

libyui-mga
1.0.4

Generated by Doxygen 1.8.8

Sat Oct 18 2014 12:26:09

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	YCBTableItem Class Reference	5
3.1.1	Detailed Description	6
3.1.2	Constructor & Destructor Documentation	6
3.1.2.1	YCBTableItem	6
3.1.2.2	YCBTableItem	6
3.2	YMGA_CBTable Class Reference	7
3.2.1	Detailed Description	9
3.2.2	Constructor & Destructor Documentation	9
3.2.2.1	YMGA_CBTable	9
3.2.3	Member Function Documentation	9
3.2.3.1	addItem	9
3.2.3.2	cellChanged	9
3.2.3.3	changedItem	10
3.2.3.4	checkItem	10
3.2.3.5	getProperty	10
3.2.3.6	hasColumn	10
3.2.3.7	immediateMode	10
3.2.3.8	keepSorting	11
3.2.3.9	propertySet	11
3.2.3.10	setChangedItem	11
3.2.3.11	setKeepSorting	11
3.2.3.12	setProperty	12

3.2.3.13	setTableHeader	12
3.2.3.14	tableMode	12
3.2.3.15	toCByTableWidgetItem	12
3.2.3.16	userInputProperty	13
3.2.3.17	YItemIteratorToYItem	13
3.3	YMGA_CBTablePrivate Struct Reference	14
3.3.1	Detailed Description	14
3.4	YMGAAboutDialog Class Reference	14
3.4.1	Detailed Description	15
3.4.2	Constructor & Destructor Documentation	15
3.4.2.1	YMGAAboutDialog	15
3.4.3	Member Function Documentation	15
3.4.3.1	show	15
3.5	YMGAAboutDialogPrivate Class Reference	16
3.5.1	Detailed Description	17
3.6	YMGAMessageBox Class Reference	17
3.6.1	Detailed Description	17
3.6.2	Member Enumeration Documentation	17
3.6.2.1	DLG_BUTTON	17
3.6.2.2	DLG_MODE	18
3.6.3	Constructor & Destructor Documentation	18
3.6.3.1	YMGAMessageBox	18
3.6.4	Member Function Documentation	18
3.6.4.1	setButtonLabel	18
3.6.4.2	setDefaultButton	18
3.6.4.3	setIcon	19
3.6.4.4	setMinSize	19
3.6.4.5	setText	19
3.6.4.6	setTitle	19
3.6.4.7	show	19
3.7	YMGAMessageBoxPrivate Class Reference	20
3.7.1	Detailed Description	21
3.8	YMGAWidgetFactory Class Reference	21
3.8.1	Detailed Description	23
3.8.2	Constructor & Destructor Documentation	23
3.8.2.1	YMGAWidgetFactory	23
3.8.3	Member Function Documentation	23

3.8.3.1	createAboutDialog	23
3.8.3.2	createCBTable	23
3.8.3.3	createDialogBox	24
3.8.3.4	createInfoBox	24
3.8.3.5	createMessageBox	25
3.8.3.6	createWarningBox	26
3.8.3.7	getYCancelEvent	27
3.8.3.8	getYDebugEvent	27
3.8.3.9	getYKeyEvent	27
3.8.3.10	getYMenuEvent	27
3.8.3.11	getYMGAWidgetFactory	27
3.8.3.12	getYTimeoutEvent	28
3.8.3.13	getYWidgetEvent	28

Index**29**

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

YExternalWidgetFactory	
YMGAWidgetFactory	21
YMGA_CBTablePrivate	14
YMGAAboutDialog	14
YMGAAboutDialogPrivate	16
YMGAMessageBox	17
YMGAMessageBoxPrivate	20
YSelectionWidget	
YMGA_CBTable	7
YTableItem	
YCBTableItem	5

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

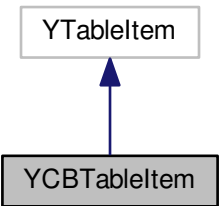
YCBTableItem	5
YMGA_CBTable	
See document of YMGA_CBTable Table: Selection list with multiple columns	7
YMGA_CBTablePrivate	14
YMGAAboutDialog	14
YMGAAboutDialogPrivate	16
YMGAMessageBox	17
YMGAMessageBoxPrivate	20
YMGAWidgetFactory	
Abstract extension widget factory to create widget extensions	21

Chapter 3

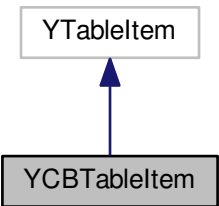
Class Documentation

3.1 YCBTableItem Class Reference

Inheritance diagram for YCBTableItem:



Collaboration diagram for YCBTableItem:



Public Member Functions

- [YCBTableItem](#) ()

Default constructor.

- [YCBTableItem](#) (const std::string &label_0, const std::string &label_1=std::string(), const std::string &label_2=std::string(), const std::string &label_3=std::string(), const std::string &label_4=std::string(), const std::string &label_5=std::string(), const std::string &label_6=std::string(), const std::string &label_7=std::string(), const std::string &label_8=std::string(), const std::string &label_9=std::string())

Convenience constructor for table items without any icons.

- virtual [~YCBTableItem](#) ()

Destructor.

- void **check** (bool val=true)
- bool **checked** ()

3.1.1 Detailed Description

Definition at line 43 of file [YMGA_CBTable.h](#).

3.1.2 Constructor & Destructor Documentation

3.1.2.1 [YCBTableItem::YCBTableItem](#) () [inline]

Default constructor.

Use addCell() to give it any content.

Definition at line 49 of file [YMGA_CBTable.h](#).

3.1.2.2 [YCBTableItem::YCBTableItem](#) (const std::string &label_0, const std::string &label_1 = std::string(), const std::string &label_2 = std::string(), const std::string &label_3 = std::string(), const std::string &label_4 = std::string(), const std::string &label_5 = std::string(), const std::string &label_6 = std::string(), const std::string &label_7 = std::string(), const std::string &label_8 = std::string(), const std::string &label_9 = std::string()) [inline]

Convenience constructor for table items without any icons.

This will create up to 10 (0..9) cells. Empty cells for empty labels at the end of the labels are not created, but empty cells in between are.

```
new YCBTableItem( "one", "two", "", "", "five" );
```

will create an item with 5 cells:

```
cell[0] ==> "one"
cell[1] ==> "two"
cell[2] ==> ""
cell[3] ==> ""
cell[4] ==> "five"
```

Definition at line 67 of file [YMGA_CBTable.h](#).

The documentation for this class was generated from the following file:

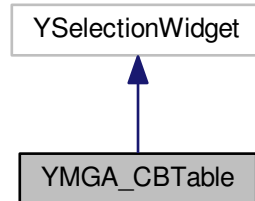
- /home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGA_CBTable.h

3.2 YMGA_CBTable Class Reference

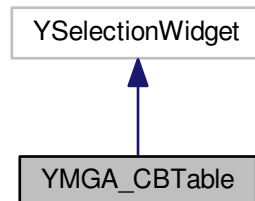
See document of [YMGA_CBTable](#) Table: Selection list with multiple columns.

```
#include <YMGA_CBTable.h>
```

Inheritance diagram for YMGA_CBTable:



Collaboration diagram for YMGA_CBTable:



Public Member Functions

- virtual [~YMGA_CBTable](#) ()
Destructor.
- virtual const char * [widgetClass](#) () const
Return a descriptive name of this widget class for logging, debugging etc.
- int [columns](#) () const
Return the number of columns of this table.
- bool [hasColumn](#) (int column) const
Return 'true' if this table has a column no.
- std::string [header](#) (int column) const
Return the header text for the specified column.

- YAlignmentType [alignment](#) (int column) const
Return the alignment for the specified column.
- bool [immediateMode](#) () const
Deliver even more events than with notify() set.
- void [setImmediateMode](#) (bool [immediateMode](#)=true)
Set [immediateMode\(\)](#) on or off.
- bool [keepSorting](#) () const
Return 'true' if the sort order is to be kept in item insertion order, i.e.
- virtual void [setKeepSorting](#) (bool [keepSorting](#))
Switch between sorting by item insertion order (keepSorting: true) or allowing the user to sort by an arbitrary column (by clicking on the column header).
- bool [hasMultiSelection](#) () const
Return 'true' if the user can select multiple items at the same time (e.g., with shift-click or ctrl-click).
- YCBTableMode [tableMode](#) ()
returns the YCBTable checkbox position mode
- virtual void [cellChanged](#) (const YTableCell *cell)=0
Notification that a cell (its text and/or its icon) was changed from the outside.
- virtual void [checkItem](#) (YItem *item, bool checked=true)=0
check/uncheck Item from application.
- virtual bool [setProperty](#) (const std::string &propertyName, const YPropertyValue &val)
Set a property.
- virtual YPropertyValue [getProperty](#) (const std::string &propertyName)
Get a property.
- virtual const YPropertySet & [propertySet](#) ()
Return this class's property set.
- const char * [userInputProperty](#) ()
The name of the widget property that will return user input.
- virtual void [addItem](#) (YItem *item_disown)
Add one item.
- void [setTableHeader](#) (YTableHeader *newHeader)
Exchange the previous table header with a new one.
- virtual YItem * [item](#) (int index) const
From YSelectionWidget returns the item at index 'index' (from 0) or 0 if there is no such item.
- virtual void [setChangedItem](#) (YCBTableItem *plItem)
When derived classes emit YWidgetEvent with reason ValueChanged they have to set which item is changed.
- virtual YCBTableItem * [changedItem](#) ()
Return the item which value is changed (e.g.
- YItemIterator [nextItem](#) (YItemIterator currentIterator)
YSelectionWidget does not implement the increment of iterator and bindings seem not to work with iterator++, next function just returns the iterator icrementation, NOTE that it does not check input parameter, just increment it.
- YItem * [YItemIteratorToYItem](#) (YItemIterator iter)
useful cast for bindings.
- YCBTableItem * [toCBYTableItem](#) (YItem *item)
useful cast for bindings.

Protected Member Functions

- [YMGA_CBTable](#) (YWidget *parent, YTableHeader *header, YCBTableMode mode)

Constructor.

3.2.1 Detailed Description

See document of [YMGA_CBTable](#) Table: Selection list with multiple columns.

The user can select exactly one row (with all its columns) from that list. Each cell (each column within each row) has a label text and an optional icon (*).

This widget is similar to SelectionBox, but it has several columns for each item (each row). If just one column is desired, consider using SelectionBox instead.

Note: This is not something like a spread sheet, and it doesn't pretend or want to be. Actions are performed on rows, not on individual cells (columns within one row).

(*) Not all UIs (in particular not text-based UIs) support displaying icons, so an icon should never be an exclusive means to display any kind of information.

Definition at line 114 of file [YMGA_CBTable.h](#).

3.2.2 Constructor & Destructor Documentation

3.2.2.1 `YMGA_CBTable::YMGA_CBTable (YWidget * parent, YTableHeader * header, YCBTableMode mode)` `[protected]`

Constructor.

'header' describes the table's headers: Number of columns, column headings, and column alignment. The widget assumes ownership of this object and will delete it when appropriate. The header cannot be changed after creating the widget.

'mode' indicates whether the checkbox is in the first or in the last column.

header must contains also header for checkbox column (empty string is allowed if not wanted)

Definition at line 55 of file [YMGA_CBTable.cc](#).

3.2.3 Member Function Documentation

3.2.3.1 `void YMGA_CBTable::addItem (YItem * item_disown)` `[virtual]`

Add one item.

This widget assumes ownership of the item object and will delete it in its destructor.

NOTE: For tree items, call this only for the toplevel items; all non-toplevel items are already owned by their respective parent items. Adding them to the parent widget will clash with this ownership.

Reimplementation of YSelectionWidget::addItem.

Definition at line 92 of file [YMGA_CBTable.cc](#).

3.2.3.2 `virtual void YMGA_CBTable::cellChanged (const YTableCell * cell)` `[pure virtual]`

Notification that a cell (its text and/or its icon) was changed from the outside.

Applications are required to call this whenever a table cell is changed after adding the corresponding table item (the row) to the table widget.

Derived classes are required to implement this and update the display accordingly.

Note that the position of this cell can be retrieved with `cell->column()` and `cell->itemIndex()`.

3.2.3.3 YCDataTableItem * YMGA_CDataTable::changedItem () [virtual]

Return the item which value is changed (e.g. checkbox).

Definition at line 75 of file [YMGA_CDataTable.cc](#).

3.2.3.4 virtual void YMGA_CDataTable::checkItem (YItem * *item*, bool *checked* = true) [pure virtual]

check/uncheck Item from application.

Derived classes are required to implement this and update the display accordingly.

Note that `item->check(true)` does not update the table

3.2.3.5 YPropertyValue YMGA_CDataTable::getProperty (const std::string & *propertyName*) [virtual]

Get a property.

Reimplemented from YWidget.

This method may throw YUIPropertyExceptions.

Definition at line 249 of file [YMGA_CDataTable.cc](#).

Here is the call graph for this function:



3.2.3.6 bool YMGA_CDataTable::hasColumn (int *column*) const

Return 'true' if this table has a column no.

'column' (counting from 0 on).

Definition at line 117 of file [YMGA_CDataTable.cc](#).

3.2.3.7 bool YMGA_CDataTable::immediateMode () const

Deliver even more events than with `notify()` set.

With "notify" alone, a table widget sends an ActivatedEvent when the user double-clicks an item or presses the "space" key on it. It does not send an event when the user just sends another item.

With "immediate", it also sends a SelectionChangedEvent when the user selects another item. "immediate" implicitly includes "notify".

Definition at line 135 of file [YMGA_CBTable.cc](#).

3.2.3.8 bool YMGA_CBTable::keepSorting () const

Return 'true' if the sort order is to be kept in item insertion order, i.e.
if sorting the table by clicking on a column header should be disabled.

Definition at line 150 of file [YMGA_CBTable.cc](#).

3.2.3.9 const YPropertySet & YMGA_CBTable::propertySet () [virtual]

Return this class's property set.

This also initializes the property upon the first call.

Reimplemented from YWidget.

Definition at line 188 of file [YMGA_CBTable.cc](#).

3.2.3.10 void YMGA_CBTable::setChangedItem (YCBTableItem * *pltem*) [virtual]

When derived classes emit YWidgetEvent with reason ValueChanged they have to set which item is changed.

Who manages the event have to use [changedItem\(\)](#) to get it.

Derived classes can overwrite this function, but they should call this base class function in the new implementation.

Definition at line 80 of file [YMGA_CBTable.cc](#).

3.2.3.11 void YMGA_CBTable::setKeepSorting (bool *keepSorting*) [virtual]

Switch between sorting by item insertion order (keepSorting: true) or allowing the user to sort by an arbitrary column (by clicking on the column header).

Derived classes can overwrite this function, but they should call this base class function in the new implementation.

Definition at line 156 of file [YMGA_CBTable.cc](#).

Here is the call graph for this function:



3.2.3.12 `bool YMGA_CBTable::setProperty (const std::string & propertyName, const YPropertyValue & val) [virtual]`

Set a property.

Reimplemented from YWidget.

This function may throw YUIPropertyExceptions.

This function returns 'true' if the value was successfully set and 'false' if that value requires special handling (not in error cases: those are covered by exceptions).

Definition at line 222 of file [YMGA_CBTable.cc](#).

Here is the call graph for this function:



3.2.3.13 `void YMGA_CBTable::setTableHeader (YTableHeader * newHeader)`

Exchange the previous table header with a new one.

This will delete the old YTableHeader object.

If the new header has a different number of columns than the old one, all items will implicitly be deleted.

Definition at line 99 of file [YMGA_CBTable.cc](#).

3.2.3.14 `YCBTableMode YMGA_CBTable::tableMode ()`

returns the YCBTable checkbox position mode

Definition at line 86 of file [YMGA_CBTable.cc](#).

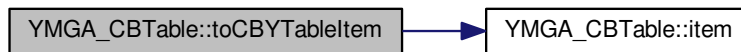
3.2.3.15 `YCBTableItem * YMGA_CBTable::toCBYTableItem (YItem * item)`

useful cast for bindings.

it just performs a `dynamic_cast`

Definition at line 183 of file [YMGA_CBTable.cc](#).

Here is the call graph for this function:



3.2.3.16 `const char* YMGA_CBTable::userInputProperty () [inline]`

The name of the widget property that will return user input.

Inherited from YWidget.

Definition at line 272 of file [YMGA_CBTable.h](#).

3.2.3.17 `YItem * YMGA_CBTable::YItemIteratorToYItem (YItemIterator iter)`

useful cast for bindings.

it does not any assumption on iter, so it is up to the user to check if it is valid, it just returns *it.

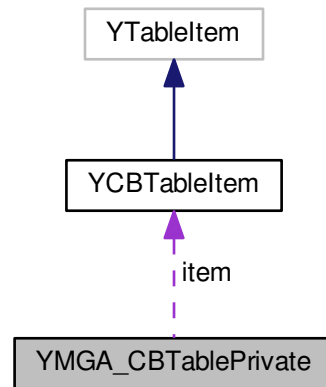
Definition at line 178 of file [YMGA_CBTable.cc](#).

The documentation for this class was generated from the following files:

- [/home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGA_CBTable.h](#)
- [/home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGA_CBTable.cc](#)

3.3 YMGA_CTablePrivate Struct Reference

Collaboration diagram for YMGA_CTablePrivate:



Public Member Functions

- **YMGA_CTablePrivate** (YTableHeader *header)

Public Attributes

- YTableHeader * **header**
- bool **keepSorting**
- bool **immediateMode**
- YCTableMode **mode**
- [YCTableItem](#) * **item**

3.3.1 Detailed Description

Definition at line 33 of file [YMGA_CTable.cc](#).

The documentation for this struct was generated from the following file:

- /home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGA_CTable.cc

3.4 YMGAAboutDialog Class Reference

Public Types

- enum **DLG_MODE** { **CLASSIC**, **TABBED** }

Public Member Functions

- [YMGAAboutDialog](#) (const std::string &name, const std::string &version, const std::string &license, const std::string &authors, const std::string &description, const std::string &logo, const std::string &icon=std::string(), const std::string &credits=std::string(), const std::string &information=std::string())

The constructor.

- void [show](#) (DLG_MODE type=TABBED)

it actually shows the about dialog

3.4.1 Detailed Description

Definition at line 26 of file [YMGAAboutDialog.h](#).

3.4.2 Constructor & Destructor Documentation

3.4.2.1 `YMGAAboutDialog::YMGAAboutDialog (const std::string & name, const std::string & version, const std::string & license, const std::string & authors, const std::string & description, const std::string & logo, const std::string & icon = std::string(), const std::string & credits = std::string(), const std::string & information = std::string())`

The constructor.

Note that this object is not a widget, so you must deleted it, if allocated on the heap.

Parameters

<i>name</i>	the application name
<i>version</i>	the application version
<i>license</i>	the application license, the short length one (e.g. GPLv2, GPLv3, LGPLv2+, etc)
<i>authors</i>	the string providing the list of authors; it could be html-formatted
<i>description</i>	the string providing a brief description of the application
<i>logo</i>	the string providing the file path for the application logo (high-res image)
<i>icon</i>	the string providing the file path for the application icon (low-res image)
<i>credits</i>	optional, the application credits, they can be html-formatted
<i>information</i>	optional, other extra informations, they can be html-formatted

Definition at line 78 of file [YMGAAboutDialog.cc](#).

3.4.3 Member Function Documentation

3.4.3.1 `void YMGAAboutDialog::show (YMGAAboutDialog::DLG_MODE type = TABBED)`

it actually shows the about dialog

Parameters

<i>type</i>	optional, DLG_MODE: defaulting to CLASSIC if not defined
-------------	--

See also

Classic()
 Tabbed()
 YMGAAboutDialog::DLG_MODE

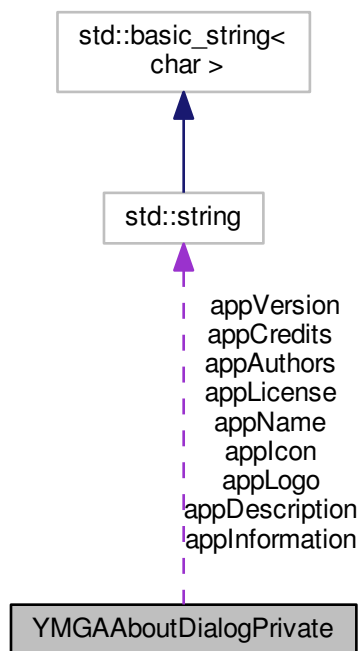
Definition at line 425 of file [YMGAAboutDialog.cc](#).

The documentation for this class was generated from the following files:

- /home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGAAboutDialog.h
- /home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGAAboutDialog.cc

3.5 YMGAAboutDialogPrivate Class Reference

Collaboration diagram for YMGAAboutDialogPrivate:



Public Attributes

- `std::string` **appName**
- `std::string` **appVersion**
- `std::string` **appLicense**
- `std::string` **appAuthors**
- `std::string` **appDescription**
- `std::string` **appLogo**
- `std::string` **applcon**
- `std::string` **appCredits**
- `std::string` **appInformation**
- `YDialog *` **mainDialog**

3.5.1 Detailed Description

Definition at line 48 of file [YMGAAboutDialog.cc](#).

The documentation for this class was generated from the following file:

- [/home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGAAboutDialog.cc](#)

3.6 YMGAMessageBox Class Reference

Public Types

- enum [DLG_BUTTON](#) { [B_ONE](#) = 0, [B_TWO](#) = 1 }
- enum [DLG_MODE](#) { [D_NORMAL](#), [D_INFO](#), [D_WARNING](#) }

Public Member Functions

- [YMGAMessageBox](#) ([DLG_BUTTON](#) b_num=[B_ONE](#), [DLG_MODE](#) dlg_mode=[D_NORMAL](#))
The constructor.
- virtual [~YMGAMessageBox](#) ()
Destructor.
- void [setIcon](#) (const std::string &icon)
sets the message box icon (full path)
- void [setTitle](#) (const std::string &title)
sets the message box title
- void [setText](#) (const std::string &text, bool useRichText=false)
sets the message box text information
- void [setMinSize](#) (YLayoutSize_t minWidth, YLayoutSize_t minHeight)
sets the dilaog box minimum size according to YWidgetFactory::createMinSize()
- void [setButtonLabel](#) (const std::string &label, [DLG_BUTTON](#) button=[B_ONE](#))
sets the message box button name (empty string is assigned by default)
- void [setDefaultButton](#) ([DLG_BUTTON](#) button=[B_ONE](#))
sets the message box default button
- [DLG_BUTTON show](#) ()
it shows the message box dialog using data set by user.

3.6.1 Detailed Description

Definition at line 34 of file [YMGAMsgBox.h](#).

3.6.2 Member Enumeration Documentation

3.6.2.1 enum YMGAMessageBox::DLG_BUTTON

Enumerator

- B_ONE** One button dialog, or button one pressed.
B_TWO two buttons dialog, or button two pressed

Definition at line 37 of file [YMGAMsgBox.h](#).

3.6.2.2 enum YMGAMessageBox::DLG_MODE

Enumerator

D_NORMAL Normal dialog.

D_INFO Info dialog.

D_WARNING Warning dialog.

Definition at line 44 of file [YMGAMsgBox.h](#).

3.6.3 Constructor & Destructor Documentation

3.6.3.1 YMGAMessageBox::YMGAMessageBox (YMGAMessageBox::DLG_BUTTON *b_num* = B_ONE, YMGAMessageBox::DLG_MODE *dlg_mode* = D_NORMAL)

The constructor.

Note that this object is not a widget, so you must deleted it, if allocated on the heap.

Parameters

<i>b_num</i>	B_ONE: one button only, B_TWO Two buttons dialog.
<i>dlg_mode</i>	NORMAL, INFO, WARNING dialog

See also

[YMGAMessageBox::DLG_BUTTON](#)

[YMGAMessageBox::DLG_MODE](#)

Definition at line 70 of file [YMGAMsgBox.cc](#).

3.6.4 Member Function Documentation

3.6.4.1 void YMGAMessageBox::setButtonLabel (const std::string & *label*, DLG_BUTTON *button* = B_ONE)

sets the message box button name (empty string is assigned by default)

Parameters

<i>label</i>	button name
<i>button</i>	button to set (B_ONE, or B_TWO if configured for two buttons)

Definition at line 118 of file [YMGAMsgBox.cc](#).

3.6.4.2 void YMGAMessageBox::setDefaultButton (DLG_BUTTON *button* = B_ONE)

sets the message box default button

Parameters

<i>button</i>	button to set (B_ONE, or B_TWO if configured for two buttons)
---------------	---

Definition at line 126 of file [YMGAMsgBox.cc](#).

3.6.4.3 void YMGAMessageBox::setIcon (const std::string & icon)

sets the message box icon (full path)

Parameters

<i>icon</i>	icon pathname
-------------	---------------

Definition at line 89 of file [YMGAMsgBox.cc](#).

3.6.4.4 void YMGAMessageBox::setMinSize (YLayoutSize_t minWidth, YLayoutSize_t minHeight)

sets the dilaog box minimum size according to YWidgetFactory::createMinSize()

Parameters

<i>minWidth</i>	dialog min width
<i>minHeight</i>	dialog min height

Definition at line 106 of file [YMGAMsgBox.cc](#).

3.6.4.5 void YMGAMessageBox::setText (const std::string & text, bool useRichText = false)

sets the message box text information

Parameters

<i>text</i>	dialog content text
<i>useRichText</i>	set text using rich text if true

Definition at line 100 of file [YMGAMsgBox.cc](#).

3.6.4.6 void YMGAMessageBox::setTitle (const std::string & title)

sets the message box title

Parameters

<i>title</i>	title text
--------------	------------

Definition at line 95 of file [YMGAMsgBox.cc](#).

3.6.4.7 YMGAMessageBox::DLG_BUTTON YMGAMessageBox::show ()

it shows the message box dialog using data set by user.

Returns

which button has been pressed to leave the dialog (B_ONE or B_TWO)

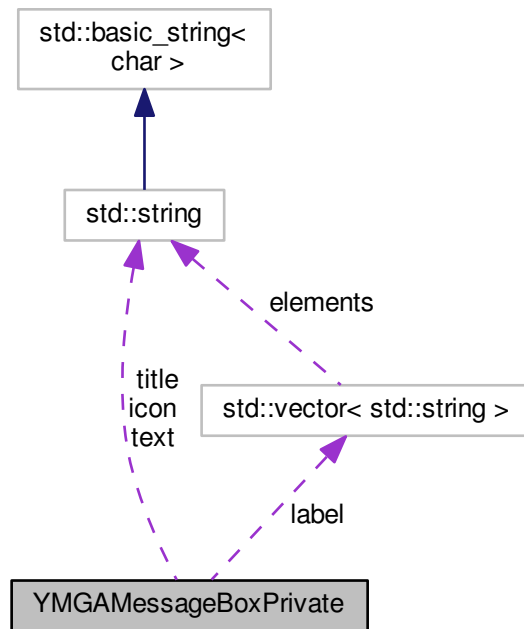
Definition at line 135 of file [YMGAMsgBox.cc](#).

The documentation for this class was generated from the following files:

- /home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGAMsgBox.h
- /home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGAMsgBox.cc

3.7 YMGAMessageBoxPrivate Class Reference

Collaboration diagram for YMGAMessageBoxPrivate:

**Public Attributes**

- `std::string` **title**
- `std::string` **text**
- `std::string` **icon**
- `bool` **useRichText**
- `YLayoutSize_t` **minWidth**
- `YLayoutSize_t` **minHeight**
- [YMGAMessageBox::DLG_BUTTON](#) **buttons**

- [YMGAMessageBox::DLG_MODE](#) **mode**
- [YMGAMessageBox::DLG_BUTTON](#) **default_button**
- `std::vector< std::string >` **label**

3.7.1 Detailed Description

Definition at line 47 of file [YMGAMsgBox.cc](#).

The documentation for this class was generated from the following file:

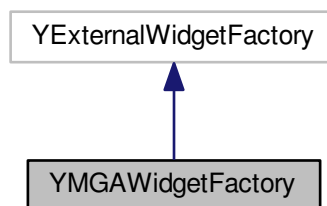
- `/home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGAMsgBox.cc`

3.8 YMGAWidgetFactory Class Reference

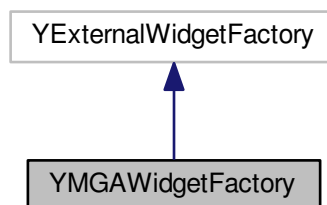
Abstract extension widget factory to create widget extensions.

```
#include <YMGAWidgetExtensionFactory.h>
```

Inheritance diagram for YMGAWidgetFactory:



Collaboration diagram for YMGAWidgetFactory:



Public Member Functions

- [YMGAAboutDialog](#) * [createAboutDialog](#) (const std::string &appname, const std::string &appversion, const std::string &applicense, const std::string &appauthors, const std::string &appdescription, const std::string &applogo, const std::string &appicon=std::string(), const std::string &appcredits=std::string(), const std::string &appinfo=std::string())
creates an About Dialog providing the information passed as arguments Remember to remove this instance after.
- virtual [YMGACBTable](#) * [createCBTable](#) (YWidget *parent, YTableHeader *header_disown, YCBTableMode mode=YCBTableCheckBoxOnFirstColumn)=0
creates a check boxed table
- [YMGAMessageBox](#) * [createDialogBox](#) ([YMGAMessageBox::DLG_BUTTON](#) button_number=[YMGAMessageBox::B_ONE](#), [YMGAMessageBox::DLG_MODE](#) dialog_mode=[YMGAMessageBox::D_NORMAL](#))
creates a dialog box dialog, use the instance to set dialog information and call show() to use it.
- [YMGAMessageBox](#) * [createMessageBox](#) (const std::string &title, const std::string &text, bool useRichText, const std::string &btn_label)
creates an messagebox dialog, use the instance to set dialog information and call show() to use it.
- [YMGAMessageBox](#) * [createInfoBox](#) (const std::string &title, const std::string &text, bool useRichText, const std::string &btn_label)
creates an Info messagebox dialog, use the instance to set dialog information and call show() to use it.
- [YMGAMessageBox](#) * [createWarningBox](#) (const std::string &title, const std::string &text, bool useRichText, const std::string &btn_label)
creates an Warning messagebox dialog, use the instance to set dialog information and call show() to use it.

Static Public Member Functions

- static [YMGAWidgetFactory](#) * [getYMGAWidgetFactory](#) (YExternalWidgetFactory *instance)
casts useful for bindings from YExternalWidgetFactory to YMGAWidgetFactory**
- static YWidgetEvent * [getYWidgetEvent](#) (YEvent *event)
casts useful for bindings from YEvent to YWidgetEvent**
- static YKeyEvent * [getYKeyEvent](#) (YEvent *event)
casts useful for bindings from YEvent to YKeyEvent**
- static YMenuEvent * [getYMenuEvent](#) (YEvent *event)
casts useful for bindings from YEvent to YMenuEvent**
- static YCancelEvent * [getYCancelEvent](#) (YEvent *event)
casts useful for bindings from YEvent to YCancelEvent**
- static YDebugEvent * [getYDebugEvent](#) (YEvent *event)
casts useful for bindings from YEvent to YDebugEvent**
- static YTimeoutEvent * [getYTimeoutEvent](#) (YEvent *event)
casts useful for bindings from YEvent to YTimeoutEvent**

Protected Member Functions

- [YMGAWidgetFactory](#) ()
Constructor.
- virtual [~YMGAWidgetFactory](#) ()
Destructor.

Friends

- class **YExternalWidgets**

3.8.1 Detailed Description

Abstract extension widget factory to create widget extensions.

Refer to the respective widget's documentation (in the header file) for documentation about the function parameters.

Definition at line 52 of file [YMGAWidgetExtensionFactory.h](#).

3.8.2 Constructor & Destructor Documentation

3.8.2.1 YMGAWidgetFactory::YMGAWidgetFactory () [protected]

Constructor.

Use YExternalWidgets::widgetExtensionFactory() to get the singleton for this class.

Definition at line 41 of file [YMGAWidgetExtensionFactory.cc](#).

3.8.3 Member Function Documentation

3.8.3.1 YMGAAboutDialog * YMGAWidgetFactory::createAboutDialog (const std::string & *appname*, const std::string & *appversion*, const std::string & *applicense*, const std::string & *appauthors*, const std::string & *appdescription*, const std::string & *applogo*, const std::string & *appicon* = std::string(), const std::string & *appcredits* = std::string(), const std::string & *appinfo* = std::string())

creates an About Dialog providing the information passed as arguments Remember to remove this instance after.

Parameters

<i>appname</i>	the application name
<i>appver</i>	the application version
<i>applicense</i>	the application license, short-length (e.g. GPLv2, GPLv3, LGPL, LGPLv2+, etc)
<i>appauthors</i>	the application authors
<i>appdescription</i>	a brief description of the application
<i>applogo</i>	the file path to the application logo
<i>appicon</i>	the file path to the application icon
<i>appcredits</i>	optional, the credits
<i>appinfo</i>	optional, other extra information

Returns

[YMGAAboutDialog](#) instance

Definition at line 52 of file [YMGAWidgetExtensionFactory.cc](#).

3.8.3.2 virtual YMGA_CBTable* YMGAWidgetFactory::createCBTable (YWidget * *parent*, YTableHeader * *header_disown*, YCBTableMode *mode* = YCBTableCheckBoxOnFirstColumn) [pure virtual]

creates a check boxed table

Parameters

<i>parent</i>	parent widget
<i>header_disown</i>	YTable header
<i>mode</i>	check box position (YCBTableCheckBoxOnFirstColumn, YCBTableCheckBoxOnLastColumn)

Returns

[YMGA_CBTable](#) widget instance

3.8.3.3 `YMGAMessageBox * YMGAWidgetFactory::createDialogBox (YMGAMessageBox::DLG_BUTTON
button_number = YMGAMessageBox::B_ONE, YMGAMessageBox::DLG_MODE dialog_mode =
YMGAMessageBox::D_NORMAL)`

creates a dialog box dialog, use the instance to set dialog information and call show() to use it.

Remember to remove this instance after.

Parameters

<i>button_number</i>	one button (such as "ok" for instance) B_ONE or two buttons (such as "ok" and "cancel") B_TWO
<i>dialog_mode</i>	dialog type (D_NORMAL, D_INFO, D_WARNING)

Returns

[YMGAMessageBox](#) dialog instance

Definition at line 68 of file [YMGAWidgetExtensionFactory.cc](#).

3.8.3.4 `YMGAMessageBox * YMGAWidgetFactory::createInfoBox (const std::string & title, const std::string & text, bool
useRichText, const std::string & btn_label)`

creates an Info messagebox dialog, use the instance to set dialog information and call show() to use it.

Remember to remove this instance after.

Parameters

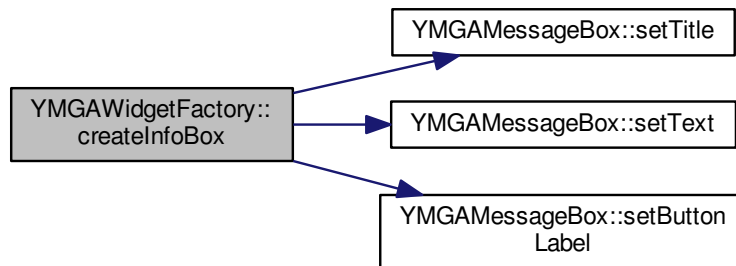
<i>title</i>	dialog title
<i>text</i>	dialog content text
<i>useRichText</i>	set text using rich text if true
<i>btn_label</i>	button label (such as Ok, for instance)

Returns

[YMGAMessageBox](#) dialog instance

Definition at line 84 of file [YMGAWidgetExtensionFactory.cc](#).

Here is the call graph for this function:



3.8.3.5 `YMGAMessageBox * YMGAWidgetFactory::createMessageBox (const std::string & title, const std::string & text, bool useRichText, const std::string & btn_label)`

creates an messagebox dialog, use the instance to set dialog information and call `show()` to use it.

Remember to remove this instance after.

Parameters

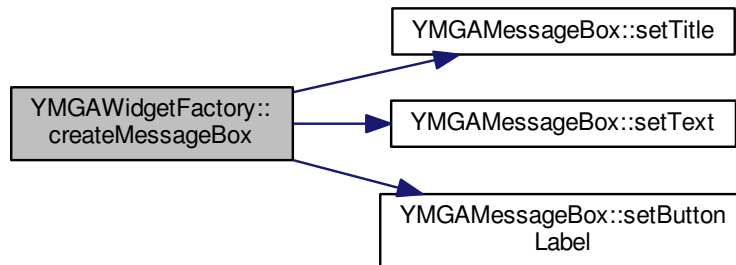
<i>title</i>	dialog title
<i>text</i>	dialog content text
<i>useRichText</i>	set text using rich text if true
<i>btn_label</i>	button label (such as Ok, for instance)

Returns

[YMGAMessageBox](#) dialog instance

Definition at line 75 of file [YMGAWidgetExtensionFactory.cc](#).

Here is the call graph for this function:



3.8.3.6 YMGAMessageBox * YMGAWidgetFactory::createWarningBox (const std::string & *title*, const std::string & *text*, bool *useRichText*, const std::string & *btn_label*)

creates an Warning messagebox dialog, use the instance to set dialog information and call `show()` to use it.

casts

Remember to remove this instance after.

Parameters

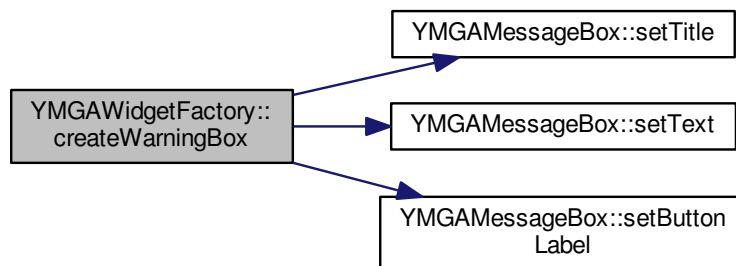
<i>title</i>	dialog title
<i>text</i>	dialog content text
<i>useRichText</i>	set text using rich text if true
<i>btn_label</i>	button label (such as Ok, for instance)

Returns

[YMGAMessageBox](#) dialog instance

Definition at line 93 of file [YMGAWidgetExtensionFactory.cc](#).

Here is the call graph for this function:



3.8.3.7 YCancelEvent * YMGAWidgetFactory::getYCancelEvent (YEvent * *event*) [static]

casts useful for bindings from YEvent* to YCancelEvent*

Definition at line 123 of file [YMGAWidgetExtensionFactory.cc](#).

3.8.3.8 YDebugEvent * YMGAWidgetFactory::getYDebugEvent (YEvent * *event*) [static]

casts useful for bindings from YEvent* to YDebugEvent*

Definition at line 128 of file [YMGAWidgetExtensionFactory.cc](#).

3.8.3.9 YKeyEvent * YMGAWidgetFactory::getYKeyEvent (YEvent * *event*) [static]

casts useful for bindings from YEvent* to YKeyEvent*

Definition at line 113 of file [YMGAWidgetExtensionFactory.cc](#).

3.8.3.10 YMenuEvent * YMGAWidgetFactory::getYMenuEvent (YEvent * *event*) [static]

casts useful for bindings from YEvent* to YMenuEvent*

Definition at line 118 of file [YMGAWidgetExtensionFactory.cc](#).

3.8.3.11 YMGAWidgetFactory * YMGAWidgetFactory::getYMGAWidgetFactory (YExternalWidgetFactory * *instance*) [static]

casts useful for bindings from YExternalWidgetFactory* to YMGAWidgetFactory*

Definition at line 103 of file [YMGAWidgetExtensionFactory.cc](#).

3.8.3.12 `YTimeoutEvent * YMGAWidgetFactory::getYTimeoutEvent (YEvent * event)` `[static]`

casts useful for bindings from YEvent* to YTimeoutEvent*

Definition at line 133 of file [YMGAWidgetExtensionFactory.cc](#).

3.8.3.13 `YWidgetEvent * YMGAWidgetFactory::getYWidgetEvent (YEvent * event)` `[static]`

casts useful for bindings from YEvent* to YWidgetEvent*

Definition at line 108 of file [YMGAWidgetExtensionFactory.cc](#).

The documentation for this class was generated from the following files:

- [/home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGAWidgetExtensionFactory.h](#)
- [/home/iurt/rpmbuild/BUILD/libyui-mga-1.0.4/src/YMGAWidgetExtensionFactory.cc](#)

Index

B_ONE

YMGAMessageBox, [17](#)

B_TWO

YMGAMessageBox, [17](#)

D_INFO

YMGAMessageBox, [18](#)

D_NORMAL

YMGAMessageBox, [18](#)

D_WARNING

YMGAMessageBox, [18](#)

YMGAMessageBox

B_ONE, [17](#)

B_TWO, [17](#)

D_INFO, [18](#)

D_NORMAL, [18](#)

D_WARNING, [18](#)