## graphicreport1triangle.new()

### Description

### Prototype

*graphicreport1triangle*.new ( graphicreport1triangle *me*, graphicreport1formpage *page*, printform1triangle *graphic*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1triangle | |
| page | None | graphicreport1formpage | |
| graphic | None | printform1triangle | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| form | graphicreport1form | |
| formnode | dlistnode | |
| graphic | printform1triangle | |
| page | graphicreport1formpage | |
| pagenode | dlistnode | |
| remove | function | |
| setnext | function | |
| type | type | |

# Methods

## remove()

### Description

### Prototype

*graphicreport1trianglevar*.remove ( graphicreport1triangle *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1triangle | |

## setnext()

### Description

### Prototype

*graphicreport1trianglevar*.setnext ( graphicreport1triangle *me*, type(graphicreport1graphic) *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | graphicreport1triangle | |
| next | None | type(graphicreport1graphic) | |

# __gr_hasaggregate()

## Description

## Prototype

__gr_hasaggregate ( string *title* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| title | None | string | |

# __gr_hascalculation()

## Description

## Prototype

__gr_hascalculation ( string *title* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| title | None | string | |

# __gr_updatecontroldatasource()

## Description

## Prototype

__gr_updatecontroldatasource ( array *controlsources*, array *parsedstatement*, type(graphicreport1formcontrol) *c* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| controlsources | None | array | |
| parsedstate-ment | None | array | |
| c | None | type(graphicreport1formcontrol) | |

# datasourceinusegr()

## Description

## Prototype

`datasourceinusegr` ( dataform1datasource *`src`*, graphicreport1 *`gr`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| src | None | dataform1datasource | |
| gr | None | graphicreport1 | |

# datasourceinusegr_ca()

## Description

## Prototype

`datasourceinusegr_ca` ( dataform1datasource *`src`*, graphicreport1 *`gr`*, array *`controls`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| src | None | dataform1datasource | |
| gr | None | graphicreport1 | |
| controls | None | array | |

# fixgraphicreportcontrolsources()

## Description

## Prototype

`fixgraphicreportcontrolsources` ( graphicreport1 *`report`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | graphicreport1 | |

# graphicreport1mergeforms()

## Description

## Prototype

`graphicreport1mergeforms` ( graphicreport1 *`report`*, graphicreport1 *`newreportcontent`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | graphicreport1 | |
| newreportcontent | None | graphicreport1 | |
| error | None | integer | |

# loadgraphicreport()

## Description

## Prototype

`loadgraphicreport` ( string *`filename`*, integer *`pagewidth`*, integer *`pageheight`*, integer *`defpagebackcolor`*, string *`defbooleanformat`*, string *`defintegerformat`*, string *`defnumberformat`*, string *`defdateformat`*, string *`deftimeformat`*, string *`defdatetimeformat`*, SBLocaledateinfo *`datelocale`*, SBLNumSettings *`numlocale`*, dring *`datasources`*, array *`tables`*, ppcstype1 *`ppcs`*, boolean *`createdisplayform`*, integer *`error`*, string *`errortext`*, wxwindow *`window`*, type(wxdialogparent) *`parentwindow`*, array *`messages`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| pagewidth | `210000` | integer | |
| pageheight | `297000` | integer | |
| defpagebackcolor | `16777215` | integer | |
| defbooleanformat | `T\|F` | string | |
| defintegerformat | `.` | string | |
| defnumberformat | `999999.00` | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimeformat | None | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| createdisplayform | `.false` | boolean | |
| error | None | integer | |
| errortext | None | string | |
| window | None | wxwindow | |
| parentwindow | None | type(wxdialogparent) | |
| messages | None | array | |

# loadgraphicreport1fromstring()

## Description

## Prototype

`loadgraphicreport1fromstring` ( string *`formcontent`*, integer *`pagewidth`*, integer *`pageheight`*, integer *`defpagebackcolor`*, string *`defbooleanformat`*, string *`defintegerformat`*, string *`defnumberformat`*, string *`defdateformat`*, string *`deftimeformat`*, string *`defdatetimeformat`*, SBLlocaledateinfo *`datelocale`*, SBLNumSettings *`numlocale`*, dring *`datasources`*, array *`tables`*, ppcstype1 *`ppcs`*, boolean *`createdisplayform`*, integer *`error`*, string *`errortext`*, wxwindow *`window`*, type(wxdialogparent) *`parentwindow`*, array *`messages`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| formcontent | None | string | |
| pagewidth | `210000` | integer | |
| pageheight | `297000` | integer | |
| defpagebackcolor | `16777215` | integer | |
| defbooleanformat | `T\|F` | string | |
| defintegerformat | `.` | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defnumberformat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimeformat | None | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| createdisplayform | `.false` | boolean | |
| error | None | integer | |
| errortext | None | string | |
| window | None | wxwindow | |
| parentwindow | None | type(wxdialogparent) | |
| messages | None | array | |

# loadgrxmlreportfromstring()

## Description

## Prototype

`loadgrxmlreportfromstring` ( string *`reportcontent`*, array *`messages`*, integer *`error`*, string *`errortext`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| reportcontent | None | string | |
| messages | None | array | |
| error | None | integer | |
| errortext | None | string | |

# parseselecttocontrolsources()

## Description

## Prototype

`parseselecttocontrolsources` ( graphicreport1 *`report`*, array *`parsedstatement`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | graphicreport1 | |
| parsedstate-ment | None | array | |

# report1_graphicreport_output_groupfooter()

## Description

## Prototype

`report1_graphicreport_output_groupfooter` ( report1group *`group`*, report1groupinst *`groupinst`*, graphicreport1 *`graphicreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| group | None | report1group | |
| groupinst | None | report1groupinst | |
| graphicreport | None | graphicreport1 | |

# report1_graphicreport_output_groupheader()

## Description

## Prototype

`report1_graphicreport_output_groupheader` ( report1group *`group`*, report1groupinst *`groupinst`*, graphicreport1 *`graphicreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| group | None | report1group | |
| groupinst | None | report1groupinst | |
| graphicreport | None | graphicreport1 | |

# report1_graphicreport_output_pagefooter()

## Description

## Prototype

`report1_graphicreport_output_pagefooter` ( report1 *`report`*, report1inst *`reportinst`*, graphicreport1 *`graphicreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| graphicreport | None | graphicreport1 | |

# report1_graphicreport_output_pageheader()

## Description

## Prototype

`report1_graphicreport_output_pageheader` ( report1 *`report`*, report1inst *`reportinst`*, graphicreport1 *`graphicreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| graphicreport | None | graphicreport1 | |

# report1_graphicreport_output_reportfooter()

## Description

## Prototype

`report1_graphicreport_output_reportfooter` ( report1 *`report`*, report1inst *`reportinst`*, graphicreport1 *`graphicreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| graphicreport | None | graphicreport1 | |

# report1_graphicreport_output_reportheader()

## Description

## Prototype

`report1_graphicreport_output_reportheader` ( report1 *`report`*, report1inst *`reportinst`*, graphicreport1 *`graphicreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| graphicreport | None | graphicreport1 | |

# report1_graphicreport_outputrow()

## Description

## Prototype

`report1_graphicreport_outputrow` ( report1 *`report`*, report1inst *`reportinst`*, array *`columns`*, array *`currcolvals`*, boolean *`reading`*, graphicreport1 *`graphicreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| columns | None | array | |
| currcolvals | None | array | |
| reading | None | boolean | |
| graphicreport | None | graphicreport1 | |

# savegraphicreport()

## Description

## Prototype

`savegraphicreport` ( graphicreport1 *`gr`*, string *`filename`*, integer *`tabsize`*, string *`origfilename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| gr | None | graphicreport1 | |
| filename | None | string | |
| tabsize | 2 | integer | |
| origfilename | None | string | |
| error | None | integer | |

# savegraphicreport1ctrlsasstring()

## Description

## Prototype

`savegraphicreport1ctrlsasstring` ( graphicreport1 *`gr`*, array *`controls`*, integer *`tabsize`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| gr | None | graphicreport1 | |
| controls | None | array | |
| tabsize | 2 | integer | |
| error | None | integer | |

# tableinusegr()

## Description

## Prototype

`tableinusegr` ( dataform1table *`table`*, graphicreport1 *`gr`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | dataform1table | |
| gr | None | graphicreport1 | |

# tableinusegr_ca()

## Description

## Prototype

`tableinusegr_ca` ( dataform1table *`table`*, graphicreport1 *`gr`*, array *`controls`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | dataform1table | |
| gr | None | graphicreport1 | |
| controls | None | array | |

# Chapter 40. httpclientlib

This library provides HTTP 1.0 compliant GET and POST functionality for interacting with web browsers. Results are returned in appropriate objects. Any content type can be retrieved.

# httpcookie

## Description

## Type Tags

None

## Object Value

Objects of type httpcookie have no value, and it is an error to try to get or set this value.

## httpcookie.new()

### Description

### Prototype

*`httpcookie`*.new ( httpcookie *`me`*, string *`name`*, string *`value`*, string *`path`*, datetime *`expires`*, string *`domain`*, integer *`max_age`*, boolean *`secure`*, boolean *`httponly`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httpcookie | |
| name | None | string | |
| value | None | string | |
| path | None | string | |
| expires | None | datetime | |
| domain | None | string | |
| max_age | None | integer | |
| secure | `.false` | boolean | |
| httponly | `.false` | boolean | |

## Properties

| Property | Type | Description |
|---|---|---|
| domain | string | |

| Property | Type | Description |
|---|---|---|
| expires | datetime | |
| gettext | function | |
| httponly | boolean | |
| max_age | integer | |
| name | string | |
| parsetext | function | |
| path | string | |
| secure | boolean | |
| type | type | |
| value | string | |

# Methods

## gettext()

### Description

### Prototype

*httpcookievar*.gettext ( httpcookie *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httpcookie | |

## parsetext()

### Description

### Prototype

*httpcookievar*.parsetext ( httpcookie *me*, string *s* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httpcookie | |
| s | None | string | |

# httpentityheader

## Description

# Type Tags

None

# Object Value

Objects of type httpentityheader have no value, and it is an error to try to get or set this value.

# httpentityheader.new()

## Description

## Prototype

*httpentityheader*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| allow | string | |
| content_encoding | string | |
| content_language | string | |
| content_length | string | |
| content_location | string | |
| content_md5 | string | |
| content_range | string | |
| content_type | string | |
| expires | string | |
| last_modified | string | |
| type | type | |

# httpgeneralheader

## Description

## Type Tags

None

# Object Value

Objects of type httpgeneralheader have no value, and it is an error to try to get or set this value.

# httpgeneralheader.new()

## Description

## Prototype

*httpgeneralheader*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| cache_control | string | |
| connection | string | |
| date_ | string | |
| pragma | string | |
| trailer | string | |
| transfer_encoding | string | |
| type | type | |
| upgrade | string | |
| via | string | |
| warning | string | |

# httprequest

# Description

# Type Tags

None

# Object Value

Objects of type httprequest have no value, and it is an error to try to get or set this value.

# httprequest.new()

## Description

## Prototype

*httprequest*.new ( httprequest *me*, string *useragent*, string *content_type*, string *protocol* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httprequest | |
| useragent | `sim-pol-http-clientlib/1.0 (http://www.simpol.com)` | string | |
| content_type | `application/x-www-form-urlencoded` | string | |
| protocol | `1.0` | string | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| entitybody | blob | |
| entityheader | httpentityheader | |
| generalheader | httpgeneralheader | |
| getrequestblob | function | |
| protocol | string | |
| requestheader | httprequestheader | |
| requesttype | string | |
| setcookie | function | |
| settunnelurl | function | |
| tunnelurl | URL | |
| type | type | |
| url | URL | |

# Methods

## getrequestblob()

### Description

### Prototype

*httprequestvar*.getrequestblob ( httprequest *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httprequest | |

## setcookie()

### Description

### Prototype

*httprequestvar*.setcookie ( httprequest *me*, string *name*, string *value*, string *path*, datetime *expires* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httprequest | |
| name | None | string | |
| value | None | string | |
| path | None | string | |
| expires | None | datetime | |

## settunnelurl()

### Description

### Prototype

*httprequestvar*.settunnelurl ( httprequest *me*, URL *tunnelurl* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httprequest | |
| tunnelurl | None | URL | |

# httprequestheader

# Description

# Type Tags

None

# Object Value

Objects of type httprequestheader have no value, and it is an error to try to get or set this value.

# httprequestheader.new()

## Description

## Prototype

*httprequestheader*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| accept | string | |
| accept_charset | string | |
| accept_encoding | string | |
| accept_language | string | |
| authmethod | string | |
| authorization | string | |
| cookies | array | |
| expect | string | |
| from | string | |
| getcookieblob | function | |
| host | string | |
| if_match | string | |
| if_modified_since | string | |
| if_none_match | string | |
| if_range | string | |
| if_unmodified_since | string | |
| max_forwards | string | |
| password | string | |
| proxy_authorization | string | |
| range | string | |

| Property | Type | Description |
|---|---|---|
| referer | string | |
| te | string | |
| type | type | |
| user_agent | string | |
| username | string | |
| xtraheader-fields | array | |

# Methods

## getcookieblob()

### Description

### Prototype

*httprequestheadervar*.getcookieblob ( httprequestheader *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httprequestheader | |

# httpresponse

# Description

# Type Tags

None

# Object Value

Objects of type httpresponse have no value, and it is an error to try to get or set this value.

# httpresponse.new()

## Description

## Prototype

*httpresponse*.new ( httpresponse *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httpresponse | |

## Properties

| Property | Type | Description |
|---|---|---|
| entitybody | blob | |
| entityheader | httpentityheader | |
| errorstatus | string | |
| fullheader | string | |
| generalheader | httpgeneralheader | |
| httpversion | string | |
| readheader | function | |
| reasonphrase | string | |
| responseheader | httpresponseheader | |
| statuscode | integer | |
| statusline | string | |
| type | type | |

## Methods

### readheader()

#### Description

#### Prototype

*`httpresponsevar`*`.readheader` ( httpresponse *`me`*, string *`sHeader`* )

#### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | httpresponse | |
| sHeader | None | string | |

# httpresponseheader

## Description

## Type Tags

None

# Object Value

Objects of type httpresponseheader have no value, and it is an error to try to get or set this value.

# httpresponseheader.new()

## Description

## Prototype

*httpresponseheader*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| accept_ranges | string | |
| age | string | |
| cookies | array | |
| etag | string | |
| location | string | |
| proxy_authenticate | string | |
| retry_after | string | |
| server | string | |
| type | type | |
| vary | string | |
| www_authenticate | string | |

# httpdelete()

# Description

# Prototype

httpdelete ( string *sUrl*, string *variables* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sUrl | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| variables | None | string | |

# httpget()

## Description

## Prototype

`httpget` ( string *`sUrl`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sUrl | None | string | |

# httphead()

## Description

## Prototype

`httphead` ( string *`sUrl`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sUrl | None | string | |

# httppost()

## Description

## Prototype

`httppost` ( string *`sUrl`*, string *`variables`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sUrl | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| variables | None | string | |

# httpput()

## Description

## Prototype

`httpput` ( string *`sUrl`*, string *`variables`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sUrl | None | string | |
| variables | None | string | |

# httpsendreceive()

## Description

## Prototype

`httpsendreceive` ( httprequest *`request`*, boolean *`debug`*, LogManager *`logManager`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| request | None | httprequest | |
| debug | `.false` | boolean | |
| logManager | None | LogManager | |

# Chapter 41. ieeelib

## fromieee4()

### Description

### Prototype

`fromieee4` ( blob *b*, boolean *lbo* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| b | None | blob | |
| lbo | None | boolean | |

## fromieee8()

### Description

### Prototype

`fromieee8` ( blob *b*, boolean *lbo* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| b | None | blob | |
| lbo | None | boolean | |

## toieee4()

### Description

### Prototype

`toieee4` ( number *n*, boolean *lbo* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| n | None | number | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lbo | None | boolean | |

# toieee8()

## Description

## Prototype

`toieee8` ( number *n*, boolean *lbo* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| n | None | number | |
| lbo | None | boolean | |

# Chapter 42. imagelib

Thi9s library was started by Simpol Limited and extended significantly and contributed by John Roberts. It provides functionality for saving blobs as BMPs and XPM files, as well as reading them from such files.

# BMP

## Description

## Type Tags

None

## Object Value

Objects of type BMP have no value, and it is an error to try to get or set this value.

## BMP.new()

### Description

### Prototype

*BMP*.new ( BMP *me*, string *filename*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BMP | |
| filename | None | string | |
| name | None | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| backcolor | integer | |
| charsperpixel | integer | |
| data | blob | |
| filename | string | |
| getdefbackcol-or | function | |
| getheader | function | |
| header | BMP_header | |
| height | integer | |

| Property | Type | Description |
|---|---|---|
| infoheader | BMP_infoheader | |
| name | string | |
| palette | colorpalette | |
| readfromfile | function | |
| rereadpalette | function | |
| type | type | |
| width | integer | |
| writetofile | function | |

# Methods

## getdefbackcolor()

### Description

### Prototype

*BMPvar*.getdefbackcolor ( BMP *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BMP | |

## getheader()

### Description

### Prototype

*BMPvar*.getheader ( BMP *bmp*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bmp | None | BMP | |
| error | None | integer | |

## readfromfile()

### Description

### Prototype

*BMPvar*.readfromfile ( BMP *me*, string *filename*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BMP | |
| filename | None | string | |
| error | None | integer | |

## rereadpalette()

### Description

### Prototype

*BMPvar*.rereadpalette ( BMP *me*, boolean *buildstrindx*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BMP | |
| buildstrindx | .false | boolean | |
| error | None | integer | |

## writetofile()

### Description

### Prototype

*BMPvar*.writetofile ( BMP *bmp*, string *filename*, integer *bmpbits*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bmp | None | BMP | |
| filename | None | string | |
| bmpbits | None | integer | |
| error | None | integer | |

# BMP_header

## Description

## Type Tags

None

# Object Value

Objects of type BMP_header have no value, and it is an error to try to get or set this value.

# BMP_header.new()

## Description

## Prototype

*BMP_header*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| dataoffset | integer | |
| filesize | integer | |
| id | blob | |
| reserved | integer | |
| type | type | |

# BMP_infoheader

# Description

# Type Tags

None

# Object Value

Objects of type BMP_infoheader have no value, and it is an error to try to get or set this value.

# BMP_infoheader.new()

## Description

## Prototype

*BMP_infoheader*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| bits | integer | |
| colors | integer | |
| compression | integer | |
| imagesize | integer | |
| importantcol-ors | integer | |
| infosize | integer | |
| planes | integer | |
| type | type | |
| xresolution | integer | |
| yresolution | integer | |

# XPM

## Description

## Type Tags

None

## Object Value

Objects of type XPM have no value, and it is an error to try to get or set this value.

## XPM.new()

### Description

### Prototype

*XPM*.new ( XPM *me*, string *filename*, string *name*, integer *maxcolors*, integer *charsperpixel* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | XPM | |
| filename | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | |
| maxcolors | `16777215` | integer | |
| charsperpixel | `1` | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| backcolor | integer | |
| charsperpixel | integer | |
| colors | integer | |
| data | blob | |
| filename | string | |
| getdefbackcol-or | function | |
| height | integer | |
| id | blob | |
| name | string | |
| palette | colorpalette | |
| readfromfile | function | |
| rereadpalette | function | |
| type | type | |
| width | integer | |
| writetofile | function | |

# Methods

## getdefbackcolor()

### Description

### Prototype

*XPMvar*.`getdefbackcolor` ( XPM *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | XPM | |

## readfromfile()

### Description

### Prototype

*XPMvar*.readfromfile ( XPM *me*, string *filename*, string *xpmstr*, XPMcolorlist *x11colors*, integer *backgroundcolor*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | XPM | |
| filename | None | string | |
| xpmstr | None | string | |
| x11colors | None | XPMcolorlist | |
| background-color | 12632256 | integer | |
| error | None | integer | |

## rereadpalette()

### Description

### Prototype

*XPMvar*.rereadpalette ( XPM *me*, boolean *buildstrindx*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | XPM | |
| buildstrindx | .true | boolean | |
| error | None | integer | |

## writetofile()

### Description

### Prototype

*XPMvar*.writetofile ( XPM *me*, string *filename*, XPMcolorlist *colorlist*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | XPM | |
| filename | None | string | |
| colorlist | None | XPMcolorlist | |
| error | None | integer | |

# XPMcolorlist

## Description

## Type Tags

None

## Object Value

Objects of type XPMcolorlist have no value, and it is an error to try to get or set this value.

## XPMcolorlist.new()

### Description

### Prototype

*XPMcolorlist*.new ( XPMcolorlist *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | XPMcolorlist | |

## Properties

| Property | Type | Description |
|---|---|---|
| colors | array | |
| type | type | |

# blobtoBMP()

## Description

## Prototype

blobtoBMP ( string *filename*, blob *data*, integer *width*, integer *height*, integer *backcolor*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| data | None | blob | |
| width | None | integer | |
| height | None | integer | |
| backcolor | None | integer | |
| error | None | integer | |

# blobtoBMPfile()

## Description

## Prototype

`blobtoBMPfile` ( string *`filename`*, blob *`data`*, integer *`width`*, integer *`height`*, integer *`backcolor`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| data | None | blob | |
| width | None | integer | |
| height | None | integer | |
| backcolor | None | integer | |

# blobtoXPM()

## Description

## Prototype

`blobtoXPM` ( string *`filename`*, blob *`data`*, integer *`width`*, integer *`height`*, integer *`backcolor`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| data | None | blob | |
| width | None | integer | |
| height | None | integer | |
| backcolor | None | integer | |

# blobtoXPMfile()

## Description

## Prototype

`blobtoXPMfile` ( string *`filename`*, blob *`data`*, integer *`width`*, integer *`height`*, integer *`backcolor`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| data | None | blob | |
| width | None | integer | |
| height | None | integer | |
| backcolor | None | integer | |
| error | None | integer | |

# Chapter 43. INT

IMplements the SBL-compatible INT() function.

# INT()

## Description

Returns the integer value of the number passed. It truncates, it does not round the value. Supplied for compatibility with Superbase BASIC Language.

## Prototype

`INT` ( number *n* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| n | None | number | |

# Chapter 44. iplib

## ipstringtointeger()

### Description

### Prototype

`ipstringtointeger` ( string *s* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# Chapter 45. jpeglib

This library is a starting point for providing JPEG support. Although JPEG support ofr images is built-in, this library shows how to analyze and retrieve informaiton from the image file itself. As an example, it provides a function for retrieving the width and height of a JPEG image.

# getjpegsize()

## Description

## Prototype

`getjpegsize` ( string *`sFilename`*, integer *`iWidth`*, integer *`iHeight`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFilename | None | string | |
| iWidth | None | integer | |
| iHeight | None | integer | |

# smexec_getjpegsize()

## Description

## Prototype

`smexec_getjpegsize` ( string *`sFilename`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFilename | None | string | |

# Chapter 46. json

# JSON_ARRAY

## Description

## Type Tags

JSON_VALUE

## Object Value

Objects of type JSON_ARRAY have no value, and it is an error to try to get or set this value.

# JSON_ARRAY.new()

## Description

## Prototype

*JSON_ARRAY*.new ( JSON_ARRAY *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_ARRAY | |

# Properties

| Property | Type | Description |
|---|---|---|
| elements | array | |
| getjson | function | |
| type | type | |

# Methods

## getjson()

### Description

### Prototype

*JSON_ARRAYvar*.getjson ( JSON_ARRAY *me*, integer *encoding* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_ARRAY | |
| encoding | None | integer | |

# JSON_BOOLEAN

## Description

## Type Tags

JSON_VALUE

## Object Value

Objects of type JSON_BOOLEAN have no value, and it is an error to try to get or set this value.

## JSON_BOOLEAN.new()

### Description

### Prototype

*JSON_BOOLEAN*.new ( JSON_BOOLEAN *me*, boolean *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_BOOLEAN | |
| value | None | boolean | |

## Properties

| Property | Type | Description |
|---|---|---|
| getjson | function | |
| type | type | |
| value | boolean | |

## Methods

### getjson()

#### Description

### Prototype

*JSON_BOOLEANvar*.getjson ( JSON_BOOLEAN *me*, integer *encoding* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_BOOLEAN | |
| encoding | None | integer | |

# JSON_FALSE

## Description

## Type Tags

JSON_VALUE

## Object Value

Objects of type JSON_FALSE have no value, and it is an error to try to get or set this value.

## JSON_FALSE.new()

### Description

### Prototype

*JSON_FALSE*.new ( JSON_FALSE *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_FALSE | |

## Properties

| Property | Type | Description |
|---|---|---|
| getjson | function | |
| type | type | |
| value | boolean | |

## Methods

### getjson()

#### Description

### Prototype

*JSON_FALSEvar*.getjson ( JSON_FALSE *me*, integer *encoding* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_FALSE | |
| encoding | None | integer | |

# JSON_MEMBERS

## Description

## Type Tags

None

## Object Value

Objects of type JSON_MEMBERS have no value, and it is an error to try to get or set this value.

## JSON_MEMBERS.new()

### Description

### Prototype

*JSON_MEMBERS*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| addmember | function | |
| getfirst | function | |
| members | list | |
| type | type | |

## Methods

### addmember()

#### Description

### Prototype

*JSON_MEMBERSvar*.addmember ( JSON_MEMBERS *me*, string *name*, type(JSON_VALUE) *value*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_MEMBERS | |
| name | None | string | |
| value | None | type(JSON_VALUE) | |
| error | None | integer | |

## getfirst()

### Description

### Prototype

*JSON_MEMBERSvar*.getfirst ( JSON_MEMBERS *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_MEMBERS | |

# JSON_NULL

# Description

# Type Tags

JSON_VALUE

# Object Value

Objects of type JSON_NULL have no value, and it is an error to try to get or set this value.

# JSON_NULL.new()

## Description

## Prototype

*JSON_NULL*.new ( JSON_NULL *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_NULL | |

## Properties

| Property | Type | Description |
|---|---|---|
| getjson | function | |
| type | type | |
| value | boolean | |

## Methods

### getjson()

#### Description

#### Prototype

*JSON_NULLvar*.getjson ( JSON_NULL *me*, integer *encoding* )

#### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_NULL | |
| encoding | None | integer | |

# JSON_NUMBER

## Description

## Type Tags

JSON_VALUE

## Object Value

Objects of type JSON_NUMBER have no value, and it is an error to try to get or set this value.

## JSON_NUMBER.new()

### Description

### Prototype

*JSON_NUMBER*.new ( JSON_NUMBER *me*, number *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_NUMBER | |
| value | None | number | |

## Properties

| Property | Type | Description |
|---|---|---|
| getjson | function | |
| type | type | |
| value | number | |

## Methods

### getjson()

#### Description

#### Prototype

*JSON_NUMBERvar*.getjson ( JSON_NUMBER *me*, integer *encoding* )

#### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_NUMBER | |
| encoding | None | integer | |

# JSON_OBJECT

## Description

## Type Tags

JSON_VALUE

## Object Value

Objects of type JSON_OBJECT have no value, and it is an error to try to get or set this value.

# JSON_OBJECT.new()

## Description

## Prototype

*JSON_OBJECT*.new ( JSON_OBJECT *me*, boolean *isroot*, integer *encoding* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_OBJECT | |
| isroot | .false | boolean | |
| encoding | 0 | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| encoding | integer | |
| getjson | function | |
| isroot | boolean | |
| members | JSON_MEMBERS | |
| type | type | |

# Methods

## getjson()

### Description

### Prototype

*JSON_OBJECTvar*.getjson ( JSON_OBJECT *me*, integer *encoding* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_OBJECT | |
| encoding | None | integer | |

# JSON_PAIR

## Description

# Type Tags

None

# Object Value

Objects of type JSON_PAIR have no value, and it is an error to try to get or set this value.

# JSON_PAIR.new()

## Description

## Prototype

*JSON_PAIR*.new ( JSON_PAIR *me*, JSON_MEMBERS *jsonmembers*, string *name*, type(JSON_VALUE) *value*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_PAIR | |
| jsonmembers | None | JSON_MEMBERS | |
| name | None | string | |
| value | None | type(JSON_VALUE) | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| getnext | function | |
| name | string | |
| node | listnode | |
| type | type | |
| value | type(JSON_VALUE) | |

# Methods

## getnext()

### Description

### Prototype

*JSON_PAIRvar*.getnext ( JSON_PAIR *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_PAIR | |

# JSON_STRING

## Description

## Type Tags

JSON_VALUE

## Object Value

Objects of type JSON_STRING have no value, and it is an error to try to get or set this value.

## JSON_STRING.new()

### Description

### Prototype

*JSON_STRING*.new ( JSON_STRING *me*, blob *value*, integer *encoding* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_STRING | |
| value | None | blob | |
| encoding | 0 | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| assignsimpol-string | function | |
| encoding | integer | |
| getjson | function | |
| getsimpolstring | function | |
| type | type | |
| value | blob | |

# Methods

## assignsimpolstring()

### Description

### Prototype

*JSON_STRINGvar*.assignsimpolstring ( JSON_STRING *me*, string *s* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_STRING | |
| s | None | string | |

## getjson()

### Description

### Prototype

*JSON_STRINGvar*.getjson ( JSON_STRING *me*, integer *encoding* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_STRING | |
| encoding | None | integer | |

## getsimpolstring()

### Description

### Prototype

*JSON_STRINGvar*.getsimpolstring ( JSON_STRING *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_STRING | |

# JSON_TRUE

## Description

# Type Tags

JSON_VALUE

# Object Value

Objects of type JSON_TRUE have no value, and it is an error to try to get or set this value.

# JSON_TRUE.new()

## Description

## Prototype

*JSON_TRUE*.new ( JSON_TRUE *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_TRUE | |

# Properties

| Property | Type | Description |
|---|---|---|
| getjson | function | |
| type | type | |
| value | boolean | |

# Methods

## getjson()

### Description

### Prototype

*JSON_TRUEvar*.getjson ( JSON_TRUE *me*, integer *encoding* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | JSON_TRUE | |
| encoding | None | integer | |

# db1recordtojson()

## Description

## Prototype

`db1recordtojson` ( type(db1record) *`r`*, integer *`encoding`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |
| encoding | None | integer | |

# parsejson()

## Description

## Prototype

`parsejson` ( blob *`b`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| b | None | blob | |
| error | None | integer | |

# simpolstringtojsonblob()

## Description

## Prototype

`simpolstringtojsonblob` ( string *`s`*, integer *`encoding`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| encoding | `0` | integer | |

# simpolstringtojsonformat()

## Description

## Prototype

`simpolstringtojsonformat` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# Chapter 47. labelslib

# labeldef1

## Description

## Type Tags

None

## Object Value

Objects of type labeldef1 have no value, and it is an error to try to get or set this value.

# labeldef1.new()

## Description

## Prototype

*labeldef1*.new ( labeldef1 *me*, printform1 *template*, string *whereclause*, string *orderclause*, string *labelname*, string *labelcode*, integer *countacross*, integer *countdown*, integer *labelwidth*, integer *labelheight*, integer *paperwidth*, integer *paperheight*, integer *leftmargin*, integer *topmargin*, integer *horizgap*, integer *vertgap*, boolean *outputacross*, integer *backgroundrgb*, boolean *loading*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labeldef1 | |
| template | None | printform1 | |
| whereclause | None | string | |
| orderclause | None | string | |
| labelname | `14 Up` | string | |
| labelcode | `Avery L7163` | string | |
| countacross | `2` | integer | |
| countdown | `7` | integer | |
| labelwidth | `99100` | integer | |
| labelheight | `38100` | integer | |
| paperwidth | `210000` | integer | |
| paperheight | `297000` | integer | |
| leftmargin | `4650` | integer | |
| topmargin | `15870` | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| horizgap | `2500` | integer | |
| vertgap | `0` | integer | |
| outputacross | `.true` | boolean | |
| backgroundrgb | `16777215` | integer | |
| loading | `.false` | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| backgroundrgb | integer | |
| clearallcol-lapsiblecon-trols | function | |
| clearcollapsi-blecontrol | function | |
| collapsiblecon-trols | array | |
| countacross | integer | |
| countdown | integer | |
| dirty | boolean | |
| filename | string | |
| horizgap | integer | |
| labelcode | string | |
| labelheight | integer | |
| labelname | string | |
| labelwidth | integer | |
| leftmargin | integer | |
| loading | boolean | |
| orderclause | string | |
| outputacross | boolean | |
| paperheight | integer | |
| paperwidth | integer | |
| setcollapsible-control | function | |
| setprintform | function | |
| template | printform1 | |
| topmargin | integer | |
| type | type | |
| vertgap | integer | |

| Property | Type | Description |
|---|---|---|
| whereclause | string | |

# Methods

## clearallcollapsiblecontrols()

### Description

### Prototype

*labeldef1var*.clearallcollapsiblecontrols ( labeldef1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labeldef1 | |

## clearcollapsiblecontrol()

### Description

### Prototype

*labeldef1var*.clearcollapsiblecontrol ( labeldef1 *me*, string *controlname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labeldef1 | |
| controlname | None | string | |

## setcollapsiblecontrol()

### Description

### Prototype

*labeldef1var*.setcollapsiblecontrol ( labeldef1 *me*, string *controlname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labeldef1 | |
| controlname | None | string | |

## setprintform()

### Description

### Prototype

*labeldef1var*.setprintform ( labeldef1 *me*, printform1 *template*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labeldef1 | |
| template | None | printform1 | |
| error | None | integer | |

# labelinstance

## Description

## Type Tags

None

## Object Value

Objects of type labelinstance have no value, and it is an error to try to get or set this value.

## labelinstance.new()

### Description

### Prototype

*labelinstance*.new ( labelinstance *me*, labelmaker1 *labelmaker*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labelinstance | |
| labelmaker | None | labelmaker1 | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _private | __labelinstanceprivate | |
| controlcounter | integer | |
| controlmap-ping | array | |

| Property | Type | Description |
|---|---|---|
| currleftoffset | integer | |
| currpage | wxprintpage | |
| currpagenum-ber | integer | |
| currtemplate | wxprintpagetemplate | |
| currtopoffset | integer | |
| getemptylabel | function | |
| labeldef | labeldef1 | |
| labelmaker | labelmaker1 | |
| outputlabel | function | |
| printform | printform1 | |
| printout | wxprintout | |
| report | report1 | |
| type | type | |

# labelmaker1

## Description

## Type Tags

None

## Object Value

Objects of type labelmaker1 have no value, and it is an error to try to get or set this value.

## labelmaker1.new()

### Description

### Prototype

*labelmaker1*.new ( labelmaker1 *me*, string *title*, string *defbooleanformat*, string *defintegerformat*, string *defnumberformat*, string *defdateformat*, string *deftimeformat*, string *defdatetimeformat*, SBLlocaledateinfo *datelocale*, SBLNumSettings *numlocale*, dring *datasources*, array *tables*, ppcstype1 *ppcs* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labelmaker1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| title | `Superbase NG Labels Output` | string | |
| defbooleanformat | `T\|F` | string | |
| defintegerformat | `.` | string | |
| defnumberformat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimeformat | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |

# Properties

| Property | Type | Description |
|---|---|---|
| centeroverdisplay | boolean | |
| datasources | dring | |
| datelocale | SBLlocaledateinfo | |
| defbooleanformat | string | |
| defdateformat | string | |
| defdatetimeformat | string | |
| defintegerformat | string | |
| defnumberformat | string | |
| deftimeformat | string | |
| dialogdata | string | |
| gauge | gaugedialog | |
| gdi | GDI | |
| labeldef | labeldef1 | |
| numlocale | SBLNumSettings | |
| openlabeldef | function | |

| Property | Type | Description |
|---|---|---|
| outputtarget | integer | |
| ppcs | ppcstype1 | |
| run | function | |
| savelabeldef | function | |
| setlabeldef | function | |
| startat100percent | boolean | |
| tables | array | |
| title | string | |
| type | type | |
| usegauge | boolean | |
| winspool | WINSPOOL | |

# Methods

## openlabeldef()

### Description

### Prototype

*labelmaker1var*.openlabeldef ( labelmaker1 *me*, string *filename*, integer *paperwidth*, integer *paperheight*, boolean *createdisplayform*, integer *error*, string *errortext* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labelmaker1 | |
| filename | None | string | |
| paperwidth | 210000 | integer | |
| paperheight | 297000 | integer | |
| createdisplay-form | .false | boolean | |
| error | None | integer | |
| errortext | None | string | |

## run()

### Description

### Prototype

*labelmaker1var*.run ( labelmaker1 *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labelmaker1 | |
| error | None | integer | |

## savelabeldef()

### Description

### Prototype

*labelmaker1var*.savelabeldef ( labelmaker1 *me*, string *filename*, integer *tabsize*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labelmaker1 | |
| filename | None | string | |
| tabsize | 2 | integer | |
| error | None | integer | |

## setlabeldef()

### Description

### Prototype

*labelmaker1var*.setlabeldef ( labelmaker1 *me*, labeldef1 *labeldef*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | labelmaker1 | |
| labeldef | None | labeldef1 | |
| error | None | integer | |

# __labelstransferpagechunktoprintout()

# Description

# Prototype

__labelstransferpagechunktoprintout ( printform1page *prfrmpage*, labelinstance *labelinst* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| prfrmpage | None | printform1page | |
| labelinst | None | labelinstance | |

# openlabeldef()

## Description

## Prototype

`openlabeldef` ( string *`filename`*, integer *`pagewidth`*, integer *`pageheight`*, integer *`defpagebackcolor`*, string *`defbooleanformat`*, string *`defintegerformat`*, string *`defnumberformat`*, string *`defdateformat`*, string *`deftimeformat`*, string *`defdatetimeformat`*, SBLlocaledateinfo *`datelocale`*, SBLNumSettings *`numlocale`*, dring *`datasources`*, array *`tables`*, ppcstype1 *`ppcs`*, boolean *`createdisplayform`*, integer *`error`*, string *`errortext`*, type(wxdialogparent) *`parentwindow`*, array *`messages`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| pagewidth | `210000` | integer | |
| pageheight | `297000` | integer | |
| defpageback-color | `16777215` | integer | |
| defbooleanfor-mat | `T\|F` | string | |
| defintegerfor-mat | `.` | string | |
| defnumberfor-mat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimefor-mat | None | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| createdisplay-form | `.false` | boolean | |
| error | None | integer | |
| errortext | None | string | |
| parentwindow | None | type(wxdialogparent) | |
| messages | None | array | |

# report1_labelsreport_outputrow()

## Description

## Prototype

`report1_labelsreport_outputrow` ( report1 *`report`*, report1inst *`reportinst`*, array *`columns`*, array *`currcolvals`*, boolean *`reading`*, labelinstance *`labelinst`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| columns | None | array | |
| currcolvals | None | array | |
| reading | None | boolean | |
| labelinst | None | labelinstance | |

# report1_labelsreport_reportfooter()

## Description

## Prototype

`report1_labelsreport_reportfooter` ( report1 *`report`*, report1inst *`reportinst`*, labelinstance *`labelinst`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| labelinst | None | labelinstance | |

# report1_labelsreport_reportheader()

## Description

## Prototype

`report1_labelsreport_reportheader` ( report1 *`report`*, report1inst *`reportinst`*, labelinstance *`labelinst`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| labelinst | None | labelinstance | |

# savelabeldef()

## Description

## Prototype

`savelabeldef` ( labeldef1 *`labeldef`*, string *`filename`*, string *`origfilename`*, integer *`tabsize`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| labeldef | None | labeldef1 | |
| filename | None | string | |
| origfilename | None | string | |
| tabsize | 2 | integer | |
| error | None | integer | |

# Chapter 48. logmanager

## LogEntry

### Description

## Type Tags

None

## Object Value

Objects of type LogEntry have no value, and it is an error to try to get or set this value.

## LogEntry.new()

### Description

### Prototype

*`LogEntry`*.new ( LogEntry *`me`*, datetime *`dt`*, string *`message`*, string *`filename`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | LogEntry | |
| dt | None | datetime | |
| message | None | string | |
| filename | None | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| dt | datetime | |
| filename | string | |
| message | string | |
| type | type | |

# LogManager

## Description

# Type Tags

None

# Object Value

Objects of type LogManager have no value, and it is an error to try to get or set this value.

# LogManager.new()

## Description

## Prototype

*LogManager*.new ( LogManager *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | LogManager | |

# Properties

| Property | Type | Description |
|---|---|---|
| endLogProcessing | function | |
| logQueue | queue | |
| onAddEntry | event | |
| procLogAccess | lock1 | |
| processingLogs | boolean | |
| type | type | |
| writeLogEntry | function | |

# Methods

## endLogProcessing()

### Description

### Prototype

*LogManagervar*.endLogProcessing ( LogManager *me*, integer *accessCode* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | LogManager | |
| accessCode | None | integer | |

## writeLogEntry()

### Description

### Prototype

*LogManagervar*.writeLogEntry ( LogManager *me*, string *filename*, string *message*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | LogManager | |
| filename | None | string | |
| message | None | string | |
| error | None | integer | |

# Chapter 49. libxmldom1

## DOMDocument

### Description

### Type Tags

DOMNode

### Object Value

Objects of type DOMDocument have no value, and it is an error to try to get or set this value.

## DOMDocument.new()

### Description

### Prototype

*DOMDocument*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _docprivate | LIBXMLDocumentPrivate | |
| _node | DOMNode | |
| createAttribute | function | |
| createAttributeNS | function | |
| createCDATASection | function | |
| createComment | function | |
| createDocumentFragment | function | |
| createElement | function | |
| createElementNS | function | |
| createEntityReference | function | |

| Property | Type | Description |
|---|---|---|
| createProcessingInstruction | function | |
| createTextNode | function | |
| getDoctype | function | |
| getDocumentElement | function | |
| getDocumentURI | function | |
| getElementById | function | |
| getElementsByTagName | function | |
| getElementsByTagNameNS | function | |
| getImplementation | function | |
| getXmlEncoding | function | |
| getXmlStandalone | function | |
| getXmlVersion | function | |
| importNode | function | |
| saveHTMLFile | function | |
| saveXMLFile | function | |
| setDebugMode | function | |
| setDocumentURI | function | |
| setXmlStandalone | function | |
| setXmlVersion | function | |
| type | type | |
| validate | function | |

# Methods

## createAttribute()

### Description

### Prototype

*DOMDocumentvar*.createAttribute ( DOMDocument *me*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| name | None | string | |

## createAttributeNS()

### Description

### Prototype

*DOMDocumentvar*.createAttributeNS ( DOMDocument *me*, string *namespaceURI*, string *qualifiedName* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| namespaceURI | None | string | |
| qualifiedName | None | string | |

## createCDATASection()

### Description

### Prototype

*DOMDocumentvar*.createCDATASection ( DOMDocument *me*, string *data* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| data | None | string | |

## createComment()

### Description

### Prototype

*DOMDocumentvar*.createComment ( DOMDocument *me*, string *data* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| data | None | string | |

# createDocumentFragment()

## Description

## Prototype

*DOMDocumentvar*.createDocumentFragment ( DOMDocument *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

# createElement()

## Description

## Prototype

*DOMDocumentvar*.createElement ( DOMDocument *me*, string *tagName* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| tagName | None | string | |

# createElementNS()

## Description

## Prototype

*DOMDocumentvar*.createElementNS ( DOMDocument *me*, string *namespaceURI*, string *qualifiedName* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| namespaceURI | None | string | |
| qualifiedName | None | string | |

# createEntityReference()

## Description

## Prototype

*DOMDocumentvar*.createEntityReference ( DOMDocument *me*, string *data* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| data | None | string | |

# createProcessingInstruction()

## Description

## Prototype

*DOMDocumentvar*.createProcessingInstruction ( DOMDocument *me*, string *target*, string *data* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| target | None | string | |
| data | None | string | |

# createTextNode()

## Description

## Prototype

*DOMDocumentvar*.createTextNode ( DOMDocument *me*, string *data* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| data | None | string | |

# getDoctype()

## Description

### Prototype

*DOMDocumentvar*.getDoctype ( DOMDocument *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

## getDocumentElement()

### Description

### Prototype

*DOMDocumentvar*.getDocumentElement ( DOMDocument *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

## getDocumentURI()

### Description

### Prototype

*DOMDocumentvar*.getDocumentURI ( DOMDocument *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

## getElementById()

### Description

### Prototype

*DOMDocumentvar*.getElementById ( DOMDocument *me*, string *elementId* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| elementId | None | string | |

## getElementsByTagName()

### Description

### Prototype

*DOMDocumentvar*.getElementsByTagName ( DOMDocument *me*, string *tagname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| tagname | None | string | |

## getElementsByTagNameNS()

### Description

### Prototype

*DOMDocumentvar*.getElementsByTagNameNS ( DOMDocument *me*, string *namespaceURI*, string *localName* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| namespaceURI | None | string | |
| localName | None | string | |

## getImplementation()

### Description

### Prototype

*DOMDocumentvar*.getImplementation ( DOMDocument *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

# getXmlEncoding()

## Description

## Prototype

*DOMDocumentvar*.getXmlEncoding ( DOMDocument *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

# getXmlStandalone()

## Description

## Prototype

*DOMDocumentvar*.getXmlStandalone ( DOMDocument *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

# getXmlVersion()

## Description

## Prototype

*DOMDocumentvar*.getXmlVersion ( DOMDocument *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |

# importNode()

## Description

## Prototype

*DOMDocumentvar*.importNode ( DOMDocument *me*, type(DOMNode) *importedNode*, boolean *deep* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| importedNode | None | type(DOMNode) | |
| deep | None | boolean | |

## saveHTMLFile()

### Description

### Prototype

*DOMDocumentvar*.saveHTMLFile ( DOMDocument *me*, string *filePath*, boolean *format*, string *encoding* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| filePath | None | string | |
| format | .true | boolean | |
| encoding | None | string | |

## saveXMLFile()

### Description

### Prototype

*DOMDocumentvar*.saveXMLFile ( DOMDocument *me*, string *filePath*, boolean *format*, string *encoding* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| filePath | None | string | |
| format | .true | boolean | |
| encoding | None | string | |

## setDebugMode()

### Description

### Prototype

*DOMDocumentvar*.setDebugMode ( DOMDocument *me*, boolean *debugMode* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| debugMode | None | boolean | |

## setDocumentURI()

### Description

### Prototype

*DOMDocumentvar*.setDocumentURI ( DOMDocument *me*, string *documentURI* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| documentURI | None | string | |

## setXmlStandalone()

### Description

### Prototype

*DOMDocumentvar*.setXmlStandalone ( DOMDocument *me*, boolean *xmlStandalone* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| xmlStandalone | None | boolean | |

## setXmlVersion()

### Description

### Prototype

*DOMDocumentvar*.setXmlVersion ( DOMDocument *me*, string *xmlVersion* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| xmlVersion | None | string | |

## validate()

### Description

### Prototype

*DOMDocumentvar*.validate ( DOMDocument *me*, string *outputInfo* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocument | |
| outputInfo | None | string | |

# DOMImplementation

## Description

## Type Tags

None

## Object Value

Objects of type DOMImplementation have no value, and it is an error to try to get or set this value.

## DOMImplementation.new()

### Description

### Prototype

*DOMImplementation*.new ( DOMImplementation *me*, string *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | string | |

# Properties

| Property | Type | Description |
|---|---|---|
| _implprivate | LIBXMLImplementationPrivate | |
| createDocument | function | |
| createDocumentFromHTMLFile | function | |
| createDocumentFromHTMLString | function | |
| createDocumentFromXMLFile | function | |
| createDocumentFromXMLString | function | |
| createDocumentType | function | |
| getLastMsg | function | |
| getProductInfo | function | |
| hasFeature | function | |
| new | function | |
| type | type | |

# Methods

## createDocument()

### Description

### Prototype

*`DOMImplementationvar`*`.createDocument` ( DOMImplementation *`me`*, string *`namespaceURI`*, string *`qualifiedName`*, DOMDocumentType *`doctype`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| namespaceURI | None | string | |
| qualifiedName | None | string | |
| doctype | None | DOMDocumentType | |

# createDocumentFromHTMLFile()

## Description

## Prototype

*DOMImplementationvar*.createDocumentFromHTMLFile ( DOMImplementation *me*, string *filePath*, boolean *keepBlanks* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |
| filePath | None | string | |
| keepBlanks | .false | boolean | |

# createDocumentFromHTMLString()

## Description

## Prototype

*DOMImplementationvar*.createDocumentFromHTMLString ( DOMImplementation *me*, string *xmlString*, boolean *keepBlanks* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |
| xmlString | None | string | |
| keepBlanks | .false | boolean | |

# createDocumentFromXMLFile()

## Description

## Prototype

*DOMImplementationvar*.createDocumentFromXMLFile ( DOMImplementation *me*, string *filePath*, boolean *keepBlanks* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |
| filePath | None | string | |
| keepBlanks | `.false` | boolean | |

## createDocumentFromXMLString()

### Description

### Prototype

*`DOMImplementationvar`*`.createDocumentFromXMLString` ( DOMImplementation *`me`*, string *`xmlString`*, boolean *`keepBlanks`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |
| xmlString | None | string | |
| keepBlanks | `.false` | boolean | |

## createDocumentType()

### Description

### Prototype

*`DOMImplementationvar`*`.createDocumentType` ( DOMImplementation *`me`*, string *`qualifiedName`*, string *`publicId`*, string *`systemId`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |
| qualifiedName | None | string | |
| publicId | None | string | |
| systemId | None | string | |

## getLastMsg()

### Description

### Prototype

*`DOMImplementationvar`*`.getLastMsg` ( DOMImplementation *`me`*, type(DOMNode) *`node`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |
| node | None | type(DOMNode) | |

## getProductInfo()

### Description

### Prototype

*DOMImplementationvar*.getProductInfo ( DOMImplementation *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |

## hasFeature()

### Description

### Prototype

*DOMImplementationvar*.hasFeature ( DOMImplementation *me*, string *feature*, string *version* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMImplementation | |
| feature | None | string | |
| version | None | string | |

# DOMNode

# Description

# Type Tags

DOMNode

# Object Value

Objects of type DOMNode have no value, and it is an error to try to get or set this value.

# DOMNode.new()

## Description

## Prototype

*DOMNode*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _debugInfo | LIBXMLNodeDebugInfo | |
| _private | LIBXMLNodePrivate | |
| appendChild | function | |
| cloneNode | function | |
| dumpContent | function | |
| getAttributes | function | |
| getChildNodes | function | |
| getFirstChild | function | |
| getLastChild | function | |
| getLocalName | function | |
| getNamespaceURI | function | |
| getNextSibling | function | |
| getNodeName | function | |
| getNodeType | function | |
| getNodeValue | function | |
| getOwnerDocument | function | |
| getParent | function | |
| getPrefix | function | |
| getPreviousSibling | function | |
| hasAttributes | function | |
| hasChildNodes | function | |
| insertBefore | function | |
| isBlank | function | |
| isSameNode | function | |

| Property | Type | Description |
|---|---|---|
| removeChild | function | |
| replaceChild | function | |
| selectFirstN-ode | function | |
| selectNodes | function | |
| setNodeValue | function | |
| type | type | |

# Methods

## appendChild()

### Description

### Prototype

*DOMNodevar*.appendChild ( DOMNode *me*, type(DOMNode) *newChild* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |
| newChild | None | type(DOMNode) | |

## cloneNode()

### Description

### Prototype

*DOMNodevar*.cloneNode ( DOMNode *me*, boolean *deep* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |
| deep | None | boolean | |

## dumpContent()

### Description

### Prototype

*DOMNodevar*.dumpContent ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getAttributes()

### Description

### Prototype

*DOMNodevar*.getAttributes ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getChildNodes()

### Description

### Prototype

*DOMNodevar*.getChildNodes ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getFirstChild()

### Description

### Prototype

*DOMNodevar*.getFirstChild ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getLastChild()

### Description

### Prototype

*DOMNodevar*.getLastChild ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getLocalName()

### Description

### Prototype

*DOMNodevar*.getLocalName ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getNamespaceURI()

### Description

### Prototype

*DOMNodevar*.getNamespaceURI ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getNextSibling()

### Description

### Prototype

*DOMNodevar*.getNextSibling ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

# getNodeName()

## Description

## Prototype

*DOMNodevar*.getNodeName ( DOMNode *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

# getNodeType()

## Description

## Prototype

*DOMNodevar*.getNodeType ( DOMNode *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

# getNodeValue()

## Description

## Prototype

*DOMNodevar*.getNodeValue ( DOMNode *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

# getOwnerDocument()

## Description

## Prototype

*DOMNodevar*.getOwnerDocument ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getParent()

### Description

### Prototype

*DOMNodevar*.getParent ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getPrefix()

### Description

### Prototype

*DOMNodevar*.getPrefix ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## getPreviousSibling()

### Description

### Prototype

*DOMNodevar*.getPreviousSibling ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## hasAttributes()

### Description

### Prototype

*DOMNodevar*.hasAttributes ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## hasChildNodes()

### Description

### Prototype

*DOMNodevar*.hasChildNodes ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

## insertBefore()

### Description

### Prototype

*DOMNodevar*.insertBefore ( DOMNode *me*, type(DOMNode) *newChild*, type(DOMNode) *refChild* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |
| newChild | None | type(DOMNode) | |
| refChild | None | type(DOMNode) | |

## isBlank()

### Description

### Prototype

*DOMNodevar*.isBlank ( DOMNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |

# isSameNode()

## Description

## Prototype

*DOMNodevar*.isSameNode ( DOMNode *me*, type(DOMNode) *node* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |
| node | None | type(DOMNode) | |

# removeChild()

## Description

## Prototype

*DOMNodevar*.removeChild ( DOMNode *me*, type(DOMNode) *oldChild* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |
| oldChild | None | type(DOMNode) | |

# replaceChild()

## Description

## Prototype

*DOMNodevar*.replaceChild ( DOMNode *me*, type(DOMNode) *newChild*, type(DOMNode) *oldChild* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |
| newChild | None | type(DOMNode) | |
| oldChild | None | type(DOMNode) | |

# selectFirstNode()

## Description

### Prototype

*DOMNodevar*.selectFirstNode ( DOMNode *me*, string *query* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |
| query | None | string | |

## selectNodes()

### Description

### Prototype

*DOMNodevar*.selectNodes ( DOMNode *me*, string *query* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |
| query | None | string | |

## setNodeValue()

### Description

### Prototype

*DOMNodevar*.setNodeValue ( DOMNode *me*, string *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNode | |
| value | None | string | |

# LIBXMLNodeDebugInfo

# Description

# Type Tags

None

# Object Value

Objects of type LIBXMLNodeDebugInfo have no value, and it is an error to try to get or set this value.

# LIBXMLNodeDebugInfo.new()

## Description

## Prototype

*LIBXMLNodeDebugInfo*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| name | string | |
| nodetype | integer | |
| type | type | |
| value | string | |

# Chapter 50. libxmldom2

# DOMAttr

## Description

## Type Tags

DOMNode

## Object Value

Objects of type DOMAttr have no value, and it is an error to try to get or set this value.

# DOMAttr.new()

## Description

## Prototype

*DOMAttr*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _node | DOMNode | |
| getName | function | |
| getOwnerElement | function | |
| getSpecified | function | |
| getValue | function | |
| isId | function | |
| setValue | function | |
| type | type | |

# Methods

## getName()

### Description

### Prototype

*DOMAttrvar*.getName ( DOMAttr *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMAttr | |

## getOwnerElement()

### Description

### Prototype

*DOMAttrvar*.getOwnerElement ( DOMAttr *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMAttr | |

## getSpecified()

### Description

### Prototype

*DOMAttrvar*.getSpecified ( DOMAttr *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMAttr | |

## getValue()

### Description

### Prototype

*DOMAttrvar*.getValue ( DOMAttr *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMAttr | |

## isId()

### Description

### Prototype

*DOMAttrvar*.isId ( DOMAttr *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMAttr | |

## setValue()

### Description

### Prototype

*DOMAttrvar*.setValue ( DOMAttr *me*, string *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMAttr | |
| value | None | string | |

# DOMCDATASection

# Description

# Type Tags

DOMNode, DOMCharacterData, DOMText

# Object Value

Objects of type DOMCDATASection have no value, and it is an error to try to get or set this value.

# DOMCDATASection.new()

## Description

## Prototype

*DOMCDATASection*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _text | DOMText | |
| type | type | |

# DOMCharacterData

## Description

## Type Tags

DOMNode

## Object Value

Objects of type DOMCharacterData have no value, and it is an error to try to get or set this value.

## DOMCharacterData.new()

### Description

### Prototype

*`DOMCharacterData`*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _node | DOMNode | |
| appendData | function | |
| deleteData | function | |
| getData | function | |
| getLength | function | |
| insertData | function | |
| replaceData | function | |
| setData | function | |

| Property | Type | Description |
|---|---|---|
| substringData | function | |
| type | type | |

# Methods

## appendData()

### Description

### Prototype

*DOMCharacterDatavar*.appendData ( DOMCharacterData *me*, string *arg* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMCharacterData | |
| arg | None | string | |

## deleteData()

### Description

### Prototype

*DOMCharacterDatavar*.deleteData ( DOMCharacterData *me*, integer *offset*, integer *count* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMCharacterData | |
| offset | None | integer | |
| count | None | integer | |

## getData()

### Description

### Prototype

*DOMCharacterDatavar*.getData ( DOMCharacterData *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMCharacterData | |

# getLength()

## Description

## Prototype

*DOMCharacterDatavar*.getLength ( DOMCharacterData *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMCharacterData | |

# insertData()

## Description

## Prototype

*DOMCharacterDatavar*.insertData ( DOMCharacterData *me*, integer *offset*, string *arg* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMCharacterData | |
| offset | None | integer | |
| arg | None | string | |

# replaceData()

## Description

## Prototype

*DOMCharacterDatavar*.replaceData ( DOMCharacterData *me*, integer *offset*, integer *count*, string *arg* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMCharacterData | |
| offset | None | integer | |
| count | None | integer | |
| arg | None | string | |

# setData()

## Description

### Prototype

*DOMCharacterDatavar*.setData ( DOMCharacterData *me*, string *data* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMCharacterData | |
| data | None | string | |

## substringData()

### Description

### Prototype

*DOMCharacterDatavar*.substringData ( DOMCharacterData *me*, integer *offset*, integer *count* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMCharacterData | |
| offset | None | integer | |
| count | None | integer | |

# DOMComment

# Description

# Type Tags

DOMCharacterData, DOMNode

# Object Value

Objects of type DOMComment have no value, and it is an error to try to get or set this value.

# DOMComment.new()

## Description

## Prototype

*DOMComment*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _characterData | DOMCharacterData | |
| type | type | |

# DOMDocumentFragment

## Description

## Type Tags

DOMNode

## Object Value

Objects of type DOMDocumentFragment have no value, and it is an error to try to get or set this value.

## DOMDocumentFragment.new()

### Description

### Prototype

*DOMDocumentFragment*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _node | DOMNode | |
| type | type | |

# DOMDocumentType

## Description

# Type Tags

DOMNode

# Object Value

Objects of type DOMDocumentType have no value, and it is an error to try to get or set this value.

# DOMDocumentType.new()

## Description

## Prototype

*DOMDocumentType*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _node | DOMNode | |
| getEntities | function | |
| getInternalSub-set | function | |
| getName | function | |
| getNotations | function | |
| getPublicId | function | |
| getSystemId | function | |
| type | type | |

# Methods

## getEntities()

### Description

### Prototype

*DOMDocumentTypevar*.getEntities ( DOMDocumentType *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocumentType | |

# getInternalSubset()

## Description

## Prototype

*DOMDocumentTypevar*.getInternalSubset ( DOMDocumentType *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocumentType | |

# getName()

## Description

## Prototype

*DOMDocumentTypevar*.getName ( DOMDocumentType *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocumentType | |

# getNotations()

## Description

## Prototype

*DOMDocumentTypevar*.getNotations ( DOMDocumentType *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocumentType | |

# getPublicId()

## Description

## Prototype

*DOMDocumentTypevar*.getPublicId ( DOMDocumentType *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocumentType | |

## getSystemId()

### Description

### Prototype

*DOMDocumentTypevar*.getSystemId ( DOMDocumentType *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMDocumentType | |

# DOMElement

## Description

## Type Tags

DOMNode

## Object Value

Objects of type DOMElement have no value, and it is an error to try to get or set this value.

## DOMElement.new()

### Description

### Prototype

*DOMElement*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _node | DOMNode | |

| Property | Type | Description |
|---|---|---|
| getAttribute | function | |
| getAttributeNS | function | |
| getAttributeNode | function | |
| getAttributeNodeNS | function | |
| getElementsByTagName | function | |
| getElementsByTagNameNS | function | |
| getTagName | function | |
| hasAttribute | function | |
| hasAttributeNS | function | |
| removeAttribute | function | |
| removeAttributeNS | function | |
| removeAttributeNode | function | |
| setAttribute | function | |
| setAttributeNS | function | |
| setAttributeNode | function | |
| setAttributeNodeNS | function | |
| type | type | |

# Methods

## getAttribute()

### Description

### Prototype

*DOMElementvar*.getAttribute ( DOMElement *me*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| name | None | string | |

# getAttributeNS()

## Description

## Prototype

*DOMElementvar*.getAttributeNS ( DOMElement *me*, string *namespaceURI*, string *localName* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| namespaceURI | None | string | |
| localName | None | string | |

# getAttributeNode()

## Description

## Prototype

*DOMElementvar*.getAttributeNode ( DOMElement *me*, string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| name | None | string | |

# getAttributeNodeNS()

## Description

## Prototype

*DOMElementvar*.getAttributeNodeNS ( DOMElement *me*, string *namespaceURI*, string *localName* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| namespaceURI | None | string | |
| localName | None | string | |

# getElementsByTagName()

## Description

## Prototype

*DOMElementvar*.getElementsByTagName ( DOMElement *me*, string *tagname* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| tagname | None | string | |

# getElementsByTagNameNS()

## Description

## Prototype

*DOMElementvar*.getElementsByTagNameNS ( DOMElement *me*, string *namespaceURI*, string *localName* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| namespaceURI | None | string | |
| localName | None | string | |

# getTagName()

## Description

## Prototype

*DOMElementvar*.getTagName ( DOMElement *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |

# hasAttribute()

## Description

### Prototype

*DOMElementvar*.hasAttribute ( DOMElement *me*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| name | None | string | |

## hasAttributeNS()

### Description

### Prototype

*DOMElementvar*.hasAttributeNS ( DOMElement *me*, string *namespaceURI*, string *localName* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| namespaceURI | None | string | |
| localName | None | string | |

## removeAttribute()

### Description

### Prototype

*DOMElementvar*.removeAttribute ( DOMElement *me*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| name | None | string | |

## removeAttributeNS()

### Description

### Prototype

*DOMElementvar*.removeAttributeNS ( DOMElement *me*, string *namespaceURI*, string *localName* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| namespaceURI | None | string | |
| localName | None | string | |

## removeAttributeNode()

### Description

### Prototype

*DOMElementvar*.removeAttributeNode ( DOMElement *me*, DOMAttr *oldAttr* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| oldAttr | None | DOMAttr | |

## setAttribute()

### Description

### Prototype

*DOMElementvar*.setAttribute ( DOMElement *me*, string *name*, string *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| name | None | string | |
| value | None | string | |

## setAttributeNS()

### Description

### Prototype

*DOMElementvar*.setAttributeNS ( DOMElement *me*, string *namespaceURI*, string *qualifiedName*, string *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| namespaceURI | None | string | |
| qualifiedName | None | string | |
| value | None | string | |

## setAttributeNode()

### Description

### Prototype

*DOMElementvar*.setAttributeNode ( DOMElement *me*, DOMAttr *newAttr* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| newAttr | None | DOMAttr | |

## setAttributeNodeNS()

### Description

### Prototype

*DOMElementvar*.setAttributeNodeNS ( DOMElement *me*, DOMAttr *newAttr* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMElement | |
| newAttr | None | DOMAttr | |

# DOMEntity

# Description

# Type Tags

DOMNode

# Object Value

Objects of type DOMEntity have no value, and it is an error to try to get or set this value.

# DOMEntity.new()

## Description

## Prototype

*DOMEntity*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _node | DOMNode | |
| getNotation-Name | function | |
| getPublicId | function | |
| getSystemId | function | |
| type | type | |

# Methods

## getNotationName()

### Description

### Prototype

*DOMEntityvar*.getNotationName ( DOMEntity *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMEntity | |

## getPublicId()

### Description

### Prototype

*DOMEntityvar*.getPublicId ( DOMEntity *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMEntity | |

## getSystemId()

### Description

### Prototype

*DOMEntityvar*.getSystemId ( DOMEntity *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMEntity | |

# DOMEntityReference

## Description

## Type Tags

DOMNode

## Object Value

Objects of type DOMEntityReference have no value, and it is an error to try to get or set this value.

## DOMEntityReference.new()

### Description

### Prototype

*DOMEntityReference*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _node | DOMNode | |

| Property | Type | Description |
|---|---|---|
| type | type | |

# DOMNamedNodeMap

## Description

## Type Tags

None

## Object Value

Objects of type DOMNamedNodeMap have no value, and it is an error to try to get or set this value.

## DOMNamedNodeMap.new()

### Description

### Prototype

*DOMNamedNodeMap*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _store | LIBXMLStore | |
| getLength | function | |
| getNamedItem | function | |
| get-NamedItemNS | function | |
| item | function | |
| remove-NamedItem | function | |
| remove-NamedItemNS | function | |
| setNamedItem | function | |
| setNamedItem-NS | function | |

| Property | Type | Description |
|---|---|---|
| type | type | |

# Methods

## getLength()

### Description

### Prototype

*DOMNamedNodeMapvar*.getLength ( DOMNamedNodeMap *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNamedNodeMap | |

## getNamedItem()

### Description

### Prototype

*DOMNamedNodeMapvar*.getNamedItem ( DOMNamedNodeMap *me*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNamedNodeMap | |
| name | None | string | |

## getNamedItemNS()

### Description

### Prototype

*DOMNamedNodeMapvar*.getNamedItemNS ( DOMNamedNodeMap *me*, string *namespaceURI*, string *localName* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNamedNodeMap | |
| namespaceURI | None | string | |
| localName | None | string | |

# item()

## Description

## Prototype

*DOMNamedNodeMapvar*.item ( DOMNamedNodeMap *me*, integer *index* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNamedNodeMap | |
| index | None | integer | |

# removeNamedItem()

## Description

## Prototype

*DOMNamedNodeMapvar*.removeNamedItem ( DOMNamedNodeMap *me*, string *name* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNamedNodeMap | |
| name | None | string | |

# removeNamedItemNS()

## Description

## Prototype

*DOMNamedNodeMapvar*.removeNamedItemNS ( DOMNamedNodeMap *me*, string *namespaceURI*, string *localName* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNamedNodeMap | |
| namespaceURI | None | string | |
| localName | None | string | |

# setNamedItem()

## Description

### Prototype

*DOMNamedNodeMapvar*.setNamedItem ( DOMNamedNodeMap *me*, type(DOMNode) *node* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNamedNodeMap | |
| node | None | type(DOMNode) | |

## setNamedItemNS()

### Description

### Prototype

*DOMNamedNodeMapvar*.setNamedItemNS ( DOMNamedNodeMap *me*, type(DOMNode) *node* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNamedNodeMap | |
| node | None | type(DOMNode) | |

# DOMNodeList

## Description

## Type Tags

None

## Object Value

Objects of type DOMNodeList have no value, and it is an error to try to get or set this value.

## DOMNodeList.new()

### Description

### Prototype

*DOMNodeList*.new ()

### Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _store | LIBXMLStore | |
| getLength | function | |
| item | function | |
| type | type | |

# Methods

## getLength()

### Description

### Prototype

*DOMNodeListvar*.getLength ( DOMNodeList *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNodeList | |

## item()

### Description

### Prototype

*DOMNodeListvar*.item ( DOMNodeList *me*, integer *index* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNodeList | |
| index | None | integer | |

# DOMNotation

# Description

# Type Tags

DOMNode

# Object Value

Objects of type DOMNotation have no value, and it is an error to try to get or set this value.

# DOMNotation.new()

## Description

## Prototype

*DOMNotation*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _node | DOMNode | |
| getPublicId | function | |
| getSystemId | function | |
| type | type | |

# Methods

## getPublicId()

### Description

### Prototype

*DOMNotationvar*.getPublicId ( DOMNotation *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNotation | |

## getSystemId()

### Description

### Prototype

*DOMNotationvar*.getSystemId ( DOMNotation *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMNotation | |

# DOMProcessingInstruction

## Description

## Type Tags

DOMNode

## Object Value

Objects of type DOMProcessingInstruction have no value, and it is an error to try to get or set this value.

## DOMProcessingInstruction.new()

### Description

### Prototype

*DOMProcessingInstruction*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _node | DOMNode | |
| getData | function | |
| getTarget | function | |
| setData | function | |
| type | type | |

## Methods

### getData()

#### Description

### Prototype

*DOMProcessingInstructionvar*.getData ( DOMProcessingInstruction *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMProcessingInstruction | |

## getTarget()

### Description

### Prototype

*DOMProcessingInstructionvar*.getTarget ( DOMProcessingInstruction *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMProcessingInstruction | |

## setData()

### Description

### Prototype

*DOMProcessingInstructionvar*.setData ( DOMProcessingInstruction *me*, string *data* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMProcessingInstruction | |
| data | None | string | |

# DOMText

# Description

# Type Tags

DOMNode, DOMCharacterData

# Object Value

Objects of type DOMText have no value, and it is an error to try to get or set this value.

# DOMText.new()

## Description

## Prototype

*DOMText*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _characterData | DOMCharacterData | |
| splitText | function | |
| type | type | |

# Methods

## splitText()

### Description

### Prototype

*DOMTextvar*.splitText ( DOMText *me*, integer *offset* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | DOMText | |
| offset | None | integer | |

# Chapter 51. libxmlutil

## GetNodeValue()

### Description

### Prototype

`GetNodeValue` ( type(DOMNode) *`nodeRoot`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nodeRoot | None | type(DOMNode) | |

## lxmlDOMIsValidXMLName()

### Description

### Prototype

`lxmlDOMIsValidXMLName` ( string *`name`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| name | None | string | |

## lxmlUTF8Decode()

### Description

### Prototype

`lxmlUTF8Decode` ( blob *`input`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| input | None | blob | |

# lxmlUTF8Encode()

## Description

## Prototype

`lxmlUTF8Encode` ( string *`input`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| input | None | string | |

# lxmlXSLTransformFile()

## Description

## Prototype

`lxmlXSLTransformFile` ( string *`xmlFilePath`*, string *`xslFilePath`*, string *`output-FilePath`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| xmlFilePath | None | string | |
| xslFilePath | None | string | |
| outputFilePath | None | string | |

# Chapter 52. lists

The lists library provides implementations for commonlyh used data structures, such as singly- and doubly-linked lists and rings, a stack and a queue.

# dlist

## Description

## Type Tags

list, dlist

## Object Value

Objects of type dlist have no value, and it is an error to try to get or set this value.

# dlist.new()

### Description

### Prototype

*dlist*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| add | function | |
| clear | function | |
| count | integer | |
| find | function | |
| getfirst | function | |
| getlast | function | |
| head | dlistnode | |
| insert | function | |
| remove | function | |

| Property | Type | Description |
|---|---|---|
| search | function | |
| tail | dlistnode | |
| type | type | |

# Methods

## add()

### Description

### Prototype

*dlistvar*.add ( dlist *me*, dlistnode *t1* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlist | |
| t1 | None | dlistnode | |

## clear()

### Description

### Prototype

*dlistvar*.clear ( dlist *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlist | |

## find()

### Description

### Prototype

*dlistvar*.find ( dlist *me*, dlistnode *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlist | |
| t | None | dlistnode | |

# getfirst()

## Description

## Prototype

*dlistvar*.getfirst ( dlist *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlist | |

# getlast()

## Description

## Prototype

*dlistvar*.getlast ( dlist *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlist | |

# insert()

## Description

## Prototype

*dlistvar*.insert ( dlist *me*, dlistnode *t1*, dlistnode *after* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlist | |
| t1 | None | dlistnode | |
| after | None | dlistnode | |

# remove()

## Description

## Prototype

*dlistvar*.remove ( dlist *me*, dlistnode *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlist | |
| t | None | dlistnode | |

## search()

### Description

### Prototype

*dlistvar*.search ( dlist *me*, type(*) *keyval* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlist | |
| keyval | None | type(*) | |

# dlistnode

# Description

# Type Tags

node, dnode

# Object Value

Objects of type dlistnode have no value, and it is an error to try to get or set this value.

# dlistnode.new()

## Description

## Prototype

*dlistnode*.new ( dlistnode *me*, type(dlist) *parent* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlistnode | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(dlist) | |

# Properties

| Property | Type | Description |
|---|---|---|
| getnext | function | |
| getprev | function | |
| key | type(*) | |
| next | type(node) | |
| parent | type(dlist) | |
| prev | type(dnode) | |
| setnext | function | |
| type | type | |

# Methods

## getnext()

### Description

### Prototype

*dlistnodevar*.getnext ( dlistnode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlistnode | |

## getprev()

### Description

### Prototype

*dlistnodevar*.getprev ( dlistnode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlistnode | |

## setnext()

### Description

### Prototype

*dlistnodevar*.setnext ( dlistnode *me*, dlistnode *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dlistnode | |
| next | None | dlistnode | |

# dring

## Description

## Type Tags

list, dlist

## Object Value

Objects of type dring have no value, and it is an error to try to get or set this value.

## dring.new()

### Description

### Prototype

*dring*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| add | function | |
| clear | function | |
| count | integer | |
| find | function | |
| getfirst | function | |

| Property | Type | Description |
|---|---|---|
| getlast | function | |
| head | dlistnode | |
| insert | function | |
| remove | function | |
| search | function | |
| tail | dlistnode | |
| type | type | |

# Methods

## add()

### Description

### Prototype

*dringvar*.add ( dring *me*, dlistnode *t1* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dring | |
| t1 | None | dlistnode | |

## clear()

### Description

### Prototype

*dringvar*.clear ( dring *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dring | |

## find()

### Description

### Prototype

*dringvar*.find ( dring *me*, dlistnode *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dring | |
| t | None | dlistnode | |

## getfirst()

### Description

### Prototype

*dringvar*.getfirst ( dring *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dring | |

## getlast()

### Description

### Prototype

*dringvar*.getlast ( dring *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dring | |

## insert()

### Description

### Prototype

*dringvar*.insert ( dring *me*, dlistnode *t1*, dlistnode *after* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dring | |
| t1 | None | dlistnode | |
| after | None | dlistnode | |

## remove()

### Description

### Prototype

*dringvar*.remove ( dring *me*, dlistnode *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dring | |
| t | None | dlistnode | |

## search()

### Description

### Prototype

*dringvar*.search ( dring *me*, type(*) *keyval* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | dring | |
| keyval | None | type(*) | |

# list

# Description

# Type Tags

list

# Object Value

Objects of type list have no value, and it is an error to try to get or set this value.

# list.new()

## Description

## Prototype

*list*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| add | function | |
| clear | function | |
| count | integer | |
| find | function | |
| getfirst | function | |
| head | listnode | |
| remove | function | |
| search | function | |
| tail | listnode | |
| type | type | |

# Methods

## add()

### Description

### Prototype

*listvar*.add ( list *me*, listnode *t1* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | list | |
| t1 | None | listnode | |

## clear()

### Description

### Prototype

*listvar*.clear ( list *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | list | |

## find()

### Description

### Prototype

*listvar*.find ( list *me*, listnode *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | list | |
| t | None | listnode | |

## getfirst()

### Description

### Prototype

*listvar*.getfirst ( list *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | list | |

## remove()

### Description

### Prototype

*listvar*.remove ( list *me*, listnode *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | list | |
| t | None | listnode | |

### search()

#### Description

#### Prototype

*listvar*.search ( list *me*, type(*) *keyval* )

#### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | list | |
| keyval | None | type(*) | |

# listnode

## Description

## Type Tags

node

## Object Value

Objects of type listnode have no value, and it is an error to try to get or set this value.

## listnode.new()

### Description

### Prototype

*listnode*.new ( listnode *me*, type(list) *parent* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | listnode | |
| parent | None | type(list) | |

## Properties

| Property | Type | Description |
|---|---|---|
| getnext | function | |
| key | type(*) | |
| next | type(node) | |

| Property | Type | Description |
|---|---|---|
| parent | type(list) | |
| type | type | |

# Methods

## getnext()

### Description

### Prototype

*listnodevar*.getnext ( listnode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | listnode | |

# queue

# Description

# Type Tags

None

# Object Value

Objects of type queue have no value, and it is an error to try to get or set this value.

# queue.new()

## Description

## Prototype

*queue*.new ( queue *me*, integer *max*, boolean *endless* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | queue | |
| max | .inf | integer | |
| endless | .false | boolean | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | queueprivate | |
| add | function | |
| clear | function | |
| endless | boolean | |
| get | function | |
| getcount | function | |
| isempty | function | |
| remove | function | |
| type | type | |

# Methods

## add()

### Description

### Prototype

*queuevar*.add ( queue *me*, type(*) *item* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | queue | |
| item | None | type(*) | |

## clear()

### Description

### Prototype

*queuevar*.clear ( queue *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | queue | |

# get()

## Description

## Prototype

*queuevar*.get ( queue *me*, integer *index* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | queue | |
| index | None | integer | |

# getcount()

## Description

## Prototype

*queuevar*.getcount ( queue *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | queue | |

# isempty()

## Description

## Prototype

*queuevar*.isempty ( queue *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | queue | |

# remove()

## Description

## Prototype

*queuevar*.remove ( queue *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | queue | |

# ring

## Description

## Type Tags

list

## Object Value

Objects of type ring have no value, and it is an error to try to get or set this value.

## ring.new()

### Description

### Prototype

*ring*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| add | function | |
| clear | function | |
| count | integer | |
| find | function | |
| getfirst | function | |
| head | listnode | |
| remove | function | |
| search | function | |
| tail | listnode | |
| type | type | |

# Methods

## add()

### Description

### Prototype

*ringvar*.add ( ring *me*, listnode *t1* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ring | |
| t1 | None | listnode | |

## clear()

### Description

### Prototype

*ringvar*.clear ( ring *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ring | |

## find()

### Description

### Prototype

*ringvar*.find ( ring *me*, listnode *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ring | |
| t | None | listnode | |

## getfirst()

### Description

### Prototype

*ringvar*.getfirst ( ring *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ring | |

## remove()

### Description

### Prototype

*ringvar*.remove ( ring *me*, listnode *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ring | |
| t | None | listnode | |

## search()

### Description

### Prototype

*ringvar*.search ( ring *me*, type(*) *keyval* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ring | |
| keyval | None | type(*) | |

# stack

# Description

# Type Tags

None

# Object Value

Objects of type stack have no value, and it is an error to try to get or set this value.

# stack.new()

## Description

## Prototype

*stack*.new ( stack *me*, integer *max* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | stack | |
| max | .inf | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | stackprivate | |
| clear | function | |
| getcount | function | |
| getnextitem | function | |
| gettopitem | function | |
| isempty | function | |
| pop | function | |
| push | function | |
| type | type | |

# Methods

## clear()

### Description

### Prototype

*stackvar*.clear ( stack *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | stack | |

# getcount()

## Description

## Prototype

*stackvar*.getcount ( stack *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | stack | |

# getnextitem()

## Description

## Prototype

*stackvar*.getnextitem ( stack *me*, type(*) *searchitem* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | stack | |
| searchitem | None | type(*) | |

# gettopitem()

## Description

## Prototype

*stackvar*.gettopitem ( stack *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | stack | |

# isempty()

## Description

## Prototype

*stackvar*.isempty ( stack *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | stack | |

## pop()

### Description

### Prototype

*stackvar*.pop ( stack *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | stack | |

## push()

### Description

### Prototype

*stackvar*.push ( stack *me*, type(*) *item* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | stack | |
| item | None | type(*) | |

# Chapter 53. LTRIM

Implements the SBL-compatible LTRIM function. For a more flexible function see the ltrim() function in the stringlib library.

# LTRIM()

## Description

Returns the string value passed with all leading space characters removed.

## Prototype

`LTRIM` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# Chapter 54. mathlib

The mathlib supplies the various math routines used by certain packages, such as the trigonometric functions: pi(), sin(), cos(), tan(), cosecant(), secant(), cotangetn(), arcsin(), arccos(), and arctan(), as well as the degrees_to_radians() and radians_to_degrees() functions for converting back and forth from these two unit types. It also supplies ceil(), floor(), factorial(), sqrt() abs(), int(), and others.

# abs()

## Description

## Prototype

`abs` ( anyvalue *n* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| n | None | anyvalue | |

# arccos()

## Description

## Prototype

`arccos` ( number *x* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| x | None | number | |

# arcsin()

## Description

## Prototype

`arcsin` ( number *x* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| x | None | number | |

# arctan()

## Description

## Prototype

`arctan` ( number *x* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| x | None | number | |

# arctan2()

## Description

## Prototype

`arctan2` ( number *y*, number *x* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| y | None | number | |
| x | None | number | |

# ceil()

## Description

## Prototype

`ceil` ( number *nValue*, integer *iMultiple* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nValue | None | number | |
| iMultiple | None | integer | |

# cos()

## Description

## Prototype

`cos` ( number *`angle_degrees`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| angle_degrees | None | number | |

# cosecant()

## Description

## Prototype

`cosecant` ( number *`angle_degrees`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| angle_degrees | None | number | |

# cotangent()

## Description

## Prototype

`cotangent` ( number *`angle_degrees`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| angle_degrees | None | number | |

# degrees_to_radians()

## Description

## Prototype

`degrees_to_radians` ( number *`value`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | number | |

# factorial()

## Description

## Prototype

`factorial` ( integer *`x`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| x | None | integer | |

# floor()

## Description

## Prototype

`floor` ( number *`nValue`*, integer *`iMultiple`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nValue | None | number | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| iMultiple | None | integer | |

# int()

## Description

## Prototype

`int` ( anyvalue *`v`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| v | None | anyvalue | |

# intnearest()

## Description

## Prototype

`intnearest` ( integer *`iNum1`*, integer *`iNum2`*, integer *`iNum3`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| iNum1 | None | integer | |
| iNum2 | None | integer | |
| iNum3 | None | integer | |

# pi()

## Description

## Prototype

`pi` ( integer *`places`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| places | `50` | integer | |

# radians_to_degrees()

## Description

## Prototype

`radians_to_degrees` ( number *`value`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | number | |

# raisetopower()

## Description

## Prototype

`raisetopower` ( number *`value`*, integer *`power`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | number | |
| power | None | integer | |

# round()

## Description

## Prototype

`round` ( number *`nValue`*, number *`iMultiple`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nValue | None | number | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| iMultiple | None | number | |

# secant()

## Description

## Prototype

`secant` ( number *`angle_degrees`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| angle_degrees | None | number | |

# sin()

## Description

## Prototype

`sin` ( number *`angle_degrees`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| angle_degrees | None | number | |

# sqrt()

## Description

## Prototype

`sqrt` ( number *`n`*, integer *`accuracy`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| n | None | number | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| accuracy | `14` | integer | |

# tan()

## Description

## Prototype

`tan` ( number *`angle_degrees`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| angle_degrees | None | number | |

# Chapter 55. mrulib

The mrulib provides a viable Most Recently Used (MRU) capability for storing the information in a config file (conflib) and managing a submenu of items.

# MRUList

## Description

## Type Tags

None

## Object Value

Objects of type MRUList have no value, and it is an error to try to get or set this value.

## MRUList.new()

### Description

### Prototype

*`MRUList`*.new ( MRUList *`me`*, string *`sectionname`*, string *`inifilename`*, wxmenu *`rootmenu`*, integer *`maxmenuentries`*, integer *`maxentriestotal`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| sectionname | None | string | |
| inifilename | None | string | |
| rootmenu | None | wxmenu | |
| maxmenuentries | 8 | integer | |
| maxentriestotal | 99 | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | MRUListprivate | |
| additem | function | |

| Property | Type | Description |
|---|---|---|
| count | function | |
| finditem | function | |
| get | function | |
| getlistreference | function | |
| inifilename | string | |
| maxentriestotal | integer | |
| maxmenuen-tries | integer | |
| new | function | |
| onselect | event | |
| readconfig | function | |
| removeitem | function | |
| rootmenu | wxmenu | |
| sectionname | string | |
| setlistreference | function | |
| showdialog | function | |
| type | type | |
| updatemenu | function | |
| writeconfig | function | |

# Methods

## additem()

### Description

### Prototype

*MRUListvar*.additem ( MRUList *me*, string *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| value | None | string | |

## count()

### Description

### Prototype

*MRUListvar*.count ( MRUList *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |

## finditem()

### Description

### Prototype

*MRUListvar*.finditem ( MRUList *me*, string *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| value | None | string | |

## get()

### Description

### Prototype

*MRUListvar*.get ( MRUList *me*, integer *index*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| index | None | integer | |
| error | None | integer | |

## getlistreference()

### Description

### Prototype

*MRUListvar*.getlistreference ( MRUList *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |

# readconfig()

## Description

## Prototype

*MRUListvar*.readconfig ( MRUList *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| error | None | integer | |

# removeitem()

## Description

## Prototype

*MRUListvar*.removeitem ( MRUList *me*, integer *index*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| index | None | integer | |
| error | None | integer | |

# setlistreference()

## Description

## Prototype

*MRUListvar*.setlistreference ( MRUList *me*, dlist *entries* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| entries | None | dlist | |

# showdialog()

## Description

### Prototype

*MRUListvar*.showdialog ( MRUList *me*, type(wxdialogparent) *parent*, string *captiontext* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| parent | None | type(wxdialogparent) | |
| captiontext | None | string | |

## updatemenu()

### Description

### Prototype

*MRUListvar*.updatemenu ( MRUList *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| error | None | integer | |

## writeconfig()

### Description

### Prototype

*MRUListvar*.writeconfig ( MRUList *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | MRUList | |
| error | None | integer | |

# Chapter 56. netinfolib

This library currently only provides the functionality for getting the user name (as was logged onto the OS), for WIndows and Linux.

# w32IPAdapterInfo

## Description

## Type Tags

None

## Object Value

Objects of type w32IPAdapterInfo have no value, and it is an error to try to get or set this value.

# w32IPAdapterInfo.new()

## Description

## Prototype

*w32IPAdapterInfo*.new ()

## Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| AdapterName | string | |
| AdapterType | integer | |
| Address | blob | |
| Description | string | |
| DhcpEnabled | boolean | |
| dhcpserveraddr | string | |
| dhcpserver-mask | string | |
| gatewayaddr | string | |
| gatewaymask | string | |
| havewins | boolean | |

| Property | Type | Description |
|---|---|---|
| ipaddr | string | |
| ipaddrmask | string | |
| primarywinss-rvraddr | string | |
| primarywinss-rvrmask | string | |
| secondary-winssrvraddr | string | |
| secondary-winssrvrmask | string | |
| type | type | |

# getcomputername_win32()

## Description

## Prototype

`getcomputername_win32` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# getusername()

## Description

## Prototype

`getusername` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# getusername_posix()

## Description

## Prototype

`getusername_posix` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# getusername_win32()

## Description

## Prototype

`getusername_win32` ( integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# win32_getadapterinfo()

## Description

## Prototype

`win32_getadapterinfo` ()

## Parameters

None

# Chapter 57. objset

The objset is a full set implementation using binary trees to store set elements in sorted order (UNICODE order). New applications should use fastset instead, since it provides the same functionality only faster.

# objset

## Description

## Type Tags

None

## Object Value

Objects of type objset have no value, and it is an error to try to get or set this value.

# objset.new()

### Description

### Prototype

*objset*.new ( objset *me*, string *name* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objset | |
| name | None | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| addelement | function | |
| count | integer | |
| deleteelement | function | |
| difference | function | |
| distinct | boolean | |
| elements | objsetelementref | |
| findelement-bykey | function | |
| getcount | function | |

| Property | Type | Description |
|---|---|---|
| getfirst | function | |
| intersect | function | |
| name | string | |
| totalcount | integer | |
| type | type | |
| unite | function | |

# Methods

## addelement()

### Description

### Prototype

*objsetvar*.addelement ( objset *me*, string *key*, type(*) *element* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objset | |
| key | None | string | |
| element | None | type(*) | |

## deleteelement()

### Description

### Prototype

*objsetvar*.deleteelement ( objset *me*, objsetelementref *elementref* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objset | |
| elementref | None | objsetelementref | |

## difference()

### Description

### Prototype

*objsetvar*.difference ( objset *me*, objset *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objset | |
| t | None | objset | |

## findelementbykey()

### Description

### Prototype

*objsetvar*.findelementbykey ( objset *me*, string *key* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objset | |
| key | None | string | |

## getcount()

### Description

### Prototype

*objsetvar*.getcount ( objset *me*, string *key* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objset | |
| key | None | string | |

## getfirst()

### Description

### Prototype

*objsetvar*.getfirst ( objset *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objset | |

## intersect()

### Description

### Prototype

*objsetvar*.intersect ( objset *me*, objset *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objset | |
| t | None | objset | |

## unite()

### Description

### Prototype

*objsetvar*.unite ( objset *me*, objset *t* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objset | |
| t | None | objset | |

# objsetelement

# Description

# Type Tags

None

# Object Value

Objects of type objsetelement have no value, and it is an error to try to get or set this value.

# objsetelement.new()

## Description

## Prototype

*objsetelement*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| element | type(*) | |
| getfirst | function | |
| getnext | function | |
| key | string | |
| left | objsetelementref | |
| parent | objsetelementref | |
| refcount | integer | |
| right | objsetelementref | |
| type | type | |

# Methods

## getfirst()

### Description

### Prototype

*objsetelementvar*.getfirst ( objsetelement *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objsetelement | |

## getnext()

### Description

### Prototype

*objsetelementvar*.getnext ( objsetelement *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | objsetelement | |

# objsetelementref

## Description

## Type Tags

None

## Object Value

Objects of type objsetelementref have no value, and it is an error to try to get or set this value.

## objsetelementref.new()

### Description

### Prototype

*objsetelementref*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| t | objsetelement | |
| type | type | |

# Chapter 58. odbcsql1

## odbcsql1

### Description

## Type Tags

sqlq1

## Object Value

Objects of type odbcsql1 have no value, and it is an error to try to get or set this value.

## odbcsql1.new()

### Description

### Prototype

*odbcsql1*.new ( odbcsql1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |

## Properties

| Property | Type | Description |
|---|---|---|
| clearsettings | function | |
| columns | array | |
| connection | odbc1connection | |
| defdisplayformats | array | |
| executed | boolean | |
| findcolumndatatype | function | |
| findcolumnsource | function | |
| getcolumncount | function | |

| Property | Type | Description |
|---|---|---|
| getcolumn-datatype | function | |
| getcolumndis-playformat | function | |
| getcolum-ntableandfield-names | function | |
| getcolumntitle | function | |
| getcolumnval-ue | function | |
| getrow | function | |
| gotallrows | boolean | |
| gotrow | boolean | |
| hastable | function | |
| new | function | |
| orderclause | string | |
| prepare | function | |
| prepared | boolean | |
| rowdata | array | |
| selectclause | string | |
| setcolumndis-playformat | function | |
| setdefaultfor-mats | function | |
| setodbcconnec-tion | function | |
| setorderclause | function | |
| setselectclause | function | |
| setwhereclause | function | |
| statement | odbc1statement | |
| type | type | |
| whereclause | string | |

# Methods

## clearsettings()

### Description

### Prototype

*odbcsql1var*.clearsettings ( odbcsql1 *me*, boolean *gotrow*, boolean *gotallrows*, boolean *prepared*, boolean *executed* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| gotrow | `.false` | boolean | |
| gotallrows | `.false` | boolean | |
| prepared | `.false` | boolean | |
| executed | `.false` | boolean | |

## findcolumndatatype()

### Description

### Prototype

*`odbcsql1var`*`.findcolumndatatype` ( odbcsql1 *`me`*, integer *`colno`*, string *`errormessage`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| colno | None | integer | |
| errormessage | None | string | |

## findcolumnsource()

### Description

### Prototype

*`odbcsql1var`*`.findcolumnsource` ( odbcsql1 *`me`*, integer *`colno`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| colno | None | integer | |

## getcolumncount()

### Description

### Prototype

*`odbcsql1var`*`.getcolumncount` ( odbcsql1 *`me`*, string *`errormessage`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| errormessage | None | string | |

## getcolumndatatype()

### Description

### Prototype

*odbcsql1var*.getcolumndatatype ( odbcsql1 *me*, integer *colno*, string *errormessage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| colno | None | integer | |
| errormessage | None | string | |

## getcolumndisplayformat()

### Description

### Prototype

*odbcsql1var*.getcolumndisplayformat ( odbcsql1 *me*, integer *colno*, string *errormessage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| colno | None | integer | |
| errormessage | None | string | |

## getcolumntableandfieldnames()

### Description

### Prototype

*odbcsql1var*.getcolumntableandfieldnames ( odbcsql1 *me*, integer *colno*, string *tablename*, string *fieldname*, string *errormessage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| colno | None | integer | |
| tablename | None | string | |
| fieldname | None | string | |
| errormessage | None | string | |

## getcolumntitle()

### Description

### Prototype

*odbcsql1var*.getcolumntitle ( odbcsql1 *me*, integer *colno*, string *errormessage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| colno | None | integer | |
| errormessage | None | string | |

## getcolumnvalue()

### Description

### Prototype

*odbcsql1var*.getcolumnvalue ( odbcsql1 *me*, integer *colno*, string *errormessage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| colno | None | integer | |
| errormessage | None | string | |

## getrow()

### Description

### Prototype

*odbcsql1var*.getrow ( odbcsql1 *me*, string *errormessage*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| errormessage | None | string | |
| error | None | integer | |

## hastable()

### Description

### Prototype

*odbcsql1var*.hastable ( odbcsql1 *me*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| tablename | None | string | |

## prepare()

### Description

### Prototype

*odbcsql1var*.prepare ( odbcsql1 *me*, string *errormessage*, integer *errorindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| errormessage | None | string | |
| errorindex | None | integer | |

## setcolumndisplayformat()

### Description

### Prototype

*odbcsql1var*.setcolumndisplayformat ( odbcsql1 *me*, integer *colno*, string *displayformat*, string *errormessage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| colno | None | integer | |
| displayformat | None | string | |
| errormessage | None | string | |

# setdefaultformats()

## Description

## Prototype

*odbcsql1var*.setdefaultformats ( odbcsql1 *me*, string *defnumberformat*, string *defdateformat*, string *deftimeformat*, string *defdatetimeformat*, string *defintegerformat*, string *defbooleanformat* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| defnumberformat | None | string | |
| defdateformat | None | string | |
| deftimeformat | None | string | |
| defdatetimeformat | None | string | |
| defintegerformat | None | string | |
| defbooleanformat | None | string | |

# setodbcconnection()

## Description

## Prototype

*odbcsql1var*.setodbcconnection ( odbcsql1 *me*, odbc1connection *connection* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| connection | None | odbc1connection | |

# setorderclause()

## Description

### Prototype

*odbcsql1var*.setorderclause ( odbcsql1 *me*, string *orderclause* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| orderclause | None | string | |

## setselectclause()

### Description

### Prototype

*odbcsql1var*.setselectclause ( odbcsql1 *me*, string *selectclause*, string *errormessage*, integer *errorindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| selectclause | None | string | |
| errormessage | None | string | |
| errorindex | None | integer | |

## setwhereclause()

### Description

### Prototype

*odbcsql1var*.setwhereclause ( odbcsql1 *me*, string *whereclause*, string *errormessage*, integer *errorindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | odbcsql1 | |
| whereclause | None | string | |
| errormessage | None | string | |
| errorindex | None | integer | |

# Chapter 59. PAD

Implements the SBL-compatible PAD() function, and adds the LPAD function for left padding.

# LPAD()

## Description

This function inserts space characters to the left of the string value passed until it is the length passed in the *iLength*. If the length of the text passed in *s* is longer than the value passed in *iLength*, then the text will be truncated from the right to the desired length. The result is returned to the calling function.

## Prototype

LPAD ( string *s*, integer *iLength* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| iLength | None | integer | |

# PAD()

## Description

This function inserts space characters to the right of the string value passed until it is the length passed in the *iLength*. If the length of the text passed in *s* is longer than the value passed in *iLength*, then the text will be reduced from the right side until it is of the desired length. The result is returned to the calling function.

## Prototype

PAD ( string *s*, integer *iLength* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| iLength | None | integer | |

# Chapter 60. parsenum

## parsenum

### Description

### Type Tags

None

### Object Value

Objects of type parsenum have no value, and it is an error to try to get or set this value.

## parsenum.new()

### Description

### Prototype

*parsenum*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| nTagval | number | |
| sFront | string | |
| sOurval | string | |
| type | type | |

# NumberInWords()

## Description

## Prototype

NumberInWords ( number *nAmt* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nAmt | None | number | |

# Chapter 61. printformlib

This library provides a number of types and functions for creating printouts and for converting a form to a printout.

# pagesetupinfo

## Description

## Type Tags

None

## Object Value

Objects of type pagesetupinfo have no value, and it is an error to try to get or set this value.

# pagesetupinfo.new()

### Description

### Prototype

*pagesetupinfo*.new ( pagesetupinfo *me*, integer *papertype*, integer *units*, boolean *isportrait*, wxfont *font* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | pagesetupinfo | |
| papertype | 4 | integer | |
| units | 1 | integer | |
| isportrait | .true | boolean | |
| font | None | wxfont | |

## Properties

| Property | Type | Description |
|---|---|---|
| ConvertFromUnitsToMicrometers | function | |
| ConvertToUnitsFromMicrometers | function | |
| FindPaperType | function | |

| Property | Type | Description |
|---|---|---|
| SetFont | function | |
| SetMargins | function | |
| SetOrientation | function | |
| SetPaperType | function | |
| SetUnits | function | |
| bottommargin | integer | |
| font | wxfont | |
| leftmargin | integer | |
| pageheight | integer | |
| pagewidth | integer | |
| papertype | integer | |
| papertypes | array | |
| portrait | boolean | |
| rightmargin | integer | |
| topmargin | integer | |
| type | type | |
| units_id | integer | |

# Methods

## ConvertFromUnitsToMicrometers()

### Description

### Prototype

*pagesetupinfovar*.ConvertFromUnitsToMicrometers ( pagesetupinfo *me*, number *value*, integer *unitsid* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | pagesetupinfo | |
| value | None | number | |
| unitsid | None | integer | |

## ConvertToUnitsFromMicrometers()

### Description

### Prototype

*pagesetupinfovar*.ConvertToUnitsFromMicrometers ( pagesetupinfo *me*, integer *value*, integer *unitsid* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | pagesetupinfo | |
| value | None | integer | |
| unitsid | None | integer | |

## FindPaperType()

### Description

### Prototype

*pagesetupinfovar*.FindPaperType ( pagesetupinfo *me*, integer *width*, integer *height*, string *papername*, boolean *uselandscape* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | pagesetupinfo | |
| width | None | integer | |
| height | None | integer | |
| papername | None | string | |
| uselandscape | None | boolean | |

## SetFont()

### Description

### Prototype

*pagesetupinfovar*.SetFont ( pagesetupinfo *me*, wxfont *font*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | pagesetupinfo | |
| font | None | wxfont | |
| error | None | integer | |

## SetMargins()

### Description

### Prototype

*pagesetupinfovar*.SetMargins ( pagesetupinfo *me*, number *left*, number *top*, number *right*, number *bottom* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | pagesetupinfo | |
| left | None | number | |
| top | None | number | |
| right | None | number | |
| bottom | None | number | |

## SetOrientation()

### Description

### Prototype

*pagesetupinfovar*.SetOrientation ( pagesetupinfo *me*, boolean *portrait*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | pagesetupinfo | |
| portrait | .true | boolean | |
| error | None | integer | |

## SetPaperType()

### Description

### Prototype

*pagesetupinfovar*.SetPaperType ( pagesetupinfo *me*, integer *papertype*, integer *pagewidth*, integer *pageheight*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | pagesetupinfo | |
| papertype | None | integer | |
| pagewidth | None | integer | |
| pageheight | None | integer | |
| error | None | integer | |

## SetUnits()

### Description

### Prototype

*pagesetupinfovar*.SetUnits ( pagesetupinfo *me*, integer *units_id*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | pagesetupinfo | |
| units_id | None | integer | |
| error | None | integer | |

# printformparams

## Description

## Type Tags

None

## Object Value

Objects of type printformparams have no value, and it is an error to try to get or set this value.

## printformparams.new()

### Description

### Prototype

*printformparams*.new ( printformparams *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printformparams | |

## Properties

| Property | Type | Description |
|---|---|---|
| background | boolean | |
| baseparams | printoutparams | |
| buttons | boolean | |
| combotext | boolean | |

| Property | Type | Description |
|---|---|---|
| edittext | boolean | |
| graphics | boolean | |
| images | boolean | |
| labels | boolean | |
| printcombo-textbackground | boolean | |
| printedit-textbackground | boolean | |
| printlabelback-ground | boolean | |
| showposhelp | boolean | |
| type | type | |
| usedpi | integer | |

# printoutparams

## Description

## Type Tags

None

## Object Value

Objects of type printoutparams have no value, and it is an error to try to get or set this value.

## printoutparams.new()

### Description

### Prototype

*printoutparams*.new ( printoutparams *me*, boolean *showdialog*, string *title*, boolean *print-preview*, string *dialogdata* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | printoutparams | |
| showdialog | .true | boolean | |
| title | Printing | string | |
| printpreview | .false | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dialogdata | None | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| dialogdata | string | |
| font | wxfont | |
| printpreview | boolean | |
| showdialog | boolean | |
| title | string | |
| type | type | |

# getpapertypefromwindowsid()

## Description

Given the Windows paper ID, this function returns the wxPaper ID that matches.

## Prototype

`getpapertypefromwindowsid` ( integer *`id`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| id | None | integer | |

# pagesetup()

## Description

This displays a dialog box where the user can select their desired page setup information, including paper type, paper width and height, left, top, right and bottom margins, page orientation, and font.

## Prototype

`pagesetup` ( type(wxdialogparent) *`parent`*, pagesetupinfo *`psinfo`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| psinfo | None | pagesetupinfo | |

# papersizeinfo()

## Description

## Prototype

`papersizeinfo` ()

## Parameters

None

# printrecord()

## Description

## Prototype

`printrecord` ( type(db1record) *`r`*, array *`formatstrings`*, pagesetupinfo *`psinfo`*, printoutparams *`params`*, SBLlocaledateinfo *`datelocale`*, SBLNumSettings *`numlocale`*, wxprintout *`printout`*, integer *`pageno`*, boolean *`closeprintout`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| r | None | type(db1record) | |
| formatstrings | None | array | |
| psinfo | None | pagesetupinfo | |
| params | None | printoutparams | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| printout | None | wxprintout | |
| pageno | `1` | integer | |
| closeprintout | `.true` | boolean | |
| error | None | integer | |

# printtext()

## Description

## Prototype

`printtext` ( string *`s`*, pagesetupinfo *`psinfo`*, printoutparams *`params`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| psinfo | None | pagesetupinfo | |
| params | None | printoutparams | |
| error | None | integer | |

# printwxform()

## Description

## Prototype

`printwxform` ( wxform *`f`*, printformparams *`params`*, pagesetupinfo *`psinfo`*, wxprintout *`printout`*, integer *`pageno`*, boolean *`closeprintout`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| f | None | wxform | |
| params | None | printformparams | |
| psinfo | None | pagesetupinfo | |
| printout | None | wxprintout | |
| pageno | `1` | integer | |
| closeprintout | `.true` | boolean | |
| error | None | integer | |

# rendertext()

## Description

## Prototype

`rendertext` ( wxprintout *`po`*, integer *`pageno`*, string *`s`*, pagesetupinfo *`psinfo`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| po | None | wxprintout | |
| pageno | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| psinfo | None | pagesetupinfo | |
| error | None | integer | |

# renderwxform()

## Description

## Prototype

`renderwxform` ( wxprintout *`po`*, wxform *`f`*, printformparams *`params`*, pagesetupinfo *`psinfo`*, string *`pagename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| po | None | wxprintout | |
| f | None | wxform | |
| params | None | printformparams | |
| psinfo | None | pagesetupinfo | |
| pagename | `page1` | string | |
| error | None | integer | |

# updatewxdialogdata()

## Description

## Prototype

`updatewxdialogdata` ( string *`dialogdata`*, pagesetupinfo *`psinfo`*, integer *`papertype`*, boolean *`isportrait`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dialogdata | None | string | |
| psinfo | None | pagesetupinfo | |
| papertype | None | integer | |
| isportrait | `.true` | boolean | |

# Chapter 62. quickreportlib

## quickreport1

### Description

### Type Tags

report1

### Object Value

Objects of type quickreport1 have no value, and it is an error to try to get or set this value.

## quickreport1.new()

### Description

### Prototype

*quickreport1*.new ( quickreport1 *me*, integer *paperwidth*, integer *paperheight*, integer *marginleft*, integer *margintop*, integer *marginright*, integer *marginbottom*, integer *outputtarget*, boolean *showpagenumber*, boolean *showdate*, boolean *showtitle*, string *title*, wxfont *pagefont*, string *defbooleanformat*, string *defintegerformat*, string *defnumberformat*, string *defdateformat*, string *deftimeformat*, string *defdatetimeformat*, SBLocaledateinfo *datelocale*, SBLNumSettings *numlocale*, pagesetupinfo *psinfo*, boolean *loading*, type(sqlq1) *query*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| paperwidth | 210000 | integer | |
| paperheight | 297000 | integer | |
| marginleft | 20000 | integer | |
| margintop | 12800 | integer | |
| marginright | 20000 | integer | |
| marginbottom | 12800 | integer | |
| outputtarget | 1 | integer | |
| showpagenumber | .false | boolean | |
| showdate | .false | boolean | |
| showtitle | .false | boolean | |
| title | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| pagefont | None | wxfont | |
| defbooleanformat | `T\|F` | string | |
| defintegerformat | `.` | string | |
| defnumberformat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimeformat | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| psinfo | None | pagesetupinfo | |
| loading | `.false` | boolean | |
| query | None | type(sqlq1) | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| addaggregate | function | |
| addcolumninfo | function | |
| adddatasource | function | |
| addgroup | function | |
| addtable | function | |
| bolditalicfont | wxfont | |
| bolditalicunderlinefont | wxfont | |
| boldunderlinefont | wxfont | |
| centeroverdisplay | boolean | |
| clipoutput | array | |
| columninfo | dring | |
| columns | array | |
| csvexportconverter | dbCSVExport | |
| currpage | wxprintpage | |
| currpagenumber | integer | |

| Property | Type | Description |
|---|---|---|
| currrownumber | integer | |
| currtemplate | wxprintpagetemplate | |
| currtopofpage | integer | |
| datasources | dring | |
| defbooleanfor-mat | string | |
| defdateformat | string | |
| defdatetimefor-mat | string | |
| defintegerfor-mat | string | |
| defnumberfor-mat | string | |
| deftimeformat | string | |
| dialogdata | string | |
| dirty | boolean | |
| dpix | integer | |
| dpiy | integer | |
| filename | string | |
| finddatasource | function | |
| findtable | function | |
| fontheightratio | number | |
| fontwidthratio | number | |
| footerlinecount | integer | |
| fpo | fsfileoutputstream | |
| gauge | gaugedialog | |
| gdi | GDI | |
| getcolumnin-fobycolno | function | |
| getprinttextex-tent | function | |
| getwrapheight | function | |
| getwrapheight2 | function | |
| headerfont | wxfont | |
| headeroutput | boolean | |
| italicfont | wxfont | |
| italicunderline-font | wxfont | |
| lastreportedpa-genumber | integer | |

| Property | Type | Description |
|---|---|---|
| loading | boolean | |
| locale | localeinfo | |
| marginbottom | integer | |
| marginleft | integer | |
| marginright | integer | |
| margintop | integer | |
| onaftergroup | event | |
| onafterrow | event | |
| onbeforegroup | event | |
| onbeforerow | event | |
| onoutputfooter | event | |
| onoutputheader | event | |
| onoutputre-portfooter | event | |
| onoutputre-portheader | event | |
| onpagechange | event | |
| origfont | wxfont | |
| outputextraline | function | |
| outputfilename | string | |
| outputrowodd | boolean | |
| outputtarget | integer | |
| pagefont | wxfont | |
| paperheight | integer | |
| paperwidth | integer | |
| printout | wxprintout | |
| psinfo | pagesetupinfo | |
| qrimportcon-verter | dbQRImport | |
| removedata-source | function | |
| removetable | function | |
| report | report1 | |
| reportheader-output | boolean | |
| rowheight | integer | |
| rowheightad-justment | number | |
| run | function | |

| Property | Type | Description |
|---|---|---|
| sbmeexport-converter | dbSBMEExport | |
| setfont | function | |
| setrowheigh-tadjustment | function | |
| showdate | boolean | |
| showpagenum-ber | boolean | |
| showprinter-dialog | boolean | |
| showtitle | boolean | |
| startat100percent | boolean | |
| stylefilename | string | |
| suppressout-putmessages | boolean | |
| suppress-rowoutput | boolean | |
| tables | dring | |
| targettable-name | string | |
| tbodyoutput | boolean | |
| title | string | |
| type | type | |
| underlinefont | wxfont | |
| usegauge | boolean | |
| usewrapheight2 | boolean | |
| valid | boolean | |
| winspool | WINSPOOL | |
| wrapcharcoun-tkludgevalue | number | |

# Methods

## addaggregate()

### Description

### Prototype

*`quickreport1var`*`.addaggregate` ( quickreport1 *`me`*, report1group *`group`*, integer *`aggregate-type`*, integer *`colno`*, type *`datatype`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| group | None | report1group | |
| aggregatetype | None | integer | |
| colno | None | integer | |
| datatype | None | type | |
| error | None | integer | |

## addcolumninfo()

### Description

### Prototype

*quickreport1var*.addcolumninfo ( quickreport1 *me*, integer *columnstart*, integer *columnwidth*, string *alignment*, boolean *wrap*, string *displayformat*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| columnstart | None | integer | |
| columnwidth | None | integer | |
| alignment | `left,top` | string | |
| wrap | `.false` | boolean | |
| displayformat | None | string | |
| error | None | integer | |

## adddatasource()

### Description

### Prototype

*quickreport1var*.adddatasource ( quickreport1 *me*, type *sourcetype*, string *source*, type(*) *datasource*, string *username*, string *password*, integer *codepage*, integer *retry*, integer *timeout*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sourcetype | None | type | |
| source | None | string | |
| datasource | None | type(*) | |
| username | None | string | |
| password | None | string | |
| codepage | None | integer | |
| retry | `1000000` | integer | |
| timeout | `5000000` | integer | |
| error | None | integer | |

# addgroup()

## Description

## Prototype

*`quickreport1var`*`.addgroup` ( quickreport1 *`me`*, string *`name`*, integer *`colno`*, type *`datatype`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| name | None | string | |
| colno | None | integer | |
| datatype | None | type | |
| error | None | integer | |

# addtable()

## Description

## Prototype

*`quickreport1var`*`.addtable` ( quickreport1 *`me`*, type(db1table) *`table`*, quickreport1datasource *`source`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| table | None | type(db1table) | |
| source | None | quickreport1datasource | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# finddatasource()

## Description

## Prototype

*quickreport1var*.finddatasource ( quickreport1 *me*, string *sourcename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| sourcename | None | string | |

# findtable()

## Description

## Prototype

*quickreport1var*.findtable ( quickreport1 *me*, string *tablename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| tablename | None | string | |

# getcolumninfobycolno()

## Description

## Prototype

*quickreport1var*.getcolumninfobycolno ( quickreport1 *me*, integer *colno*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| colno | None | integer | |
| error | None | integer | |

# getprinttextextent()

## Description

## Prototype

*quickreport1var*.getprinttextextent ( quickreport1 *me*, wxfont *font*, string *s*, integer *width*, integer *height*, integer *descent*, integer *extleading*, integer *maxwidth*, integer *charcount* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| font | None | wxfont | |
| s | None | string | |
| width | None | integer | |
| height | None | integer | |
| descent | None | integer | |
| extleading | None | integer | |
| maxwidth | None | integer | |
| charcount | None | integer | |

# getwrapheight()

## Description

## Prototype

*quickreport1var*.getwrapheight ( quickreport1 *me*, wxfont *font*, string *s*, integer *width*, integer *origheight* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| font | None | wxfont | |
| s | None | string | |
| width | None | integer | |
| origheight | None | integer | |

# getwrapheight2()

## Description

### Prototype

*quickreport1var*.getwrapheight2 ( quickreport1 *me*, wxfont *font*, string *s*, integer *width*, integer *origheight* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| font | None | wxfont | |
| s | None | string | |
| width | None | integer | |
| origheight | None | integer | |

## outputextraline()

### Description

### Prototype

*quickreport1var*.outputextraline ( quickreport1 *me*, quickreportextraoutputinfo *outputinfo*, string *fontcharacteristics*, boolean *incrementtopofpage*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| outputinfo | None | quickreportextraoutputinfo | |
| fontcharacteristics | None | string | |
| incrementtopofpage | .true | boolean | |
| error | None | integer | |

## removedatasource()

### Description

### Prototype

*quickreport1var*.removedatasource ( quickreport1 *me*, quickreport1datasource *datasource* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| datasource | None | quickreport1datasource | |

# removetable()

## Description

## Prototype

*quickreport1var*.removetable ( quickreport1 *me*, quickreport1table *table* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| table | None | quickreport1table | |

# run()

## Description

## Prototype

*quickreport1var*.run ( quickreport1 *me*, string *errmsg*, integer *erridx*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| errmsg | None | string | |
| erridx | None | integer | |
| error | None | integer | |

# setfont()

## Description

## Prototype

*quickreport1var*.setfont ( quickreport1 *me*, wxfont *font* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| font | None | wxfont | |

# setrowheightadjustment()

## Description

### Prototype

*quickreport1var*.setrowheightadjustment ( quickreport1 *me*, number *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1 | |
| value | None | number | |

# quickreport1datasource

## Description

## Type Tags

None

## Object Value

Objects of type quickreport1datasource have no value, and it is an error to try to get or set this value.

## quickreport1datasource.new()

### Description

### Prototype

*quickreport1datasource*.new ( quickreport1datasource *me*, quickreport1 *quickreport*, type *sourcetype*, string *source*, type(*) *datasource*, string *username*, string *password*, integer *retry*, integer *timeout*, integer *codepage*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1datasource | |
| quickreport | None | quickreport1 | |
| sourcetype | None | type | |
| source | None | string | |
| datasource | None | type(*) | |
| username | None | string | |
| password | None | string | |
| retry | 1000000 | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| timeout | `5000000` | integer | |
| codepage | None | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| codepage | integer | |
| datasource | type(*) | |
| opentable | function | |
| password | string | |
| quickreport | quickreport1 | |
| reportnode | dlistnode | |
| retry | integer | |
| source | string | |
| sourcetype | type | |
| tables | dring | |
| timeout | integer | |
| type | type | |
| username | string | |

# Methods

## opentable()

### Description

### Prototype

*`quickreport1datasourcevar`*.`opentable` ( quickreport1datasource *`me`*, string *`tablename`*, string *`password`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1datasource | |
| tablename | None | string | |
| password | None | string | |
| error | None | integer | |

# quickreport1table

## Description

## Type Tags

None

## Object Value

Objects of type quickreport1table have no value, and it is an error to try to get or set this value.

## quickreport1table.new()

### Description

### Prototype

*quickreport1table*.new ( quickreport1table *me*, type(db1table) *table*, quickreport1 *quickreport*, dring *parent*, quickreport1datasource *datasource*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1table | |
| table | None | type(db1table) | |
| quickreport | None | quickreport1 | |
| parent | None | dring | |
| datasource | None | quickreport1datasource | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| datasource | quickreport1datasource | |
| datasourcenode | dlistnode | |
| fieldinfo | array | |
| gettablename | function | |
| parentnode | dlistnode | |
| report | quickreport1 | |
| table | type(db1table) | |

| Property | Type | Description |
|---|---|---|
| type | type | |

# Methods

## gettablename()

### Description

### Prototype

*quickreport1tablevar*.gettablename ( quickreport1table *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreport1table | |

# quickreportextraoutputinfo

# Description

# Type Tags

None

# Object Value

Objects of type quickreportextraoutputinfo have no value, and it is an error to try to get or set this value.

# quickreportextraoutputinfo.new()

## Description

## Prototype

*quickreportextraoutputinfo*.new ( quickreportextraoutputinfo *me*, quickreport1 *quickreport*, string *text*, integer *left*, integer *width*, integer *height*, string *alignment*, string *printname*, string *name*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreportextraoutputinfo | |
| quickreport | None | quickreport1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| text | None | string | |
| left | None | integer | |
| width | None | integer | |
| height | None | integer | |
| alignment | `left,top` | string | |
| printname | None | string | |
| name | None | string | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| alignment | string | |
| height | integer | |
| left | integer | |
| name | string | |
| printname | string | |
| quickreport | quickreport1 | |
| text | string | |
| type | type | |
| width | integer | |

# convert_dpi_mcm()

## Description

## Prototype

`convert_dpi_mcm` ( integer *`value`*, integer *`dpi`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | integer | |
| dpi | None | integer | |

# convert_mcm_dpi()

## Description

## Prototype

`convert_mcm_dpi` ( integer *`value`*, integer *`dpi`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | integer | |
| dpi | None | integer | |

# getprinttextextent()

## Description

## Prototype

`getprinttextextent` ( wxfont *`font`*, string *`s`*, integer *`width`*, integer *`height`*, integer *`descent`*, integer *`extleading`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| font | None | wxfont | |
| s | None | string | |
| width | None | integer | |
| height | None | integer | |
| descent | None | integer | |
| extleading | None | integer | |

# loadquickreport()

## Description

This function is called to load a Quick Report that has been stored on disk. To make use of already opened tables, pass the *`datasources`* dring and the *`tables`* array.

## Prototype

`loadquickreport` ( string *`filename`*, integer *`pagewidth`*, integer *`pageheight`*, integer *`defpagebackcolor`*, string *`defbooleanformat`*, string *`defintegerformat`*, string *`defnumberformat`*, string *`defdateformat`*, string *`deftimeformat`*, string *`defdatetimeformat`*, SBLlocaledateinfo *`datelocale`*, SBLNumSettings *`numlocale`*, dring *`datasources`*, array *`tables`*, ppcstype1 *`ppcs`*, integer *`error`*, string *`errortext`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| pagewidth | `210000` | integer | |
| pageheight | `297000` | integer | |
| defpageback-color | `16777215` | integer | |
| defbooleanfor-mat | `T\|F` | string | |
| defintegerfor-mat | `.` | string | |
| defnumberfor-mat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimefor-mat | None | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| error | None | integer | |
| errortext | None | string | |

# report1_quickreport_output_groupfooter()

## Description

## Prototype

`report1_quickreport_output_groupfooter` ( report1group *`group`*, report1groupinst *`groupinst`*, quickreport1 *`quickreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| group | None | report1group | |
| groupinst | None | report1groupinst | |
| quickreport | None | quickreport1 | |

# report1_quickreport_output_groupheader()

## Description

## Prototype

`report1_quickreport_output_groupheader` ( report1group *`group`*, report1groupinst *`groupinst`*, quickreport1 *`quickreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| group | None | report1group | |
| groupinst | None | report1groupinst | |
| quickreport | None | quickreport1 | |

# report1_quickreport_output_reportfooter()

## Description

## Prototype

`report1_quickreport_output_reportfooter` ( report1 *`report`*, report1inst *`reportinst`*, quickreport1 *`quickreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| quickreport | None | quickreport1 | |

# report1_quickreport_output_reportheader()

## Description

## Prototype

`report1_quickreport_output_reportheader` ( report1 *`report`*, report1inst *`reportinst`*, quickreport1 *`quickreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| quickreport | None | quickreport1 | |

# report1_quickreport_outputpageheader()

## Description

## Prototype

`report1_quickreport_outputpageheader` ( report1 *`report`*, report1inst *`reportinst`*, quickreport1 *`quickreport`*, array *`currcolvals`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| quickreport | None | quickreport1 | |
| currcolvals | None | array | |

# report1_quickreport_outputrow()

## Description

## Prototype

`report1_quickreport_outputrow` ( report1 *`report`*, report1inst *`reportinst`*, array *`columns`*, array *`currcolvals`*, boolean *`reading`*, quickreport1 *`quickreport`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| report | None | report1 | |
| reportinst | None | report1inst | |
| columns | None | array | |
| currcolvals | None | array | |
| reading | None | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| quickreport | None | quickreport1 | |

# savequickreport()

## Description

This saves a Quick Report as an XML file to the file name passed.

## Prototype

`savequickreport` ( quickreport1 *`qr`*, string *`filename`*, integer *`tabsize`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| qr | None | quickreport1 | |
| filename | None | string | |
| tabsize | 2 | integer | |
| error | None | integer | |

# Chapter 63. random

This library supplies the random type for generating pseudo-random numbers between 0 and 1.

# random

## Description

## Type Tags

None

## Object Value

Objects of type random have no value, and it is an error to try to get or set this value.

# random.new()

### Description

### Prototype

*random*.new ( random *me*, integer *seed* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | random | |
| seed | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _private | randomprivate | |
| next | function | |
| reseed | function | |
| type | type | |

## Methods

### next()

#### Description

### Prototype

*randomvar*.next ( random *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | random | |

## reseed()

### Description

### Prototype

*randomvar*.reseed ( random *me*, integer *seed* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | random | |
| seed | None | integer | |

# Chapter 64. recordview

## trecordview

### Description

### Type Tags

None

### Object Value

Objects of type trecordview have no value, and it is an error to try to get or set this value.

## trecordview.new()

### Description

### Prototype

*trecordview*.new ( trecordview *me*, integer *width*, integer *height*, type(db1table) *table*, type(db1index) *idx*, type(wxcontainer) *window*, string *defboolean*, string *definteger*, string *defnumber*, string *defdate*, string *deftime*, string *defdatetime*, SBLlocaledateinfo *datelocale*, SBLNumSettings *numlocale*, wxfont *font*, type(db1record) *record*, tdataview *dataview*, boolean *useview*, string *inifilename*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| width | None | integer | |
| height | None | integer | |
| table | None | type(db1table) | |
| idx | None | type(db1index) | |
| window | None | type(wxcontainer) | |
| defboolean | T\|F | string | |
| definteger | . | string | |
| defnumber | 999999.00 | string | |
| defdate | yyyy.0m.0d | string | |
| deftime | hh:mm:ss | string | |
| defdatetime | None | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| font | None | wxfont | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| record | None | type(db1record) | |
| dataview | None | tdataview | |
| useview | `.false` | boolean | |
| inifilename | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | trecordviewprivate | |
| assignfilterobject | function | |
| clear | function | |
| columninfo | array | |
| columninfoview | array | |
| currentview | tdataview | |
| datelocale | SBLlocaledateinfo | |
| defboolean | string | |
| defdate | string | |
| defdatetime | string | |
| definteger | string | |
| defnumber | string | |
| deftime | string | |
| deleterecord | function | |
| displayformat | string | |
| displaymessages | boolean | |
| duplicaterecord | function | |
| filter | dataform1filter | |
| font | wxfont | |
| form | dataform1 | |
| getcolinfo | function | |
| getindexdisplayformat | function | |
| grid | dataform1grid | |
| gridinfo | trecordviewgridinfo | |
| height | integer | |

| Property | Type | Description |
|---|---|---|
| hide | function | |
| idx | type(db1index) | |
| inifilename | string | |
| isdirty | boolean | |
| isreadonly | boolean | |
| lastusedrecord | type(db1record) | |
| messagetitle | string | |
| newrecord | function | |
| numlocale | SBLNumSettings | |
| ondelete | event | |
| onnewrecord | event | |
| onsave | event | |
| record | type(db1record) | |
| resize | function | |
| saverecord | function | |
| selectcurrent | function | |
| selectfforward | function | |
| selectfirst | function | |
| selectkey | function | |
| selectlast | function | |
| selectnext | function | |
| selectprevious | function | |
| selectrewind | function | |
| setdataview | function | |
| setfilter | function | |
| setinitialrecord | function | |
| setlastuse-drecord | function | |
| setreadonly | function | |
| settable | function | |
| show | function | |
| showview | function | |
| table | type(db1table) | |
| temprecord | type(db1record) | |
| type | type | |
| updategrid | function | |
| useview | boolean | |
| width | integer | |

| Property | Type | Description |
|---|---|---|
| window | type(wxcontainer) | |

# Methods

## assignfilterobject()

### Description

### Prototype

*trecordviewvar*.assignfilterobject ( trecordview *me*, dataform1filter *dffilter* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| dffilter | None | dataform1filter | |

## clear()

### Description

### Prototype

*trecordviewvar*.clear ( trecordview *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |

## deleterecord()

### Description

### Prototype

*trecordviewvar*.deleterecord ( trecordview *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| error | None | integer | |

# duplicaterecord()

## Description

## Prototype

*trecordviewvar*.duplicaterecord ( trecordview *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| error | None | integer | |

# getcolinfo()

## Description

## Prototype

*trecordviewvar*.getcolinfo ( trecordview *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |

# getindexdisplayformat()

## Description

## Prototype

*trecordviewvar*.getindexdisplayformat ( trecordview *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |

# hide()

## Description

## Prototype

*trecordviewvar*.hide ( trecordview *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |

## newrecord()

### Description

### Prototype

*trecordviewvar*.newrecord ( trecordview *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |

## resize()

### Description

### Prototype

*trecordviewvar*.resize ( trecordview *me*, integer *width*, integer *height* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| width | None | integer | |
| height | None | integer | |

## saverecord()

### Description

### Prototype

*trecordviewvar*.saverecord ( trecordview *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| error | None | integer | |

# selectcurrent()

## Description

## Prototype

*trecordviewvar*.selectcurrent ( trecordview *me*, type(db1index) *index*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| index | None | type(db1index) | |
| lock | .false | boolean | |
| error | None | integer | |

# selectfforward()

## Description

## Prototype

*trecordviewvar*.selectfforward ( trecordview *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| error | None | integer | |

# selectfirst()

## Description

## Prototype

*trecordviewvar*.selectfirst ( trecordview *me*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| lock | .false | boolean | |
| error | None | integer | |

# selectkey()

## Description

## Prototype

*trecordviewvar*.selectkey ( trecordview *me*, anyvalue *value*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| value | None | anyvalue | |
| lock | .false | boolean | |
| error | None | integer | |

# selectlast()

## Description

## Prototype

*trecordviewvar*.selectlast ( trecordview *me*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| lock | .false | boolean | |
| error | None | integer | |

# selectnext()

## Description

## Prototype

*trecordviewvar*.selectnext ( trecordview *me*, boolean *lock*, boolean *ignorefastselection*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| lock | .false | boolean | |
| ignorefastselection | .false | boolean | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# selectprevious()

## Description

## Prototype

*trecordviewvar*.selectprevious ( trecordview *me*, boolean *lock*, boolean *ignorefastselection*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| lock | .false | boolean | |
| ignorefastselection | .false | boolean | |
| error | None | integer | |

# selectrewind()

## Description

## Prototype

*trecordviewvar*.selectrewind ( trecordview *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| error | None | integer | |

# setdataview()

## Description

## Prototype

*trecordviewvar*.setdataview ( trecordview *me*, array *fieldlist*, string *name*, tdataview *dataview*, boolean *useview*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| fieldlist | None | array | |
| name | None | string | |
| dataview | None | tdataview | |
| useview | `.true` | boolean | |
| error | None | integer | |

# setfilter()

## Description

## Prototype

*trecordviewvar*.`setfilter` ( trecordview *me*, string *filter*, string *errtext*, integer *errindex* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| filter | None | string | |
| errtext | None | string | |
| errindex | None | integer | |

# setinitialrecord()

## Description

## Prototype

*trecordviewvar*.`setinitialrecord` ( trecordview *me*, type(db1record) *r*, boolean *dorefresh*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| r | None | type(db1record) | |
| dorefresh | `.false` | boolean | |
| error | None | integer | |

# setlastusedrecord()

## Description

### Prototype

*trecordviewvar*.setlastusedrecord ( trecordview *me*, type(db1record) *record* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| record | None | type(db1record) | |

## setreadonly()

### Description

### Prototype

*trecordviewvar*.setreadonly ( trecordview *me*, boolean *setreadonly* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| setreadonly | .false | boolean | |

## show()

### Description

### Prototype

*trecordviewvar*.show ( trecordview *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |

## showview()

### Description

### Prototype

*trecordviewvar*.showview ( trecordview *me*, boolean *show* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| show | `.true` | boolean | |

# updategrid()

## Description

## Prototype

*`trecordviewvar`*.`updategrid` ( trecordview *`me`*, type(db1record) *`r`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | trecordview | |
| r | None | type(db1record) | |

# Chapter 65. registrylib

## win32_registry

### Description

### Type Tags

None

### Object Value

Objects of type win32_registry have no value, and it is an error to try to get or set this value.

## win32_registry.new()

### Description

### Prototype

*win32_registry*.new ( win32_registry *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |

## Properties

| Property | Type | Description |
|---|---|---|
| closekey | function | |
| createkeyex | function | |
| deletekey | function | |
| deletevalue | function | |
| enumkeyex | function | |
| enumvalue | function | |
| enumvalue_getbufsize | function | |
| getvalue | function | |
| openkeyex | function | |
| queryvalueex | function | |
| regfuncs | sharedlibrary | |
| setvalueex_dword | function | |

| Property | Type | Description |
|---|---|---|
| setvalueex_string | function | |
| type | type | |

# Methods

## closekey()

### Description

### Prototype

*win32_registryvar*.closekey ( win32_registry *me*, integer *hkey*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| error | None | integer | |

## createkeyex()

### Description

### Prototype

*win32_registryvar*.createkeyex ( win32_registry *me*, integer *hkey*, string *subkey*, integer *options*, integer *samaccess*, integer *disposition*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| subkey | None | string | |
| options | `0` | integer | |
| samaccess | `983103` | integer | |
| disposition | None | integer | |
| error | None | integer | |

## deletekey()

### Description

### Prototype

*win32_registryvar*.deletekey ( win32_registry *me*, integer *hkey*, string *keyname*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| keyname | None | string | |
| error | None | integer | |

## deletevalue()

### Description

### Prototype

*win32_registryvar*.deletevalue ( win32_registry *me*, integer *hkey*, string *valuename*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| valuename | None | string | |
| error | None | integer | |

## enumkeyex()

### Description

### Prototype

*win32_registryvar*.enumkeyex ( win32_registry *me*, integer *hkey*, integer *index*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| index | None | integer | |
| error | None | integer | |

# enumvalue()

## Description

## Prototype

*win32_registryvar*.enumvalue ( win32_registry *me*, integer *hkey*, integer *index*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| index | None | integer | |
| error | None | integer | |

# getvalue()

## Description

## Prototype

*win32_registryvar*.getvalue ( win32_registry *me*, integer *hkey*, string *subkey*, string *valuename*, integer *flags*, integer *valuetype*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| subkey | None | string | |
| valuename | None | string | |
| flags | 65535 | integer | |
| valuetype | None | integer | |
| error | None | integer | |

# openkeyex()

## Description

## Prototype

*win32_registryvar*.openkeyex ( win32_registry *me*, integer *hkey*, string *subkey*, integer *samaccess*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| subkey | None | string | |
| samaccess | `983103` | integer | |
| error | None | integer | |

## queryvalueex()

### Description

### Prototype

*`win32_registryvar`*.`queryvalueex` ( win32_registry *`me`*, integer *`hkey`*, string *`valuename`*, integer *`valuetype`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| valuename | None | string | |
| valuetype | None | integer | |
| error | None | integer | |

## setvalueex_dword()

### Description

### Prototype

*`win32_registryvar`*.`setvalueex_dword` ( win32_registry *`me`*, integer *`hkey`*, string *`valuename`*, integer *`value`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| valuename | None | string | |
| value | None | integer | |
| error | None | integer | |

# setvalueex_string()

## Description

## Prototype

*win32_registryvar*.setvalueex_string ( win32_registry *me*, integer *hkey*, string *valuename*, string *value*, integer *valuetype*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | win32_registry | |
| hkey | None | integer | |
| valuename | None | string | |
| value | None | string | |
| valuetype | 1 | integer | |
| error | None | integer | |

# Chapter 66. reorglib

The reorglib provides the core functionality used by the command line reorganize command and it is also used by SIMPOL Personal for its reorganize functionality. This provides a method of repacking the database for basic maintenance on *.sbm databases.

# reorginfo

## Description

## Type Tags

None

## Object Value

Objects of type reorginfo have no value, and it is an error to try to get or set this value.

# reorginfo.new()

### Description

### Prototype

*reorginfo*.new ( reorginfo *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | reorginfo | |

## Properties

| Property | Type | Description |
|---|---|---|
| deferindexes | boolean | |
| endtime | datetime | |
| getdate | function | |
| gettime | function | |
| interval | integer | |
| logfile | string | |
| maxinterval | integer | |
| minimumup-dateinterval | datetime | |
| readcachesize | integer | |

| Property | Type | Description |
|---|---|---|
| recordcount | integer | |
| starttime | datetime | |
| type | type | |
| updateinterval | integer | |
| usercanceled | boolean | |
| userreference | type(*) | |
| userupdate | function | |
| writecachesize | integer | |

# Methods

## getdate()

### Description

### Prototype

*reorginfovar*.`getdate` ( datetime *`dt`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | datetime | |

## gettime()

### Description

### Prototype

*reorginfovar*.`gettime` ( datetime *`dt`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | datetime | |

# getvalidtempname()

# Description

# Prototype

`getvalidtempname` ()

## Parameters

None

# reorg_one_table()

## Description

## Prototype

`reorg_one_table` ( type(db1table) *`tablein`*, sbme1 *`sbmout`*, string *`targettablename`*, reorginfo *`info`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tablein | None | type(db1table) | |
| sbmout | None | sbme1 | |
| targettable-name | None | string | |
| info | None | reorginfo | |
| error | None | integer | |

# writereorglogentry()

## Description

## Prototype

`writereorglogentry` ( string *`logfile`*, string *`message`*, boolean *`append`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| logfile | None | string | |
| message | None | string | |
| append | `.true` | boolean | |

# Chapter 67. repguilib

## qrwdefaults

### Description

## Type Tags

None

## Object Value

Objects of type qrwdefaults have no value, and it is an error to try to get or set this value.

## qrwdefaults.new()

### Description

### Prototype

*qrwdefaults*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| currentfont | wxfont | |
| defbooleanfor-mat | string | |
| defdateformat | string | |
| defdatelocale | SBLlocaledateinfo | |
| defdatetimefor-mat | string | |
| defintegerfor-mat | string | |
| defnumberfor-mat | string | |
| defnumericlo-cale | SBLNumSettings | |
| deftimeformat | string | |

| Property | Type | Description |
|---|---|---|
| displayheight | integer | |
| displaywidth | integer | |
| getdefaultdis-playformat | function | |
| type | type | |

# Methods

## getdefaultdisplayformat()

### Description

### Prototype

*qrwdefaultsvar*.getdefaultdisplayformat ( qrwdefaults *me*, type *datatype* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | qrwdefaults | |
| datatype | None | type | |

# qrwfieldinfo

# Description

# Type Tags

None

# Object Value

Objects of type qrwfieldinfo have no value, and it is an error to try to get or set this value.

# qrwfieldinfo.new()

## Description

## Prototype

*qrwfieldinfo*.new ( qrwfieldinfo *me*, string *name*, string *displayformat*, type *datatype*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | qrwfieldinfo | |
| name | None | string | |
| displayformat | None | string | |
| datatype | None | type | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| datatype | type | |
| displayformat | string | |
| name | string | |
| type | type | |

# quickreportwindow

## Description

## Type Tags

None

## Object Value

Objects of type quickreportwindow have no value, and it is an error to try to get or set this value.

## quickreportwindow.new()

### Description

### Prototype

*quickreportwindow*.new ( quickreportwindow *me*, string *defbooleanformat*, string *defintegerformat*, string *defnumberformat*, string *defdateformat*, string *deftimeformat*, string *defdatetimeformat*, SBLlocaledateinfo *defdatelocale*, SBLNumSettings *defnumericlocale*, dring *datasources*, array *tables*, ppcstype1 *ppcs*, pagesetupinfo *psinfo*, boolean *createform* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreportwindow | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defbooleanformat | `T\|F` | string | |
| defintegerformat | `.` | string | |
| defnumberformat | `999999.00` | string | |
| defdateformat | `yyyy.0m.0d` | string | |
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimeformat | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |
| defdatelocale | None | SBLlocaledateinfo | |
| defnumericlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| psinfo | None | pagesetupinfo | |
| createform | `.true` | boolean | |

# Properties

| Property | Type | Description |
|---|---|---|
| adddatasource | function | |
| addtable | function | |
| createdf1 | function | |
| currentsavepath | string | |
| datasources | dring | |
| defaults | qrwdefaults | |
| f | dataform1 | |
| fieldinfo | array | |
| filename | string | |
| finddatasource | function | |
| findtable | function | |
| gap | integer | |
| isportrait | boolean | |
| last_units_id | integer | |
| locale | localeinfo | |
| margin_bottom | integer | |
| margin_left | integer | |

| Property | Type | Description |
|---|---|---|
| margin_right | integer | |
| margin_top | integer | |
| onclose | event | |
| outputtarget-names | array | |
| paperheight | integer | |
| papertype | integer | |
| paperwidth | integer | |
| parsejoinclause | function | |
| ppcs | ppcstype1 | |
| report | quickreport1 | |
| reportisdirty | boolean | |
| rowheightad-justment | number | |
| setrowheigh-tadjustment | function | |
| stdfont | wxfont | |
| tables | array | |
| tabsize | integer | |
| type | type | |
| usercanceled | boolean | |
| warningfont | wxfont | |

# Methods

## adddatasource()

### Description

### Prototype

*quickreportwindowvar*.adddatasource ( quickreportwindow *me*, type *sourcetype*, string *source*, type(*) *datasource*, string *username*, string *password*, integer *codepage*, integer *retry*, integer *timeout* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreportwindow | |
| sourcetype | None | type | |
| source | None | string | |
| datasource | None | type(*) | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| username | None | string | |
| password | None | string | |
| codepage | None | integer | |
| retry | None | integer | |
| timeout | None | integer | |

# addtable()

## Description

## Prototype

*quickreportwindowvar*.addtable ( quickreportwindow *me*, type(db1table) *table*, datasource-info *dsinfo*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreportwindow | |
| table | None | type(db1table) | |
| dsinfo | None | datasourceinfo | |
| error | None | integer | |

# createdf1()

## Description

## Prototype

*quickreportwindowvar*.createdf1 ( quickreportwindow *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreportwindow | |

# finddatasource()

## Description

## Prototype

*quickreportwindowvar*.finddatasource ( quickreportwindow *me*, string *sourcename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreportwindow | |
| sourcename | None | string | |

## findtable()

### Description

### Prototype

*quickreportwindowvar*.findtable ( quickreportwindow *me*, string *tablename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreportwindow | |
| tablename | None | string | |

## parsejoinclause()

### Description

### Prototype

*quickreportwindowvar*.parsejoinclause ( quickreportwindow *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreportwindow | |

## setrowheightadjustment()

### Description

### Prototype

*quickreportwindowvar*.setrowheightadjustment ( quickreportwindow *me*, number *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | quickreportwindow | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | number | |

# updatewindow

## Description

## Type Tags

None

## Object Value

Objects of type updatewindow have no value, and it is an error to try to get or set this value.

## updatewindow.new()

### Description

### Prototype

*updatewindow*.new ( updatewindow *me*, update1 *update*, integer *tabsize*, string *currentsavepath*, dring *datasources*, array *tables*, ppcstype1 *ppcs* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | updatewindow | |
| update | None | update1 | |
| tabsize | 2 | integer | |
| currentsavepath | None | string | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |

## Properties

| Property | Type | Description |
|---|---|---|
| adddatasource | function | |
| addtable | function | |
| calcok | boolean | |
| calculation | string | |

| Property | Type | Description |
|---|---|---|
| cur-rentsavepath | string | |
| datasources | dring | |
| filename | string | |
| finddatasource | function | |
| findtable | function | |
| form | wxform | |
| ppcs | ppcstype1 | |
| tablename | string | |
| tables | array | |
| tabsize | integer | |
| type | type | |
| update | update1 | |
| updateisdirty | boolean | |
| usercanceled | boolean | |
| warningfont | wxfont | |
| whereclause | string | |
| whereok | boolean | |

# Methods

## adddatasource()

### Description

### Prototype

*updatewindowvar*.adddatasource ( updatewindow *me*, type *sourcetype*, string *source*, type(*) *datasource*, string *username*, string *password*, integer *codepage*, integer *retry*, integer *timeout* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | updatewindow | |
| sourcetype | None | type | |
| source | None | string | |
| datasource | None | type(*) | |
| username | None | string | |
| password | None | string | |
| codepage | None | integer | |
| retry | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| timeout | None | integer | |

# addtable()

## Description

## Prototype

*updatewindowvar*.addtable ( updatewindow *me*, type(db1table) *table*, datasourceinfo *dsinfo*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | updatewindow | |
| table | None | type(db1table) | |
| dsinfo | None | datasourceinfo | |
| error | None | integer | |

# finddatasource()

## Description

## Prototype

*updatewindowvar*.finddatasource ( updatewindow *me*, string *sourcename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | updatewindow | |
| sourcename | None | string | |

# findtable()

## Description

## Prototype

*updatewindowvar*.findtable ( updatewindow *me*, string *tablename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | updatewindow | |
| tablename | None | string | |

# __quickreport()

## Description

## Prototype

__quickreport ( quickreport1 *qr*, type(wxdialogparent) *parent*, string *defbooleanformat*, string *defintegerformat*, string *defnumberformat*, string *defdateformat*, string *deftimeformat*, string *defdatetimeformat*, SBLlocaledateinfo *defdatelocale*, SBLNumSettings *defnumericlocale*, dring *datasources*, array *tables*, ppcstype1 *ppcs*, pagesetupinfo *psinfo*, number *rowheightadjustment*, string *currentpath*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| qr | None | quickreport1 | |
| parent | None | type(wxdialogparent) | |
| defbooleanformat | None | string | |
| defintegerformat | None | string | |
| defnumberformat | None | string | |
| defdateformat | None | string | |
| deftimeformat | None | string | |
| defdatetimeformat | None | string | |
| defdatelocale | None | SBLlocaledateinfo | |
| defnumericlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| psinfo | None | pagesetupinfo | |
| rowheightadjustment | None | number | |
| currentpath | None | string | |
| error | None | integer | |

# __update()

## Description

## Prototype

__update ( update1 *update*, type(wxdialogparent) *parent*, dring *datasources*, array *tables*, ppcstype1 *ppcs*, string *currentsavepath*, integer *tabsize*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| update | None | update1 | |
| parent | None | type(wxdialogparent) | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| currentsavepath | None | string | |
| tabsize | None | integer | |
| error | None | integer | |

# guigetsimplefilter()

## Description

## Prototype

guigetsimplefilter ( type(wxdialogparent) *w*, string *filter*, string *tablename*, string *defbooleanformat*, string *defintegerformat*, string *defnumberformat*, string *defdateformat*, string *deftimeformat*, string *defdatetimeformat*, SBLlocaledateinfo *defdatelocale*, SBLNumSettings *defnumericlocale*, dring *datasources*, array *tables*, ppcstype1 *ppcs*, string *caption* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| w | None | type(wxdialogparent) | |
| filter | None | string | |
| tablename | None | string | |
| defbooleanformat | T\|F | string | |
| defintegerformat | . | string | |
| defnumberformat | 999999.00 | string | |
| defdateformat | yyyy.0m.0d | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| deftimeformat | `hh:mm:ss` | string | |
| defdatetimeformat | `yyyy-mm-dd hh:mm:ss.nnnnnn` | string | |
| defdatelocale | None | SBLlocaledateinfo | |
| defnumericlocale | None | SBLNumSettings | |
| datasources | None | dring | |
| tables | None | array | |
| ppcs | None | ppcstype1 | |
| caption | `Superbase NG Filter` | string | |

# indexorderpicker()

## Description

## Prototype

`indexorderpicker` ( type(db1table) *`table`*, type(wxdialogparent) *`parent`*, string *`captiontext`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| table | None | type(db1table) | |
| parent | None | type(wxdialogparent) | |
| captiontext | `Select an index` | string | |

# Chapter 68. replace

This library implements the replace() function, for replacing all instances of one substring with another in a third substring.

# replace()

## Description

## Prototype

`replace` ( string *`s`*, string *`sOldvalue`*, string *`sNewvalue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| sOldvalue | None | string | |
| sNewvalue | None | string | |

# replaceold()

## Description

## Prototype

`replaceold` ( string *`s`*, string *`sOldvalue`*, string *`sNewvalue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| sOldvalue | None | string | |
| sNewvalue | None | string | |

# Chapter 69. reportlib

# report1

## Description

## Type Tags

sqlq1, report1aggregatecontainer, report1

## Object Value

Objects of type report1 have no value, and it is an error to try to get or set this value.

# report1.new()

## Description

## Prototype

*report1*.new ( report1 *me*, SBLNumSettings *numlocale*, SBLlocaledateinfo *datelocale*, type(sqlq1) *query* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |
| numlocale | None | SBLNumSettings | |
| datelocale | None | SBLlocaledateinfo | |
| query | None | type(sqlq1) | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addaggregate | function | |
| addgroup | function | |
| aggregates | dring | |
| continue | boolean | |
| datelocale | SBLlocaledateinfo | |

| Property | Type | Description |
|---|---|---|
| delete | boolean | |
| distinct | boolean | |
| findaggregate | function | |
| groups | dring | |
| numlocale | SBLNumSettings | |
| ondeleterow | event | |
| onoutputrow | event | |
| onreportend | event | |
| onreportstart | event | |
| orderclause | string | |
| outputpartialre-sults | boolean | |
| outputtarget | type(reportoutputtarget) | |
| query | type(sqlq1) | |
| removeaggre-gate | function | |
| removegroup | function | |
| run | function | |
| setdelete | function | |
| setdistinct | function | |
| setorderclause | function | |
| type | type | |

# Methods

## addaggregate()

### Description

### Prototype

*report1var*.addaggregate ( report1 *me*, function *calc*, integer *colno*, type *datatype*, integer *typeid*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |
| calc | None | function | |
| colno | None | integer | |
| datatype | None | type | |
| typeid | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# addgroup()

## Description

## Prototype

*report1var*.addgroup ( report1 *me*, string *name*, integer *colno*, type *datatype*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |
| name | None | string | |
| colno | None | integer | |
| datatype | None | type | |
| error | None | integer | |

# findaggregate()

## Description

## Prototype

*report1var*.findaggregate ( report1 *me*, integer *colno*, integer *typeid* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |
| colno | None | integer | |
| typeid | None | integer | |

# removeaggregate()

## Description

## Prototype

*report1var*.removeaggregate ( report1 *me*, report1aggregate *aggregate* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggregate | None | report1aggregate | |

# removegroup()

## Description

## Prototype

*report1var*.removegroup ( report1 *me*, report1group *group* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |
| group | None | report1group | |

# run()

## Description

## Prototype

*report1var*.run ( report1 *me*, string *errmsg*, integer *erridx*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |
| errmsg | None | string | |
| erridx | None | integer | |
| error | None | integer | |

# setdelete()

## Description

## Prototype

*report1var*.setdelete ( report1 *me*, boolean *delete* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| delete | `.false` | boolean | |

## setdistinct()

### Description

### Prototype

*`report1var`*`.setdistinct` ( report1 *`me`*, boolean *`distinct`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |
| distinct | `.false` | boolean | |

## setorderclause()

### Description

### Prototype

*`report1var`*`.setorderclause` ( report1 *`me`*, string *`orderclause`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1 | |
| orderclause | None | string | |

# report1aggregate

# Description

# Type Tags

None

# Object Value

Objects of type report1aggregate have no value, and it is an error to try to get or set this value.

## report1aggregate.new()

### Description

## Prototype

*report1aggregate*.new ( report1aggregate *me*, type(report1aggregatecontainer) *container*, function *getvalue*, integer *colno*, type *datatype*, integer *typeid*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1aggregate | |
| container | None | type(report1aggregatecontainer) | |
| getvalue | None | function | |
| colno | None | integer | |
| datatype | None | type | |
| typeid | None | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| colno | integer | |
| container | type(report1aggregatecontainer) | |
| datatype | type | |
| getvalue | function | |
| node | dlistnode | |
| onupdate | event | |
| type | type | |
| typeid | integer | |

# report1aggregatevalue

## Description

## Type Tags

None

## Object Value

Objects of type report1aggregatevalue have no value, and it is an error to try to get or set this value.

# report1aggregatevalue.new()

## Description

## Prototype

*report1aggregatevalue*.new ( report1aggregatevalue *me*, report1aggregate *parent*, type(report1aggregatevaluecontainer) *container*, anyvalue *value*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1aggregatevalue | |
| parent | None | report1aggregate | |
| container | None | type(report1aggregatevaluecontainer) | |
| value | None | anyvalue | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| container | type(report1aggregatevaluecontainer) | |
| getvalue | function | |
| node | dlistnode | |
| parent | report1aggregate | |
| type | type | |
| value | anyvalue | |
| valueholder | type(*) | |
| valueinfostub | type(*) | |

# report1group

## Description

## Type Tags

report1aggregatecontainer

## Object Value

Objects of type report1group have no value, and it is an error to try to get or set this value.

# report1group.new()

## Description

## Prototype

*report1group*.new ( report1group *me*, report1 *report*, string *name*, integer *colno*, type *datatype*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1group | |
| report | None | report1 | |
| name | None | string | |
| colno | None | integer | |
| datatype | None | type | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addaggregate | function | |
| aggregates | dring | |
| colno | integer | |
| datatype | type | |
| findaggregate | function | |
| getinstance | function | |
| name | string | |
| ongroupend | event | |
| ongroupstart | event | |
| removeaggregate | function | |
| report | report1 | |
| reportnode | dlistnode | |
| suppressaftergroupoutput | boolean | |
| suppressbeforegroupoutput | boolean | |
| type | type | |

# Methods

## addaggregate()

### Description

### Prototype

*report1groupvar*.addaggregate ( report1group *me*, function *getvalue*, integer *colno*, type *datatype*, integer *typeid*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1group | |
| getvalue | None | function | |
| colno | None | integer | |
| datatype | None | type | |
| typeid | None | integer | |
| error | None | integer | |

## findaggregate()

### Description

### Prototype

*report1groupvar*.findaggregate ( report1group *me*, integer *colno*, integer *typeid* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1group | |
| colno | None | integer | |
| typeid | None | integer | |

## getinstance()

### Description

### Prototype

*report1groupvar*.getinstance ( report1group *me*, anyvalue *value*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1group | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | anyvalue | |
| error | None | integer | |

## removeaggregate()

### Description

### Prototype

*report1groupvar*.removeaggregate ( report1group *me*, report1aggregate *aggregate* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1group | |
| aggregate | None | report1aggregate | |

# report1groupinst

## Description

## Type Tags

report1aggregatevaluecontainer

## Object Value

Objects of type report1groupinst have no value, and it is an error to try to get or set this value.

## report1groupinst.new()

### Description

### Prototype

*report1groupinst*.new ( report1groupinst *me*, report1inst *reportinst*, report1group *group*, anyvalue *groupvalue*, array *columns*, array *currcolvals*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1groupinst | |
| reportinst | None | report1inst | |
| group | None | report1group | |
| groupvalue | None | anyvalue | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| columns | None | array | |
| currcolvals | None | array | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addaggvalue | function | |
| aggregateval-ues | dring | |
| columns | array | |
| count | integer | |
| currcolvals | array | |
| group | report1group | |
| groupvalue | anyvalue | |
| remove | function | |
| reportinst | report1inst | |
| reportinstnode | dlistnode | |
| type | type | |

# Methods

## addaggvalue()

### Description

### Prototype

*report1groupinstvar*.addaggvalue ( report1groupinst *me*, report1aggregate *aggregate*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1groupinst | |
| aggregate | None | report1aggregate | |
| error | None | integer | |

## remove()

### Description

### Prototype

*report1groupinstvar*.remove ( report1groupinst *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1groupinst | |

# report1inst

## Description

## Type Tags

report1aggregatevaluecontainer

## Object Value

Objects of type report1inst have no value, and it is an error to try to get or set this value.

## report1inst.new()

### Description

### Prototype

*report1inst*.new ( report1inst *me*, report1 *report*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1inst | |
| report | None | report1 | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| addaggvalue | function | |
| addgroupinst | function | |
| aggregateval-ues | dring | |

| Property | Type | Description |
|---|---|---|
| cleargroups | function | |
| count | integer | |
| groupinstances | dring | |
| report | report1 | |
| type | type | |

# Methods

## addaggvalue()

### Description

### Prototype

*report1instvar*.addaggvalue ( report1inst *me*, report1aggregate *aggregate*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1inst | |
| aggregate | None | report1aggregate | |
| error | None | integer | |

## addgroupinst()

### Description

### Prototype

*report1instvar*.addgroupinst ( report1inst *me*, report1group *group*, anyvalue *groupvalue*, array *columns*, array *currcolvals*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1inst | |
| group | None | report1group | |
| groupvalue | None | anyvalue | |
| columns | None | array | |
| currcolvals | None | array | |
| error | None | integer | |

## cleargroups()

### Description

### Prototype

*report1instvar*.cleargroups ( report1inst *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | report1inst | |

# update1

## Description

## Type Tags

None

## Object Value

Objects of type update1 have no value, and it is an error to try to get or set this value.

## update1.new()

### Description

### Prototype

*update1*.new ( update1 *me*, string *calculation*, string *whereclause*, string *tablename*, string *updatefilename*, string *errormessage*, integer *errorindex*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1 | |
| calculation | None | string | |
| whereclause | None | string | |
| tablename | None | string | |
| updatefilename | None | string | |
| errormessage | None | string | |
| errorindex | None | integer | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |

| Property | Type | Description |
|---|---|---|
| __ | type(*) | |
| adddatasource | function | |
| addtable | function | |
| calculation | string | |
| continue | boolean | |
| datasource | update1datasource | |
| dirty | boolean | |
| fields | dring | |
| filename | string | |
| parsecalcula-tion | function | |
| query | sqlq1 | |
| run | function | |
| setcalculation | function | |
| table | update1table | |
| tablename | string | |
| type | type | |

# Methods

## adddatasource()

### Description

### Prototype

*update1var*.`adddatasource` ( update1 *`me`*, type *`sourcetype`*, string *`source`*, type(*) *`data-source`*, string *`username`*, string *`password`*, integer *`retry`*, integer *`timeout`*, integer *`codepage`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1 | |
| sourcetype | None | type | |
| source | None | string | |
| datasource | None | type(*) | |
| username | None | string | |
| password | None | string | |
| retry | `1000000` | integer | |
| timeout | `5000000` | integer | |
| codepage | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# addtable()

## Description

## Prototype

*update1var*.addtable ( update1 *me*, type(db1table) *table*, string *aliasname*, update1datasource *source*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1 | |
| table | None | type(db1table) | |
| aliasname | None | string | |
| source | None | update1datasource | |
| error | None | integer | |

# parsecalculation()

## Description

## Prototype

*update1var*.parsecalculation ( update1 *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1 | |

# run()

## Description

## Prototype

*update1var*.run ( update1 *me*, string *errormessage*, integer *errorindex*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| errormessage | None | string | |
| errorindex | None | integer | |
| error | None | integer | |

## setcalculation()

### Description

### Prototype

*update1var*.setcalculation ( update1 *me*, string *calculation* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1 | |
| calculation | None | string | |

# update1datasource

# Description

# Type Tags

None

# Object Value

Objects of type update1datasource have no value, and it is an error to try to get or set this value.

# update1datasource.new()

## Description

## Prototype

*update1datasource*.new ( update1datasource *me*, update1 *update*, type *sourcetype*, string *source*, type(*) *datasource*, string *username*, string *password*, integer *retry*, integer *timeout*, integer *codepage*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1datasource | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| update | None | update1 | |
| sourcetype | None | type | |
| source | None | string | |
| datasource | None | type(*) | |
| username | None | string | |
| password | None | string | |
| retry | `1000000` | integer | |
| timeout | `5000000` | integer | |
| codepage | None | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| codepage | integer | |
| datasource | type(*) | |
| opentable | function | |
| password | string | |
| retry | integer | |
| source | string | |
| sourcetype | type | |
| tables | dring | |
| timeout | integer | |
| type | type | |
| update | update1 | |
| updatenode | dlistnode | |
| username | string | |

# Methods

## opentable()

### Description

### Prototype

*update1datasourcevar*.`opentable` ( update1datasource *`me`*, string *`tablename`*, string *`password`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1datasource | |
| tablename | None | string | |
| password | None | string | |
| error | None | integer | |

# update1fieldinfo

## Description

## Type Tags

None

## Object Value

Objects of type update1fieldinfo have no value, and it is an error to try to get or set this value.

## update1fieldinfo.new()

### Description

### Prototype

*update1fieldinfo*.new ( update1fieldinfo *me*, update1 *update*, string *calculation*, string *fieldname*, string *tablename*, integer *colno*, type(db1field) *field*, boolean *designmode*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1fieldinfo | |
| update | None | update1 | |
| calculation | None | string | |
| fieldname | None | string | |
| tablename | None | string | |
| colno | None | integer | |
| field | None | type(db1field) | |
| designmode | `.false` | boolean | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| calculation | string | |
| colno | integer | |
| field | type(db1field) | |
| fieldname | string | |
| node | dlistnode | |
| tablename | string | |
| type | type | |
| update | update1 | |

# update1table

## Description

## Type Tags

None

## Object Value

Objects of type update1table have no value, and it is an error to try to get or set this value.

## update1table.new()

### Description

### Prototype

*update1table*.new ( update1table *me*, type(db1table) *table*, update1 *update*, dring *parent*, update1datasource *datasource*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1table | |
| table | None | type(db1table) | |
| update | None | update1 | |
| parent | None | dring | |
| datasource | None | update1datasource | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| datasource | update1datasource | |
| datasourcenode | dlistnode | |
| gettablename | function | |
| table | type(db1table) | |
| type | type | |
| update | update1 | |

# Methods

## gettablename()

### Description

### Prototype

*update1tablevar*.gettablename ( update1table *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | update1table | |

# __parsecalccomponent()

# Description

# Prototype

__parsecalccomponent ( update1 *update*, string *s*, string *tablename*, integer *colno*, boolean *designmode* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| update | None | update1 | |
| s | None | string | |
| tablename | None | string | |
| colno | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| designmode | `.false` | boolean | |

# __uparrowoutputrow()

## Description

## Prototype

`__uparrowoutputrow` ( integer *`spaces`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| spaces | None | integer | |

# __update_parsecalculation()

## Description

## Prototype

`__update_parsecalculation` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# getaggsortposbyid()

## Description

## Prototype

`getaggsortposbyid` ( integer *`typeid`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| typeid | None | integer | |

# getaggtypeidfromstring()

## Description

## Prototype

`getaggtypeidfromstring` ( string *`typename`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| typename | None | string | |

# getaggtypestring()

## Description

## Prototype

`getaggtypestring` ( integer *`typeid`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| typeid | None | integer | |

# loadupdatefile()

## Description

## Prototype

`loadupdatefile` ( string *`filename`*, dring *`datasources`*, array *`tables`*, ppcstype1 *`ppcs`*, string *`errortext`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| datasources | None | dring | |
| tables | None | array | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| ppcs | None | ppcstype1 | |
| errortext | None | string | |
| error | None | integer | |

# parseorderclause()

## Description

## Prototype

`parseorderclause` ( string *`orderclause`*, report1 *`report`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| orderclause | None | string | |
| report | None | report1 | |
| error | None | integer | |

# report1_agg_getval_count()

## Description

## Prototype

`report1_agg_getval_count` ( report1aggregatevalue *`aggvalue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggvalue | None | report1aggregatevalue | |

# report1_agg_getval_mean()

## Description

## Prototype

`report1_agg_getval_mean` ( report1aggregatevalue *`aggvalue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggvalue | None | report1aggregatevalue | |

# report1_agg_getval_median()

## Description

## Prototype

`report1_agg_getval_median` ( report1aggregatevalue *`aggvalue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggvalue | None | report1aggregatevalue | |

# report1_agg_getval_mode()

## Description

## Prototype

`report1_agg_getval_mode` ( report1aggregatevalue *`aggvalue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggvalue | None | report1aggregatevalue | |

# report1_agg_getval_sum()

## Description

## Prototype

`report1_agg_getval_sum` ( report1aggregatevalue *`aggvalue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggvalue | None | report1aggregatevalue | |

# report1_agg_getval_variance()

## Description

## Prototype

`report1_agg_getval_variance` ( report1aggregatevalue *`aggvalue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggvalue | None | report1aggregatevalue | |

# report1_agg_update_count()

## Description

## Prototype

`report1_agg_update_count` ( report1aggregate *`aggregate`*, report1aggregatevalue *`aggvalue`*, anyvalue *`value`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggregate | None | report1aggregate | |
| aggvalue | None | report1aggregatevalue | |
| value | None | anyvalue | |

# report1_agg_update_mean()

## Description

## Prototype

`report1_agg_update_mean` ( report1aggregate *`aggregate`*, report1aggregatevalue *`aggvalue`*, anyvalue *`value`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggregate | None | report1aggregate | |
| aggvalue | None | report1aggregatevalue | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | anyvalue | |

# report1_agg_update_median()

## Description

## Prototype

`report1_agg_update_median` ( report1aggregate *`aggregate`*, report1aggregatevalue *`aggvalue`*, anyvalue *`value`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggregate | None | report1aggregate | |
| aggvalue | None | report1aggregatevalue | |
| value | None | anyvalue | |

# report1_agg_update_mode()

## Description

## Prototype

`report1_agg_update_mode` ( report1aggregate *`aggregate`*, report1aggregatevalue *`aggvalue`*, anyvalue *`value`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggregate | None | report1aggregate | |
| aggvalue | None | report1aggregatevalue | |
| value | None | anyvalue | |

# report1_agg_update_sum()

## Description

## Prototype

`report1_agg_update_sum` ( report1aggregate *`aggregate`*, report1aggregatevalue *`aggvalue`*, anyvalue *`value`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggregate | None | report1aggregate | |
| aggvalue | None | report1aggregatevalue | |
| value | None | anyvalue | |

# report1_agg_update_variance()

## Description

## Prototype

`report1_agg_update_variance` ( report1aggregate *`aggregate`*, report1aggregatevalue *`aggvalue`*, anyvalue *`value`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| aggregate | None | report1aggregate | |
| aggvalue | None | report1aggregatevalue | |
| value | None | anyvalue | |

# reportinsertionsort()

## Description

## Prototype

`reportinsertionsort` ( array *`a`*, integer *`colno`*, integer *`lower`*, integer *`upper`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| colno | None | integer | |
| lower | None | integer | |
| upper | None | integer | |

# reportquicksortrit()

## Description

## Prototype

`reportquicksortrit` ( array *`a`*, integer *`colno`*, integer *`base`*, integer *`count`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| colno | None | integer | |
| base | None | integer | |
| count | None | integer | |

# saveupdatefile()

## Description

## Prototype

`saveupdatefile` ( update1 *`update`*, string *`filename`*, integer *`tabsize`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| update | None | update1 | |
| filename | None | string | |
| tabsize | 2 | integer | |
| error | None | integer | |

# Chapter 70. rsalib

This library provides a SIMPOL-language RSA library implementation, inclduing the routines for key generation.

# RSAdecrypt()

## Description

## Prototype

RSAdecrypt ( string *s*, integer *publickey*, integer *privatekey* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| publickey | None | integer | |
| privatekey | None | integer | |

# RSAencrypt()

## Description

## Prototype

RSAencrypt ( string *s*, integer *publickey* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| publickey | None | integer | |

# RSAgetkey()

## Description

## Prototype

RSAgetkey ( integer *bcount*, integer *publickey*, integer *privatekey* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bcount | None | integer | |
| publickey | None | integer | |
| privatekey | None | integer | |

# RSAreversedecrypt()

## Description

## Prototype

`RSAreversedecrypt` ( string *`s`*, integer *`publickey`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| publickey | None | integer | |

# RSAreverseencrypt()

## Description

## Prototype

`RSAreverseencrypt` ( string *`s`*, integer *`publickey`*, integer *`privatekey`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| publickey | None | integer | |
| privatekey | None | integer | |

# extRSAdecrypt()

## Description

## Prototype

`extRSAdecrypt` ( string *`s`*, string *`spublickey`*, string *`sprivatekey`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| spublickey | None | string | |
| sprivatekey | None | string | |

# extRSAencrypt()

## Description

## Prototype

`extRSAencrypt` ( string *`s`*, string *`spublickey`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| spublickey | None | string | |

# extRSAgetkey()

## Description

## Prototype

`extRSAgetkey` ( string *`sbitcount`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sbitcount | None | string | |

# extRSAreversedecrypt()

## Description

## Prototype

`extRSAreversedecrypt` ( string *`s`*, string *`spublickey`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| spublickey | None | string | |

# extRSAreverseencrypt()

## Description

## Prototype

`extRSAreverseencrypt` ( string *`s`*, string *`spublickey`*, string *`sprivatekey`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| spublickey | None | string | |
| sprivatekey | None | string | |

# Chapter 71. sbislib

This library provides some drop-in replacements for the Superbase Internet Server SBIS library functionality for web server applications. Some of the functions are quite useful, especially the HTML_Include(), which is used extensively in the Simpol Limited web system.

# HTML_GetValue()

## Description

## Prototype

`HTML_GetValue` ( cgicall *`cgi`*, string *`sName`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| cgi | None | cgicall | |
| sName | None | string | |

# HTML_Include()

## Description

## Prototype

`HTML_Include` ( cgicall *`cgi`*, string *`sFpiname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| cgi | None | cgicall | |
| sFpiname | None | string | |

# HTML_Page()

## Description

## Prototype

`HTML_Page` ( string *`sFuncname`*, cgicall *`cgi`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFuncname | None | string | |
| cgi | None | cgicall | |

# HTML_Path()

## Description

## Prototype

`HTML_Path` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# HTML_Read()

## Description

## Prototype

`HTML_Read` ( string *`sFpiname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFpiname | None | string | |

# HTML_SetValue()

## Description

## Prototype

`HTML_SetValue` ( cgicall *`cgi`*, string *`sName`*, string *`sValue`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| cgi | None | cgicall | |
| sName | None | string | |
| sValue | None | string | |

# Chapter 72. SBLDateLib

This provides a set of SBL-compatible functions for working with dates (the SBL versions used the $ sign in the name, which is not permitted in SIMPOL, so these are replaced by the letters STR): DAY, DAYS, DAYSTR(), MONTH(), MONTHSTR(), YEAR(), and DATESTR(). It also supplies a function for converting from a string to a date string2date().

# DATESTR()

## Description

Like all of the functions that are named with all capital letters, this function was designed to provide a drop-in replacement for the Superbase BASIC Language function of the same name `DATE$()`. As such it has a few idiosyncracies, just like the original. If the *`format`* parameter contains a comma or four `m` characters such that the date will be output with the name of the month spelled out, then the return value will have a leading space prepended. If a comma is included, then the output will contain a comma plus a space. The supported pattern character combinations are:

- dd – Numeric representation of the day of the month, without leading zeros
- 0d – Numeric representation of the day of the month, with leading zeros
- mm – Numeric representation of the month, without leading zeros
- 0m – Numeric representation of the month, with leading zeros
- mmm – Three-letter abbreviation for the name of the month
- mmmm – Full name of the month
- yy – Year represented by only the last two digits
- yyyy – Year represented by all four digits

Typical pattern strings are:

- "dd/mm/yy"
- "0d/0m/yy"
- "mm/dd/yy"
- "0m/0d/yy"
- "yyyy0m0d"
- "dd mmmm,yyyy"

## Prototype

`DATESTR` ( date *`dt`*, string *`format`*, SBLlocaledateinfo *`ldiLocale`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | date | |
| format | None | string | |
| ldiLocale | None | SBLlocaledateinfo | |

# DAY()

## Description

This function is for Superbase BASIC Language compatibility. It returns the day of the month or 0 if the date parameter was `.nul`.

## Prototype

`DAY` ( date *`dt`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | date | |

# DAYS()

## Description

This function implements the `DAYS()` function for Superbase BASIC Language SBL compatibility. It calls the `SBLDays()` function to read the date string and return a date object. It then retrieves the integer value of the date and adjusts it to what it should be for an SBL date. There are a few differences between the Julian date values in SBL and in SIMPOL. SBL implements the Gregorian calendar reform between September 2nd and September 14th, 1752 and considers dates that fall in the range of 3 September, 1752 and 13 September, 1752 to be illegal. That is mostly okay, except that it unfortunately still applies correct Gregorian formatting to the dates prior to that point resulting in 11 days too many in the Julian value. SBL also considers the date 1 January 0001 to be the value 1, while SIMPOL considers the same date to be the value 0. SIMPOL also assumes a correct astronomical calendar throughout and any adjustments for incorrectly calculated calendars are considered a localization issue. Returns the date corrected for SBL compatibility.

## Prototype

`DAYS` ( string *`value`*, string *`format`*, SBLlocaledateinfo *`ldiLocale`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| format | None | string | |
| ldiLocale | None | SBLlocaledateinfo | |
| error | None | integer | |

# DAYSTR()

## Description

This function returns the day of the week as a string from the locale information based on the date passed. If the date is `.nul` or the locale object is `.nul` then the return value is the string `".nul"`.

## Prototype

`DAYSTR` ( date *`dt`*, SBLlocaledateinfo *`ldiLocale`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | date | |
| ldiLocale | None | SBLlocaledateinfo | |

# MONTH()

## Description

This function implements the `MONTH()` function for SBL compatibility. It returns the month component of a date object as an integer or the value `0` if the date is equal to `.nul`.

## Prototype

`MONTH` ( date *`dt`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | date | |

# MONTHSTR()

## Description

This function implements the `MONTH$()` function for SBL compatibility. It returns the month component of a date object as a string from the locale information based on the date value passed. If the date value is `.nul` or the locale object is `.nul` then the return value is the string `".nul"`.

## Prototype

MONTHSTR ( date *dt*, SBLlocaledateinfo *ldiLocale* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | date | |
| ldiLocale | None | SBLlocaledateinfo | |

# YEAR()

## Description

This function implements the YEAR() function for SBL compatibility. It returns the year component of a date object as an integer or 0 if the date value is equal to .nul

## Prototype

YEAR ( date *dt* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | date | |

# string2date()

## Description

This function evaluates the date string passed in *t* using the format string passed in *format* and uses the centurybase information from the locale information if the year is provided as a 2-digit year. The locale information will also be used if the format string indicates the month is not provided as digits. From all of that an attempt will be made to determine the date provided and assign it to a new date object, which is then returned. The return value will normally be the date that corresponds to the values passed. If problems occur parsing the date, then an incorrect date or even a non-initialized (.nul) date object may be returned.

## Prototype

string2date ( string *t*, string *format*, SBLlocaledateinfo *datelocale*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| t | None | string | |
| format | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| datelocale | None | SBLlocaledateinfo | |
| error | None | integer | |

# Chapter 73. sblexten

This is a fairly faithful conversion of the sblexten.sbp from the Superbase distribution. It provides a number of useful functions.

# between()

## Description

## Prototype

`between` ( type(=) *`tValue`*, type(=) *`tRangeLo`*, type(=) *`tRangeHi`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tValue | None | type(=) | |
| tRangeLo | None | type(=) | |
| tRangeHi | None | type(=) | |

# blobBetween()

## Description

## Prototype

`blobBetween` ( blob *`bValue`*, blob *`bRangeLo`*, blob *`bRangeHi`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bValue | None | blob | |
| bRangeLo | None | blob | |
| bRangeHi | None | blob | |

# blobSwap()

## Description

## Prototype

`blobSwap` ( blob *`b1`*, blob *`b2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| b1 | None | blob | |
| b2 | None | blob | |

# boolSwap()

## Description

## Prototype

`boolSwap` ( boolean *`b1`*, boolean *`b2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| b1 | None | boolean | |
| b2 | None | boolean | |

# ceil()

## Description

## Prototype

`ceil` ( number *`nValue`*, integer *`iMultiple`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nValue | None | number | |
| iMultiple | None | integer | |

# checkformat()

## Description

## Prototype

`checkformat` ( string *`sSrc`*, string *`sFormat`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSrc | None | string | |
| sFormat | None | string | |

# countstr()

## Description

## Prototype

`countstr` ( string *`sSrc`*, string *`sSearch`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSrc | None | string | |
| sSearch | None | string | |

# floor()

## Description

## Prototype

`floor` ( number *`nValue`*, integer *`iMultiple`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nValue | None | number | |
| iMultiple | None | integer | |

# intAverage()

## Description

## Prototype

`intAverage` ( integer *`iNum1`*, integer *`iNum2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| iNum1 | None | integer | |
| iNum2 | None | integer | |

# intBetween()

## Description

## Prototype

`intBetween` ( integer *`iValue`*, integer *`iRangeLo`*, integer *`iRangeHi`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| iValue | None | integer | |
| iRangeLo | None | integer | |
| iRangeHi | None | integer | |

# intHighest()

## Description

## Prototype

`intHighest` ( integer *`i1`*, integer *`i2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| i1 | None | integer | |
| i2 | None | integer | |

# intNearest()

## Description

## Prototype

`intNearest` ( integer *`iNum1`*, integer *`iNum2`*, integer *`iNum3`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| iNum1 | None | integer | |
| iNum2 | None | integer | |
| iNum3 | None | integer | |

# intPercent()

## Description

## Prototype

`intPercent` ( number *`nNum1`*, number *`nNum2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nNum1 | None | number | |
| nNum2 | None | number | |

# intSwap()

## Description

## Prototype

`intSwap` ( integer *`iNum1`*, integer *`iNum2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| iNum1 | None | integer | |
| iNum2 | None | integer | |

# isleapyear()

## Description

Checks if the date object passed falls in a leap year and returns either `.true` or `.false`.

## Prototype

`isleapyear` ( date *`dEval`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dEval | None | date | |

# numAverage()

## Description

## Prototype

`numAverage` ( number *`nNum1`*, number *`nNum2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nNum1 | None | number | |
| nNum2 | None | number | |

# numBetween()

## Description

## Prototype

`numBetween` ( number *`nValue`*, number *`nRangeLo`*, number *`nRangeHi`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nValue | None | number | |
| nRangeLo | None | number | |
| nRangeHi | None | number | |

# numHighest()

## Description

## Prototype

`numHighest` ( number *`n1`*, number *`n2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| n1 | None | number | |
| n2 | None | number | |

# numNearest()

## Description

## Prototype

`numNearest` ( number *`nNum1`*, number *`nNum2`*, number *`nNum3`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nNum1 | None | number | |
| nNum2 | None | number | |
| nNum3 | None | number | |

# numPercent()

## Description

## Prototype

`numPercent` ( number *`nNum1`*, number *`nNum2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nNum1 | None | number | |
| nNum2 | None | number | |

# numSwap()

## Description

## Prototype

numSwap ( number *nNum1*, number *nNum2* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nNum1 | None | number | |
| nNum2 | None | number | |

# numposition()

## Description

## Prototype

numposition ( integer *i* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| i | None | integer | |

# round()

## Description

## Prototype

round ( number *nValue*, integer *iMultiple* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| nValue | None | number | |
| iMultiple | None | integer | |

# strBetween()

## Description

## Prototype

`strBetween` ( string *`sValue`*, string *`sRangeLo`*, string *`sRangeHi`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sValue | None | string | |
| sRangeLo | None | string | |
| sRangeHi | None | string | |

# strSwap()

## Description

## Prototype

`strSwap` ( string *`s1`*, string *`s2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s1 | None | string | |
| s2 | None | string | |

# swap()

## Description

## Prototype

`swap` ( anyvalue *`t1`*, anyvalue *`t2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| t1 | None | anyvalue | |
| t2 | None | anyvalue | |

# Chapter 74. sbllib

This library implements many of the Superbase FN() functions, such as: FN_Alpha(), FN_Dec(), FN_Hex(), FN_Fact(), and so on. See the source code for details.

# FN_Alpha()

## Description

## Prototype

`FN_Alpha` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# FN_Dec()

## Description

## Prototype

`FN_Dec` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# FN_Ext()

## Description

## Prototype

`FN_Ext` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# FN_Fact()

## Description

## Prototype

`FN_Fact` ( integer *`i`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| i | None | integer | |

# FN_Hex()

## Description

## Prototype

`FN_Hex` ( integer *`i`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| i | None | integer | |

# FN_Name()

## Description

## Prototype

`FN_Name` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# FN_Numeric()

## Description

## Prototype

`FN_Numeric` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# FN_Path()

## Description

## Prototype

`FN_Path` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# FN_Phone()

## Description

## Prototype

`FN_Phone` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# FN_Root()

## Description

## Prototype

`FN_Root` ( string *s* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# Chapter 75. SBLlocaledateinfo

This library implements the SBLlocaledateinfo type, which encodes the locale info required by many of the SBL date functions. It is included in the SBLDateLib.

# SBLdateinfo

## Description

## Type Tags

None

## Object Value

Objects of type SBLdateinfo have no value, and it is an error to try to get or set this value.

## SBLdateinfo.new()

### Description

### Prototype

*SBLdateinfo*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| index | integer | |
| name | string | |
| next | SBLdateinfo | |
| type | type | |

# SBLlocaledateinfo

## Description

## Type Tags

None

# Object Value

Objects of type SBLlocaledateinfo have no value, and it is an error to try to get or set this value.

# SBLlocaledateinfo.new()

## Description

## Prototype

*SBLlocaledateinfo*.new ( SBLlocaledateinfo *me*, string *sDaynames*, string *sMonthnames*, string *sMonthsAbbrev*, integer *centurybase*, string *format* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | SBLlocaledateinfo | |
| sDaynames | `Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday` | string | |
| sMonthnames | `January,February,March,April,May,June,July,August,September,October,Novemb` | string | |
| sMonthsAb-brev | `Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec` | string | |
| centurybase | `1930` | integer | |
| format | `yyyy0m0d` | string | |

# Properties

| Property | Type | Description |
|---|---|---|
| centurybase | integer | |
| finddayname | function | |
| findmonthab-brev | function | |
| findmonth-name | function | |
| firstdayname | SBLdateinfo | |
| firstmonth-name | SBLdateinfo | |
| firstmonthsab-brev | SBLdateinfo | |
| format | string | |
| getdayname | function | |
| getmonthab-brev | function | |
| getmonthname | function | |
| type | type | |

# Methods

## finddayname()

### Description

### Prototype

*SBL localedateinfovar*.finddayname ( SBLlocaledateinfo *me*, string *sName* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | SBLlocaledateinfo | |
| sName | None | string | |

## findmonthabbrev()

### Description

### Prototype

*SBL localedateinfovar*.findmonthabbrev ( SBLlocaledateinfo *me*, string *sName* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | SBLlocaledateinfo | |
| sName | None | string | |

## findmonthname()

### Description

### Prototype

*SBL localedateinfovar*.findmonthname ( SBLlocaledateinfo *me*, string *sName* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | SBLlocaledateinfo | |
| sName | None | string | |

## getdayname()

### Description

### Prototype

*SBL**localedateinfovar*.getdayname ( SBLlocaledateinfo *me*, date *dt* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | SBLlocaledateinfo | |
| dt | None | date | |

## getmonthabbrev()

### Description

### Prototype

*SBL**localedateinfovar*.getmonthabbrev ( SBLlocaledateinfo *me*, date *dt* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | SBLlocaledateinfo | |
| dt | None | date | |

## getmonthname()

### Description

### Prototype

*SBL**localedateinfovar*.getmonthname ( SBLlocaledateinfo *me*, date *dt* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | SBLlocaledateinfo | |
| dt | None | date | |

# Chapter 76. SBLTimeLib

This library implements an SBL-compatible time library, with functions like: TIMESTR(), TIMEVAL(), HRS(), MINS(), SECS(), THOUSECS, and string2time() for converting from a string version of a time to a time value.

# HRS()

## Description

This function implements the `HRS()` function for SBL compatibility. Returns the hours component of the time passed in *t*.

## Prototype

`HRS` ( time *t* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| t | None | time | |

# MINS()

## Description

This function implements the `MINS()` function for SBL compatibility. Returns the minutes component of the time passed in *t*.

## Prototype

`MINS` ( time *t* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| t | None | time | |

# SECS()

## Description

This function implements the `SECS()` function for SBL compatibility. Returns the seconds component of the time passed in *t*.

## Prototype

SECS ( time *t* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| t | None | time | |

# THOUSECS()

## Description

This function implements the THOUSECS() function for SBL compatibility. Returns the milliseconds component of the time passed in *t*.

## Prototype

THOUSECS ( time *t* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| t | None | time | |

# TIMESTR()

## Description

This function implements the TIME$() function for SBL compatibility. The return value will be either the empty string (""), some string containing the word ".nul" one or more times, or the time formatted as a string based on the format string passed. The supported pattern characters are:

- hh – Hours without leading zeros (12-hour clock)
- 0h – Hours with leading zeros (12-hour clock)
- mm – Minutes with leading zeros
- ss – Seconds with leading zeros
- ss.s – Seconds with leading zeros, a decimal point, and the milliseconds padded with zeros
- a – 12-hour clock values with am or pm designator

If the time format begins with a | character, then it is using a newer extended time format and will be passed to the extTIMESTR() function for processing.

## Prototype

TIMESTR ( time *t*, string *sFormat* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| t | None | time | |
| sFormat | None | string | |

# TIMEVAL()

## Description

This function implements the `TIMEVAL()` function for SBL compatibility. It uses the `string2time()` function to parse the value in *`sValue`*. If the value in the *`sFormat`* parameter is invalid or any other problem occurs, `.nul` is returned otherwise the time in milliseconds is returned as an integer.

## Prototype

`TIMEVAL` ( string *`sValue`*, string *`sFormat`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sValue | None | string | |
| sFormat | None | string | |

# extTIMESTR()

## Description

This function provides a wrapper to the `strftime()` function which can be found on Windows, Linux, and OS-X. This wrapper only supports the time components since it is called using a time object. All of the format items are included for completeness. It formats the time *`t`* according to the format specification *`sFormat`* and returns the result. The format specification is a string and may contain special character sequences called conversion specifications, each of which is introduced by a '%' character and terminated by some other character known as a conversion specifier character. All other character sequences are ordinary character sequences.

The characters of ordinary character sequences are copied verbatim from *`sFormat`* to the return value. However, the characters of conversion specifications are replaced as follows:

- %a – The abbreviated name of the day of the week according to the current operating system locale.
- %A – The full name of the day of the week according to the current operating system locale.
- %b – The abbreviated month name according to the current operating system locale.
- %B – The full month name according to the current operating system locale.
- %c – The preferred date and time representation for the current operating system locale.

- %C – The century number (year/100) as a 2-digit integer.
- %d – The day of the month as a decimal number (range 01 to 31).
- %D – Equivalent to %m/%d/%y (American-specific format. It should be noted that in other countries %d/%m/%y is far more common which means that in international context this format is ambiguous and should not be used).
- %e – Like %d, the day of the month as a decimal number, but a leading zero is replaced by a space.
- %E – Modifier: use alternative format, see below.
- %F – Equivalent to %Y-%m-%d (the ISO 8601 date format).
- %G – The ISO 8601 week-based year with century as a decimal number. The 4-digit year corresponding to the ISO week number (see %V). This has the same format and value as %Y, except that if the ISO week number belongs to the previous or next year, that year is used instead.
- %g – Like %G, but without century, that is, with a 2-digit year (00-99).
- %h – Equivalent to %b.
- %H – The hour as a decimal number using a 24-hour clock (range 00 to 23).
- %I – The hour as a decimal number using a 12-hour clock (range 01 to 12).
- %j – The day of the year as a decimal number (range 001 to 366).
- %k – The hour (24-hour clock) as a decimal number (range 0 to 23); single digits are preceded by a blank. (See also %H.)
- %l – The hour (12-hour clock) as a decimal number (range 1 to 12); single digits are preceded by a blank. (See also %I.)
- %m – The month as a decimal number (range 01 to 12).
- %M – The minute as a decimal number (range 00 to 59).
- %n – A newline character.
- %O – Modifier: use alternative format, see below.
- %p – Either "AM" or "PM" according to the given time value, or the corresponding strings for the current locale. Noon is treated as "PM" and midnight as "AM".
- %P – Like %p but in lowercase: "am" or "pm" or a corresponding string for the current locale.
- %r – The time in a.m. or p.m. notation. In the POSIX locale this is equivalent to %I:%M:%S %p.
- %R – The time in 24-hour notation (%H:%M). For a version including the seconds, see %T below.
- %s – The number of seconds since the Epoch, 1970-01-01 00:00:00 +0000 (UTC).
- %S – The second as a decimal number (range 00 to 60). (The range is up to 60 to allow for occasional leap seconds.)
- %t – A tab character.
- %T – The time in 24-hour notation (%H:%M:%S).

- %u – The day of the week as a decimal, range 1 to 7, Monday being 1. See also %w.
- %U – The week number of the current year as a decimal number, range 00 to 53, starting with the first Sunday as the first day of week 01. See also %V and %W.
- %V – The ISO 8601 week number of the current year as a decimal number, range 01 to 53, where week 1 is the first week that has at least 4 days in the new year. See also %U and %W.
- %w – The day of the week as a decimal, range 0 to 6, Sunday being 0. See also %u.
- %W – The week number of the current year as a decimal number, range 00 to 53, starting with the first Monday as the first day of week 01.
- %x – The preferred date representation for the current locale without the time.
- %X – The preferred time representation for the current locale without the date.
- %y – The year as a decimal number without a century (range 00 to 99).
- %Y – The year as a decimal number including the century.
- %z – The +hhmm or -hhmm numeric timezone (that is, the hour and minute offset from UTC).
- %Z – The timezone name or abbreviation.
- %% – A literal '%' character.

Some conversion specifications can be modified by preceding the conversion specifier character by the E or O modifier to indicate that an alternative format should be used. If the alternative format or specification does not exist for the current locale, the behavior will be as if the unmodified conversion specification were used. The Single UNIX Specification mentions %Ec, %EC, %Ex, %EX, %Ey, %EY, %Od, %Oe, %OH, %OI, %Om, %OM, %OS, %Ou, %OU, %OV, %Ow, %OW, %Oy, where the effect of the O modifier is to use alternative numeric symbols (say, roman numerals), and that of the E modifier is to use a locale-dependent alternative representation.

## Prototype

`extTIMESTR` ( time *`t`*, string *`sFormat`*, sharedlibraryfunction *`strftime`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| t | None | time | |
| sFormat | None | string | |
| strftime | None | sharedlibraryfunction | |

# string2time()

## Description

This function attempts to interpret the value in *`sValue`* using the format string in *`sFormat`* and convert that to a valid time, which will be the return value. The hours and minutes values are required or else the return value will be `.nul`.

# Prototype

`string2time` ( string *`sValue`*, string *`sFormat`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sValue | None | string | |
| sFormat | None | string | |

# Chapter 77. sbnglib

This library incorporates a number of useful types for interacting wit the data-aware form system and for communicating information with other functions, such as the datasourceinfo type, the tbinfo type, and the implementation of option button grouping for non-data-aware forms.

# datasourceinfo

## Description

## Type Tags

None

## Object Value

Objects of type datasourceinfo have no value, and it is an error to try to get or set this value.

# datasourceinfo.new()

### Description

### Prototype

*datasourceinfo*.new ( datasourceinfo *me*, dring *parent*, type *sourcetype*, string *source*, type(*) *datasource*, string *username*, string *password*, integer *retry*, integer *timeout*, integer *codepage*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | datasourceinfo | |
| parent | None | dring | |
| sourcetype | None | type | |
| source | None | string | |
| datasource | None | type(*) | |
| username | None | string | |
| password | None | string | |
| retry | `1000000` | integer | |
| timeout | `5000000` | integer | |
| codepage | None | integer | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| appnode | dlistnode | |
| codepage | integer | |
| datasource | type(*) | |
| findsource-name | function | |
| password | string | |
| reinit | function | |
| retry | integer | |
| source | string | |
| sourcetype | type | |
| tables | array | |
| timeout | integer | |
| type | type | |
| username | string | |

# Methods

## findsourcename()

### Description

### Prototype

*datasourceinfovar*.`findsourcename` ( datasourceinfo *me*, string *sourcename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | datasourceinfo | |
| sourcename | None | string | |

## reinit()

### Description

### Prototype

*datasourceinfovar*.`reinit` ( datasourceinfo *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | datasourceinfo | |
| error | None | integer | |

# tbinfo

## Description

## Type Tags

None

## Object Value

Objects of type tbinfo have no value, and it is an error to try to get or set this value.

## tbinfo.new()

### Description

### Prototype

*tbinfo*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| gettablename | function | |
| hastemplate | boolean | |
| source | datasourceinfo | |
| table | type(db1table) | |
| type | type | |

## Methods

### gettablename()

#### Description

### Prototype

*tbinfovar*.gettablename ( tbinfo *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | tbinfo | |

# fletcher16()

## Description

## Prototype

fletcher16 ( blob *data* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| data | None | blob | |

# fletcher32()

## Description

## Prototype

fletcher32 ( blob *data* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| data | None | blob | |

# tablestatus()

## Description

## Prototype

tablestatus ( tbinfo *tinfo*, string *eol* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| tinfo | None | tbinfo | |
| eol | | string | |

# validppcscodepage()

## Description

## Prototype

`validppcscodepage` ( integer *`codepage`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| codepage | None | integer | |

# Chapter 78. sendkeys

This is a SENDKEYS implementation written in SIMPOL and which is specific to Windows. It does not currently have an implementation for Linux.

# sendkeyshelper

## Description

## Type Tags

None

## Object Value

Objects of type sendkeyshelper have no value, and it is an error to try to get or set this value.

# sendkeyshelper.new()

### Description

### Prototype

*sendkeyshelper*.new ( sendkeyshelper *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sendkeyshelper | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| GetVersion | sharedlibraryfunction | |
| KeyCode | function | |
| KeyDown | function | |
| KeyUp | function | |
| MapVirtu-alKeyA | sharedlibraryfunction | |
| MapVirtu-alKeyW | sharedlibraryfunction | |

| Property | Type | Description |
|---|---|---|
| VkKeyScanA | sharedlibraryfunction | |
| VkKeyScanW | sharedlibraryfunction | |
| bNt | boolean | |
| keybd_event | sharedlibraryfunction | |
| type | type | |

# Methods

## KeyCode()

### Description

### Prototype

*sendkeyshelpervar*.KeyCode ( sendkeyshelper *me*, string *char* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sendkeyshelper | |
| char | None | string | |

## KeyDown()

### Description

### Prototype

*sendkeyshelpervar*.KeyDown ( sendkeyshelper *me*, integer *vKey*, integer *scancode* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sendkeyshelper | |
| vKey | None | integer | |
| scancode | 0 | integer | |

## KeyUp()

### Description

### Prototype

*sendkeyshelpervar*.KeyUp ( sendkeyshelper *me*, integer *vKey*, integer *scancode* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sendkeyshelper | |
| vKey | None | integer | |
| scancode | `0` | integer | |

# sendkeys()

## Description

## Prototype

`sendkeys` ( string *`chars`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| chars | None | string | |
| error | None | integer | |

# Chapter 79. sendmail

The sendmail library has one function, the sendmail() function. It provides an easy interface to the sending of SMTP emails. If all that is required is a text email, then this is the easiest way to do it.

# sendmail()

## Description

## Prototype

`sendmail` ( string *`sFrom`*, string *`sEmailaddr`*, string *`sSubject`*, string *`sMessage`*, string *`sMailhost`*, string *`sHostname`*, string *`sTimezone`*, string *`sReplyto`*, string *`authtype`*, string *`username`*, string *`password`*, boolean *`usehtml`*, array *`attachments`*, boolean *`debug`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFrom | None | string | |
| sEmailaddr | None | string | |
| sSubject | None | string | |
| sMessage | None | string | |
| sMailhost | None | string | |
| sHostname | None | string | |
| sTimezone | None | string | |
| sReplyto | None | string | |
| authtype | None | string | |
| username | None | string | |
| password | None | string | |
| usehtml | `.false` | boolean | |
| attachments | None | array | |
| debug | `.false` | boolean | |

# sendmail_sb()

## Description

## Prototype

`sendmail_sb` ( string *`sFrom`*, string *`sEmailaddr`*, string *`sSubject`*, string *`sMessage`*, string *`sMailhost`*, string *`sHostname`*, string *`sTimezone`*, string *`sReplyto`*, string *`authtype`*, string *`username`*, string *`password`*, string *`usehtml`*, string *`attachmentlist`*, string *`sDebug`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFrom | None | string | |
| sEmailaddr | None | string | |
| sSubject | None | string | |
| sMessage | None | string | |
| sMailhost | None | string | |
| sHostname | None | string | |
| sTimezone | None | string | |
| sReplyto | None | string | |
| authtype | None | string | |
| username | None | string | |
| password | None | string | |
| usehtml | None | string | |
| attachmentlist | None | string | |
| sDebug | None | string | |

# Chapter 80. serialize

This library provides a method serializing objects in SIMPOL to a file, and then reloading them.

# loadserializeddata()

## Description

## Prototype

`loadserializeddata` ( string *`filename`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| error | None | integer | |

# readserializeddata()

## Description

## Prototype

`readserializeddata` ( type(*) *`object`*, blob *`b`*, integer *`position`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| object | None | type(*) | |
| b | None | blob | |
| position | `1` | integer | |
| error | None | integer | |

# serialize()

## Description

## Prototype

`serialize` ( type(*) *`object`*, fsfileoutputstream *`fpo`*, string *`filename`*, boolean *`append`*, integer *`error`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| object | None | type(*) | |
| fpo | None | fsfileoutputstream | |
| filename | None | string | |
| append | `.true` | boolean | |
| error | None | integer | |

# Chapter 81. sessionid

## sessionid

### Description

## Type Tags

None

## Object Value

Objects of type sessionid have no value, and it is an error to try to get or set this value.

## sessionid.new()

### Description

### Prototype

*sessionid*.new ( sessionid *me*, string *sIpaddress*, string *sEncID*, integer *iValidDuration*, integer *iPublicKey*, integer *iPrivateKey* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sessionid | |
| sIpaddress | None | string | |
| sEncID | None | string | |
| iValidDuration | None | integer | |
| iPublicKey | None | integer | |
| iPrivateKey | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| dtCreated | datetime | |
| iValidDuration | integer | |
| makeid | function | |
| new | function | |
| reverseid | function | |

| Property | Type | Description |
|---|---|---|
| sEncID | string | |
| sID | string | |
| sIpaddress | string | |
| type | type | |
| valid | function | |

# Methods

## makeid()

### Description

### Prototype

*sessionidvar*.makeid ( sessionid *me*, integer *iPublicKey* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sessionid | |
| iPublicKey | None | integer | |

## reverseid()

### Description

### Prototype

*sessionidvar*.reverseid ( sessionid *me*, integer *iPublicKey*, integer *iPrivateKey* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sessionid | |
| iPublicKey | None | integer | |
| iPrivateKey | None | integer | |

## valid()

### Description

### Prototype

*sessionidvar*.valid ( sessionid *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sessionid | |

# Chapter 82. sessionid2

## sessionid2

### Description

## Type Tags

None

## Object Value

Objects of type sessionid2 have no value, and it is an error to try to get or set this value.

## sessionid2.new()

### Description

### Prototype

*sessionid2*.new ( sessionid2 *me*, string *sIpaddress*, string *sEncID*, integer *iPublicKey*, integer *iPrivateKey* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sessionid2 | |
| sIpaddress | None | string | |
| sEncID | None | string | |
| iPublicKey | None | integer | |
| iPrivateKey | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| dtCreated | datetime | |
| makeid | function | |
| new | function | |
| reverseid | function | |
| sEncID | string | |
| sID | string | |
| sIpaddress | string | |

| Property | Type | Description |
|---|---|---|
| type | type | |

# Methods

## makeid()

### Description

### Prototype

*sessionid2var*.makeid ( sessionid2 *me*, integer *iPublicKey* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sessionid2 | |
| iPublicKey | None | integer | |

## reverseid()

### Description

### Prototype

*sessionid2var*.reverseid ( sessionid2 *me*, integer *iPublicKey*, integer *iPrivateKey* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sessionid2 | |
| iPublicKey | None | integer | |
| iPrivateKey | None | integer | |

# Chapter 83. shellexecute

This library is a wrapper for the Windows API SHellExecute() function. It is very handy for starting the appropriate program for a given content type. Use as an alternative to the !execute() system function. This is a Windows only library.

# shellexecute()

## Description

This is a windows-only function that provides a method of calling the Windows API ShellExecute() function. It can be used to call the appropriate program for a file with a given extension without knowing which program is responsible for loading it.

## Prototype

`shellexecute` ( integer *`hwnd`*, string *`operation`*, string *`filename`*, string *`params`*, string *`directory`*, integer *`showcmd`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| hwnd | `0` | integer | |
| operation | `open` | string | |
| filename | None | string | |
| params | None | string | |
| directory | None | string | |
| showcmd | `1` | integer | |

# Chapter 84. simpollib

## findfunction()

### Description

### Prototype

`findfunction` ( string *`sFuncname`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFuncname | None | string | |

## findproperty()

### Description

### Prototype

`findproperty` ( string *`propname`*, type(*) *`object`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| propname | None | string | |
| object | None | type(*) | |

## getfunction()

### Description

This attempts to locate a function name in any of the loaded modules. A module is any compiled unit (library) of which there may be many. This will also find functions that are not exported, so it is important to make sure that the function that you are searching for has a unique name. If found it returns the function as a reference, otheriwse returns `.nul`.

### Prototype

`getfunction` ( string *`sFuncname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sFuncname | None | string | |

# gettype()

## Description

## Prototype

`gettype` ( string *`sTypename`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sTypename | None | string | |

# hasproperty()

## Description

## Prototype

`hasproperty` ( type(*) *`object`*, string *`propname`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| object | None | type(*) | |
| propname | None | string | |

# inarray()

## Description

## Prototype

`inarray` ( array *`a`*, type(*) *`item`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| item | None | type(*) | |

# isproperty()

## Description

Checks to see if the string passed in is the name of a property of the type of the object that was passed. Returns either `.true` or `.false`.

## Prototype

`isproperty` ( type(*) *`object`*, string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| object | None | type(*) | |
| s | None | string | |

# Chapter 85. smtpclientlib

This library provides a more direct access to the sending of SMTP emails than the sendmail library, and it is used by the sendmail library. It currently only implements text-based emails. See the source code for further information.

# smtpmessage

## Description

## Type Tags

None

## Object Value

Objects of type smtpmessage have no value, and it is an error to try to get or set this value.

# smtpmessage.new()

### Description

### Prototype

*`smtpmessage`*.new ( smtpmessage *`me`*, string *`contenttype`*, string *`transferencoding`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | smtpmessage | |
| contenttype | `text/plain; charset="iso-8859-15"` | string | |
| transferencoding | `8bit` | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| addattachment | function | |
| addrecip | function | |
| attachments | set | |
| authtype | integer | |
| bcc | set | |
| body | string | |

| Property | Type | Description |
|---|---|---|
| boundary | string | |
| cc | set | |
| comment | string | |
| contenttype | string | |
| createmime-boundary | function | |
| debug | boolean | |
| dt | datetime | |
| from | string | |
| getaddress | function | |
| hostname | string | |
| mailhost | string | |
| messageid | string | |
| password | string | |
| preparemsg | function | |
| replyto | string | |
| send | function | |
| send8bit | function | |
| setauthinfo | function | |
| subject | string | |
| timezone | string | |
| to | set | |
| transferencod-ing | string | |
| type | type | |
| usehtml | boolean | |
| username | string | |

# Methods

## addattachment()

### Description

### Prototype

*smtpmessagevar*.addattachment ( smtpmessage *me*, string *filename*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | smtpmessage | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| filename | None | string | |
| error | None | integer | |

## addrecip()

### Description

### Prototype

*smtpmessagevar*.addrecip ( smtpmessage *me*, string *reciptype*, string *address* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | smtpmessage | |
| reciptype | None | string | |
| address | None | string | |

## createmimeboundary()

### Description

### Prototype

*smtpmessagevar*.createmimeboundary ( smtpmessage *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | smtpmessage | |

## getaddress()

### Description

### Prototype

*smtpmessagevar*.getaddress ( smtpmessage *me*, string *s* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | smtpmessage | |
| s | None | string | |

# send()

## Description

## Prototype

*smtpmessagevar*.send ( smtpmessage *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | smtpmessage | |

# send8bit()

## Description

## Prototype

*smtpmessagevar*.send8bit ( smtpmessage *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | smtpmessage | |

# setauthinfo()

## Description

## Prototype

*smtpmessagevar*.setauthinfo ( smtpmessage *me*, integer *authtype*, string *username*, string *password*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | smtpmessage | |
| authtype | None | integer | |
| username | None | string | |
| password | None | string | |
| error | None | integer | |

# Chapter 86. smtpdatelib

This is an implementation of the SMTP date format with functions for delivering various types of formatting depending on requirements. See the source for full details.

# smtptimezoneinfo

## Description

## Type Tags

None

## Object Value

Objects of type smtptimezoneinfo have no value, and it is an error to try to get or set this value.

## smtptimezoneinfo.new()

### Description

### Prototype

*smtptimezoneinfo*.new ( smtptimezoneinfo *me*, integer *index*, string *timezone*, string *dsttimezone* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | smtptimezoneinfo | |
| index | None | integer | |
| timezone | None | string | |
| dsttimezone | None | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| dsttimezone | string | |
| index | integer | |
| timezone | string | |
| type | type | |

# gettimezoneinformation()

## Description

## Prototype

```
gettimezoneinformation ()
```

## Parameters

None

# smtp_datestr()

## Description

This returns the datetime object passed in valid SMTP date format. This depends on the format string passed. The format specification is quite extensive. The era portion is currently unsupported.

- d – Numeric representation of the day of the month, without leading zeros
- dd – Numeric representation of the day of the month, with leading zeros
- ddd – Three-letter abbreviation for the day of the week
- dddd – Full name of the day of the week
- M – Numeric representation of the month, without leading zeros
- MM – Numeric representation of the month, with leading zeros
- MMM – Three-letter abbreviation for the name of the month
- MMMM – Full name of the month
- y – Year represented by only the last two digits, without leading zeros if less than 10
- yy – Year represented by only the last two digits
- yyyy – Year represented by all four digits
- gg – Period/era string
- h – Hours without leading zeros (12-hour clock)
- hh – Hours with leading zeros (12-hour clock)
- H – Hours without leading zeros (24-hour clock)
- HH – Hours with leading zeros (24-hour clock)
- m – Minutes without leading zeros

- mm – Minutes with leading zeros
- s – Seconds without leading zeros
- ss – Seconds with leading zeros
- t – One-character time-marker string (for example, "a" and "p")
- tt – Multicharacter time-marker string (for example, "AM" and "PM")

## Prototype

`smtp_datestr` ( datetime *`dt`*, string *`sFormat`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dt | None | datetime | |
| sFormat | None | string | |

# smtp_timezonelist()

## Description

## Prototype

`smtp_timezonelist` ()

## Parameters

None

# Chapter 87. sortlib

This is a library that implements various sort algorithms, including insertionsort, quicksort,

# AVLTree

## Description

## Type Tags

BinarySearchTree

## Object Value

Objects of type AVLTree have no value, and it is an error to try to get or set this value.

## AVLTree.new()

### Description

### Prototype

*`AVLTree`*.new ( AVLTree *`me`*, boolean *`allowDuplicates`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | AVLTree | |
| allowDuplicates | `.false` | boolean | |

## Properties

| Property | Type | Description |
|---|---|---|
| put | function | |
| rebalance | function | |
| rotateLeft | function | |
| rotateRight | function | |
| tree | BinarySearchTree | |
| type | type | |
| updateBalance | function | |

# Methods

## put()

### Description

### Prototype

*AVLTreevar*.put ( AVLTree *me*, anyvalue *key*, type(*) *data* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | AVLTree | |
| key | None | anyvalue | |
| data | None | type(*) | |

## rebalance()

### Description

### Prototype

*AVLTreevar*.rebalance ( AVLTree *me*, TreeNode *node* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | AVLTree | |
| node | None | TreeNode | |

## rotateLeft()

### Description

### Prototype

*AVLTreevar*.rotateLeft ( AVLTree *me*, TreeNode *rotRoot* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | AVLTree | |
| rotRoot | None | TreeNode | |

## rotateRight()

### Description

### Prototype

*AVLTreevar*.rotateRight ( AVLTree *me*, TreeNode *rotRoot* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | AVLTree | |
| rotRoot | None | TreeNode | |

## updateBalance()

### Description

### Prototype

*AVLTreevar*.updateBalance ( AVLTree *me*, TreeNode *node* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | AVLTree | |
| node | None | TreeNode | |

# BinarySearchTree

# Description

# Type Tags

BinarySearchTree

# Object Value

Objects of type BinarySearchTree have no value, and it is an error to try to get or set this value.

# BinarySearchTree.new()

## Description

## Prototype

*BinarySearchTree*.new ( BinarySearchTree *me*, boolean *allowDuplicates* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BinarySearchTree | |
| allowDuplicates | `.false` | boolean | |

# Properties

| Property | Type | Description |
|---|---|---|
| allowDuplicates | boolean | |
| count | integer | |
| delete | function | |
| findnode | function | |
| get | function | |
| getfirst | function | |
| length | function | |
| put | function | |
| root | TreeNode | |
| type | type | |

# Methods

## delete()

### Description

### Prototype

*BinarySearchTreevar*.`delete` ( BinarySearchTree *`me`*, anyvalue *`key`*, integer *`error`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BinarySearchTree | |
| key | None | anyvalue | |
| error | None | integer | |

## findnode()

### Description

### Prototype

*BinarySearchTreevar*.findnode ( BinarySearchTree *me*, anyvalue *key* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BinarySearchTree | |
| key | None | anyvalue | |

## get()

### Description

### Prototype

*BinarySearchTreevar*.get ( BinarySearchTree *me*, anyvalue *key* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BinarySearchTree | |
| key | None | anyvalue | |

## getfirst()

### Description

### Prototype

*BinarySearchTreevar*.getfirst ( BinarySearchTree *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BinarySearchTree | |

## put()

### Description

### Prototype

*BinarySearchTreevar*.put ( BinarySearchTree *me*, anyvalue *key*, type(*) *data* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | BinarySearchTree | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| key | None | anyvalue | |
| data | None | type(*) | |

# TreeNode

## Description

## Type Tags

TreeNode

## Object Value

Objects of type TreeNode have no value, and it is an error to try to get or set this value.

## TreeNode.new()

### Description

### Prototype

*TreeNode*.new ( TreeNode *me*, anyvalue *key*, type(*) *data*, TreeNode *left*, TreeNode *right*, TreeNode *parent* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |
| key | None | anyvalue | |
| data | None | type(*) | |
| left | None | TreeNode | |
| right | None | TreeNode | |
| parent | None | TreeNode | |

## Properties

| Property | Type | Description |
|---|---|---|
| balanceFactor | integer | |
| data | type(*) | |
| duplist | dlist | |
| findMin | function | |

| Property | Type | Description |
|---|---|---|
| findSuccessor | function | |
| hasAnyChildren | function | |
| hasBothChildren | function | |
| hasLeftChild | function | |
| hasRightChild | function | |
| isLeaf | function | |
| isLeftChild | function | |
| isRightChild | function | |
| isRoot | function | |
| key | anyvalue | |
| leftChild | TreeNode | |
| parent | TreeNode | |
| replaceNodeData | function | |
| rightChild | TreeNode | |
| spliceOut | function | |
| type | type | |

# Methods

## findMin()

### Description

### Prototype

*TreeNodevar*.findMin ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## findSuccessor()

### Description

### Prototype

*TreeNodevar*.findSuccessor ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## hasAnyChildren()

### Description

### Prototype

*TreeNodevar*.hasAnyChildren ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## hasBothChildren()

### Description

### Prototype

*TreeNodevar*.hasBothChildren ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## hasLeftChild()

### Description

### Prototype

*TreeNodevar*.hasLeftChild ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## hasRightChild()

### Description

### Prototype

*TreeNodevar*.hasRightChild ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## isLeaf()

### Description

### Prototype

*TreeNodevar*.isLeaf ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## isLeftChild()

### Description

### Prototype

*TreeNodevar*.isLeftChild ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## isRightChild()

### Description

### Prototype

*TreeNodevar*.isRightChild ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## isRoot()

### Description

### Prototype

*TreeNodevar*.isRoot ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

## replaceNodeData()

### Description

### Prototype

*TreeNodevar*.replaceNodeData ( TreeNode *me*, anyvalue *key*, type(*) *data*, TreeNode *leftChild*, TreeNode *rightChild* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |
| key | None | anyvalue | |
| data | None | type(*) | |
| leftChild | None | TreeNode | |
| rightChild | None | TreeNode | |

## spliceOut()

### Description

### Prototype

*TreeNodevar*.spliceOut ( TreeNode *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | TreeNode | |

# duplicateTreeNode

# Description

# Type Tags

None

# Object Value

Objects of type duplicateTreeNode have no value, and it is an error to try to get or set this value.

# duplicateTreeNode.new()

## Description

## Prototype

*duplicateTreeNode*.new ( duplicateTreeNode *me*, dlist *parent*, anyvalue *key*, type(*) *data* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | duplicateTreeNode | |
| parent | None | dlist | |
| key | None | anyvalue | |
| data | None | type(*) | |

# Properties

| Property | Type | Description |
|---|---|---|
| data | type(*) | |
| duplistnode | dlistnode | |
| key | anyvalue | |
| type | type | |

# QuickSort()

# Description

# Prototype

QuickSort ( array *a*, integer *L*, integer *R* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| L | None | integer | |
| R | None | integer | |

# SelectionSort()

## Description

## Prototype

`SelectionSort` ( array *`a`*, integer *`lower`*, integer *`upper`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| lower | None | integer | |
| upper | None | integer | |

# binarysearch()

## Description

## Prototype

`binarysearch` ( array *`a`*, anyvalue *`value`*, integer *`low`*, integer *`high`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| value | None | anyvalue | |
| low | None | integer | |
| high | None | integer | |

# combsort11()

## Description

## Prototype

`combsort11` ( array *`a`*, integer *`lower`*, integer *`upper`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| lower | None | integer | |
| upper | None | integer | |

# insertionsort()

## Description

## Prototype

`insertionsort` ( array *`a`*, integer *`lower`*, integer *`upper`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| lower | None | integer | |
| upper | None | integer | |

# objinsertionsort()

## Description

## Prototype

`objinsertionsort` ( array *`a`*, integer *`lower`*, integer *`upper`*, function *`compare`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| lower | None | integer | |
| upper | None | integer | |
| compare | None | function | |

# qsort()

## Description

## Prototype

`qsort` ( array *`a`*, integer *`base`*, integer *`count`*, function *`compare`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| base | None | integer | |
| count | None | integer | |
| compare | None | function | |

# quicksorterr()

## Description

## Prototype

`quicksorterr` ( array *`a`*, integer *`base`*, integer *`count`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| base | None | integer | |
| count | None | integer | |

# quicksortr()

## Description

## Prototype

`quicksortr` ( array *`a`*, integer *`base`*, integer *`count`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| base | None | integer | |
| count | None | integer | |

# quicksortri()

## Description

## Prototype

`quicksortri` ( array *`a`*, integer *`base`*, integer *`count`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| base | None | integer | |
| count | None | integer | |

# quicksortrit()

## Description

## Prototype

`quicksortrit` ( array *`a`*, integer *`base`*, integer *`count`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| a | None | array | |
| base | None | integer | |
| count | None | integer | |

# standardcompare()

## Description

## Prototype

`standardcompare` ( anyvalue *`elem1`*, anyvalue *`elem2`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| elem1 | None | anyvalue | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| elem2 | None | anyvalue | |

# Chapter 88. soundlib

## winsound

### Description

## Type Tags

None

## Object Value

Objects of type winsound have no value, and it is an error to try to get or set this value.

## winsound.new()

### Description

### Prototype

*winsound*.new ( winsound *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | winsound | |
| error | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _private | winsoundprivate | |
| playsoundfile | function | |
| playsystem-sound | function | |
| type | type | |

## Methods

### playsoundfile()

#### Description

### Prototype

*winsoundvar*.playsoundfile ( winsound *me*, string *sound*, boolean *sync* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | winsound | |
| sound | None | string | |
| sync | .true | boolean | |

## playsystemsound()

### Description

### Prototype

*winsoundvar*.playsystemsound ( winsound *me*, string *sound* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | winsound | |
| sound | None | string | |

# Chapter 89. sql1

This is the main interface to the SQL92 query engine for running reports against SIMPOL databases (both SBME and PPCS). The sqlq1 type is used to create and run queries.

# sqlq1

## Description

## Type Tags

sqlq1

## Object Value

Objects of type sqlq1 have no value, and it is an error to try to get or set this value.

## sqlq1.new()

### Description

### Prototype

*sqlq1*.new ( sqlq1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |

## Properties

| Property | Type | Description |
|---|---|---|
| __create-columnnode-trees | function | |
| __createkey-words | function | |
| __createnode-tree | function | |
| __makeplan | function | |
| __pre-parecolumnar-ray | function | |

| Property | Type | Description |
|---|---|---|
| __tokenize-clause | function | |
| adddb1table | function | |
| clearsettings | function | |
| columns | array | |
| defdisplayfor-mats | array | |
| filterfalse | boolean | |
| findcolumn-datatype | function | |
| findcolumn-source | function | |
| firstcolumn | __sql__column | |
| firstsql1table | sql1_table | |
| firstsqltable | __sql__table | |
| getcolumn-count | function | |
| getcolumn-datatype | function | |
| getcolumndis-playformat | function | |
| getcolum-ntableandfield-names | function | |
| getcolumntitle | function | |
| getcolumnval-ue | function | |
| getrow | function | |
| gettableindex | function | |
| gotallrows | boolean | |
| gotrow | boolean | |
| hastable | function | |
| keywords | array | |
| lastsql1table | sql1_table | |
| new | function | |
| nulliszero | boolean | |
| prepare | function | |
| prepared | boolean | |
| q1pi | q1planinst | |
| retrieverecord | function | |
| selectclause | string | |

| Property | Type | Description |
|---|---|---|
| selecttokens | array | |
| setcolumndis-playformat | function | |
| setdefaultfor-mats | function | |
| setnulliszero | function | |
| setselectclause | function | |
| setwhereclause | function | |
| sqlnode | __sql__parsetreenode | |
| type | type | |
| whereclause | string | |
| wheretokens | array | |

# Methods

## __createcolumnnodetrees()

### Description

### Prototype

*sqlq1var*.__createcolumnnodetrees ( sqlq1 *me*, array *tokens*, __sql__table *firstsqltable*, string *errormessage*, integer *errorindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| tokens | None | array | |
| firstsqltable | None | __sql__table | |
| errormessage | None | string | |
| errorindex | None | integer | |

## __createkeywords()

### Description

### Prototype

*sqlq1var*.__createkeywords ( sqlq1 *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |

# __createnodetree()

## Description

## Prototype

*sqlq1var*.__createnodetree ( sqlq1 *me*, array *tokens*, __sql__table *firstsqltable*, __sql__column *firstcolumn*, string *errormessage*, integer *errorindex* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| tokens | None | array | |
| firstsqltable | None | __sql__table | |
| firstcolumn | None | __sql__column | |
| errormessage | None | string | |
| errorindex | None | integer | |

# __makeplan()

## Description

## Prototype

*sqlq1var*.__makeplan ( sqlq1 *me*, __sql__table *firstsqltable*, __sql__parsetreenode *sqlnode*, __sql__column *firstcolumn* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| firstsqltable | None | __sql__table | |
| sqlnode | None | __sql__parsetreenode | |
| firstcolumn | None | __sql__column | |

# __preparecolumnarray()

## Description

## Prototype

*sqlq1var*.__preparecolumnarray ( sqlq1 *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |

# __tokenizeclause()

## Description

## Prototype

*sqlq1var*.__tokenizeclause ( sqlq1 *me*, string *clause*, array *tokens*, string *clausename*, string *errormessage*, integer *errorindex* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| clause | None | string | |
| tokens | None | array | |
| clausename | None | string | |
| errormessage | None | string | |
| errorindex | None | integer | |

# adddb1table()

## Description

## Prototype

*sqlq1var*.adddb1table ( sqlq1 *me*, type(db1table) *table*, string *aliasname* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| table | None | type(db1table) | |
| aliasname | None | string | |

# clearsettings()

## Description

## Prototype

*sqlq1var*.clearsettings ( sqlq1 *me*, boolean *gotrow*, boolean *gotallrows*, boolean *prepared* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| gotrow | `.false` | boolean | |
| gotallrows | `.false` | boolean | |
| prepared | `.false` | boolean | |

# findcolumndatatype()

## Description

## Prototype

*`sqlq1var`*`.findcolumndatatype` ( sqlq1 *`me`*, integer *`colno`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| colno | None | integer | |

# findcolumnsource()

## Description

## Prototype

*`sqlq1var`*`.findcolumnsource` ( sqlq1 *`me`*, integer *`colno`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| colno | None | integer | |

# getcolumncount()

## Description

## Prototype

*`sqlq1var`*`.getcolumncount` ( sqlq1 *`me`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |

# getcolumndatatype()

## Description

## Prototype

*sqlq1var*.getcolumndatatype ( sqlq1 *me*, integer *colno*, string *errormessage* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| colno | None | integer | |
| errormessage | None | string | |

# getcolumndisplayformat()

## Description

## Prototype

*sqlq1var*.getcolumndisplayformat ( sqlq1 *me*, integer *colno*, string *errormessage* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| colno | None | integer | |
| errormessage | None | string | |

# getcolumntableandfieldnames()

## Description

## Prototype

*sqlq1var*.getcolumntableandfieldnames ( sqlq1 *me*, integer *colno*, string *tablename*, string *fieldname*, string *errormessage* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| colno | None | integer | |
| tablename | None | string | |
| fieldname | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| errormessage | None | string | |

# getcolumntitle()

## Description

## Prototype

*sqlq1var*.getcolumntitle ( sqlq1 *me*, integer *colno*, string *errormessage* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| colno | None | integer | |
| errormessage | None | string | |

# getcolumnvalue()

## Description

## Prototype

*sqlq1var*.getcolumnvalue ( sqlq1 *me*, integer *colno*, string *errormessage* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| colno | None | integer | |
| errormessage | None | string | |

# getrow()

## Description

## Prototype

*sqlq1var*.getrow ( sqlq1 *me*, string *errormessage*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| errormessage | None | string | |
| error | None | integer | |

# gettableindex()

## Description

## Prototype

*sqlq1var*.gettableindex ( sqlq1 *me*, string *tablename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| tablename | None | string | |

# hastable()

## Description

## Prototype

*sqlq1var*.hastable ( sqlq1 *me*, string *tablename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| tablename | None | string | |

# prepare()

## Description

## Prototype

*sqlq1var*.prepare ( sqlq1 *me*, string *errormessage*, integer *errorindex* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| errormessage | None | string | |
| errorindex | None | integer | |

# retrieverecord()

## Description

### Prototype

*sqlq1var*.retrieverecord ( sqlq1 *me*, integer *tableindex*, string *errormessage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| tableindex | 1 | integer | |
| errormessage | None | string | |

## setcolumndisplayformat()

### Description

### Prototype

*sqlq1var*.setcolumndisplayformat ( sqlq1 *me*, integer *colno*, string *displayformat*, string *errormessage* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| colno | None | integer | |
| displayformat | None | string | |
| errormessage | None | string | |

## setdefaultformats()

### Description

### Prototype

*sqlq1var*.setdefaultformats ( sqlq1 *me*, string *defnumberformat*, string *defdateformat*, string *deftimeformat*, string *defdatetimeformat*, string *defintegerformat*, string *defbooleanformat* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| defnumberformat | None | string | |
| defdateformat | None | string | |
| deftimeformat | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defdatetimeformat | None | string | |
| defintegerformat | None | string | |
| defbooleanformat | None | string | |

# setnulliszero()

## Description

## Prototype

*sqlq1var*.setnulliszero ( sqlq1 *me*, boolean *nulliszero* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| nulliszero | .true | boolean | |

# setselectclause()

## Description

## Prototype

*sqlq1var*.setselectclause ( sqlq1 *me*, string *selectclause*, string *errormessage*, integer *errorindex* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| selectclause | None | string | |
| errormessage | None | string | |
| errorindex | None | integer | |

# setwhereclause()

## Description

## Prototype

*sqlq1var*.setwhereclause ( sqlq1 *me*, string *whereclause*, string *errormessage*, integer *errorindex* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sqlq1 | |
| whereclause | None | string | |
| errormessage | None | string | |
| errorindex | None | integer | |

# Chapter 90. STR

This is an implementation of the STR$() function from SBL. It is as compatible as possible, including many of the strange effects as found in Superbase's STR$() function. It does not currently support scientific notation.

# SBLNumSettings

## Description

## Type Tags

None

## Object Value

Objects of type SBLNumSettings have no value, and it is an error to try to get or set this value.

## SBLNumSettings.new()

### Description

### Prototype

*SBLNumSettings*.new ( SBLNumSettings *me*, string *sCS*, string *sTS*, string *sDS*, boolean *bCS* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | SBLNumSettings | |
| sCS | £ | string | |
| sTS | , | string | |
| sDS | . | string | |
| bCS | .false | boolean | |

## Properties

| Property | Type | Description |
|---|---|---|
| CurrencySuffix | boolean | |
| CurrencySymbol | string | |
| DecimalSeparator | string | |

| Property | Type | Description |
|---|---|---|
| Thousand-sSeparator | string | |
| type | type | |

# STR()

## Description

Return the text equivalent of a numeric expression by evaluating a pattern string, the value, and the locale information. The pattern or format string controls various features, such as if a currency symbol, percent symbol, or thousands separator is displayed, and how many digits are shown on either side of the decimal symbol. Valid pattern symbols include:

- "9" – displays digits if they are present
- "0" – displays a 0 as a placeholder if no digit was present for that position
- "*" – fill unused digits with the asterisk symbol (only valid on the left side of the decimal point) – typically used for check writing systems
- "-" – allow room for the sign character
- "+" – allow room for the sign character and display even positive values with a leading +
- "$" – displays the currency symbol as defined in the locale information directly before or after the value (depends on locale information)
- " $" – displays the currency symbol as defined in the locale information separated by a space before or after the value (depends on locale information)
- "%" – displays the percent symbol after the value
- "(" – shows negative values in parentheses
- "z" – shows zero values as blank spaces
- "." – represents the decimal point when in combination with digit characters, when used alone it means to use as much space as required for the value, and no more
- "," – formats using the defined thousands separator character (every three), as defined in the locale information

## Prototype

STR ( number *value*, string *pattern*, SBLNumSettings *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| value | None | number | |
| pattern | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | SBLNumSettings | |

# Chapter 91. stringlib

This is a very useful library of functions for working with strings. This includes parsing functions, the ltrim() and rtrim() functions, the formatlinebreaks() function for formatting output to fit in a specific length, the nondigits() and nondigitsordecimal() functions for returning a string containing all the characters to ignore when converting to a value using .toval(), and the multiinstr() function that looks for the first instance of several different characters in a string.

# MakeNotNull()

## Description

## Prototype

`MakeNotNull` ( string *`sValue`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sValue | None | string | |

# afterstr()

## Description

This function searches the input string for the search string and if it is found, returns the text following the first instance of the search text, otherwise it returns the empty string.

## Prototype

`afterstr` ( string *`sInput`*, string *`sSearch`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sInput | None | string | |
| sSearch | None | string | |

# beforestr()

## Description

This function searches the input string for the search string and if it is found, returns the text up to and including the search text from the left of the string. If it does not find the search text it returns the whole string.

## Prototype

`beforestr` ( string *`sInput`*, string *`sSearch`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sInput | None | string | |
| sSearch | None | string | |

# fcase()

## Description

This function converts the incoming string to first case (where each area of text separated by spaces has the initial character capitalized).

## Prototype

`fcase` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# findfileencoding()

## Description

This function attempts to detect if the string passed is Unicode and if so which encoding it uses. This supports using a byte order mark as well as not having one. It is meant to be used on the initial chunk of a file, and it is recommended to pass around 240 bytes or so if possible which should be read from the start of file using 1 byte per character.

### Note

It is horribly difficult to determine encoding this way. It should only be used as a starting point for determining the encoding type. So if the return value is 1, it is not 100% sure that it is ANSI. Asking the user is always best.

## Prototype

`findfileencoding` ( string *`s`*, boolean *`hasBOM`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| hasBOM | None | boolean | |

# formatlinebreaks()

## Description

This function processes the incoming text and rewrites it to fit certain requirements, such as being indented a certain amount and having lines of a certain maximum length. It is commonly used to reformat content for pretty printers or when converting from one format to another or when outputting some results into a text-based format.

### Note

It is important to realize that this code will not remove existing line breaks, only break lines that are too long. To use this code to reformat existing text, remove all line breaks from the input first!

## Prototype

`formatlinebreaks` ( string *`sIndent`*, string *`sOrig`*, integer *`iLinewidth`*, string *`sEOL`*, boolean *`preservewhitespace`*, boolean *`skipindentonfirstline`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sIndent | None | string | |
| sOrig | None | string | |
| iLinewidth | None | integer | |
| sEOL | | string | |
| preservewhite-space | `.false` | boolean | |
| skipindenton-firstline | `.false` | boolean | |

# getlastitem()

## Description

This function steps through a string looking for a specific separator character. When it finds the last instance, or if it finds no instance it returns the remainder (or all) of the string. This is useful if you have a list of items in a string and you only need to retrieve the last item, such as when looking for the message responded to in an NNTP References header field, where the message responded to will always be the

last entry. Returns a string containing the last substring found following the last instance of the sSeparator character, or the whole string if no instance of the sSeparator character is found.

## Prototype

`getlastitem` ( string *`sSrc`*, string *`sSeparator`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSrc | None | string | |
| sSeparator | None | string | |

# getline()

## Description

This function simply returns a string value representing the text starting at *`iPos`* until the next *`sEOLchar`* or the end of the string and then increments the *`iPos`* value to point to the first character beyond the *`sEOLchar`* or to the end of the string.

## Prototype

`getline` ( string *`sSrc`*, integer *`iPos`*, string *`sEOLchar`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSrc | None | string | |
| iPos | None | integer | |
| sEOLchar | | string | |

# getnumericvalue()

## Description

## Prototype

`getnumericvalue` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# iseolchar()

## Description

This function evaluates the first character of the string passed and returns `.true` or `.false` depending on whether the character is an end of line character (carriage return 0x0D or linefeed 0x0A).

## Prototype

`iseolchar` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# ismatchingpattern()

## Description

This function compares the string passed in s against the pattern. The pattern consists of either "A" for alpha characters, "9" for digits, and otherwise the character in the pattern must match the character in the input string. For example: "999-99-9999" would match a US Social Security number. The UK postcodes system has 6 patterns: "AA9A 9AA", "A9A 9AA", "A9 9AA", "A99 9AA", "AA9 9AA", "AA99 9AA", and each pattern is valid with or without the space, so either each would need to be checked twice, or the space could be extracted first and removed from the patterns.

## Prototype

`ismatchingpattern` ( string *`s`*, string *`pattern`*, boolean *`partialmatchok`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| pattern | None | string | |
| partialmatchok | `.false` | boolean | |

# isspace()

## Description

This function evaluates the first character of the string passed and returns `.true` or `.false` depending on whether the character is either a space, a tab or something else.

## Prototype

`isspace` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# iswhitespace()

## Description

This function evaluates the first character of the string passed and returns `.true` or `.false` depending on whether the character is one of either space, tab, carriage return, or linefeed.

## Prototype

`iswhitespace` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# laststr()

## Description

This function searches the input string for each instance of the search string and returns the text following the last instance of the search string. No change is made to the original input string.

## Prototype

`laststr` ( string *`sInput`*, string *`sSearch`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sInput | None | string | |
| sSearch | None | string | |

# lpad()

## Description

This function pads a string on the left by inserting spaces or optionally characters matching the one passed in *`c`* until it is the same length as the value passed in *`n`*. If the string is longer than *`n`*, it will be truncated from the right to the size passed in *`n`*.

## Prototype

`lpad` ( string *`s`*, integer *`n`*, string *`c`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| n | None | integer | |
| c | | string | |

# ltrim()

## Description

This function calls `isspace()` or else compares with the contents of *`sCharlist`* for each character from the left of the string until it reaches a character that is not considered to be a space (space and tab) and then returns the remaining string. The return value is a string with all leading spaces and tab characters removed up until the first character that is not one of those two. If the optional *`sCharlist`* parameter is passed, then the characters that comprise the *`sCharlist`* will be used instead of the `isspace()` function to decide whether a character should be trimmed.

## Prototype

`ltrim` ( string *`sString`*, string *`sCharlist`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sString | None | string | |
| sCharlist | None | string | |

# multiinstr()

## Description

This function performs a .instr() for each of the characters in the searchchars variable against the src string. The character found closest to the beginning of the string is returned in termchar and its position is the return value of the function. This is particularly useful when any of several characters may be expected when parsing a string and it is important to find the first one, but without the expense of stepping through the string character by character. Returns the position of the character returned in termchar counted from the first character of the string. If none of the characters passed is found, then 0 will be returned and termchar will be the empty string.

## Prototype

`multiinstr` ( string *`src`*, string *`searchchars`*, string *`termchar`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| src | None | string | |
| searchchars | None | string | |
| termchar | None | string | |

# nondigits()

## Description

This function is meant to be used to return a string that can be used by `.toval()` in the *`ignorechars`* position to make sure that a value is converted to an integer.

## Prototype

`nondigits` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# nondigitsordecimal()

## Description

This function is meant to be used to return a string that can be used by `.toval()` in the *`ignorechars`* position to make sure that a value is converted to a number or integer depending on input. Returns the string minus anything that is a digit in the range of 0-9, the decimal point and the square brackets used for repeating decimals. This does not handle localized decimal characters.

## Prototype

`nondigitsordecimal` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# onechar2twochar()

## Description

## Prototype

`onechar2twochar` ( string *`sSrc`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSrc | None | string | |

# parseitem()

## Description

This function searches through a string looking for a specific target value. If it finds it, it then returns the value following the target on the same line either until the end of the line or the end of the string. Returns the string containing the value of the item being sought or the empty string if the item was not found.

## Prototype

`parseitem` ( string *`s`*, string *`sTarget`*, string *`terminatorlist`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| sTarget | None | string | |
| terminatorlist | | string | |

# parsemultitoken()

## Description

## Prototype

`parsemultitoken` ( string *`sSrc`*, integer *`iPos`*, string *`sSeparators`*, string *`sTerminator`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSrc | None | string | |
| iPos | None | integer | |
| sSeparators | None | string | |
| sTerminator | None | string | |

# parsenext()

## Description

This function searches through a string looking for a specific start boundary (can be one or more characters). It then returns the substring beginning from that boundary until the end boundary is reached or the end of the string. Returns a string containing the substring found between the beginning and ending character not including the boundary characters. If the first character is not found, returns "", if the second is not found, returns the remainder of the string following the start character.

## Prototype

`parsenext` ( string *`s`*, string *`startchar`*, string *`endchar`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| startchar | None | string | |
| endchar | None | string | |

# parsestr()

## Description

This function searches the input string for the search string and if it is found, returns the text up to and including the search text from the left of the string. If it does not find the search text it returns the whole string. The original *`sInput`* is modified to reflect what is left after searching for the first instance of the delimiter string.

## Prototype

`parsestr` ( string *`sInput`*, string *`sSearch`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sInput | None | string | |
| sSearch | None | string | |

# parsetoken()

## Description

This function simply returns a string representing the text starting at *`iPos`* until the next *`sSeparator`* or the end of the string and then increments the *`iPos`* value to point to the first character beyond the *`sSeparator`* or to the string length. This is a very memory efficient way of parsing through a string.

## Prototype

`parsetoken` ( string *`sSrc`*, integer *`iPos`*, string *`sSeparator`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSrc | None | string | |
| iPos | None | integer | |
| sSeparator | None | string | |

# rpad()

## Description

This function pads a string on the right by inserting spaces or optionally characters matching the one passed in *`c`* until it is the same length as the value passed in *`n`*. If the string is longer than *`n`*, it will be truncated from the right to the size passed in *`n`*.

## Prototype

`rpad` ( string *`s`*, integer *`n`*, string *`c`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| n | None | integer | |
| c | | string | |

# rtrim()

## Description

This function calls `isspace()` or else compares with the contents of *`sCharlist`* for each character from the right of the string until it reaches a character that is not considered to be a space (space and tab) and then returns the remaining string. The return value is a string with all trailing spaces and tab characters removed up until the first character that is not one of those two. If the optional *`sCharlist`* parameter is passed, then the characters that comprise the *`sCharlist`* will be used instead of the `isspace()` function to decide whether a character should be trimmed.

## Prototype

`rtrim` ( string *`sString`*, string *`sCharlist`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sString | None | string | |
| sCharlist | None | string | |

# space()

## Description

This function creates a string containing the number of spaces as requested in the *i* parameter.

## Prototype

`space` ( integer *i* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| i | None | integer | |

# sprintf()

## Description

## Prototype

`sprintf` ( string *s*, array *svalues* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| svalues | None | array | |

# twochar2onechar()

## Description

## Prototype

`twochar2onechar` ( string *sSrc* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sSrc | None | string | |

# Chapter 92. tableview

## ttableview

### Description

### Type Tags

None

### Object Value

Objects of type ttableview have no value, and it is an error to try to get or set this value.

## ttableview.new()

### Description

### Prototype

*ttableview*.new ( ttableview *me*, integer *width*, integer *height*, type(db1table) *table*, type(db1index) *idx*, type(wxcontainer) *window*, integer *highlightbackcolor*, integer *oddbackcolor*, integer *evenbackcolor*, string *defboolean*, string *definteger*, string *defnumber*, string *defdate*, string *deftime*, string *defdatetime*, SBLlocaledateinfo *datelocale*, SBLNumSettings *numlocale*, wxfont *font*, type(db1record) *record*, tdataview *dataview*, boolean *useview*, string *inifilename*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| width | None | integer | |
| height | None | integer | |
| table | None | type(db1table) | |
| idx | None | type(db1index) | |
| window | None | type(wxcontainer) | |
| highlightbackcolor | 12632256 | integer | |
| oddbackcolor | 16777215 | integer | |
| evenbackcolor | 16777168 | integer | |
| defboolean | T\|F | string | |
| definteger | . | string | |
| defnumber | 999999.00 | string | |
| defdate | yyyy.0m.0d | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| deftime | `hh:mm:ss` | string | |
| defdatetime | None | string | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| font | None | wxfont | |
| record | None | type(db1record) | |
| dataview | None | tdataview | |
| useview | `.false` | boolean | |
| inifilename | None | string | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | ttableviewprivate | |
| assignfilterob-ject | function | |
| clear | function | |
| columninfo | array | |
| columnin-foview | array | |
| currentrow | integer | |
| currentrow-modified | boolean | |
| currentview | tdataview | |
| datelocale | SBLlocaledateinfo | |
| defboolean | string | |
| defdate | string | |
| defdatetime | string | |
| definteger | string | |
| defnumber | string | |
| deftime | string | |
| deleterecord | function | |
| displayformat | string | |
| displaymes-sages | boolean | |
| duplicaterecord | function | |
| evenbackcolor | integer | |

| Property | Type | Description |
|---|---|---|
| filter | dataform1filter | |
| font | wxfont | |
| form | dataform1 | |
| getcolinfo | function | |
| getindexdis-playformat | function | |
| grid | dataform1grid | |
| gridinfo | ttableviewgridinfo | |
| height | integer | |
| hide | function | |
| highlightback-color | integer | |
| idx | type(db1index) | |
| inifilename | string | |
| isreadonly | boolean | |
| lastusedrecord | type(db1record) | |
| messagetitle | string | |
| newrecord | function | |
| numlocale | SBLNumSettings | |
| oddbackcolor | integer | |
| ondelete | event | |
| onnewrecord | event | |
| onsave | event | |
| records | array | |
| resize | function | |
| saverecord | function | |
| selectcurrent | function | |
| selectfirst | function | |
| selectkey | function | |
| selectlast | function | |
| selectnext | function | |
| selectnextpage | function | |
| selectprevious | function | |
| selectprevious-page | function | |
| setdataview | function | |
| setfilter | function | |
| setinitialrecord | function | |

| Property | Type | Description |
|---|---|---|
| setlastuse-drecord | function | |
| setreadonly | function | |
| settable | function | |
| show | function | |
| showview | function | |
| table | type(db1table) | |
| temprecord | type(db1record) | |
| type | type | |
| updategridrow | function | |
| useview | boolean | |
| width | integer | |
| window | type(wxcontainer) | |

# Methods

## assignfilterobject()

### Description

### Prototype

*ttableviewvar*.assignfilterobject ( ttableview *me*, dataform1filter *dffilter* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| dffilter | None | dataform1filter | |

## clear()

### Description

### Prototype

*ttableviewvar*.clear ( ttableview *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |

# deleterecord()

## Description

## Prototype

*ttableviewvar*.deleterecord ( ttableview *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| error | None | integer | |

# duplicaterecord()

## Description

## Prototype

*ttableviewvar*.duplicaterecord ( ttableview *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| error | None | integer | |

# getcolinfo()

## Description

## Prototype

*ttableviewvar*.getcolinfo ( ttableview *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |

# getindexdisplayformat()

## Description

## Prototype

*ttableviewvar*.getindexdisplayformat ( ttableview *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |

## hide()

### Description

### Prototype

*ttableviewvar*.hide ( ttableview *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |

## newrecord()

### Description

### Prototype

*ttableviewvar*.newrecord ( ttableview *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |

## resize()

### Description

### Prototype

*ttableviewvar*.resize ( ttableview *me*, integer *width*, integer *height* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| width | None | integer | |
| height | None | integer | |

# saverecord()

## Description

## Prototype

*ttableviewvar*.saverecord ( ttableview *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| error | None | integer | |

# selectcurrent()

## Description

## Prototype

*ttableviewvar*.selectcurrent ( ttableview *me*, type(db1index) *index*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| index | None | type(db1index) | |
| lock | .false | boolean | |
| error | None | integer | |

# selectfirst()

## Description

## Prototype

*ttableviewvar*.selectfirst ( ttableview *me*, boolean *lock*, boolean *setcolwidths*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| lock | .false | boolean | |
| setcolwidths | .false | boolean | |
| error | None | integer | |

# selectkey()

## Description

## Prototype

*ttableviewvar*.selectkey ( ttableview *me*, anyvalue *value*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| value | None | anyvalue | |
| lock | .false | boolean | |
| error | None | integer | |

# selectlast()

## Description

## Prototype

*ttableviewvar*.selectlast ( ttableview *me*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| lock | .false | boolean | |
| error | None | integer | |

# selectnext()

## Description

## Prototype

*ttableviewvar*.selectnext ( ttableview *me*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| lock | .false | boolean | |
| error | None | integer | |

# selectnextpage()

## Description

## Prototype

*ttableviewvar*.selectnextpage ( ttableview *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| error | None | integer | |

# selectprevious()

## Description

## Prototype

*ttableviewvar*.selectprevious ( ttableview *me*, boolean *lock*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| lock | .false | boolean | |
| error | None | integer | |

# selectpreviouspage()

## Description

## Prototype

*ttableviewvar*.selectpreviouspage ( ttableview *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| error | None | integer | |

# setdataview()

## Description

### Prototype

*ttableviewvar*.setdataview ( ttableview *me*, array *fieldlist*, string *name*, tdataview *dataview*, boolean *useview*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| fieldlist | None | array | |
| name | None | string | |
| dataview | None | tdataview | |
| useview | .true | boolean | |
| error | None | integer | |

## setfilter()

### Description

### Prototype

*ttableviewvar*.setfilter ( ttableview *me*, string *filter*, string *errtext*, integer *errindex* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| filter | None | string | |
| errtext | None | string | |
| errindex | None | integer | |

## setinitialrecord()

### Description

### Prototype

*ttableviewvar*.setinitialrecord ( ttableview *me*, type(db1record) *r*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| r | None | type(db1record) | |
| error | None | integer | |

# setlastusedrecord()

## Description

## Prototype

*ttableviewvar*.setlastusedrecord ( ttableview *me*, type(db1record) *record* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| record | None | type(db1record) | |

# setreadonly()

## Description

## Prototype

*ttableviewvar*.setreadonly ( ttableview *me*, boolean *setreadonly* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| setreadonly | .false | boolean | |

# show()

## Description

## Prototype

*ttableviewvar*.show ( ttableview *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |

# showview()

## Description

## Prototype

*ttableviewvar*.showview ( ttableview *me*, boolean *show* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| show | `.true` | boolean | |

## updategridrow()

### Description

### Prototype

*`ttableviewvar`*.`updategridrow` ( ttableview *`me`*, integer *`row`*, type(db1record) *`r`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | ttableview | |
| row | None | integer | |
| r | None | type(db1record) | |

# Chapter 93. timer

This is a mulit-threaded timer implementation which can create a timer to call an event-handling function at a specific time or at a regular interval.

# timer

## Description

## Type Tags

timer

## Object Value

Objects of type timer have no value, and it is an error to try to get or set this value.

## timer.new()

### Description

### Prototype

*timer*.new ( timer *me*, datetime *alarm*, integer *repeat*, integer *interval* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | timer | |
| alarm | None | datetime | |
| repeat | None | integer | |
| interval | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | timerprivate | |
| getinfo | function | |
| ontimer | event | |
| setinfo | function | |

| Property | Type | Description |
|---|---|---|
| start | function | |
| started | boolean | |
| stop | function | |
| type | type | |

# Methods

## getinfo()

### Description

### Prototype

*timervar*.getinfo ( timer *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | timer | |

## setinfo()

### Description

### Prototype

*timervar*.setinfo ( timer *me*, datetime *alarm*, integer *repeat*, integer *interval*, integer *microinterval* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | timer | |
| alarm | None | datetime | |
| repeat | None | integer | |
| interval | None | integer | |
| microinterval | 10000 | integer | |

## start()

### Description

### Prototype

*timervar*.start ( timer *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | timer | |

## stop()

### Description

### Prototype

*timervar*.stop ( timer *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | timer | |

# timerinfo

# Description

# Type Tags

None

# Object Value

Objects of type timerinfo have no value, and it is an error to try to get or set this value.

# timerinfo.new()

## Description

## Prototype

*timerinfo*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| alarm | datetime | |

| Property | Type | Description |
|---|---|---|
| interval | integer | |
| repeat | integer | |
| type | type | |

# Chapter 94. TRIM

This is the SBL-comaptible implementation of the TRIM() function.

# TRIM()

## Description

Returns the string value passed with all trailing space characters removed.

## Prototype

`TRIM` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# Chapter 95. uisyshelp

This is a very useful library for working with the user interface. It includes functions and types for retrieving the locale information (both Windows and Linux), a listpicker() for providing a pick list, functions for retrieving the screen size, default font, system colors, and much more. See the source for details.

# localecalendar

## Description

## Type Tags

None

## Object Value

Objects of type localecalendar have no value, and it is an error to try to get or set this value.

## localecalendar.new()

### Description

### Prototype

*localecalendar*.new()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| dayname1 | string | |
| dayname1abbrev | string | |
| dayname2 | string | |
| dayname2abbrev | string | |
| dayname3 | string | |
| dayname3abbrev | string | |
| dayname4 | string | |
| dayname4abbrev | string | |
| dayname5 | string | |
| dayname5abbrev | string | |
| dayname6 | string | |
| dayname6abbrev | string | |

| Property | Type | Description |
|---|---|---|
| dayname7 | string | |
| dayname7abbrev | string | |
| getdayabbrev | function | |
| getdaynames | function | |
| getmonthab-brev | function | |
| getmonth-names | function | |
| monthname1 | string | |
| monthname10 | string | |
| monthname10abbrev | string | |
| monthname11 | string | |
| monthname11abbrev | string | |
| monthname12 | string | |
| monthname12abbrev | string | |
| monthname13 | string | |
| monthname13abbrev | string | |
| monthname1abbrev | string | |
| monthname2 | string | |
| monthname2abbrev | string | |
| monthname3 | string | |
| monthname3abbrev | string | |
| monthname4 | string | |
| monthname4abbrev | string | |
| monthname5 | string | |
| monthname5abbrev | string | |
| monthname6 | string | |
| monthname6abbrev | string | |
| monthname7 | string | |
| monthname7abbrev | string | |
| monthname8 | string | |
| monthname8abbrev | string | |
| monthname9 | string | |
| monthname9abbrev | string | |
| type | type | |

# Methods

## getdayabbrev()

### Description

### Prototype

*localecalendarvar*.getdayabbrev ( localecalendar *me*, string *sepchar* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | localecalendar | |
| sepchar | , | string | |

## getdaynames()

### Description

### Prototype

*localecalendarvar*.getdaynames ( localecalendar *me*, string *sepchar* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | localecalendar | |
| sepchar | , | string | |

## getmonthabbrev()

### Description

### Prototype

*localecalendarvar*.getmonthabbrev ( localecalendar *me*, string *sepchar* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | localecalendar | |
| sepchar | , | string | |

## getmonthnames()

### Description

### Prototype

*localecalendarvar*.getmonthnames ( localecalendar *me*, string *sepchar* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | localecalendar | |
| sepchar | , | string | |

# localecodepage

## Description

## Type Tags

None

## Object Value

Objects of type localecodepage have no value, and it is an error to try to get or set this value.

## localecodepage.new()

### Description

### Prototype

*localecodepage*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| defansicode-page | integer | |
| defcountryid | integer | |
| deflanguageid | integer | |
| defoemcode-page | integer | |
| type | type | |

# localecountry

## Description

## Type Tags

None

## Object Value

Objects of type localecountry have no value, and it is an error to try to get or set this value.

## localecountry.new()

### Description

### Prototype

*localecountry*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| countryid | integer | |
| countryname | string | |
| countryname-abbrev | string | |
| country-nameenglish | string | |
| countryname-native | string | |
| type | type | |

# localecurrency

## Description

## Type Tags

None

# Object Value

Objects of type localecurrency have no value, and it is an error to try to get or set this value.

# localecurrency.new()

## Description

## Prototype

*localecurrency*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| currencydecimalseparator | string | |
| currencydigitgroupingpattern | string | |
| currencyfractionaldigitcount | integer | |
| currencyfractionaldigitcountintl | integer | |
| currencynegativenumberformat | integer | |
| currencysymbol | string | |
| currencysymbolformat | integer | |
| currencysymbolintl | string | |
| currencythousandsseparator | string | |
| type | type | |

# localedate

## Description

# Type Tags

None

# Object Value

Objects of type localedate have no value, and it is an error to try to get or set this value.

# localedate.new()

## Description

## Prototype

*localedate*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| dateformatlong | string | |
| dateformatorderlong | integer | |
| dateformatordershort | integer | |
| dateformatshort | string | |
| dateformatyearmonth | string | |
| dateseparator | string | |
| dateshort4yearcentury | boolean | |
| dateshortdayleadingzeros | boolean | |
| dateshortmonthleadingzeros | boolean | |
| time24hourformat | boolean | |
| timeamdesignator | string | |
| timeformat | string | |
| timeleadingzeros | boolean | |

| Property | Type | Description |
|---|---|---|
| timemarkerpre-fix | boolean | |
| timepmdesig-nator | string | |
| timeseparator | string | |
| type | type | |

# localeinfo

## Description

## Type Tags

None

## Object Value

Objects of type localeinfo have no value, and it is an error to try to get or set this value.

## localeinfo.new()

### Description

### Prototype

*`localeinfo`*.new ( localeinfo *`me`*, boolean *`forcelinux`* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | localeinfo | |
| forcelinux | `.false` | boolean | |

## Properties

| Property | Type | Description |
|---|---|---|
| SBLmonthab-brev | array | |
| calendar | localecalendar | |
| currency | localecurrency | |
| datetimeinfo | localedate | |
| getSBLmonthab-brev | function | |

| Property | Type | Description |
|---|---|---|
| languageinfo | localelanguage | |
| numeric | localenumeric | |
| other | localeother | |
| signinfo | localenumericsign | |
| type | type | |

# Methods

## getSBLmonthabbrev()

### Description

### Prototype

*localeinfovar*.getSBLmonthabbrev ( localeinfo *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | localeinfo | |

# localelanguage

# Description

# Type Tags

None

# Object Value

Objects of type localelanguage have no value, and it is an error to try to get or set this value.

# localelanguage.new()

## Description

## Prototype

*localelanguage*.new ()

## Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| languagename-abbrev | string | |
| type | type | |

# localenumeric

## Description

## Type Tags

None

## Object Value

Objects of type localenumeric have no value, and it is an error to try to get or set this value.

## localenumeric.new()

### Description

### Prototype

*localenumeric*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| decimalseparator | string | |
| digitgroupingpattern | string | |
| digitsnative | string | |
| fractionaldigitcount | integer | |
| negativenumberchar | string | |
| negativenumberformat | integer | |

| Property | Type | Description |
|---|---|---|
| thousandsseparator | string | |
| type | type | |

# localenumericsign

## Description

## Type Tags

None

## Object Value

Objects of type localenumericsign have no value, and it is an error to try to get or set this value.

## localenumericsign.new()

### Description

### Prototype

*localenumericsign*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| currencynegativespacesep | boolean | |
| currencynegativesymbolprecedes | boolean | |
| currencypositivespacesep | boolean | |
| currencypositivesymbolprecedes | boolean | |
| negativesign | string | |
| negativesignposition | integer | |

| Property | Type | Description |
|---|---|---|
| positivesign | string | |
| positivesignposition | integer | |
| type | type | |

# localeother

## Description

## Type Tags

None

## Object Value

Objects of type localeother have no value, and it is an error to try to get or set this value.

## localeother.new()

### Description

### Prototype

*localeother*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| digitsubstitution | integer | |
| listseparator | string | |
| metricsystem | boolean | |
| papersize | integer | |
| type | type | |

# syscolors

## Description

# Type Tags

None

# Object Value

Objects of type syscolors have no value, and it is an error to try to get or set this value.

# syscolors.new()

## Description

## Prototype

*syscolors*.new ( syscolors *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | syscolors | |

# Properties

| Property | Type | Description |
|---|---|---|
| colors | array | |
| count | integer | |
| getsyscolor | function | |
| type | type | |

# Methods

## getsyscolor()

### Description

### Prototype

*syscolorsvar*.getsyscolor ( syscolors *me*, integer *index* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | syscolors | |
| index | None | integer | |

# sysrgb

## Description

## Type Tags

None

## Object Value

Objects of type sysrgb have no value, and it is an error to try to get or set this value.

## sysrgb.new()

### Description

### Prototype

*sysrgb*.new ( sysrgb *me*, integer *red*, integer *green*, integer *blue*, integer *value* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | sysrgb | |
| red | `0` | integer | |
| green | `0` | integer | |
| blue | `0` | integer | |
| value | None | integer | |

## Properties

| Property | Type | Description |
|---|---|---|
| blue | integer | |
| green | integer | |
| red | integer | |
| type | type | |
| value | integer | |

# windowsversion

## Description

# Type Tags

None

# Object Value

Objects of type windowsversion have no value, and it is an error to try to get or set this value.

# windowsversion.new()

## Description

## Prototype

*windowsversion*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| BuildNumber | integer | |
| CSDVersion | string | |
| MajorVersion | integer | |
| MinorVersion | integer | |
| PlatformID | integer | |
| type | type | |

# wxformoptiongroup

# Description

# Type Tags

None

# Object Value

Objects of type wxformoptiongroup have no value, and it is an error to try to get or set this value.

# wxformoptiongroup.new()

## Description

## Prototype

*wxformoptiongroup*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| addmember | function | |
| members | list | |
| select | function | |
| type | type | |

# Methods

## addmember()

### Description

### Prototype

*wxformoptiongroupvar*.addmember ( wxformoptiongroup *me*, wxformoption *ob* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformoptiongroup | |
| ob | None | wxformoption | |

## select()

### Description

### Prototype

*wxformoptiongroupvar*.select ( wxformoptiongroup *me*, wxformoptiongroupmember *member* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformoptiongroup | |
| member | None | wxformoptiongroupmember | |

# wxformoptiongroupmember

## Description

## Type Tags

None

## Object Value

Objects of type wxformoptiongroupmember have no value, and it is an error to try to get or set this value.

## wxformoptiongroupmember.new()

### Description

### Prototype

*wxformoptiongroupmember*.new ( wxformoptiongroupmember *me*, wxformoption *optionbutton*, wxformoptiongroup *parent* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformoptiongroupmember | |
| optionbutton | None | wxformoption | |
| parent | None | wxformoptiongroup | |

## Properties

| Property | Type | Description |
|---|---|---|
| node | listnode | |
| onchange | event | |
| optionbutton | wxformoption | |
| parent | wxformoptiongroup | |
| type | type | |

# adjustbitmapbackgroundcolor()

## Description

## Prototype

adjustbitmapbackgroundcolor ( wxbitmap *bitmap*, sysrgb *targetcolor*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| bitmap | None | wxbitmap | |
| targetcolor | None | sysrgb | |
| error | None | integer | |

# appactivate()

## Description

## Prototype

`appactivate` ( string *`windowtitle`*, boolean *`exacttitlematch`*, integer *`hwnd`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| windowtitle | None | string | |
| exacttitlematch | `.false` | boolean | |
| hwnd | None | integer | |

# arrowdown_20x16()

## Description

## Prototype

`arrowdown_20x16` ()

## Parameters

None

# arrowdown_disabled_20x16()

## Description

## Prototype

`arrowdown_disabled_20x16` ()

## Parameters

None

# arrowdown_focus_20x16()

## Description

## Prototype

`arrowdown_focus_20x16` ()

## Parameters

None

# arrowdown_sel_20x16()

## Description

## Prototype

`arrowdown_sel_20x16` ()

## Parameters

None

# arrowup_20x16()

## Description

## Prototype

`arrowup_20x16` ()

## Parameters

None

# arrowup_disabled_20x16()

## Description

## Prototype

`arrowup_disabled_20x16` ()

## Parameters

None

# arrowup_focus_20x16()

## Description

## Prototype

`arrowup_focus_20x16` ()

## Parameters

None

# arrowup_sel_20x16()

## Description

## Prototype

`arrowup_sel_20x16` ()

## Parameters

None

# calendar_bmp()

## Description

## Prototype

`calendar_bmp` ()

## Parameters

None

# calendar_disabled_bmp()

## Description

## Prototype

`calendar_disabled_bmp` ()

## Parameters

None

# calendar_focus_bmp()

## Description

## Prototype

`calendar_focus_bmp` ()

## Parameters

None

# calendar_selfocus_bmp()

## Description

## Prototype

`calendar_selfocus_bmp` ()

## Parameters

None

# centerdialogonparent()

## Description

## Prototype

`centerdialogonparent` ( wxdialog *d* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| d | None | wxdialog | |

# centerwindowondisplay()

## Description

## Prototype

`centerwindowondisplay` ( wxwindow *`w`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| w | None | wxwindow | |

# choicelistdialog()

## Description

## Prototype

`choicelistdialog` ( type(wxdialogparent) *`parent`*, array *`choices`*, string *`captiontext`*, boolean *`onlyone`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| choices | None | array | |
| captiontext | `Select item(s)` | string | |
| onlyone | `.false` | boolean | |
| error | None | integer | |

# datepicker()

## Description

## Prototype

`datepicker` ( date *`startdate`*, type(wxdialogparent) *`parent`*, SBLlocaledateinfo *`datelocale`*, string *`format`*, string *`title`*, integer *`initialyear`*, integer *`yearcount`*, boolean *`ascending`*, boolean *`weekstartonmonday`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| startdate | None | date | |
| parent | None | type(wxdialogparent) | |
| datelocale | None | SBLlocaledateinfo | |
| format | `0d/mm/yy` | string | |
| title | `Calendar` | string | |
| initialyear | None | integer | |
| yearcount | `20` | integer | |
| ascending | `.true` | boolean | |
| weekstarton-monday | `.true` | boolean | |

# daysinmonth()

## Description

## Prototype

`daysinmonth` ( date *`d`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| d | None | date | |

# duallist()

## Description

## Prototype

`duallist` ( function *`fillfunction`*, type(*) *`fillreference`*, string *`response`*, type(wxdialogparent) *`parent`*, string *`leftcaption`*, string *`rightcaption`*, string *`okcaption`*,

string *cancelcaption*, string *dialogcaption*, boolean *leftlistisconstant*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| fillfunction | None | function | |
| fillreference | None | type(*) | |
| response | None | string | |
| parent | None | type(wxdialogparent) | |
| leftcaption | `Item List` | string | |
| rightcaption | `Target List` | string | |
| okcaption | `OK` | string | |
| cancelcaption | `Cancel` | string | |
| dialogcaption | `Select items from the left to place on the right` | string | |
| leftlistiscon-stant | `.false` | boolean | |
| error | None | integer | |

# findchildwindow()

## Description

## Prototype

`findchildwindow` ( integer *hwnd*, string *caption*, boolean *exactmatch* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| hwnd | None | integer | |
| caption | None | string | |
| exactmatch | `.true` | boolean | |

# findcombobliststring()

## Description

## Prototype

`findcombolistst``ring` ( type(*) *`lb`*, string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lb | None | type(*) | |
| s | None | string | |

# findwindowhandle()

## Description

## Prototype

`findwindowhandle` ( string *`windowtitle`*, boolean *`exacttitlematch`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| windowtitle | None | string | |
| exacttitlematch | `.false` | boolean | |

# getcenteredwindowrect()

## Description

## Prototype

`getcenteredwindowrect` ( integer *`width`*, integer *`height`*, point *`lefttop`*, point *`rightbot-tom`*, type(wxdialogparent) *`parent`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | None | integer | |
| height | None | integer | |
| lefttop | None | point | |
| rightbottom | None | point | |
| parent | None | type(wxdialogparent) | |
| error | None | integer | |

# getdate()

## Description

## Prototype

`getdate` ( date *`defdate`*, type(wxdialogparent) *`parent`*, string *`defdateformat`*, SBLlocaledate-info *`datelocale`*, string *`captiontext`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defdate | None | date | |
| parent | None | type(wxdialogparent) | |
| defdateformat | `0d/mm/yyyy` | string | |
| datelocale | None | SBLlocaledateinfo | |
| captiontext | `Enter the date` | string | |

# getdatetime()

## Description

## Prototype

`getdatetime` ( datetime *`defdatetime`*, type(wxdialogparent) *`parent`*, string *`defdateformat`*, SBLlocaledateinfo *`datelocale`*, string *`captiontext`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| defdatetime | None | datetime | |
| parent | None | type(wxdialogparent) | |
| defdateformat | `0d/mm/yyyy` | string | |
| datelocale | None | SBLlocaledateinfo | |
| captiontext | `Enter the date and time` | string | |

# getdefaultfont()

## Description

## Prototype

`getdefaultfont` ( integer *`pointsize`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| pointsize | None | integer | |

# getdisplaysize()

## Description

## Prototype

`getdisplaysize` ( integer *`width`*, integer *`height`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | None | integer | |
| height | None | integer | |

# getdpivalues()

## Description

## Prototype

`getdpivalues` ( integer *`dpix`*, integer *`dpiy`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| dpix | None | integer | |
| dpiy | None | integer | |

# getscrollbarsizes()

## Description

## Prototype

`getscrollbarsizes` ( integer *`width`*, integer *`height`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | None | integer | |
| height | None | integer | |

# gettime()

## Description

## Prototype

`gettime` ( time *`deftime`*, type(wxdialogparent) *`parent`*, string *`captiontext`*, boolean *`showsecs`*, boolean *`useampm`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| deftime | None | time | |
| parent | None | type(wxdialogparent) | |
| captiontext | `Enter the time` | string | |
| showsecs | `.true` | boolean | |
| useampm | `.false` | boolean | |

# getusabledisplaysize()

## Description

## Prototype

`getusabledisplaysize` ( integer *`width`*, integer *`height`*, integer *`left`*, integer *`top`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| width | None | integer | |
| height | None | integer | |
| left | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| top | None | integer | |

# getuserinput()

## Description

## Prototype

`getuserinput` ( string *`message`*, string *`title`*, anyvalue *`prompt`*, string *`response`*, boolean *`passwordstyle`*, integer *`maxlength`*, type(wxdialogparent) *`parent`*, type *`datatype`*, SBLlocale-dateinfo *`datelocale`*, SBLNumSettings *`numlocale`*, string *`displayformat`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| message | None | string | |
| title | None | string | |
| prompt | None | anyvalue | |
| response | None | string | |
| passwordstyle | `.false` | boolean | |
| maxlength | `.inf` | integer | |
| parent | None | type(wxdialogparent) | |
| datatype | None | type | |
| datelocale | None | SBLlocaledateinfo | |
| numlocale | None | SBLNumSettings | |
| displayformat | None | string | |
| error | None | integer | |

# getwindowsfolder()

## Description

## Prototype

`getwindowsfolder` ( integer *`folderid`*, integer *`error`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| folderid | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# getwindowspublicdocsfolder()

## Description

## Prototype

`getwindowspublicdocsfolder` ()

## Parameters

None

# getwindowstaskbarstateandpos()

## Description

## Prototype

`getwindowstaskbarstateandpos` ( point *`lt`*, point *`rb`*, boolean *`ishidden`*, integer *`edge`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| lt | None | point | |
| rb | None | point | |
| ishidden | None | boolean | |
| edge | None | integer | |

# getwindowsversion()

## Description

## Prototype

`getwindowsversion` ()

## Parameters

None

# getwindowsversionstring()

## Description

## Prototype

`getwindowsversionstring` ( windowsversion *v* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| v | None | windowsversion | |

# listpicker()

## Description

## Prototype

`listpicker` ( function *listfiller*, type(*) *ref*, string *result*, integer *width*, integer *height*, string *listcaption*, string *listtype*, string *okbutton*, string *cancelbutton*, string *captiontext*, string *fontname*, integer *pointsize*, integer *formbackgroundrgb*, integer *textrgb*, integer *listbackground*, boolean *centeronparent*, type(wxdialogparent) *parent*, boolean *modal*, function *nonmodalonclick*, type(*) *nonmodalonclickreference*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| listfiller | None | function | |
| ref | None | type(*) | |
| result | None | string | |
| width | `400` | integer | |
| height | `250` | integer | |
| listcaption | `Items` | string | |
| listtype | `single` | string | |
| okbutton | `OK` | string | |
| cancelbutton | `Cancel` | string | |
| captiontext | `Pick one` | string | |
| fontname | None | string | |
| pointsize | None | integer | |
| formback-groundrgb | None | integer | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| textrgb | None | integer | |
| listbackground | None | integer | |
| centeronparent | `.true` | boolean | |
| parent | None | type(wxdialogparent) | |
| modal | `.true` | boolean | |
| non-modalonclick | None | function | |
| non-modalonclick-reference | None | type(*) | |
| error | None | integer | |

# messagebox()

## Description

## Prototype

`messagebox` ( type(wxdialogparent) *`parent`*, string *`message`*, string *`captiontext`*, integer *`left`*, integer *`top`*, string *`stdbuttons`*, wxfont *`font`*, integer *`timeout`*, boolean *`usestdbtns`*, string *`result`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| parent | None | type(wxdialogparent) | |
| message | None | string | |
| captiontext | None | string | |
| left | None | integer | |
| top | None | integer | |
| stdbuttons | None | string | |
| font | None | wxfont | |
| timeout | `.inf` | integer | |
| usestdbtns | `.true` | boolean | |
| result | None | string | |

# padhex()

## Description

## Prototype

`padhex` ( integer *`i`*, integer *`length`*, boolean *`include0x`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| i | None | integer | |
| length | None | integer | |
| include0x | `.false` | boolean | |

# selectcombotextitem()

## Description

## Prototype

`selectcombotextitem` ( integer *`hwnd`*, string *`text`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| hwnd | None | integer | |
| text | None | string | |

# setcontroltext()

## Description

## Prototype

`setcontroltext` ( integer *`hwnd`*, string *`text`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| hwnd | None | integer | |
| text | None | string | |

# setfocus()

## Description

## Prototype

`setfocus` ( integer *`hwnd`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| hwnd | None | integer | |

# setwindowposition()

## Description

## Prototype

`setwindowposition` ( integer *`hwnd`*, integer *`left`*, integer *`top`*, integer *`width`*, integer *`height`*, boolean *`repaint`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| hwnd | None | integer | |
| left | None | integer | |
| top | None | integer | |
| width | None | integer | |
| height | None | integer | |
| repaint | None | boolean | |

# showcopyabletextmessage()

## Description

## Prototype

`showcopyabletextmessage` ( string *`msg`*, string *`title`*, type(wxdialogparent) *`parent`*, integer *`left`*, integer *`top`*, integer *`width`*, integer *`height`*, boolean *`centeronparent`*, string *`oktext`*, wx-font *`font`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| msg | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| title | `For your informa-tion...` | string | |
| parent | None | type(wxdialogparent) | |
| left | `1` | integer | |
| top | `1` | integer | |
| width | None | integer | |
| height | None | integer | |
| centeronparent | `.true` | boolean | |
| oktext | `OK` | string | |
| font | None | wxfont | |

# windows_getactivewindow()

## Description

## Prototype

`windows_getactivewindow` ()

## Parameters

None

# windows_redrawwindow()

## Description

## Prototype

`windows_redrawwindow` ( integer *hwnd* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| hwnd | None | integer | |

# wxformoptiongroupmemberselchange()

## Description

# Prototype

`wxformoptiongroupmemberselchange` ( wxformoption *`me`*, wxformoptiongroupmember *`member`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | wxformoption | |
| member | None | wxformoptiongroupmember | |

# Chapter 96. unittest

This is a basic unit testing library. It can be used to crreate and run regression tests for functions with various cases.

# testcase

## Description

## Type Tags

testcase

## Object Value

Objects of type testcase have no value, and it is an error to try to get or set this value.

# testcase.new()

## Description

## Prototype

*testcase*.new ( testcase *me*, string *name*, function *test*, type(*) *testref* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | testcase | |
| name | None | string | |
| test | None | function | |
| testref | None | type(*) | |

# Properties

| Property | Type | Description |
|---|---|---|
| addvalue | function | |
| cleanup | function | |
| defaultTestResult | function | |
| description | string | |
| fail | function | |

| Property | Type | Description |
|---|---|---|
| failIf | function | |
| failIfAlmostEqual | function | |
| failIfEqual | function | |
| failUnless | function | |
| failUnlessAlmostEqual | function | |
| failUnlessEqual | function | |
| failUnlessError | function | |
| failureError | integer | |
| name | string | |
| reporteqerror | function | |
| reporteqerror_definition | function | |
| result | string | |
| run | function | |
| setup | function | |
| test | function | |
| testref | type(*) | |
| type | type | |
| values | list | |

# Methods

## addvalue()

### Description

### Prototype

*testcasevar*.addvalue ( testcase *me*, type(testcasevalue) *v* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | testcase | |
| v | None | type(testcasevalue) | |

## reporteqerror()

### Description

### Prototype

*testcasevar*.reporteqerror ( testcase *me*, integer *id*, type(=) *expected*, type(=) *result* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | testcase | |
| id | None | integer | |
| expected | None | type(=) | |
| result | None | type(=) | |

## reporteqerror_datetimes()

### Description

### Prototype

*testcasevar*.reporteqerror_datetimes ( testcase *me*, integer *id*, type(*) *expected*, type(*) *result*, function *evalfunc*, string *pattern* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | testcase | |
| id | None | integer | |
| expected | None | type(*) | |
| result | None | type(*) | |
| evalfunc | None | function | |
| pattern | None | string | |

## run()

### Description

### Prototype

*testcasevar*.run ( testcase *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | testcase | |

# testcasevalue

# Description

# Type Tags

testcasevalue

# Object Value

Objects of type testcasevalue have no value, and it is an error to try to get or set this value.

# testcasevalue.new()

## Description

## Prototype

*testcasevalue*.new ( testcasevalue *me*, type(testcase) *parentcase* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | testcasevalue | |
| parentcase | None | type(testcase) | |

# Properties

| Property | Type | Description |
|---|---|---|
| node | listnode | |
| parentcase | testcase | |
| type | type | |

# testresult

# Description

# Type Tags

None

# Object Value

Objects of type testresult have no value, and it is an error to try to get or set this value.

# testresult.new()

## Description

## Prototype

*testresult*.new ( testresult *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | testresult | |

# Properties

| Property | Type | Description |
|---|---|---|
| addError | function | |
| addFailure | function | |
| addSuccess | function | |
| errors | array | |
| failures | array | |
| startTest | function | |
| stop | function | |
| stopTest | function | |
| success | boolean | |
| testsrun | integer | |
| type | type | |
| wasSuccessful | function | |

# Methods

## addError()

### Description

### Prototype

*testresultvar*.addError ( testcase *test*, integer *errtype*, anyvalue *value*, string *info* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| test | None | testcase | |
| errtype | None | integer | |
| value | None | anyvalue | |
| info | None | string | |

# Chapter 97. urlendecode

This library provides URL encoding and decoding functions. Very useful in web environbments ore when interacting with the web in some way.

# isURLalpha()

## Description

## Prototype

`isURLalpha` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLalphanum()

## Description

## Prototype

`isURLalphanum` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLcontrol()

## Description

## Prototype

`isURLcontrol` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLdelim()

## Description

## Prototype

`isURLdelim` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLdigit()

## Description

## Prototype

`isURLdigit` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLencodable()

## Description

## Prototype

`isURLencodable` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLexcluded()

## Description

## Prototype

`isURLexcluded` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLlower()

## Description

## Prototype

`isURLlower` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLmark()

## Description

## Prototype

`isURLmark` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLreserved()

## Description

## Prototype

`isURLreserved` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLunreserved()

## Description

## Prototype

`isURLunreserved` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLunwise()

## Description

## Prototype

`isURLunwise` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isURLupper()

## Description

## Prototype

`isURLupper` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# urldecode()

## Description

## Prototype

`urldecode` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# urlencode()

## Description

## Prototype

`urlencode` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# Chapter 98. urllib

# URL

## Description

## Type Tags

None

## Object Value

Objects of type URL have no value, and it is an error to try to get or set this value.

## URL.new()

### Description

### Prototype

*URL*.new ( URL *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | URL | |

## Properties

| Property | Type | Description |
|---|---|---|
| authority | string | |
| display | string | |
| fragment | string | |
| path | string | |
| query | string | |
| scheme | string | |
| type | type | |

# parseurl()

## Description

# Prototype

`parseurl` ( string *`testurl`* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| testurl | None | string | |

# Chapter 99. utf8lib

This library provides conversion functions from and to the UTF8 encodng format.

# ucs2_to_utf8()

## Description

## Prototype

`ucs2_to_utf8` ( string *`s`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# utf8_to_ucs2()

## Description

## Prototype

`utf8_to_ucs2` ( string *`s`*, string *`sResult`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| sResult | None | string | |

# Chapter 100. uuencode

## DecodeBase64()

### Description

### Prototype

`DecodeBase64` ( string *s* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

## EncodeBase64()

### Description

### Prototype

`EncodeBase64` ( string *s* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

## base64decode()

### Description

### Prototype

`base64decode` ( string *s1* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s1 | None | string | |

# base64encode()

## Description

## Prototype

`base64encode` ( string *`s1`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s1 | None | string | |

# decodequotedprintable()

## Description

## Prototype

`decodequotedprintable` ( string *`s`*, string *`codepage`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| codepage | None | string | |

# encodequotedprintable()

## Description

## Prototype

`encodequotedprintable` ( string *`s`*, string *`codepage`*, boolean *`useunsafechars`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |
| codepage | None | string | |
| useunsafechars | `.false` | boolean | |

# uudecodestring()

## Description

## Prototype

`uudecodestring` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# uuencodestring()

## Description

## Prototype

`uuencodestring` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# Chapter 101. VAL

Provides an SBL-comaptible implementation of VAL().

# VAL()

## Description

## Prototype

VAL ( string *sOrigValue*, SBLNumSettings *NumSettings* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sOrigValue | None | string | |
| NumSettings | None | SBLNumSettings | |

# isSBLalpha()

## Description

## Prototype

isSBLalpha ( string *sInput* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sInput | None | string | |

# isSBLdigit()

## Description

## Prototype

isSBLdigit ( string *sInput* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sInput | None | string | |

# Chapter 102. volatable

This is a complete volatile file library that implements the vola1base as an in memory, virtual database that is compliant to SBME and db1table. It does not currently implement the table modification functionality, only table creation.

# vola1base

## Description

## Type Tags

db1base

## Object Value

Objects of type vola1base have no value, and it is an error to try to get or set this value.

# vola1base.new()

### Description

### Prototype

*vola1base*.new ( vola1base *me*, string *filename* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| filename | `untitled` | string | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | vola1baseprivate | |
| cachesize | integer | |
| commit | function | |
| committype | string | |
| filename | string | |

| Property | Type | Description |
|---|---|---|
| gettablenames | function | |
| lock | function | |
| locktype | string | |
| newtable | function | |
| opentable | function | |
| rollback | function | |
| setcachesize | function | |
| setcommittype | function | |
| type | type | |
| unlock | function | |

# Methods

## commit()

### Description

### Prototype

*vola1basevar*.commit ( vola1base *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| error | None | integer | |

## gettablenames()

### Description

### Prototype

*vola1basevar*.gettablenames ( vola1base *me*, array *tablenames*, string *pattern*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| tablenames | None | array | |
| pattern | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# lock()

## Description

## Prototype

*vola1basevar*.lock ( vola1base *me*, string *locktype*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| locktype | None | string | |
| error | None | integer | |

# newtable()

## Description

## Prototype

*vola1basevar*.newtable ( vola1base *me*, string *tablename* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| tablename | None | string | |

# opentable()

## Description

## Prototype

*vola1basevar*.opentable ( vola1base *me*, string *tablename*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| tablename | None | string | |
| error | None | integer | |

# rollback()

## Description

## Prototype

*vola1basevar*.rollback ( vola1base *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| error | None | integer | |

# setcachesize()

## Description

## Prototype

*vola1basevar*.setcachesize ( vola1base *me*, integer *cachesize* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| cachesize | None | integer | |

# setcommittype()

## Description

## Prototype

*vola1basevar*.setcommittype ( vola1base *me*, string *committype* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| committype | None | string | |

# unlock()

## Description

### Prototype

*vola1basevar*.unlock ( vola1base *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1base | |
| error | None | integer | |

# vola1field

## Description

## Type Tags

db1field

## Object Value

Objects of type vola1field have no value, and it is an error to try to get or set this value.

## vola1field.new()

### Description

### Prototype

*vola1field*.new ()

### Parameters

None

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | vola1fieldprivate | |
| datatype | type | |
| index | vola1index | |
| name | string | |
| next | vola1field | |

| Property | Type | Description |
|---|---|---|
| table | vola1table | |
| type | type | |

# vola1index

## Description

## Type Tags

db1index

## Object Value

Objects of type vola1index have no value, and it is an error to try to get or set this value.

## vola1index.new()

### Description

### Prototype

*vola1index*.new ( vola1index *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1index | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | vola1indexprivate | |
| algorithm | string | |
| field | vola1field | |
| next | vola1index | |
| precision | integer | |
| select | function | |
| selectkey | function | |
| table | vola1table | |

| Property | Type | Description |
|---|---|---|
| type | type | |
| unique | boolean | |

# Methods

## select()

### Description

### Prototype

*vola1indexvar*.`select` ( vola1index *me*, boolean *lastrecord*, string *locktype*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1index | |
| lastrecord | `.false` | boolean | |
| locktype | None | string | |
| error | None | integer | |

## selectkey()

### Description

### Prototype

*vola1indexvar*.`selectkey` ( vola1index *me*, anyvalue *value*, string *locktype*, integer *error*, boolean *found* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1index | |
| value | None | anyvalue | |
| locktype | None | string | |
| error | None | integer | |
| found | None | boolean | |

# vola1newfield

# Description

# Type Tags

None

# Object Value

Objects of type vola1newfield have no value, and it is an error to try to get or set this value.

# vola1newfield.new()

## Description

## Prototype

*vola1newfield*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | vola1newfieldprivate | |
| datatype | type | |
| index | vola1newindex | |
| name | string | |
| next | vola1newfield | |
| table | vola1newtable | |
| type | type | |

# vola1newindex

# Description

# Type Tags

None

# Object Value

Objects of type vola1newindex have no value, and it is an error to try to get or set this value.

# vola1newindex.new()

## Description

## Prototype

*vola1newindex*.new ()

## Parameters

None

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | vola1newindexprivate | |
| algorithm | string | |
| field | vola1newfield | |
| next | vola1newindex | |
| precision | integer | |
| table | vola1newtable | |
| type | type | |
| unique | boolean | |

# vola1newtable

## Description

## Type Tags

None

## Object Value

Objects of type vola1newtable have no value, and it is an error to try to get or set this value.

## vola1newtable.new()

### Description

## Prototype

*vola1newtable*.new ( vola1newtable *me*, string *tablename*, vola1base *volabase* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1newtable | |
| tablename | None | string | |
| volabase | None | vola1base | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | vola1newtableprivate | |
| create | function | |
| firstfield | vola1newfield | |
| firstindex | vola1newindex | |
| newfield | function | |
| newindex | function | |
| tablename | string | |
| type | type | |
| volabase | vola1base | |

# Methods

## create()

### Description

### Prototype

*vola1newtablevar*.create ( vola1newtable *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1newtable | |
| error | None | integer | |

## newfield()

### Description

### Prototype

*vola1newtablevar*.newfield ( vola1newtable *me*, string *name*, type *datatype*, vola1newfield *next* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1newtable | |
| name | None | string | |
| datatype | None | type | |
| next | None | vola1newfield | |

## newindex()

### Description

### Prototype

*vola1newtablevar*.newindex ( vola1newtable *me*, vola1newfield *field*, integer *precision*, boolean *unique*, string *algorithm* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1newtable | |
| field | None | vola1newfield | |
| precision | None | integer | |
| unique | .false | boolean | |
| algorithm | None | string | |

# vola1record

# Description

# Type Tags

db1record

# Object Value

Objects of type vola1record have no value, and it is an error to try to get or set this value.

## vola1record.new()

### Description

## Prototype

*vola1record*.new ( vola1record *me*, vola1table *table* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1record | |
| table | None | vola1table | |

# Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | vola1recordprivate | |
| delete | function | |
| get | function | |
| index | vola1index | |
| locktype | string | |
| put | function | |
| save | function | |
| select | function | |
| selectcurrent | function | |
| stored | boolean | |
| table | vola1table | |
| type | type | |
| unlock | function | |

# Methods

## delete()

### Description

### Prototype

*vola1recordvar*.delete ( vola1record *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1record | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# get()

## Description

## Prototype

*vola1recordvar*.get ( vola1record *me*, vola1field *field* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1record | |
| field | None | vola1field | |

# put()

## Description

## Prototype

*vola1recordvar*.put ( vola1record *me*, vola1field *field*, anyvalue *value* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1record | |
| field | None | vola1field | |
| value | None | anyvalue | |

# save()

## Description

## Prototype

*vola1recordvar*.save ( vola1record *me*, string *locktype*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1record | |
| locktype | None | string | |
| error | None | integer | |

# select()

## Description

## Prototype

*vola1recordvar*.select ( vola1record *me*, boolean *previousrecord*, string *locktype*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1record | |
| previousrecord | .false | boolean | |
| locktype | None | string | |
| error | None | integer | |

# selectcurrent()

## Description

## Prototype

*vola1recordvar*.selectcurrent ( vola1record *me*, vola1index *index*, string *locktype*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1record | |
| index | None | vola1index | |
| locktype | None | string | |
| error | None | integer | |

# unlock()

## Description

## Prototype

*vola1recordvar*.unlock ( vola1record *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1record | |
| error | None | integer | |

# vola1table

## Description

## Type Tags

db1table

## Object Value

Objects of type vola1table have no value, and it is an error to try to get or set this value.

## vola1table.new()

### Description

### Prototype

*vola1table*.new ( vola1table *me* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1table | |

## Properties

| Property | Type | Description |
|---|---|---|
| _ | type(*) | |
| __ | type(*) | |
| _private | vola1tableprivate | |
| deleteall | function | |
| firstfield | vola1field | |
| firstindex | vola1index | |
| lock | function | |
| locktype | string | |
| newrecord | function | |
| recordcount | function | |
| select | function | |
| tablename | string | |
| type | type | |
| unlock | function | |

| Property | Type | Description |
|---|---|---|
| volabase | vola1base | |

# Methods

## !()

### Description

### Prototype

*vola1tablevar*.! ( vola1table *me*, string *fieldname* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1table | |
| fieldname | None | string | |

## deleteall()

### Description

### Prototype

*vola1tablevar*.deleteall ( vola1table *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1table | |
| error | None | integer | |

## lock()

### Description

### Prototype

*vola1tablevar*.lock ( vola1table *me*, string *locktype*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1table | |
| locktype | None | string | |
| error | None | integer | |

# newrecord()

## Description

## Prototype

*vola1tablevar*.newrecord ( vola1table *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1table | |

# recordcount()

## Description

## Prototype

*vola1tablevar*.recordcount ( vola1table *me* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1table | |

# select()

## Description

## Prototype

*vola1tablevar*.select ( vola1table *me*, boolean *lastrecord*, string *locktype*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1table | |
| lastrecord | .false | boolean | |
| locktype | None | string | |
| error | None | integer | |

# unlock()

## Description

### Prototype

*vola1tablevar*.unlock ( vola1table *me*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | vola1table | |
| error | None | integer | |

# checkvola1indexentries()

## Description

## Prototype

checkvola1indexentries ( vola1index *idx* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| idx | None | vola1index | |

# checkvola1indexlist()

## Description

## Prototype

checkvola1indexlist ( vola1index *idx* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| idx | None | vola1index | |

# checkvola1indextree()

## Description

## Prototype

checkvola1indextree ( vola1index *idx* )

# Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| idx | None | vola1index | |

# Chapter 103. windowsemaillib

## windowsemailinfo

### Description

### Type Tags

None

### Object Value

Objects of type windowsemailinfo have no value, and it is an error to try to get or set this value.

## windowsemailinfo.new()

### Description

### Prototype

*windowsemailinfo*.new ( windowsemailinfo *me*, string *message*, string *subject*, string *from*, string *to*, string *cc*, string *bcc*, string *smtpserver*, string *userid*, string *password*, boolean *usehtml*, integer *serverport*, boolean *usessl*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | windowsemailinfo | |
| message | None | string | |
| subject | None | string | |
| from | None | string | |
| to | None | string | |
| cc | None | string | |
| bcc | None | string | |
| smtpserver | None | string | |
| userid | None | string | |
| password | None | string | |
| usehtml | .false | boolean | |
| serverport | 25 | integer | |
| usessl | .false | boolean | |
| error | None | integer | |

# Properties

| Property | Type | Description |
|---|---|---|
| addattachment | function | |
| attachments | array | |
| bcc | string | |
| cc | string | |
| from | string | |
| message | string | |
| msgtextfile-name | string | |
| password | string | |
| programfile-name | string | |
| retrievepro-gram | function | |
| sendmessage | function | |
| serverport | integer | |
| smtpserver | string | |
| subject | string | |
| to | string | |
| type | type | |
| usehtml | boolean | |
| userid | string | |
| usessl | boolean | |

# Methods

## addattachment()

### Description

### Prototype

*windowsemailinfovar*.addattachment ( windowsemailinfo *me*, string *attachmentfile-name*, integer *error* )

### Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | windowsemailinfo | |
| attachmentfile-name | None | string | |

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| error | None | integer | |

# retrieveprogram()

## Description

## Prototype

*windowsemailinfovar*.retrieveprogram ( windowsemailinfo *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | windowsemailinfo | |
| error | None | integer | |

# sendmessage()

## Description

## Prototype

*windowsemailinfovar*.sendmessage ( windowsemailinfo *me*, integer *error* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| me | None | windowsemailinfo | |
| error | None | integer | |

# Chapter 104. winfiledlg

This library provides a function that can be called form SBL to use the SIMPOL open and save dialogs rather than the versions supplied to the Win16 subsystem.

# opensavefile()

## Description

## Prototype

`opensavefile` ( string *`sStyle`*, string *`sMessage`*, string *`sWildcard`*, string *`sDeffilename`*, string *`sDefdir`* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| sStyle | `open,mustexist` | string | |
| sMessage | None | string | |
| sWildcard | `All files (*.*)\|*.*` | string | |
| sDeffilename | None | string | |
| sDefdir | None | string | |

# Chapter 105. xmllib

This library provides a number of functions for working with XML, particularly when parsing XML.

# decodeXMLEntities()

## Description

## Prototype

`decodeXMLEntities` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_BaseChar()

## Description

## Prototype

`isXML_BaseChar` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_CombiningChar()

## Description

## Prototype

`isXML_CombiningChar` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_Digit()

## Description

## Prototype

`isXML_Digit` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_Extender()

## Description

## Prototype

`isXML_Extender` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_Ideographic()

## Description

## Prototype

`isXML_Ideographic` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_Letter()

## Description

## Prototype

`isXML_Letter` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_Name()

## Description

## Prototype

`isXML_Name` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_NameChar()

## Description

## Prototype

`isXML_NameChar` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_NameFirstChar()

## Description

## Prototype

`isXML_NameFirstChar` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# isXML_WhiteSpace()

## Description

## Prototype

`isXML_WhiteSpace` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# makeValidXMLContent()

## Description

## Prototype

`makeValidXMLContent` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# makeXMLName()

## Description

## Prototype

`makeXMLName` ( string *s* )

## Parameters

| Parameter | Default value | Type name | Description |
|---|---|---|---|
| s | None | string | |

# Appendix A. Error Codes

## Introduction

In SIMPOL as is the case in virtually every other programming language, there are error codes in order to provide usable feedback in the event of something unforeseen occurs. Some of the error codes are fixed, others may yet change. All error values beginning with 1000000 are old Superbase based errors. An error from PPCS such as 1000010 is equivalent to the error number 10 in Superbase (end of file). Error values that begin with 2000000 are operating system errors that are passed back to you through us. Subtract 2000000 from them to get the original operating system error value.

### Note

Error variables that are passed into functions are *not* always written to! Do not assume that if there is no error that the variable will be assigned the value zero. This is not the case. Error variables are only written to if an error condition occurs!

## General Runtime Errors

| Error Number | Error | Description |
|---|---|---|
| 1 | Out of memory | |
| 2 | Busy | |
| 3 | Locked out | |
| 4 | All in use | |
| 5 | Value in use | |
| 6 | UDP error | |
| 7 | Data not found | |
| 8 | Can't send | |
| 9 | Can't receive | |
| 10 | Time out | |
| 11 | Serial error | |
| | | The above error numbers won't change. Any of the following may still change. |
| 12 | Access denied | |
| 13 | Cannot be done | |
| 14 | Object is read-only | |
| 15 | Object is in use | |
| 16 | Object is wrong type | |
| 17 | Wrong type | |
| 18 | Type not found | |

| Error Number | Error | Description |
|---|---|---|
| 19 | Function not found | |
| 20 | Property not found | |
| 21 | Object not found | |
| 22 | Local variable not dimensioned | This can happen if you call a function with more parameters than there are arguments to the function. When debugging, this can be very confusing, since the debugger will fail to even enter the `main()` function if too many parameters are being passed by the IDE to the `main()` function. If this is happening, check the content of the Command Line entry in the Project → Settings section. |
| 23 | Cyclic dependency | |
| 24 | Invalid op-code | |
| 25 | Invalid constant index | |
| 26 | Invalid local variable index | |
| 27 | Not ready | |
| 28 | String too long | You have either attempted to create a string beyond the memory constraints of the operating system or you have attempted to assign a string to an entity that will cause it to exceed its size limitations, such as assigning a string to a blob using `putstring()` where the string being assigned is too large. |
| 29 | String null or empty | |
| 30 | Number out of range | |
| 31 | Not implemented | |
| 32 | Cannot create | |
| 33 | Invalid data | |
| 34 | End of data | |
| 35 | Invalid attribute | |
| 36 | No end tag | |
| 37 | Required tag not found | |
| 38 | Required attribute not found | |

| Error Number | Error | Description |
|---|---|---|
| 39 | Incorrect number of parameters | |
| 40 | Incorrect parameter type | |
| 41 | Parameter not found | |
| 42 | Parameter used twice | |
| 43 | Parameter not object | |
| 44 | The mixture of parameters is invalid | |
| 45 | Incompatible version | |
| 46 | Module not found | |
| 47 | Module not valid SIMPOL | |
| 48 | IP address not found | |
| 49 | Window creation failed | |
| 50 | Operation not allowed with a reference | |
| 51 | Operation only allowed with a reference | |
| 52 | Not embeddable | |
| 53 | Value, not object reference | |
| 54 | Object, not object reference | |
| 55 | You have attempted to assign a value to a read-only property. | |
| 56 | Program finished — internal | |

| Error Number | Error | Description |
|---|---|---|
| 57 | ESF1: Invalid Event — internal | |
| 58 | ESF1: Invalid Object you have supplied a next control reference that is from a different form. | |
| 59 | General OS failure | |
| 60 | Debugger: Malformed statement — internal | |
| 61 | Not enough temporary space | |
| 62 | Beyond capabilities of the current version. | You have attempted to do something that is completely beyond the capabilities of the OS where the language is currently running. For example, you have attempted to read string value that is so large that the current OS can't even handle the numeric size of the required buffer. |
| 63 | Bad structure | This implies some form of file corruption when reading an SMP or a file in SBM format. |
| 64 | End of data | Information message indicating that an attempt to read past the current point has reached the end of the available data. |
| 65 | Invalid special operator. | You have used the member operator `!` in conjunction with a type for which it has not yet been defined. |
| 66 | Invalid square bracket operator | You have used the square bracket operator `[]` in conjunction with a type for which it is invalid. |
| 67 | Subscript miscount | This is an error related to the use of the square brackets operator and is type-specific. |
| 68 | Subscript mistype | |
| 69 | Subscript out of range | |
| 70 | Blob too big | You have either attempted to create or enlarge a blob beyond the memory constraints of the operating system or you have attempted to assign a blob to an entity that will cause it to exceed its size limitations, such as assigning a blob to another blob using `putblob()` where the blob being assigned is too large. |
| 71 | Blob null or empty | You have passed a null or empty blob to a function that requires a blob. |
| 72 | Invalid usage | You have attempted to run a program as Fast-CGI that isn't being called using the Fast-CGI conventions. |

| Error Number | Error | Description |
|---|---|---|
| 73 | Deadly embrace | You have attempted to lock a `lock1` object with a request to block. Other threads are currently locking this object and the program is currently in such a state that it will not be able to continue since the blocking thread will not be able to acquire the lock which will result in the entire program hanging. This situation has been detected and your program has therefore been terminated. |
| 74 | Containment type error | You have attempted to assign an object to a container that is not appropriate. This can occur if you attempt to assign a wxform to a window1 object, since a wxform can only be contained by valid wxcontainer types. |
| 75 | Containment error | |
| 76 | Invalid parameter | The parameter definition for the shared library function was syntactically incorrect. |
| 77 | Cannot find | The shared library function that you attempted to find was not in the library. |
| 78 | Invalid bit count | |
| 79 | Invalid type specified | |
| 80 | Delimiter mismatch | |
| 81 | Invalid character | |
| 82 | Duplicate element | You tried to add a duplicate element to a set that has been set to not allow duplicates. |
| 83 | Element not found | You have attempted to access an element value in a set that does not exist. |
| 84 | Wrong module | You have defined a member function for a type in a module that does not contain the type definition. |
| 85 | Type not found for function | You have defined a function as a member function for a type that does not exist in the same module. |
| 86 | Abandoned | This is the return value to use from a callback function for the report engine if the report should be cancelled. |
| 200 | Object has no value | You have attempted to retrieve the value of an object that has no value. |
| 1000 | Internal | This is an internal error. If this error occurs, please record the circumstances that lead to the error and report it to Superbase. |
| 1010 | Internal | This is an internal error. If this error occurs, please record the circumstances that lead to the error and report it to Superbase Software Limited. |

# Utility Errors

| Error Number | Error | Description |
|---|---|---|
| 400 | Nodes already linked | |
| 401 | Nodes not linked | |
| 402 | Nodes already in a list | |
| 403 | Nodes not in same list | |
| 404 | Nodes not contiguous | |

# UI Errors

| Error Number | Error | Description |
|---|---|---|
| 500 | UI startup error | |
| 501 | Unrecognized control type | |
| 502 | Failed to add item | |
| 503 | Invalid modality state | This will occur if you have tried to call the `processmodal()` of a wxdialog object when the object has been created with the *`visible`* set to `.true`. |
| 504 | Font creation failed | |
| 505 | Menu creation failed | |
| 506 | Menu insertion failed | |
| 507 | Menu not parentless | |
| 508 | Menu not in menubar | |
| 509 | Menu item not checkable | |
| 510 | Bitmap creation failed | |
| 511 | Tool bar creation failed | |
| 512 | Status bar creation failed | |

| Error Number | Error | Description |
|---|---|---|
| 513 | Tool creation failed | |
| 514 | Style doesn't make sense | |
| 515 | Common dialog creation failed | |
| 516 | Not a valid choice | |
| 517 | Mouse not captured | |
| 518 | Controls not captured | |
| 519 | Disabled | You have attempted to set focus to a disabled control |
| 520 | Invisible | You have attempted to set focus to an invisible control |
| 521 | Not valid ellipse | You have attempted to create an ellipse using invalid coordinates |
| 522 | Cannot be done for child windows | You have attempted to use a method to make changes to a child window in a way that is not supported, such as calling setstate() to minimize or maximize a child window. |
| 523 | Cancelled | The user cancelled the print out. |
| 524 | Printing error | An error occurred while attempting to print the printout. |
| 525 | Preview creation failed | An error occurred while to create the print preview. |
| 526 | Can't make icon from bitmap | |
| 527 | Child window can't have icon | |
| 528 | OLE2 Automation: object creation failed | The OLE2 subsystem was unable to start an instance of the OLE2 server |
| 529 | OLE2 Automation: instance not available | The OLE2 subsystem was unable to access a running instance of the OLE2 server |
| 530 | OLE2 Automation: operation failed | The attempted OLE2 automation operation did not succeed |
| 531 | OLE2 Automation: type not supported | The type that was returned from a call to the OLE2 subsystem is not supported |
| 532 | Command event only | This is an internal error. If this error occurs, please record the circumstances that lead to the error and report it to Superbase Software Limited. |

# Registration Errors

| Error Number | Error | Description |
|---|---|---|
| 600 | Validation error | There is a problem with your product registration number. |

# Sockets Errors

| Error Number | Error | Description |
|---|---|---|
| 701 | Port not given | You failed to provide a port number when creating the socket |
| 702 | Address not found | The IP address provided could not be reached or the name could not be resolved |
| 703 | Can't create socket | There was an error when creating the socket |
| 704 | Address error | The IP address provided was invalid |
| 705 | Connection failed | The connection failed, possibly because there is no program listening on the desired port at that address |

# PPCS Protocol Errors

| Error Number | Error | Description |
|---|---|---|
| 801 | PPCS data problem | |

# PPCS and SBME File Errors

| Error Number | Error | Description |
|---|---|---|
| 901 | Field not found | |
| 902 | Field type problem | |
| 903 | Lock required | |
| 904 | Index type not supported | |
| 905 | Method of a database file based object was called with a parameter value from a different file | |
| 906 | Code page for the target file is not supported | |

| Error Number | Error | Description |
|---|---|---|
| 907 | PPCS/SBME The record that you attempted an operation on is not currently stored and therefore the operation is not supported. | |
| 908 | Attempt to lock one or more records or in SBME to lock the table or sbme object when they are already locked. | |
| 909 | Attempt to continue an operation that is no longer possible because of a previous call to rollback. | |
| 910 | Index not found. Internal error. | |
| 911 | Field has index. You have attempted to delete a field before removing the index. | |
| 912 | Field read-only. You have attempted to modify the contents of a read-only field. | |
| 913 | Key too long. Internal error. | |
| 914 | Duplicate index entry. You have attempted to save a record with an index value that is already in use. | |

| Error Number | Error | Description |
|---|---|---|
| 915 | Index property unknown. You may be trying to use an older version of a database component with a newer index type. | |

# ODBC Errors

| Error Number | Error | Description |
|---|---|---|
| 950 | ODBC error | An ODBC error has occurred. Check the odbc1error object for specific information relating to the error. |
| 951 | Not connected | You may have attempted to do something that requires a successful ODBC connection. This can happen if you attempt to do ODBC operations after having attempted and failed to connect to a data source, but failed to check for success. |
| 952 | Already connected | You have attempted to connect to an ODBC data source using an object that is already connected. |
| 953 | Type not supported | The data type that you have attempted to pass to the ODBC data source is not supported. |
| 954 | No length given | A function that requires a length parameter was not provided one. |

# Errors from Superbase Basic Language

| Error Number | Error | Description |
|---|---|---|
| 1000001 | Not an index | |
| 1000002 | No more images for this record | |
| 1000003 | Print command failed | |
| 1000004 | Can't open the printer | |
| 1000005 | Not an open file | |
| 1000006 | Access to file not allowed | |
| 1000007 | This action requires an index | |
| 1000008 | No record with this key | |

| Error Number | Error | Description |
|---|---|---|
| 1000009 | No external file field defined | |
| 1000010 | End of file | |
| 1000011 | User cancelled | |
| 1000012 | Cannot access directory | |
| 1000013 | Cannot access disk information | |
| 1000014 | Name too long | |
| 1000015 | Can't find this file (table) | |
| 1000016 | Can't find this field | |
| 1000017 | Invalid statement | |
| 1000018 | Out of data | |
| 1000019 | Invalid date | |
| 1000020 | Data types don't match | |
| 1000021 | Operator not allowed with text | |
| 1000022 | Function not allowed with text | |
| 1000023 | Function not allowed with number | |
| 1000024 | Can't divide by zero | |
| 1000025 | Can't AND illegal values | |
| 1000026 | Can't OR illegal values | |
| 1000027 | Invalid numeric parameter | |
| 1000028 | Invalid text parameter | |
| 1000029 | Closing parentheses missing | |
| 1000030 | Opening parentheses missing | |
| 1000031 | Function syntax comma missing | |

| Error Number | Error | Description |
|---|---|---|
| 1000032 | Function argument invalid parameter | |
| 1000033 | String too long | |
| 1000034 | Misplaced command | |
| 1000035 | Formula too complex | |
| 1000036 | Variable not defined | |
| 1000037 | Field not defined | |
| 1000038 | Number out of range | |
| 1000039 | Invalid numeric format | |
| 1000040 | No file name given | |
| 1000041 | Invalid parameter | |
| 1000042 | Field already exists | |
| 1000043 | File (table) already exists | |
| 1000044 | No file (table) selected | |
| 1000045 | Invalid field type | |
| 1000046 | Invalid field name | |
| 1000047 | Data too long for field | |
| 1000048 | Field requires numeric data | |
| 1000049 | Field requires text data | |
| 1000050 | Data is required in field | |
| 1000051 | Field does not match validation | |
| 1000052 | Cannot open file (table) | |

| Error Number | Error | Description |
|---|---|---|
| 1000053 | Error deleting file (table) | |
| 1000054 | Can't record detail block operations | |
| 1000055 | Return without gosub | |
| 1000056 | NEXT without FOR | |
| 1000057 | Duplicate key found | |
| 1000058 | No record selected | |
| 1000059 | Can't find key to delete | |
| 1000060 | Can't find correct key to delete | |
| 1000061 | Can't find key while deleting | |
| 1000062 | Can't find pointer while deleting | |
| 1000063 | Memory allocation problem | |
| 1000064 | Problem reading input file | |
| 1000065 | Problem writing output file | |
| 1000066 | File contains non text characters | |
| 1000067 | Input file data invalid | |
| 1000068 | Can't open input file | |
| 1000069 | Can't open output file | |
| 1000070 | Undefined program label | |
| 1000071 | File block has been deleted | |
| 1000072 | File block is not first in chain | |

| Error Number | Error | Description |
|---|---|---|
| 1000073 | Invalid date format string | |
| 1000074 | Invalid numeric format string | |
| 1000075 | Problem reading data file | |
| 1000076 | Problem writing to data file | |
| 1000077 | Problem positioning in data file | |
| 1000078 | End of external file | |
| 1000079 | External file not found | |
| 1000080 | External file not iff | |
| 1000081 | Problem reading external file | |
| 1000082 | Index must be removed | |
| 1000083 | Incomplete field or value missing | |
| 1000084 | Incomplete statement. Check for missing operators or separators | |
| 1000085 | Statement does not give true or false | |
| 1000086 | Remove index before type change | |
| 1000087 | Can't total non numeric field | |
| 1000088 | Insufficient memory | |
| 1000089 | Internal error | |
| 1000090 | Insufficient stack space | |
| 1000091 | Error renaming file | |

| Error Number | Error | Description |
|---|---|---|
| 1000092 | Where statement should be single file | |
| 1000093 | Command must be at end of line | |
| 1000094 | Error invalid time | |
| 1000095 | Input file not open | |
| 1000096 | Field is not external | |
| 1000097 | Resume without error | |
| 1000098 | Can't save property setting | |
| 1000099 | Can't SetActive within OnActivate event | |
| 1000100 | Field is already indexed | |
| 1000101 | Array variable not dimensioned | |
| 1000102 | Invalid subscript for array | |
| 1000103 | Subscript not numeric | |
| 1000104 | Array already dimensioned | |
| 1000105 | WEND without WHILE | |
| 1000106 | Nesting depth exceeded | |
| 1000107 | This file (table) requires a later version of the software | |
| 1000108 | Invalid statement beginning | |
| 1000109 | Invalid statement ending | |
| 1000110 | Required in statement | |

| Error Number | Error | Description |
|---|---|---|
| 1000111 | Variable required in statement | |
| 1000112 | THEN or GOTO required in statement | |
| 1000113 | GOSUB or GOTO required in statement | |
| 1000114 | Invalid form file | |
| 1000115 | Group not defined | |
| 1000116 | Can't perform operation on object in a detail block | |
| 1000117 | File (table) open in another directory | |
| 1000118 | Problem opening window | |
| 1000119 | 1.2 System software required | |
| 1000120 | UDP not available - unable to load winsock | |
| 1000121 | IF without END IF | |
| 1000122 | In use by another file | |
| 1000123 | Max 255 dialog controls allowed | |
| 1000124 | Cannot open COM port | |
| 1000125 | Error during file transfer | |
| 1000126 | No carrier detected (DCD) | |
| 1000127 | File is busy | |
| 1000128 | Error unlocking file | |
| 1000129 | Command not allowed in un- | |

| Error Number | Error | Description |
|---|---|---|
| | queued message event | |
| 1000130 | Record is locked by: | |
| 1000131 | File (table) is open for exclusive access | |
| 1000132 | File (table) is open for shared access | |
| 1000133 | Too many users | |
| 1000134 | Network or file sharing inactive | |
| 1000135 | Error re-seeding index | |
| 1000136 | Cannot lock all records | |
| 1000137 | File (table) is locked by : | |
| 1000138 | Access denied: File (table) in use | |
| 1000139 | File (table) is open for exclusive write | |
| 1000140 | File (table) is open for shared write | |
| 1000141 | File (table) has been modified | |
| 1000142 | File (table) is open for read only | |
| 1000143 | State not allowed with IconComboBox | |
| 1000144 | Cannot register window classes | |
| 1000145 | Cannot create main window | |
| 1000146 | Cannot create child window | |

| Error Number | Error | Description |
|---|---|---|
| 1000147 | Invalid or duplicate IP address or port number | |
| 1000148 | Search string not found | |
| 1000149 | File in use by another instance | |
| 1000150 | Invalid margins selected | |
| 1000151 | CASE required after SELECT CASE | |
| 1000152 | END SELECT missing from SELECT CASE | |
| 1000153 | Write failed - Disk is full | |
| 1000154 | Channel already in use | |
| 1000155 | Logical field must contain Y,N,T or F | |
| 1000156 | Application not responding | |
| 1000157 | Channel not initialized | |
| 1000158 | Cannot open the Clipboard | |
| 1000159 | Floating point math error | |
| 1000160 | Application busy | |
| 1000161 | Precision error in math function | |
| 1000162 | Form requires new version of software | |
| 1000163 | Reorganize failed to complete | |
| 1000164 | Insufficient disk space | |
| 1000165 | Cannot open library | |

| Error Number | Error | Description |
|---|---|---|
| 1000166 | Invalid function argument | |
| 1000167 | Cannot find function | |
| 1000168 | Function already defined | |
| 1000169 | Invalid Superbase library | |
| 1000170 | Network control file not present | |
| 1000171 | Invalid network control file directory | |
| 1000172 | Cannot log on to network control file | |
| 1000173 | Invalid lock file | |
| 1000174 | Cannot lock database file | |
| 1000175 | Cannot lock file | |
| 1000176 | Cannot lock record | |
| 1000177 | Record is not locked | |
| 1000178 | File only accessible from lan version | |
| 1000179 | Invalid outer join syntax | |
| 1000180 | Too many outer joins | |
| 1000181 | Query optimization error | |
| 1000182 | Error from SQL system | |
| 1000183 | Invalid SQL connect command | |
| 1000184 | SQL command too long | |
| 1000185 | Not connected to SQL system | |

| Error Number | Error | Description |
|---|---|---|
| 1000186 | No SQL command has been prepared | |
| 1000187 | Invalid SQL bind syntax | |
| 1000188 | Cannot find form object | |
| 1000189 | Query optimizer not enabled | |
| 1000190 | Dialog already exists | |
| 1000191 | Cannot find dialog | |
| 1000192 | Cannot create dialog | |
| 1000193 | Invalid SQL option | |
| 1000194 | Incompatbile software version | |
| 1000195 | Error sending or receiving on UDP | |
| 1000196 | Procedure name missing | |
| 1000197 | Too many procedure arguments | |
| 1000198 | Maximum procedure depth exceeded | |
| 1000199 | Formal arguments must be simple variables | |
| 1000200 | Argument count mismatch | |
| 1000201 | Procedure name too long | |
| 1000202 | Illegal entry to procedure | |
| 1000203 | Undefined array or function | |
| 1000204 | END FUNCTION without FUNCTION | |

| Error Number | Error | Description |
|---|---|---|
| 1000205 | Variable already declared | |
| 1000206 | SUB cannot return a value | |
| 1000207 | FUNCTION must return a value | |
| 1000208 | END SUB without SUB | |
| 1000209 | CALL cannot call a function | |
| 1000210 | Not available in Superbase Personal | |
| 1000211 | Undefined Sub Procedure | |
| 1000212 | Too many programs | |
| 1000213 | Cannot close while recording | |
| 1000214 | Too many indexes. Maximum 999 | |
| 1000215 | Cannot RUN a recording macro | |
| 1000216 | Invalid macro name | |
| 1000217 | AS name required for group or total expression | |
| 1000218 | Detail block name already exists | |
| 1000219 | Invalid detail block name | |
| 1000220 | Detail block must have a name | |
| 1000221 | Detail block will not fit on current page as specified | |
| 1000222 | Detail block information for | |

| Error Number | Error | Description |
|---|---|---|
| | data object does not exist | |
| 1000223 | Invalid variable type | |
| 1000224 | Decimal and thousand separator should not be the same character | |
| 1000225 | QBE info lost on query read | |
| 1000226 | "Error sending MCI string | |
| 1000227 | Quick Report info lost on query read | |
| 1000228 | Insert query can only be single table | |
| 1000229 | Delete query can only be single table | |
| 1000230 | Too many Insert rows | |
| 1000231 | Too many Update rows | |
| 1000232 | Insert row may not contain selection marks | |
| 1000233 | Delete row may not contain selection marks | |
| 1000234 | Update row may not contain selection marks | |
| 1000235 | Query contains unlinked files (tables) | |
| 1000236 | Query contains unlinked rows | |
| 1000237 | Query selects no columns | |
| 1000238 | Nested select not supported | |

| Error Number | Error | Description |
|---|---|---|
| 1000239 | Cannot join text to numeric column | |
| 1000240 | Cannot join to derived column | |
| 1000241 | Text is not allowed in numeric column | |
| 1000242 | Number is not allowed in text column | |
| 1000243 | Example element cannot be resolved | |
| 1000244 | Expression cannot be resolved | |
| 1000245 | Report band too big | |
| 1000246 | Invalid object type for form dll function | |
| 1000247 | Specified coordinates are off page | |
| 1000248 | Fields off page not created | |
| 1000249 | Event class missing | |
| 1000250 | CALL keyword missing | |
| 1000251 | Event service procedure must be a SUB | |
| 1000252 | Equals sign missing | |
| 1000253 | Undefined event | |
| 1000254 | Wrong type of form | |
| 1000255 | Selected objects dont fit on page | |
| 1000256 | Data file has been encrypted | |
| 1000257 | Index file is already open | |

| Error Number | Error | Description |
|---|---|---|
| 1000258 | Cannot modify dBase 3 memo | |
| 1000259 | Message must be retrieved by GetMessage before reading | |
| 1000260 | Program name missing | |
| 1000261 | Program file not open | |
| 1000262 | Detail block already exists | |
| 1000263 | Detail block definition not started | |
| 1000264 | Cannot mix detail block syntax versions | |
| 1000265 | Detail block definition already started | |
| 1000266 | Valid link to parent not specified | |
| 1000267 | Inappropriate link for detail block | |
| 1000268 | Operation cannot be done for an object in a detail block | |
| 1000269 | Can't close a running program | |
| 1000270 | ORDER field in unlinked detail block must be indexed | |
| 1000271 | Non-indexed field specified in link | |
| 1000272 | Inappropriate DML executed during save process | |

| Error Number | Error | Description |
|---|---|---|
| 1000273 | Operation not possible on local variable | |
| 1000274 | Nested aggregates not permitted in calculation | |
| 1000275 | Incorrect use of DROW keyword | |
| 1000276 | Report must contain at least one band | |
| 1000277 | Cannot perform operation on requested row | |
| 1000278 | Error nesting too deep | |
| 1000279 | Could not create a new object or start object server | |
| 1000280 | Could not open ole object | |
| 1000281 | Invalid link field specification | |
| 1000282 | Detail block link cannot use unique fields | |
| 1000283 | Chosen data files (tables) don't contain any fields that can be linked | |
| 1000284 | Only one link allowed to parent detail block | |
| 1000285 | Cannot link to detail block | |
| 1000286 | Cannot use detail block as master file | |
| 1000287 | Cannot use this extension | |

| Error Number | Error | Description |
|---|---|---|
| 1000288 | Too many pending events | |
| 1000289 | No DDE request pending | |
| 1000290 | Action not supported in Trial Pack | |
| 1000291 | Keyword not supported by query optimizer | |
| 1000292 | Cannot delete detail block with children | |
| 1000293 | Can't query ANSWER table to ANSWER table | |
| 1000294 | Cannot flush pending events | |
| 1000295 | This file may be out of date | |
| 1000296 | Cannot perform operation on Report form | |
| 1000297 | Not permitted while report is open | |
| 1000298 | SUB without END SUB or FUNCTION without END FUNCTION | |
| 1000299 | No current menu | |
| 1000300 | No current Icon Bar | |
| 1000301 | Can't return event | |
| 1000302 | Procedure not supported by run-time system | |
| 1000303 | Can't LOCK/UNLOCK EVENTCLASS | |

| Error Number | Error | Description |
|---|---|---|
| 1000304 | Illegal dBase index expression | |
| 1000305 | Merge requires open file (table) | |
| 1000306 | From and To files (tables) must be different | |
| 1000307 | Data conversion format not supported | |
| 1000308 | Unique index conflict - record replaced | |
| 1000309 | Unique index conflict - record skipped | |
| 1000310 | Disparate field types - data coerced | |
| 1000311 | Too few fields for this record | |
| 1000312 | Action exceeds Trial Pack record limit | |
| 1000313 | Too many fields for this file type | |
| 1000314 | Too many records for this file type | |
| 1000315 | Record too big for this file type | |
| 1000316 | Operation prevented by matrix or vector size or shape | |
| 1000317 | Cannot be performed while a FiniteElmtArray is running | |
| 1000318 | Operation on remote connection has timed out | |
| 1000319 | Member name missing | |

| Error Number | Error | Description |
|---|---|---|
| 1000320 | Invalid type name | |
| 1000321 | Not an object variable | |
| 1000322 | Type name missing | |
| 1000323 | Object variable not SET | |
| 1000324 | Cannot find property or method | |
| 1000325 | Not a method | |
| 1000326 | Cannot assign a method | |
| 1000327 | Cannot SET an object variable to a less specific type | |
| 1000328 | String method argument expected | |
| 1000329 | Numeric method argument expected | |
| 1000330 | Method argument count mismatch | |
| 1000331 | Method does not return a value | |
| 1000332 | Invalid object | |
| 1000333 | Not an instanciable type | |
| 1000334 | Period missing | |
| 1000335 | Is a string property | |
| 1000336 | Is a numeric property | |
| 1000337 | Incomplete property or method reference | |

| Error Number | Error | Description |
|---|---|---|
| 1000338 | Type mismatch: Invalid use of object | |
| 1000339 | Invalid priority level | |
| 1000340 | Incompatible object types | |
| 1000341 | Collection member not found | |
| 1000342 | Invalid collection subscript | |
| 1000343 | Name already in use | |
| 1000344 | Property is read-only during object activation | |
| 1000345 | Invalid data received on remote connection | |
| 1000346 | Cannot connect to named server | |
| 1000347 | Type incompatible with this collection | |
| 1000348 | Cannot create object | |
| 1000349 | Cannot remove object | |
| 1000350 | Not enough dimensions to create crosstab | |
| 1000351 | Crosstab cannot contain more than 3-dimensions | |
| 1000352 | Invalid aggregate function for Crosstab | |
| 1000353 | From and To files must be specified | |
| 1000354 | Property is read-only | |

| Error Number | Error | Description |
|---|---|---|
| 1000355 | Property is write-only | |
| 1000356 | Method argument of object type expected | |
| 1000357 | Object returned by method must be assigned to a variable using SETy | |
| 1000358 | Cannot mix Crosstab and select queries | |
| 1000359 | Unable to find mail system | |
| 1000360 | Invalid mail logon password | |
| 1000361 | General failure from MAPI or VIM | |
| 1000362 | Not logged on to mail system | |
| 1000363 | Attached Variable | |
| 1000364 | No external data | |
| 1000365 | Cannot Get Object | |
| 1000366 | Cannot access server data (1) | |
| 1000367 | Invalid split window pane size | |
| 1000368 | Valid only in MDI mode | |
| 1000369 | Can't set 'Value' of non-PushButton | |
| 1000370 | Can't move or resize a maximized window | |
| 1000371 | Can't move or resize an iconized window | |

| Error Number | Error | Description |
|---|---|---|
| 1000372 | Share not running: file opened exclusive | |
| 1000373 | Not an array property | |
| 1000374 | Array property: subscript count mismatch | |
| 1000375 | Array property: string subscript expected | |
| 1000376 | Array property: numeric subscript expected | |
| 1000377 | Object variable not defined | |
| 1000378 | Element not found | |
| 1000379 | Object does not support OLE Automation | |
| 1000380 | OLE Automation Error : Member not found | |
| 1000381 | OLE Automation Error : Parameter not found | |
| 1000382 | OLE Automation Error : Type mismatch | |
| 1000383 | OLE Automation Error : Unknown name | |
| 1000384 | OLE Automation Error : Named arguments not supported | |
| 1000385 | OLE Automation Error : Invalid variable type | |

| Error Number | Error | Description |
|---|---|---|
| 1000386 | OLE Automation Error : Server error | |
| 1000387 | OLE Automation Error : Overflow | |
| 1000388 | OLE Automation Error : Invalid index | |
| 1000389 | OLE Automation Error : Invalid class | |
| 1000390 | OLE Automation Error : Array is locked | |
| 1000391 | OLE Automation Error : Parameter count mismatch | |
| 1000392 | OLE Automation Error : Parameter not optional | |
| 1000393 | OLE Automation Error : Server error | |
| 1000394 | OLE Automation Error : Invalid collection | |
| 1000395 | Invalid message identifier | |
| 1000396 | Too many recipient names listed for current mail system | |
| 1000397 | Too many attachment files listed for current mail system | |
| 1000398 | Recipient name was not found | |
| 1000399 | One of the attachment files could not be found or opened | |

| Error Number | Error | Description |
|---|---|---|
| 1000400 | Message text is too long for current mail system | |
| 1000401 | Message index number exceeds unread message count | |
| 1000402 | No attachments exist for the message | |
| 1000403 | Operation cannot be performed on a remote connection | |
| 1000404 | Remote connection not recognised | |
| 1000405 | Can't save attachment | |
| 1000406 | Can't set the polling time | |
| 1000407 | Can't find property | |
| 1000408 | Property has no default | |
| 1000409 | Unsupported COM: port | |
| 1000410 | Unsupported baud rate | |
| 1000411 | Unsupported word length | |
| 1000412 | COM: port is not open | |
| 1000413 | COM: port is already open | |
| 1000414 | COM: port is in use by another application | |
| 1000415 | Problem writing to COM port | |
| 1000416 | Problem reading from COM port | |

| Error Number | Error | Description |
|---|---|---|
| 1000417 | No style definitions in form builder .ini file | |
| 1000418 | No type definitions in form builder .ini file | |
| 1000419 | Value too large or too small for property | |
| 1000420 | Property not available for an unbound control | |
| 1000421 | Invalid for a bound property | |
| 1000422 | Invalid with Multiselect | |
| 1000423 | String too long for property | |
| 1000424 | String too short for property | |
| 1000425 | Argument to function must be object | |
| 1000426 | Can not execute with a STATIC ListSourceType | |
| 1000427 | Can not execute unless STATIC ListSourceType | |
| 1000428 | Message has not been read | |
| 1000429 | No active window | |
| 1000430 | Unable to open Crosstab library | |
| 1000431 | Dialog object needs an object name | |
| 1000432 | Wrong ListSourceType | |
| 1000433 | Field does not match specified file | |

| Error Number | Error | Description |
|---|---|---|
| 1000434 | NULL String parameter not allowed | |
| 1000435 | NetBIOS not found or insufficient NetBIOS resources available | |
| 1000436 | Failure or unexpected error from NetBIOS | |
| 1000437 | Unique index conflict - Index converted to duplicate | |
| 1000438 | Unique index conflict - field data merged with existing record | |
| 1000439 | Unique index conflict - Cannot merge with more than one record | |
| 1000440 | No output window open | |
| 1000441 | File (table) is open for exclusive access in another window | |
| 1000442 | File (table) is open for shared access in another window | |
| 1000443 | File (table) is open for exclusive write in another window | |
| 1000444 | File (table) is open for shared write in another window | |
| 1000445 | File (table) is open for read only in another window | |

| Error Number | Error | Description |
|---|---|---|
| 1000446 | Invalid Combobox type | |
| 1000447 | Filter cannot be represented by QBE | |
| 1000448 | Calculated columns must contain an expression | |
| 1000449 | Reports not allowed in Form Designer | |
| 1000450 | Forms not allowed in Report Designer | |
| 1000451 | Invalid query type for conversion to a custom report | |
| 1000452 | Object must have a name | |
| 1000453 | ODBC function failed | |
| 1000454 | Not supported on ODBC files | |
| 1000455 | Not supported on this index | |
| 1000456 | Already connected to data source | |
| 1000457 | File still open on connection | |
| 1000458 | Couldn't connect to data source | |
| 1000459 | Record not available | |
| 1000460 | Not supported on SELECT AS files | |
| 1000461 | Too many ODBC connections | |

| Error Number | Error | Description |
|---|---|---|
| 1000462 | Crosstab can only have one aggregate | |
| 1000463 | Valid only in SDI mode | |
| 1000464 | No report form in memory | |
| 1000465 | Operation cannot be performed while connection is active | |
| 1000466 | Report form does not specify any fields | |
| 1000467 | Can't nest switching of current window | |
| 1000468 | Window must have a control box to minimize or maximize | |
| 1000469 | Control not in detail block | |
| 1000470 | Property only exists for read messages | |
| 1000471 | Window is being edited in the Form Designer | |
| 1000472 | Invalid with native object | |
| 1000473 | Please enter a filename for the template | |
| 1000474 | Specified coordinates are not on the dialog | |
| 1000475 | No form in memory | |
| 1000476 | Operation cannot be performed on a volatile file | |

| Error Number | Error | Description |
|---|---|---|
| 1000477 | File with this name already open | |
| 1000478 | A Report is being edited in the Report Designer | |
| 1000479 | Array property: invalid subscript | |
| 1000480 | Entered value is not in list | |
| 1000481 | Invalid object name | |
| 1000482 | Can not remove non-primary object from detail block | |
| 1000483 | Invalid variable name | |
| 1000484 | Must merge with this file format | |
| 1000485 | Too many concurrent usages - Close a view window | |
| 1000486 | Cannot deactivate dialog box | |
| 1000487 | Can't import to a file (table) which is already open | |
| 1000488 | Cannot Order using specified file (table) | |
| 1000489 | Cannot Order using specified field | |
| 1000490 | Can't open - dBase file version too early | |
| 1000491 | File (table) not attached to detail block | |
| 1000492 | Property dialog not open | |

| Error Number | Error | Description |
|---|---|---|
| 1000493 | Starting array element required | |
| 1000494 | Object parameter: Procedure called with a less specific type | |
| 1000495 | More specific object type required | |
| 1000496 | Variable is read only | |
| 1000497 | Cannot perform operation on an unlinked detail block | |
| 1000498 | Member refers to NOTHING | |
| 1000499 | Cannot return: Formal reference parameter type does not match actual parameter | |
| 1000500 | Object not found | |
| 1000501 | Cannot DIM an array of objects | |
| 1000502 | Array member not supported | |
| 1000503 | Sub-expression evaluates to NOTHING | |
| 1000504 | Duplicate element value not allowed in distinct set | |
| 1000505 | Invalid nesting of data entry/event processes | |
| 1000506 | Command buttons cannot have both a command and | |

| Error Number | Error | Description |
|---|---|---|
| | an OnClick event | |
| 1000507 | Cannot activate a disabled window | |
| 1000508 | SELECT, ORDER or GROUP clause too large | |
| 1000509 | Can't open file - SHARE must be loaded | |
| 1000510 | No event procedure defined | |
| 1000511 | Index type not supported | |