SIMULIA

3DEXPERIENCE

ABAQUS 2026 FD04
GUI TOOLKIT REFERENCE GUIDE

DASSAULT SYSTEMES

Trademarks and Legal Notices

Trademarks

Abaqus, **3D**EXPERIENCE®, the 3DS logo, the Compass icon, IFWE, 3DEXCITE, 3DVIA, BIOVIA, CATIA, CENTRIC PLM, DELMIA, ENOVIA, GEOVIA, MEDIDATA, NETVIBES, OUTSCALE, SIMULIA and SOLIDWORKS are commercial trademarks or registered trademarks of Dassault Systèmes, a European company (Societas Europaea) incorporated under French law, and registered with the Versailles trade and companies registry under number 322 306 440, or its subsidiaries in the United States and/or other countries. All other trademarks are owned by their respective owners. Use of any Dassault Systèmes or its subsidiaries trademarks is subject to their express written approval.

Legal Notices

Abaqus and this documentation may be used or reproduced only in accordance with the terms of the software license agreement signed by the customer, or, absent such agreement, the then current software license agreement to which the documentation relates.

This documentation and the software described in this documentation are subject to change without prior notice.

Dassault Systèmes or its Affiliates shall not be responsible for the consequences of any errors or omissions that may appear in this documentation.

© Dassault Systèmes Americas Corp., 2026.

For a full list of the third-party software contained in this release, please go to the Legal Notices in the Abaqus 2026 HTML documentation, which can be obtained from a documentation installation, or in the SIMULIA Established Products 2026 Program Directory, which is available on *www.3ds.com*.

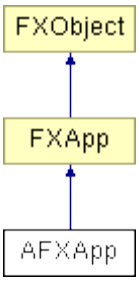
What's New

There are no enhancements in this release.

All Classes

AFXApp

This class is responsible for providing some high-level GUI control methods.



AFXApp(appName=Abaqus/CAE, vendorName=SIMULIA, productName='', majorNumber=-1, minorNumber=-1, updateNumber=-1, prerelease=False)

Constructor.

Argument	Type	Default	Description
appName	String	Abaqus/CAE	Application registry key.
vendorName	String	SIMULIA	Vendor registry key.
productName	String	''	Product name.
majorNumber	Int	-1	Version number.
minorNumber	Int	-1	Release number.
updateNumber	Int	-1	Update number.
prerelease	Bool	False	Official/Prerelease flag.

create()

Creates windows for the application.

Reimplemented from FXApp.

getAFXMainWindow()

Returns a pointer to the AFXMainWindow.

getBasePrerelease()

Returns True if the base product is a prerelease.

getBaseProductName()

Returns the base product name.

getBaseVersionNumbers(majorNumber, minorNumber, updateNumber)

Returns the base product's major, minor, and update numbers.

Argument	Type	Default	Description
majorNumber	Int		Version number.

minorNumber	Int		Release number.
updateNumber	Int		Update number.

getKernelInitializationCommand()

Returns the command string that will be issued upon application startup.

getPrerelease()

Returns True if this is a prerelease.

getProductName()

Returns the product name.

getVersionNumbers()

Returns the major, minor, and update numbers.

init(argc, argv)

Initializes the application and connects to the kernel.

Argument	Type	Default	Description
argc	Int		
argv	String		

isLocked()

Returns True if the GUI is locked or False if otherwise.

Reimplemented from FXApp.

isProductCAE()

Returns True if the base product is Abaqus/CAE.

isProductViewer()

Returns True if the base product is Abaqus/Viewer.

isLearningEdition()

Returns True if the base product is a learning edition.

lock()

Locks the GUI (normally used during command and mode processing).

run()

Runs the main application event loop until stop() is called,.

Reimplemented from FXApp.

runUntil(condition)

Run an event loop till some flag becomes non-zero.

Reimplemented from FXApp.

Argument	Type	Default	Description
condition	Int		

unlock()

Unlocks the GUI.

Class flags

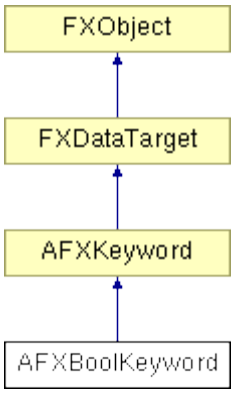
Message ID's.

ID_QUERY_GUILOCK **Used to query whether the GUI is locked.**

ID_SHOW_HOURGLASS **Used to change the cursor.**

AFXBoolKeyword

This class is designed for command keywords that have Boolean values.



AFXBoolKeyword(command, name, booleanType=ON_OFF, isRequired=False, defaultValue=False)

Constructor.

Argument	Type	Default	Description
command	AFXCommand		Host command.
name	String		Keyword name.
booleanType	Type	ON_OFF	Type of boolean used in the command.
isRequired	Bool	False	True if the keyword is a required argument of the command.
defaultValue	Bool	False	Default value.

getTypeName()

Returns the name of the keyword type.

Implements AFXKeyword.

getValue()

Returns the keyword's current value.

getValueAsString()

Returns the text string that represents the keyword's current value.

Implements AFXKeyword.

isValueChanged()

Returns True if the keyword value differs from its previous value.

Implements AFXKeyword.

setDefaultValue(defaultValue)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	Bool		Default value.

setDefaultValueByString(defaultValueString)

Sets the keyword's default value (returns True if the given text string is valid).

Argument	Type	Default	Description
defaultValueString	String		Default value in text string form.

setDefaultValueByString(defaultValueString)

Sets the keyword's default value (returns True if the given text string is valid).

Argument	Type	Default	Description
defaultValueString	String		Default value in text string form.

setValue(newValue)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	Bool		New value.

setValueByString(newValueString)

Sets the keyword's current value (returns True if the given text string is valid).

Argument	Type	Default	Description
newValueString	String		New value in text string form.

setValueByString(newValueString)

Sets the keyword's current value (returns True if the given text string is valid).

Argument	Type	Default	Description
newValueString	String		New value in text string form.

setValueToDefault(ignoreUnspecified=False)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Not used.

setValueToPrevious()

Sets the keyword value to its previous value.

Implements AFXKeyword.

syncPreviousValue()

Sets the keyword's previous value to its current value.

Implements AFXKeyword.

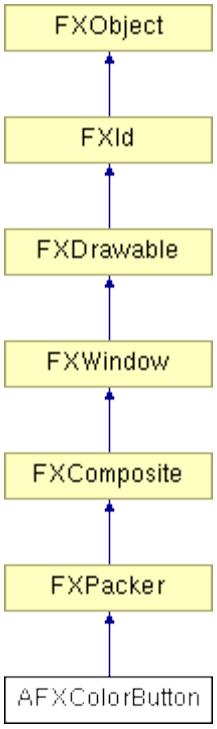
Class flags

Flags for the type of the boolean.

ON_OFF	**Keyword value will be ON or OFF.**
TRUE_FALSE	**Keyword value will be True or False.**

AFXColorButton

This class contains a label that precedes a color well, which allows the user to bring up a color dialog box by double clicking. When connected to an AFXStringKeyword, this widget will assign the value of the button's current color to the keyword in hex format (for example, "FF0000").



AFXColorButton(p, text, tgt=None, sel=0, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
text	String		Label string.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	0	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Width of the widget.
pl	Int	DEFAULT_SPACING	Left padding (margin).
pr	Int	DEFAULT_SPACING	Right padding (margin).
pt	Int	DEFAULT_SPACING	Top padding (margin).
pb	Int	DEFAULT_SPACING	Bottom padding (margin).

create()

Creates the color button widget.

Reimplemented from FXComposite.

disable()

Disables the color button.

Reimplemented from FXWindow.

enable()

Enables the color button.

Reimplemented from FXWindow.

getHelpText()

Returns the status line help text.

getLabelFont()

Returns the label font.

getLabelText()

Returns the label string.

getRGBA()

Returns the color of the button.

getTipText()

Returns the tool tip message.

setHelpText(text)

Sets the status line help text.

Argument	Type	Default	Description
text	String		

setLabelFont(fnt)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		

setLabelText(txt)

Sets the label string.

Argument	Type	Default	Description
txt	String		

setRGBA(clr)

Sets the color of the button.

Argument	Type	Default	Description
clr	FXColor		

setTipText(text)

Sets the tool tip message.

Argument	Type	Default	Description
text	String		

Class flags

Message ID's.

ID_COLORWELL **ID for color button.**

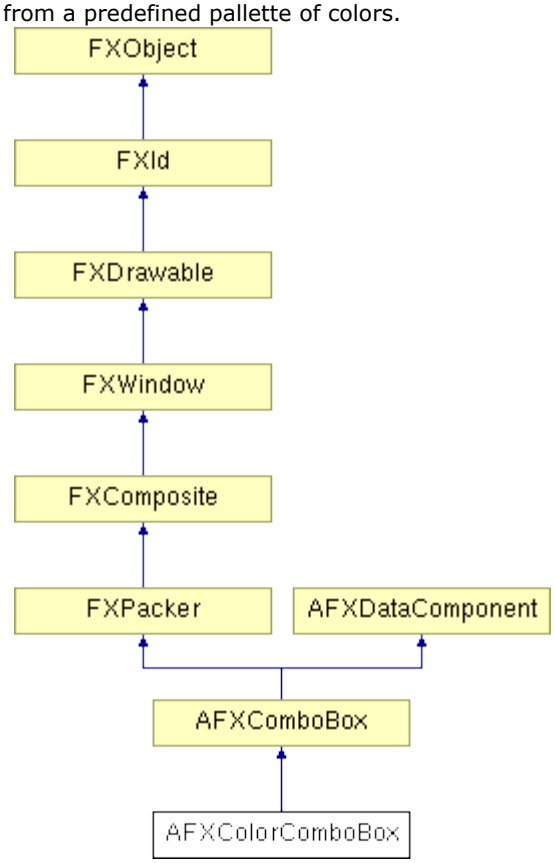
Global flags

Flags for AFX color button options.

AFXCOLORBUTTON_VERTICAL **Orient label above button.**

AFXColorComboBox

This class allows the user to select a color from a predefined pallette of colors.



AFXColorComboBox(p, text, tgt=None, sel=0, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
text	String		Label string.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	0	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Width of the widget.
pl	Int	DEFAULT_PAD	Left padding (margin).
pr	Int	DEFAULT_PAD	Right padding (margin).
pt	Int	DEFAULT_PAD	Top padding (margin).
pb	Int	DEFAULT_PAD	Bottom padding (margin).

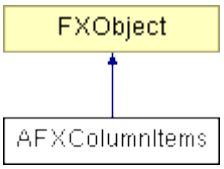
Global flags

Flags for color selector options.

AFXCOLORCOMBOBOX_INCLUDE_AS_IS	**Include "As is" color.**
AFXCOLORCOMBOBOX_INCLUDE_DEFAULT	**Include "Default" color.**

AFXColumnItems

This class connects the selected items in a single column of an AFXTable to a keyword (typically a tuple keyword).



AFXColumnItems(table, referenceColumn, opts=0)

Constructor.

Argument	Type	Default	Description
table	AFXTable		Table to use.
referenceColumn	Int		Index of the reference column.
opts	Int	0	Selection options (not used).

AFXColumnItems(referenceColumn, tgt=None, sel=0, opts=0)

Constructor for use with a keyword.

Argument	Type	Default	Description
referenceColumn	Int		Index of the reference column.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	0	Selection options (not used).

getReferenceColumn()

Returns the index of the table reference column.

getSelector()

Returns the message ID.

getTarget()

Returns the message target.

setReferenceColumn(index)

Sets the table reference column, whose selected items will be sent to the target.

Argument	Type	Default	Description
index	Int		Table column index.

setSelector(sel)

Sets the message ID.

Argument	Type	Default	Description
sel	Int		New message ID.

setTarget(tgt)

Sets the message target.

Argument	Type	Default	Description
tgt	FXObject		New message target.

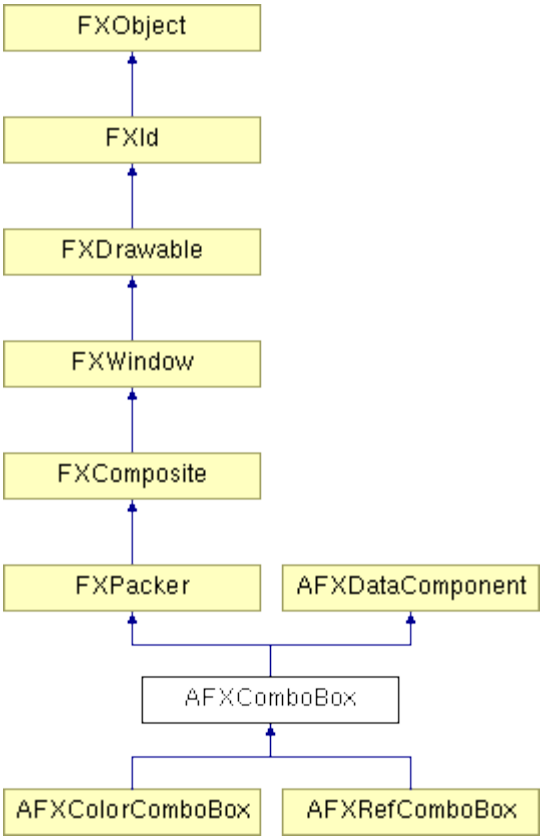
Class flags

Message ID's.

ID_TABLE **Table ID.**

AFXComboBox

This class implements a labeled combo box. It allows the user to select entries from a drop-down list. Each item has associated data set as integers as items are added.



AFXComboBox(p, ncols, nvis, text, tgt=None, sel=0, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
ncols	Int		Number of columns in the combo box (use 0 for auto-sizing).
nvis	Int		Number of visible items in the combo box's drop down list.
text	String		Label string.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	0	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	DEFAULT_PAD	Left padding (margin).
pr	Int	DEFAULT_PAD	Right padding (margin).

pt	Int	DEFAULT_PAD	Top padding (margin).
pb	Int	DEFAULT_PAD	Bottom padding (margin).

appendItem(text, sel=0)

Adds an item to the end of the list.

Argument	Type	Default	Description
text	String		Text.
sel	Int	0	Integer associated with this item.

clearItems()

Removes all items from the list.

create()

Creates the combo box.

Reimplemented from FXComposite.

disable()

Disables the combo box.

Reimplemented from FXWindow.

enable()

Enables the combo box.

Reimplemented from FXWindow.

getCheck()

Returns the state of the check button or the radio button.

getCurrentItem()

Returns the index of the current item.

getHelpText()

Returns the status line help text.

getItemData(index)

Returns the data for the specified item.

Argument	Type	Default	Description
index	Int		Index.

getItemIndexForData(data)

Returns the index of the first item with the associated data or -1 if not found.

Argument	Type	Default	Description
data			

getItemIndexForFloat(val)

Returns the index of the first item with the text evaluating to the given value.

Argument	Type	Default	Description
val	Float		

getItemProvider()

Returns the provider of the combo box's items.

getItemText(index)

Returns the text for the specified item.

Argument	Type	Default	Description
index	Int		Index.

getLabelFont()

Returns the label font.

getLabelText()

Returns the label string.

getNumColumns()

Returns the number of columns.

getNumItems()

Returns the number of items in the list.

getNumVisible()

Returns the number of visible items.

getText()

Returns the text displayed in input field.

getTipText()

Returns the tool tip message.

insertItem(index, text, sel=0)

Inserts a new item at the specified index position.

Argument	Type	Default	Description
index	Int		Index.
text	String		Text.
sel	Int	0	Integer associated with this item

isEditable()

Returns True if the text in the input field may be edited.

isItemCurrent(index)

Returns True if the item at the specified index position is the current item.

Argument	Type	Default	Description
index	Int		Index.

isReadOnlyState()

Returns True if the combo box appears in the read-only state.

removeItem(index)

Removes the item at the specified index position from the list.

Argument	Type	Default	Description
index	Int		Index.

replaceItem(index, text, sel=0)

Replaces the item at the specified index position.

Argument	Type	Default	Description
index	Int		Index.
text	String		Text.
sel	Int	0	Integer associated with this item

setCheck(state)

Sets the state of the check button or the radio button.

Argument	Type	Default	Description
state	Bool		Button state.

setCheckButtonSelector(sel)

Sets the message ID of the check button or the radio button.

Argument	Type	Default	Description
sel	Int		Selector.

setCheckButtonTarget(tgt, checkVal=False)

Sets the message target of the check button or the radio button.

Argument	Type	Default	Description
tgt	FXObject		Target.
checkVal	Bool	False	Value of check button.

setCurrentItem(index)

Sets the current item (the index is zero-based).

Argument	Type	Default	Description
index	Int		Index.

setEditable(edit=True)

Sets the editable state for the input field.

Argument	Type	Default	Description
edit	Bool	True	Editable state.

setFocusAndSelection()

Moves the focus to the input field and selects its contents if the combo box is editable.

setFocusToCheckButton()

Moves the focus to the check button or the radio button (if existed) of the widget.

setFocusToComboBox()

Moves the focus to the input field of the widget.

setHelpText(text)

Sets the status line help text.

Argument	Type	Default	Description
text	String		Help text.

setItemData(index, ptr)

Sets the data for the specified item.

Argument	Type	Default	Description
index	Int		Index.
ptr	String		Data.

setItemProvider(cp)

Sets the provider of this object items.

Argument	Type	Default	Description
cp	FXObject		Item provider.

setItemText(index, text)

Sets the text for the specified item.

Argument	Type	Default	Description
index	Int		Index.
text	String		Text.

setLabelFont(fnt)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		Label font.

setLabelText(txt)

Sets the label string.

Argument	Type	Default	Description
txt	String		Label text.

setMaxVisible(maxVis)

Sets the maximum number of visible items. The combo box will show up to the given maximum number of items in its list. If the combo box has more items, its list will show a scroll bar.

Argument	Type	Default	Description
maxVis	Int		Maximum number of visible items.

setNumColumns(cols)

Sets the number of columns in the combo box; passing zero will cause the combo box to always have the number of columns equal to the maximum item length.

Argument	Type	Default	Description
cols	Int		Number of columns.

setNumVisible(nvis)

Sets the number of visible items.

Argument	Type	Default	Description
nvis	Int		Number of visible items.

setReadOnlyState(readonly=True)

Sets the read-only state of the combo box.

Argument	Type	Default	Description

readonly	Bool	True	Read-only state.

setText(txt)

Sets the text displayed in the input field.

Argument	Type	Default	Description
txt	String		Input field text.

setTipText(text)

Sets the tool tip message.

Argument	Type	Default	Description
text	String		Tooltip text.

Class flags

Message ID's.

ID_BUTTON	**Label or button ID.**
ID_COMBO	**Combo box ID.**
ID_INCREMENT	**Up arrow button ID.**
ID_DECREMENT	**Down arrow button ID.**

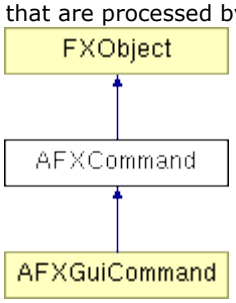
Global flags

Flags for AFX combo box options.

AFXCOMBOBOX_CHECKBUTTON	**Use a check button instead of a label.**
AFXCOMBOBOX_RADIOBUTTON	**Use a radio button instead of a label.**
AFXCOMBOBOX_VERTICAL	**Orient label or button above combo box.**
AFXCOMBOBOX_FLOAT	**Allow interaction with float keywords.**
AFXCOMBOBOX_READONLY	**Configure combo box to the read-only state.**
AFXCOMBOBOX_SPINNER	**Include spinner buttons.**

AFXCommand

This class is the abstract base class for command classes that are processed by modes.



AFXCommand(mode, method, objectName='', registerQuery=False)

Constructor.

Argument	Type	Default	Description
mode	AFXMode		Host mode.
method	String		Method.
objectName	String	''	Object name.
registerQuery	Bool	False	True if a query should be registered when the command is used for the GUI.

activate()

Activates the command; active commands will be processed during command generation.

deactivate()

Deactivates the command; inactive commands will not be processed during command generation.

getCommandString()

Returns the command string based on the current values of the active keywords.

getExpandedObjectName()

Returns the expanded object name that has all the "%s"'s replaced by the current names.

getKeyword(name)

Returns the keyword with the given name (returns 0 if none is found).

Argument	Type	Default	Description
name	String		Keyword name.

getKeyword(index)

Returns the keyword at the given index (returns 0 if the index is out-of-bounds).

Argument	Type	Default	Description

index	Int		Keyword index (0-based).

getMethod()

Returns the command's method.

getNumKeywords()

Returns the number of keywords.

getObjectName()

Returns the object name (which is not expanded and may include "%s"'s).

isActive()

Returns True if the command is active.

isQueryNeeded()

Returns True if the command needs to register a query for kernel state.

isRequired()

Returns True if this command is going to be sent even if none of its keywords has been modified, otherwise returns False.

setKeywordValuesToDefaults(ignoreUnspecified=False)

Sets the values of all keywords to their defaults.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Ignore setting the value if the default is unspecified.

setKeywordValuesToPrevious()

Sets the values of all keywords to their previous values.

setMethod(method)

Sets the command's method.

Argument	Type	Default	Description
method	String		Method.

setObjectName(objectName)

Sets the object name.

Argument	Type	Default	Description
objectName	String		Object name.

setRequired(val)

Sets this command as required or optional; if True the command will always be sent, if False the command will be sent only if it has modified keywords or if it has no keywords.

Argument	Type	Default	Description
val	Bool		

syncKeywordPreviousValues()

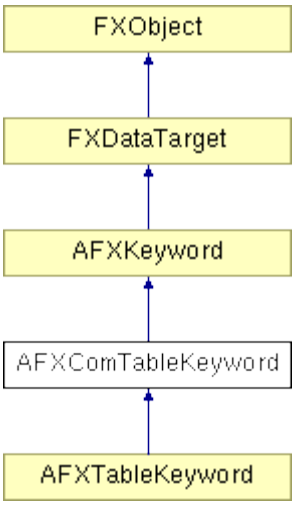
Synchronizes the current values and previous values of all keywords.

verify()

Throws an exception if any of the keywords contain invalid data.

AFXComTableKeyword

This class manages values which are sent as tables in a command.



AFXComTableKeyword(command, name, isRequired=False, minLength=0, maxLength=-1, opts=0)

Constructor.

Argument	Type	Default	Description
command	AFXCommand		Host command.
name	String		Keyword name.
isRequired	Bool	False	True if this keyword is a required argument.
minLength	Int	0	Minimum (and default) row length.
maxLength	Int	-1	Maximum row length (-1 => unlimited).
opts	Int	0	Options.

equal(index, a, b)

Returns True if the two table element values compare equal (index not used).

Argument	Type	Default	Description
index	Int		Element index (not used).
a	String		First value.
b	String		Second value.

getColumnStyle(index)

Returns the style of the column elements. Will never return AFXTABLE_STYLE_DEFAULT!

Argument	Type	Default	Description
index	Int		Column index.

getColumnType(index)

Returns the type of the column elements. Will never return AFXTABLE_TYPE_DEFAULT!

Argument	Type	Default	Description
index	Int		Column index.

getDefaultStyle()

Returns the default style for the table elements.

getDefaultType()

Returns the default type for the table elements.

getDefaultValues()

Returns the default values for this table.

getFormattedValue(row, column)

Returns the formatted value of the table element, suitable for placing in a command. If the element has AFXTABLE_EVALUATE style, and its contents are invalid, an exception will be thrown.

Argument	Type	Default	Description
row	Int		Row index.
column	Int		Column index.

getMaxNumColumns()

Returns the maximum number of columns, or -1 for unbounded.

getMinNumColumns()

Returns the minimum number of columns.

getNumColumns(row)

Returns the number of columns in the row.

Argument	Type	Default	Description
row	Int		Row index.

getNumRows()

Returns the number of rows in the table.

getRow(row)

Returns a string with the contents of a table row.

Argument	Type	Default	Description
row	Int		Row index.

getTypeName()

Returns the name of the table keyword type.

Implements AFXKeyword.

Reimplemented in AFXTableKeyword.

getValue(row, column)

Returns the value of a table element.

Argument	Type	Default	Description
row	Int		Row index.
column	Int		Column index.

getValueAsDouble()

Returns the keyword's value as a float; returns False upon failure.

getValueAsInt()

Returns the keyword's value as an integer; returns False upon failure.

getValueAsString()

Returns the formatted string that represents the current keyword value in a command.

Implements AFXKeyword.

getValueForBlank(column)

Returns the element value substituted for blank for the column.

Argument	Type	Default	Description
column	Int		Column index.

getValues()

Returns a string containing values of the tuple elements. as entered by the user.

getValuesForBlanks()

Returns a string with values substituted for blanks for all table columns.

insertColumns(index, numColumns)

Inserts columns starting at the given index.

Argument	Type	Default	Description
index	Int		Starting index.
numColumns	Int		Number of columns to insert.

insertRows(index, numRows)

Inserts rows starting at the given index.

Argument	Type	Default	Description

index	Int		Starting index.
numRows	Int		Number of rows to insert.

isValueChanged()

Returns True if the keyword value differs from its previous value.

Implements AFXKeyword.

removeColumns(index, numColumns)

Removes columns starting at the given index.

Argument	Type	Default	Description
index	Int		Starting index.
numColumns	Int		Number of columns to remove.

removeRows(index, numRows)

Removes rows starting at the given index.

Argument	Type	Default	Description
index	Int		Starting index.
numRows	Int		Number of rows to remove.

setColumnStyle(index, style)

Sets the style of the column elements.

Argument	Type	Default	Description
index	Int		Column index.
style	Int		New column style.

setColumnType(index, type)

Sets the type of the column elements.

Argument	Type	Default	Description
index	Int		Column index.
type	Int		New column type.

setDefaultStyle(style)

Sets the default style for the table elements.

Argument	Type	Default	Description
style	Int		New default style.

setDefaultType(type)

Sets the default type for table elements.

Argument	Type	Default	Description
type	Int		New default type.

setDefaultValues(values)

Sets the default values for this table.

Argument	Type	Default	Description
values	String		Sequence string with default values.

setMaxNumColumns(length)

Sets the maximum number of columns.

Argument	Type	Default	Description
length	Int		New maximum number of columns, or -1 for unbounded.

setMinNumColumns(length)

Sets the minimum number of columns.

Argument	Type	Default	Description
length	Int		New minimum length.

setNumColumnsRange(minLength, maxLength)

Sets the allowable range for the number of columns.

Argument	Type	Default	Description
minLength	Int		New minimum number of columns.
maxLength	Int		New maximum number of columns, or -1 for unbounded.

setRow(row, seq)

Sets the contents of a table row.

Argument	Type	Default	Description
row	Int		Row index.
seq	String		Sequence with elements.

setValue(row, column, value)

Sets the value of a table element.

Argument	Type	Default	Description
row	Int		Row index.
column	Int		Column index.
value	String		New value.

setValueForBlank(column, value)

Sets the element value substituted for blank for the column.

Argument	Type	Default	Description
column	Int		Column index.
value	String		New value.

setValues(values)

Sets all values for the table elements.

Argument	Type	Default	Description
values	String		Table string with new values.

setValuesForBlanks(values)

Sets the values substituted for blanks for all table columns.

Argument	Type	Default	Description
values	String		String containing comma-separated values.

setValueToDefault(ignoreUnspecified=False)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Should ignore if default is an unspecified value.

setValueToPrevious()

Sets the keyword value to its previous value.

Implements AFXKeyword.

syncPreviousValue()

Sets the keyword's previous value to its current value.

Implements AFXKeyword.

Class flags

Message ID's.

ID_TABLE	**ID for AFXTable widgets.**
ID_VALUE	**ID for widgets exchanging array strings.**
ID_PRINTSNIPPET	**For debugging.**

Global flags

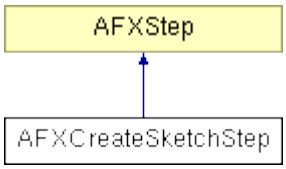
Flags for table options.

AFXTABLE_TYPE_ANY	**Any type is accepted.**
AFXTABLE_TYPE_DEFAULT	**Column type is the same as the table default type.**
AFXTABLE_TYPE_INT	**Column stores integer numbers.**
AFXTABLE_TYPE_FLOAT	**Column stores floating-point numbers.**

AFXTABLE_TYPE_STRING	**Column stores string values.**
AFXTABLE_TYPE_BOOL	**Column stores True or False.**
AFXTABLE_TYPE_MASK	**Mask for column types.**
AFXTABLE_ALLOW_EMPTY	**Allow empty values for the column elements.**
AFXTABLE_DEFAULT_IF_EMPTY	**Always substitute the default for empty values.**
AFXTABLE_EVALUATE	**Evaluate integer and float elements.**
AFXTABLE_STYLE_DEFAULT	**Use table default column style.**
AFXTABLE_STYLE_MASK	**Mask for column styles.**

AFXCreateSketchStep

This class is used to provide pick steps in GUI procedures.



AFXCreateSketchStep(owner, keyword, sheetSize, prompt='Create a sketch')

Constructor.

Argument	Type	Default	Description
owner	AFXProcedure		Procedure creating the step.
keyword	AFXObjectKeyword		Object kwd containing pick variable. Part of AFXGuiCommand.
sheetSize	float		Sketch sheet size when creating.
prompt	String	'Create a sketch'	Step's prompt displayed in prompt area.

onCancel()

Called when the step is cancelled.

Reimplemented from AFXStep.

onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.

Reimplemented from AFXStep.

onResume()

Called when the step is resumed.

Reimplemented from AFXStep.

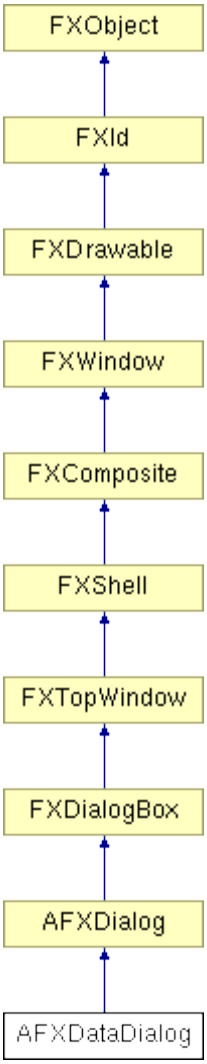
onSuspend()

Called when the step is suspended.

Reimplemented from AFXStep.

AFXDataDialog

This class is the base class for all data dialogs, which collect data from the user and typically collaborate with modes to process the data.



AFXDataDialog(mode, title, actionButtonIds=0, opts=DIALOG_NORMAL, x=0, y=0, w=0, h=0)

Constructor that creates a dialog box that occludes the main window.

Argument	Type	Default	Description
mode	AFXGuiMode		Host mode.
title	String		Title string.
actionButtonIds	Int	0	ID's of action buttons to be created.
opts	Int	DIALOG_NORMAL	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.

AFXDataDialog(mode, owner, title, actionButtonIds=0, opts=DIALOG_NORMAL, x=0, y=0, w=0, h=0)

Constructor that creates a dialog box that occludes its owner widget.

Argument	Type	Default	Description
mode	AFXGuiMode		Host mode.
owner	FXWindow		Owner widget.
title	String		Title string.
actionButtonIds	Int	0	ID's of action buttons to be created.
opts	Int	DIALOG_NORMAL	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.

addTransition(target, op, value, tgt, sel, ptr=None)

Adds a finite state transition to the dialog box. When the expression "target.getValue() op value" evaluates to True, an sel message will be sent to the tgt object.

Argument	Type	Default	Description
target	AFXIntTarget		Target.
op	AFXTransition::Operator		Operator type.
value	Int		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

addTransition(target, op, value, tgt, sel, ptr=None)

Adds a finite state transition to the dialog box. When the expression "target.getValue() op value" evaluates to True, an sel message will be sent to the tgt object.

Argument	Type	Default	Description
target	AFXFloatTarget		Target.
op	AFXTransition::Operator		Operator type.
value	Float		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

addTransition(keyword, op, value, tgt, sel, ptr=None)

Adds a finite state transition to the dialog box. When the expression "keyword.getValue() op value" evaluates to True, an sel message will be sent to the tgt object.

Argument	Type	Default	Description
keyword	AFXTogglableKeyword		Keyword.
op	AFXTransition::Operator		Operator type.

value	Int		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

addTransition(keyword, op, value, tgt, sel, ptr=None)

Adds a finite state transition to the dialog box. When the expression "keyword.getValue() op value" evaluates to True, an sel message will be sent to the tgt object.

Argument	Type	Default	Description
keyword	AFXIntKeyword		Keyword.
op	AFXTransition::Operator		Operator type.
value	Int		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

addTransition(keyword, op, value, tgt, sel, ptr=None)

Adds a finite state transition to the dialog box. When the expression "keyword.getValue() op value" evaluates to True, an sel message will be sent to the tgt object.

Argument	Type	Default	Description
keyword	AFXFloatKeyword		Keyword.
op	AFXTransition::Operator		Operator type.
value	Float		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

addTransition(keyword, op, value, tgt, sel, ptr=None)

Adds a finite state transition to the dialog box. When the expression "keyword.getValue() op value" evaluates to True, an sel message will be sent to the tgt object.

Argument	Type	Default	Description
keyword	AFXBoolKeyword		Keyword.
op	AFXTransition::Operator		Operator type.
value	Bool		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

bailout()

Performs checks to determine whether it is OK to cancel the dialog box. The implementaton of this class always returns True, and the derived class should reimplement this method to perform specific checks.

Reimplemented from AFXDialog.

getMode()

Returns the dialog box's host mode.

onKeywordError(kwd)

Handles the error that occurs when the given keyword or target contains invalid contents. This method will select the contents of the widget that is set for the keyword or target (with setWidgetForKeyword()). If no such widget is specified explicitly, it will select the contents of the widget that has the keyword or target as its message target.

Argument	Type	Default	Description
kwd	FXObject		Object that contains invalid contents.

onTableError(tableKwd, row, col)

Handles the error that occurs when the given table keyword or target contains an invalid element. This method will select the contents of the widget that is set for the element of the keyword or target (with setWidgetForKeyword()). If no such widget is specified explicitly, it will select the contents of the widget that has the keyword or target as its message target.

Argument	Type	Default	Description
tableKwd	FXObject		Object that contains invalid element.
row	Int		Row index.
col	Int		Column index.

onTupleError(tupleKwd, index)

Handles the error that occurs when the given tuple keyword or target contains an invalid element. This method will select the contents of the widget that is set for the element of the keyword or target (with setWidgetForKeyword()). If no such widget is specified explicitly, it will select the contents of the widget that has the keyword or target as its message target.

Argument	Type	Default	Description
tupleKwd	FXObject		Object that contains invalid element.
index	Int		Element index.

processUpdates()

Performs state processing during the GUI update cycles. This class provides an empty implementation of this method, and the derived class should redefine the method if it needs to process state updating.

Class flags

Message ID's.

ID_UPDATE_STATE **Used to update the state.**

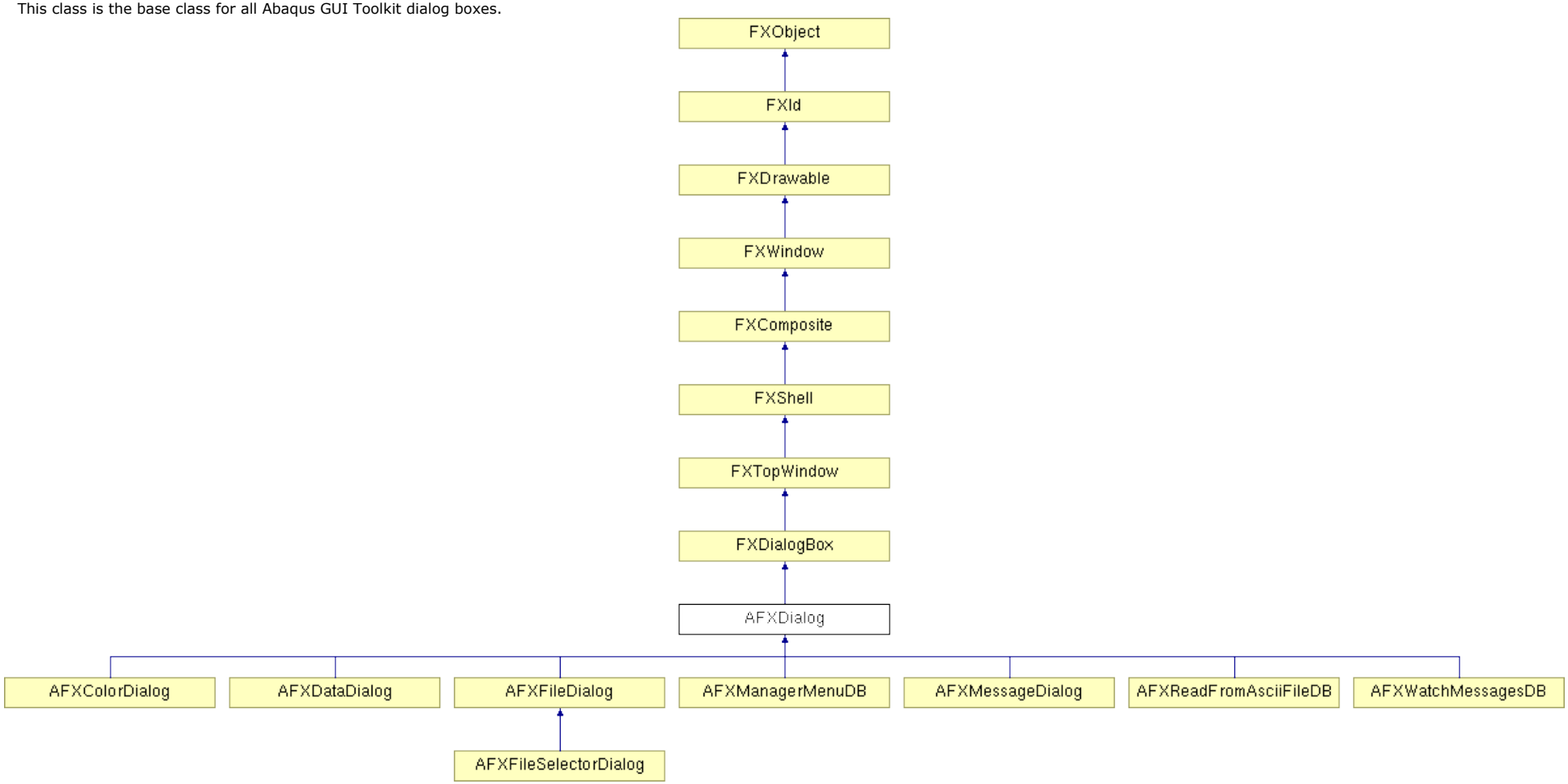
Global flags

Flags for data dialog box options.

DATADIALOG_BAILOUT **Perform bailout checks when the Cancel button is clicked.**

AFXDialog

This class is the base class for all Abaqus GUI Toolkit dialog boxes.



AFXDialog(title, actionButtonIds=0, opts=DIALOG_NORMAL, x=0, y=0, w=0, h=0)

Constructor that creates a dialog box that always occludes the main window when overlapping with the main window.

Argument	Type	Default	Description
title	String		Title string.
actionButtonIds	Int	0	ID's of action buttons to be created.
opts	Int	DIALOG_NORMAL	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.

AFXDialog(owner, title, actionButtonIds=0, opts=DIALOG_NORMAL, x=0, y=0, w=0, h=0)

Constructor that creates a dialog box that always occludes its owner widget when overlapping with the widget.

Argument	Type	Default	Description
owner	FXWindow		Owner widget.
title	String		Title string.
actionButtonIds	Int	0	ID's of action buttons to be created.
opts	Int	DIALOG_NORMAL	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.

AFXDialog(app, title, actionButtonIds=0, opts=DIALOG_NORMAL, x=0, y=0, w=0, h=0)

Constructor that creates a dialog box that may be occluded by any other windows of the application.

Argument	Type	Default	Description
app	FXApp		Application.
title	String		Title string.
actionButtonIds	Int	0	ID's of action buttons to be created.
opts	Int	DIALOG_NORMAL	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.

appendActionButton(text, tgt, sel)

Adds a custom action button in the action area of the dialog box.

Argument	Type	Default	Description
text	String		Label string.
tgt	FXObject		Message target.
sel	Int		Message ID.

appendActionButton(buttonID)

Adds a standard action button in the action area of the dialog box.

Argument	Type	Default	Description
buttonID	ButtonID		Button ID.

bailout()

Performs checks to determine whether it is OK to cancel the dialog box. The implementaton of this class always returns True, and the derived class should reimplement this method to perform specific checks.

Reimplemented in AFXDataDialog.

create()

Creates the dialog box.

Reimplemented from FXTopWindow.

createButton(parent, text, icon, tgt, sel, opts)

Creates an action area button.

Argument	Type	Default	Description
parent	FXComposite		Parent widget.
text	String		Label string.
icon	FXIcon		Icon.
tgt	FXObject		Message target.
sel	Int		Message ID.
opts	Int		Options and hints.

getActionButton(sel)

Returns the action button with the specified message ID; returns 0 if none of the action buttons has the message ID.

Argument	Type	Default	Description
sel	Int		Message ID.

getInitialFocusWidget()

Returns the widget that will receive the initial focus.

getUnpostHints()

Returns the action that the poster should perform on this dialog box when it is unposted. Possible return values are: DIALOG_UNPOST_DELETE - delete the C++ dialog box object together with the associated window. (default) DIALOG_UNPOST_DESTROY - keep the C++ dialog box object, but destroy the associated window to release resources. DIALOG_UNPOST_NOTHING - do nothing.

hide()

Unpost the dialog box.

Reimplemented from FXTopWindow.

Reimplemented in AFXManagerMenuDB, and AFXMessageDialog.

onKeywordError(kwd)

Handles the error that occurs when the given keyword or target contains invalid contents. This method will select the contents of the widget that has the keyword or target as its message target.

Argument	Type	Default	Description
kwd	FXObject		Object that contains invalid contents.

onTableError(tableKwd, row, col)

Handles the error that occurs when the given table keyword or target contains an invalid element. This method will select the contents of the widget that has the keyword or target as its message target.

Argument	Type	Default	Description
tableKwd	FXObject		Object that contains invalid element.
row	Int		Row index.
col	Int		Column index.

onTupleError(tupleKwd, index)

Handles the error that occurs when the given tuple keyword or target contains an invalid element. This method will select the contents of the widget that has the keyword or target as its message target.

Argument	Type	Default	Description
tupleKwd	FXObject		Object that contains invalid element.
index	Int		Element index.

selectContents(widget)

Selects the contents of the given widget.

Argument	Type	Default	Description
widget	FXWindow		Widget to select.

selectTableElement(widget, row, col)

Selects the given (row,col) element in the 2D array of elements displayed by the given widget.

Argument	Type	Default	Description
widget	FXWindow		Widget to select.
row	Int		Row index.
col	Int		Column index.

selectTupleElement(widget, index)

Selects the element at the given index in the sequence of elements displayed by the given widget.

Argument	Type	Default	Description
widget	FXWindow		Widget to select.
index	Int		Element index.

setInitialFocusWidget(w)

Sets the widget that will receive the initial focus.

Argument	Type	Default	Description
w	FXWindow		Widget that will receive the initial focus.

setOnPopdownTarget(target)

Sets the object to be notified when the dialog box is unposted.

Argument	Type	Default	Description
target	FXObject		Object to be notified when the dialog box is unposted.

setUnpostHints(hints)

Sets the action that the poster should perform on this dialog box when it is unposted.

Argument	Type	Default	Description
hints	Int		

show()

Posts the dialog box.

Reimplemented from FXTopWindow.

Reimplemented in AFXFileDialog, and AFXMessageDialog.

showModal(occludedWindow=None)

Posts the dialog box as a modal dialog box. The dialog box is centered against the given widget or its owner widget if 0 is given.

Argument	Type	Default	Description
occludedWindow	FXWindow	None	Widget to be occluded (0 for the owner widget).

Class flags

Message ID's.

ID_CLICKED_OK	**OK button ID.**
ID_CLICKED_CONTINUE	**Contiue button ID.**
ID_CLICKED_YES	**Yes button ID.**
ID_CLICKED_YES_TO_ALL	**Yes to All button ID.**
ID_CLICKED_APPLY	**Apply button ID.**
ID_CLICKED_DEFAULTS	**Defaults button ID.**
ID_CLICKED_NO	**No button ID.**
ID_CLICKED_CANCEL	**Cacncel button ID.**
ID_CLICKED_DISMISS	**Dismiss button ID.**

Standard action button ID's.

APPLY	**Adds an Apply button to action area.**
CANCEL	**Adds a Cancel button to action area.**
CONTINUE	**Adds a Continue button to action area.**
DEFAULTS	**Adds a Defaults button to action area.**
DISMISS	**Adds a Dismiss button to action area.**
NO	**Adds a No button to action area.**
OK	**Adds an OK button to action area.**
YES	**Adds a Yes button to action area.**
YES_TO_ALL	**Adds a Yes to All button to action area.**

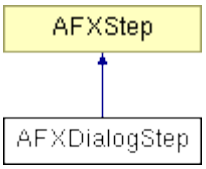
Global flags

Flags for dialog box options.

DIALOG_ACTIONS_BOTTOM	Creates the action area horizontally at the bottom of the dialog box.
DIALOG_ACTIONS_RIGHT	Creates the action area vertically at the right side of the dialog box.
DIALOG_ACTIONS_NONE	Do not create the action area.
DIALOG_ACTIONS_SEPARATOR	Creates a separator in the action area.
DIALOG_UNPOST_DELETE	Deletes the dialog box when unposted.
DIALOG_UNPOST_DESTROY	Destroys the dialog box when unposted.
DIALOG_UNPOST_NOTHING	Do nothing when unposted.
DIALOG_NORMAL	Default dialog box options.

AFXDialogStep

This class provides dialog steps in GUI procedures.



AFXDialogStep(owner, dialog, prompt)

Constructor that takes a prompt for the prompt area.

Argument	Type	Default	Description
owner	AFXProcedure		Procedure creating the step.
dialog	AFXDataDialog		Dialog box to be posted in this step.
prompt	String		

AFXDialogStep(owner, dialog)

Constructor that supplies a default prompt for the prompt area.

Argument	Type	Default	Description
owner	AFXProcedure		Procedure creating the step.
dialog	AFXDataDialog		Dialog box to be posted in this step.

onCancel()

Called when the step is cancelled.

Reimplemented from AFXStep.

onExecute()

Called to execute steps returned by getFirstStep and getNextStep.

Reimplemented from AFXStep.

onResume()

Called when the step is resumed.

Reimplemented from AFXStep.

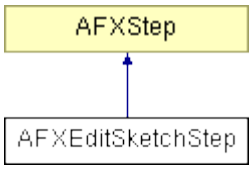
onSuspend()

Called when the step is suspended.

Reimplemented from AFXStep.

AFXEditSketchStep

This class is used to provide pick steps in GUI procedures.



AFXEditSketchStep(owner, sketchName, prompt='Edit a sketch')

Constructor.

Argument	Type	Default	Description
owner	AFXProcedure		Procedure creating the step.
sketchName	String		Name of sketch to edit, blank if create.
prompt	String	'Edit a sketch'	Step's prompt displayed in prompt area.

onCancel()

Called when the step is cancelled.

Reimplemented from AFXStep.

onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.

Reimplemented from AFXStep.

onResume()

Called when the step is resumed.

Reimplemented from AFXStep.

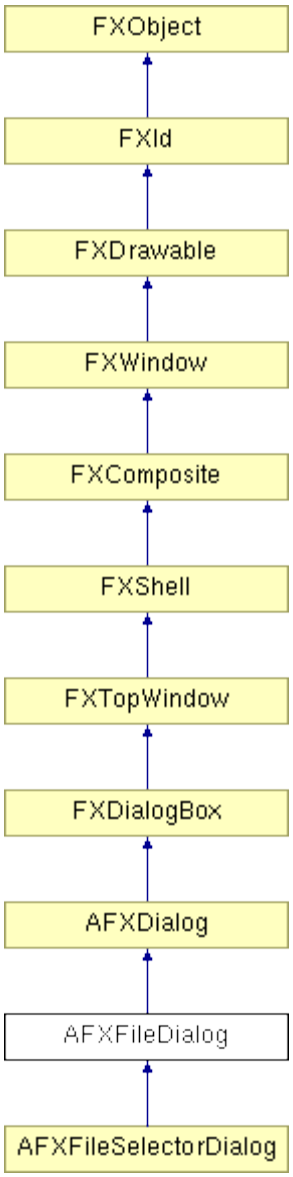
onSuspend()

Called when the step is suspended.

Reimplemented from AFXStep.

AFXFileDialog

This class contains a file selection dialog box.



AFXFileDialog(owner, title, pathNameKw, readOnlyKw, tgt=None, sel=0, mode=AFXSELECTFILE_ANY, patterns=*, patternIndexTgt=None)

Constructor that creates a dialog box that always occludes its owner widget when overlapping with the widget. The constructor expects a string keyword for storing the selected file name. If the dialog box allows multiple selection, the string keyword contains comma-separated path names of all selected files.

Argument	Type	Default	Description
owner	FXWindow		Parent widget.
title	String		Dialog title.
pathNameKw	AFXStringKeyword		Path name keyword.
readOnlyKw	AFXBoolKeyword		Read-only keyword.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.

mode	Int	AFXSELECTFILE_ANY	File selection mode.
patterns	String	*	File filter patterns.
patternIndexTgt	AFXIntTarget	None	Index used to select a file filter pattern when the dialog box is posted.

AFXFileDialog(title, pathNameKw, readOnlyKw, tgt=None, sel=0, mode=AFXSELECTFILE_ANY, patterns=*, patternIndexTgt=None)

Constructor that creates a dialog box that always occludes the main window when overlapping with the main window. The constructor expects a string keyword for storing the selected file name. If the dialog box allows multiple selection, the string keyword contains comma-separated path names of all selected files.

Argument	Type	Default	Description
title	String		Dialog title.
pathNameKw	AFXStringKeyword		Path name keyword.
readOnlyKw	AFXBoolKeyword		Read-only keyword.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
mode	Int	AFXSELECTFILE_ANY	File selection mode.
patterns	String	*	File filter patterns.
patternIndexTgt	AFXIntTarget	None	Index used to select a file filter pattern when the dialog box is posted.

AFXFileDialog(owner, title, pathNameTgt, readOnlyKw, tgt=None, sel=0, mode=AFXSELECTFILE_ANY, patterns=*, patternIndexTgt=None)

Constructor that creates a dialog box that always occludes its owner widget when overlapping with the widget. The constructor expects a string target for storing the selected file name. If the dialog box allows multiple selection, the string target contains comma-separated path names of all selected files.

Argument	Type	Default	Description
owner	FXWindow		Parent widget.
title	String		Dialog title.
pathNameTgt	AFXStringTarget		Path name target.
readOnlyKw	AFXBoolKeyword		Read-only keyword.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
mode	Int	AFXSELECTFILE_ANY	File selection mode.
patterns	String	*	File filter patterns.
patternIndexTgt	AFXIntTarget	None	Index used to select a file filter pattern when the dialog box is posted.

AFXFileDialog(title, pathNameTgt, readOnlyKw, tgt=None, sel=0, mode=AFXSELECTFILE_ANY, patterns=*, patternIndexTgt=None)

Constructor that creates a dialog box that always occludes the main window when overlapping with the main window. The constructor expects a string target for storing the selected file name. If the dialog box allows multiple selection, the string target contains comma-separated path names of all selected files.

Argument	Type	Default	Description
title	String		Dialog title.
pathNameTgt	AFXStringTarget		Path name target.
readOnlyKw	AFXBoolKeyword		Read-only keyword.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
mode	Int	AFXSELECTFILE_ANY	File selection mode.
patterns	String	*	File filter patterns.
patternIndexTgt	AFXIntTarget	None	Index used to select a file filter pattern when the dialog box is posted.

getCurrentPattern()

Returns the current pattern number.

getDirectory()

Returns the current directory.

getFileBoxStyle()

Return file list style.

getFilename()

Returns the file name.

getFilenames()

Returns an empty-string terminated list of selected file names, or 0 if none is selected.

getItemSpace()

Returns the inter-item spacing (in pixels).

getPattern()

Returns the file pattern.

getPatternList()

Returns the list of patterns.

getPatternText(patno)

Returns the pattern text for a given pattern number.

Argument	Type	Default	Description
patno	Int		

getPressedButtonId()

Returns the ID of the button that the user pressed in the dialog box.

getReadOnly()

Returns the read-only state.

getReadOnlyPatterns()

Returns the patterns that force the enabling of the read-only button.

getSelectMode()

Returns the file selection mode.

setCurrentPattern(n)

Sets the current active pattern.

Argument	Type	Default	Description
n	Int		

setDirectory(path)

Sets the current directory.

Argument	Type	Default	Description
path	String		

setFileBoxStyle(style)

Sets the file list style.

Argument	Type	Default	Description
style	Int		

setFilename(path)

Sets the file name.

Argument	Type	Default	Description
path	String		

setItemSpace(s)

Sets the inter-item spacing (in pixels).

Argument	Type	Default	Description
s	Int		

setPattern(ptrn)

Sets the file pattern.

Argument	Type	Default	Description
ptrn	String		

setPatternList(patterns)

Sets the list of file patterns.

Argument	Type	Default	Description
patterns	String		

setPatternListMaxVisible(maxVisible)

Sets the maximum number of visible items for the file pattern list.

Argument	Type	Default	Description
maxVisible	Int		

setPatternText(patno, text)

Sets the pattern text for a pattern number.

Argument	Type	Default	Description
patno	Int		
text	String		

setReadOnly(state)

Sets the initial state of read-only button.

Argument	Type	Default	Description
state	Bool		

setReadOnlyPatterns(patterns)

Sets the patterns that force the display of the read-only button; separate the entries by a newline character
.

Argument	Type	Default	Description
patterns	String		

setSelectMode(mode)

Sets the file selection mode.

Argument	Type	Default	Description
mode	Int		

show()

Posts the dialog box.

Reimplemented from AFXDialog.

showModal(occludedWindow=None)

Posts the dialog box as a modal dialog box. The dialog box is centered against the given widget or its owner widget if 0 is given.

Argument	Type	Default	Description
occludedWindow	FXWindow	None	Widget to be occluded (0 for the owner widget).

shownReadOnly()

Returns True if the read-only button is shown.

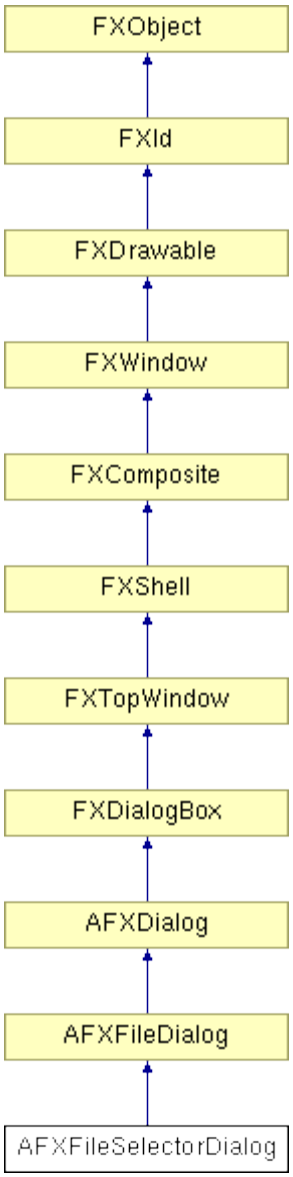
showReadOnly(show)

Shows the read-only button.

Argument	Type	Default	Description
show	Bool		

AFXFileSelectorDialog

This class extends the FXFileDialog class and is designed to work with the mode infrastructure.



AFXFileSelectorDialog(form, title, pathNameKw, readOnlyKw, mode=AFXSELECTFILE_ANY, patterns=*, patternIndexTgt=None)

Constructor typically used to create a dialog box that is posted by a mode (e.g. by getFirstDialog); a keyword is used for the pathName. If the dialog box allows multiple selection, the pathName keyword contains comma-separated path names of all selected files.

Argument	Type	Default	Description
form	AFXForm		Form.
title	String		Dialog box title.
pathNameKw	AFXStringKeyword		Path name keyword.
readOnlyKw	AFXBoolKeyword		Read-only keyword.
mode	Int	AFXSELECTFILE_ANY	File selection mode.
patterns	String	*	File filter patterns.

patternIndexTgt	AFXIntTarget	None	Index used to select a file filter pattern when the dialog box is posted.

AFXFileSelectorDialog(owner, title, pathNameKw, readOnlyKw, mode=AFXSELECTFILE_ANY, patterns=*, patternIndexTgt=None)

Constructor typically used to create a dialog box that is posted from another dialog box (e.g. from a "Select..." button); a keyword is used for the pathName. If the dialog box allows multiple selection, the pathName keyword contains comma-separated path names of all selected files.

Argument	Type	Default	Description
owner	FXWindow		Owner
title	String		Dialog box title.
pathNameKw	AFXStringKeyword		Path name keyword.
readOnlyKw	AFXBoolKeyword		Read-only keyword.
mode	Int	AFXSELECTFILE_ANY	File selection mode.
patterns	String	*	File filter patterns.
patternIndexTgt	AFXIntTarget	None	Index used to select a file filter pattern when the dialog box is posted.

AFXFileSelectorDialog(form, title, pathNameTgt, readOnlyKw, mode=AFXSELECTFILE_ANY, patterns=*, patternIndexTgt=None)

Constructor typically used to create a dialog box that is posted by a mode (e.g. by getFirstDialog); a target is used for the pathName. If the dialog box allows multiple selection, the pathName target contains comma-separated path names of all selected files.

Argument	Type	Default	Description
form	AFXForm		Form.
title	String		Dialog box title.
pathNameTgt	AFXStringTarget		Path name target.
readOnlyKw	AFXBoolKeyword		Read-only keyword.
mode	Int	AFXSELECTFILE_ANY	File selection mode.
patterns	String	*	File filter patterns.
patternIndexTgt	AFXIntTarget	None	Index used to select a file filter pattern when the dialog box is posted.

AFXFileSelectorDialog(owner, title, pathNameTgt, readOnlyKw, mode=AFXSELECTFILE_ANY, patterns=*, patternIndexTgt=None)

Constructor typically used to create a dialog box that is posted from another dialog box (e.g. from a "Select..." button); a target is used for the pathName. If the dialog box allows multiple selection, the pathName target contains comma-separated path names of all selected files.

Argument	Type	Default	Description
owner	FXWindow		Owner
title	String		Dialog box title.
pathNameTgt	AFXStringTarget		Path name target.
readOnlyKw	AFXBoolKeyword		Read-only keyword.

mode	Int	AFXSELECTFILE_ANY	File selection mode.
patterns	String	*	File filter patterns.
patternIndexTgt	AFXIntTarget	None	Index used to select a file filter pattern when the dialog box is posted.

Global flags

File selection modes

AFXSELECTFILE_ANY **A single file, existing or not (to save to).**

AFXSELECTFILE_EXISTING **An existing file (to load).**

AFXSELECTFILE_MULTIPLE **Multiple existing files.**

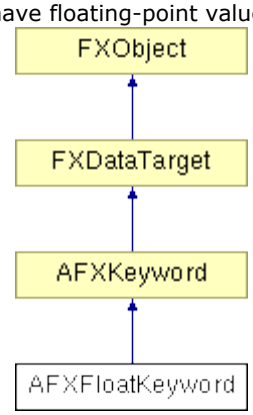
AFXSELECTFILE_MULTIPLE_ALL **Multiple existing files or directories.**

AFXSELECTFILE_DIRECTORY **Existing directory.**

AFXSELECTFILE_REMOTE_HOST **Enable opening files on a remote host.**

AFXFloatKeyword

This class is designed for the command keywords that have floating-point values.



AFXFloatKeyword(command, name, isRequired=False, defaultValue=FLOAT_DEFAULT, precision=6)

Constructor.

Argument	Type	Default	Description
command	AFXCommand		Host command.
name	String		Keyword name.
isRequired	Bool	False	True if the keyword is a required argument of the command.
defaultValue	Float	FLOAT_DEFAULT	Default value.
precision	Int	6	Precision for converting the keyword's floating-point value to a text string.

getPrecision()

Returns the precision that is used for converting the keyword's floating-point value to a text string.

getTypeName()

Returns the name of the keyword type.

Implements AFXKeyword.

getValue()

Returns the keyword's current value, or zero if the content expression is invalid.

getValueAsString()

Returns the text string that represents the keyword's current value.

Implements AFXKeyword.

isValueChanged()

Returns True if the keyword value differs from its previous value.

Implements AFXKeyword.

setDefaultValue(defaultValue)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	String		Default value.

setDefaultValue(defaultValue)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	Float		Default value.

setPrecision(precision)

Sets the precision that is used for converting the keyword's floating-point value to a text string.

Argument	Type	Default	Description
precision	Int		

setValue(newValue)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	String		New value.

setValue(newValue)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	Float		New value.

setValueToDefault(ignoreUnspecified=False)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Ignore setting the value if the default is unspecified.

setValueToPrevious()

Sets the keyword value to its previous value.

Implements AFXKeyword.

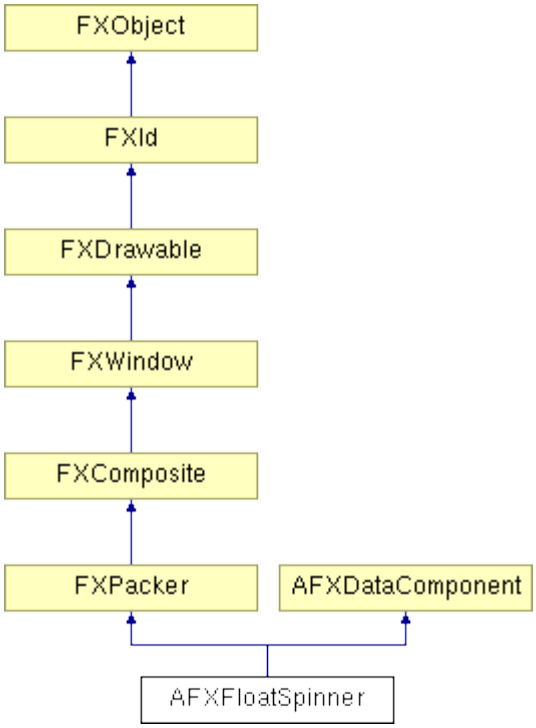
syncPreviousValue()

Sets the keyword's previous value to its current value.

Implements AFXKeyword.

AFXFloatSpinner

Convenience class for creating a labeled spinner. The label field can be a label or check button (AFXFLOATSPINNER_CHECKBUTTON option).



AFXFloatSpinner(p, ncols, labelText, tgt=None, sel=0, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
ncols	Int		Number of columns in the spinner.
labelText	String		Label preceeding spinner.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	0	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	DEFAULT_PAD	Left padding.
pr	Int	DEFAULT_PAD	Right padding.
pt	Int	DEFAULT_PAD	Top padding.
pb	Int	DEFAULT_PAD	Bottom padding.

canFocus()

Returns True if the spinner can recieve focus.

Reimplemented from FXWindow.

create()

Creates the spinner.

Reimplemented from FXComposite.

disable()

Disables the spinner.

Reimplemented from FXWindow.

enable()

Enables the spinner.

Reimplemented from FXWindow.

getCheck()

Returns the state of the check button or the radio button.

getHelpText()

Returns the status line help text.

getIncrement()

Returns the spinner increment.

getLabelFont()

Returns the label font.

getLabelText()

Returns the label string.

getRange()

Returns a sequence of floats (low, high) representing the widget's allowable minimum and maximum values.

getTipText()

Returns the tool tip message.

getValue()

Returns the spinner value.

isCheckStateChanged()

Returns True if the state of the check button or the radio button has changed by the user since it was programmatically set last time.

isEditable()

Returns True if the input field of spinner may be edited.

isReadOnlyState()

Returns True if the spinner is set to the read-only state.

setCheck(state)

Sets the state of the check button or the radio button.

Argument	Type	Default	Description
state	Bool		Button state.

setCheckButtonSelector(sel)

Sets the message ID of the check button or the radio button.

Argument	Type	Default	Description
sel	Int		Selector.

setCheckButtonTarget(tgt)

Sets the message target of the check button or the radio button.

Argument	Type	Default	Description
tgt	FXObject		Target.

setEditable(edit=True)

Sets the editable state for the input field of spinner.

Argument	Type	Default	Description
edit	Bool	True	Editable state.

setHelpText(text)

Sets the status line help text.

Argument	Type	Default	Description
text	String		Help text.

setIncrement(incr)

Sets the spinner increment.

Argument	Type	Default	Description
incr	Float		Increment.

setLabelFont(fnt)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		Label font.

setLabelText(txt)

Sets the label string.

Argument	Type	Default	Description
txt	String		Label text.

setRange(low, high)

Sets the spinner range.

Argument	Type	Default	Description
low	Float		Minimum value.
high	Float		Maximum value.

setReadOnlyState(edit=True)

Sets the read-only state of the spinner.

Argument	Type	Default	Description
edit	Bool	True	Read-only state.

setTipText(text)

Sets the tool tip message.

Argument	Type	Default	Description
text	String		Tooltip text.

setValue(val, notify=False)

Sets the spinner value.

Argument	Type	Default	Description
val	Float		Value.
notify	Bool	False	Notification flag.

Class flags

ID_BUTTON **ID for the check or radio button.**

ID_SPINNER **ID for the spinner.**

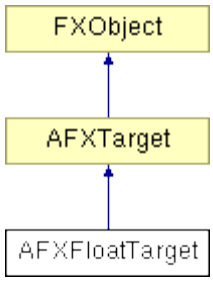
Global flags

Flags for AFX float spinner options.

AFXFLOATSPINNER_CHECKBUTTON	**Use a check button instead of a label.**
AFXFLOATSPINNER_RADIOBUTTON	**Use a radio button instead of a label.**
AFXFLOATSPINNER_VERTICAL	**Orient label or button above spinner.**
AFXFLOATSPINNER_READONLY	**Configure spinner to the read-only state.**

AFXFloatTarget

This class is designed for floating-point targets.



AFXFloatTarget(initialValue=0.0)

Constructor.

Argument	Type	Default	Description
initialValue	Float	0.0	Initial value.

getTypeName()

Returns the name of the target type ("Float").

Implements AFXTarget.

getValue()

Returns the target's current value.

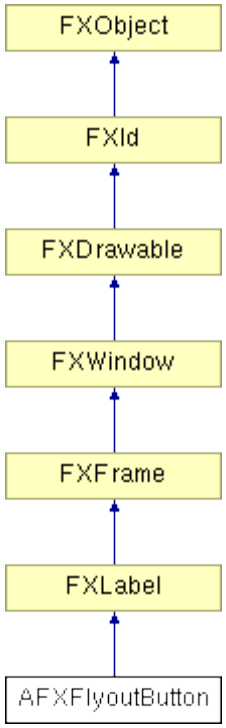
setValue(newValue)

Sets the target's current value.

Argument	Type	Default	Description
newValue	Float		New value.

AFXFlyoutButton

This class contains a button that acts like a regular FXButton when pressed and released quickly but displays a popup menu when pressed and held for a short time duration.



AFXFlyoutButton(p, pup=None, act=0, opts=AFXFLYOUT_NORMAL, x=0, y=0, w=0, h=0, pl=0, pr=0, pt=0, pb=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
pup	FXPopup	None	Popup containing flyout items.
act	Int	0	Current button index (0-based).
opts	Int	AFXFLYOUT_NORMAL	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	0	Left padding (margin).
pr	Int	0	Right padding (margin).
pt	Int	0	Top padding (margin).
pb	Int	0	Bottom padding (margin).

canFocus()

Returns True (because a flyout button can receive focus).

Reimplemented from FXWindow.

create()

Creates the flyout button.

Reimplemented from FXLabel.

detach()

Detaches server-side resources for the flyout button.

Reimplemented from FXLabel.

disable()

Disables the flyout button.

Reimplemented from FXLabel.

enable()

Enables the flyout button.

Reimplemented from FXLabel.

getButtonStyle()

Returns the flyout button style.

getCurrentItem()

Returns the current item.

getPane()

Returns the popup menu.

getState()

Returns the flyout button state.

setButtonStyle(style)

Sets the flyout button style.

Argument	Type	Default	Description
style	Int		Button style (see Flags for flyout button options.)

setCurrentItem(item)

Sets the current item.

Argument	Type	Default	Description
item	AFXFlyoutItem		Item.

setCurrentItem(index, setCheck=False)

Sets the current item and depresses the button if setCheck is True. The specified item index is 0-based, and only valid items are counted (items such as separators are not counted).

Argument	Type	Default	Description
index	Int		Index.
setCheck	Bool	False	Value of check button.

setPane(pup)

Sets the popup menu.

Argument	Type	Default	Description
pup	FXPopup		Popup menu.

setState(state)

Sets the flyout button state.

Argument	Type	Default	Description
state	Int		State (see FXButton's Button state bits).

Class flags

Message ID's.

ID_AFXFLYOUT_TIMER **ID for the popup timer.**

ID_HIDE_ITEM **ID used when hiding flyout item.**

Global flags

Flags for flyout button options.

AFXFLYOUT_AUTOGRAY **Automatically gray out when no target.**

AFXFLYOUT_AUTOHIDE **Automatically hide when no target.**

AFXFLYOUT_TOOLBAR **Toolbar style button.**

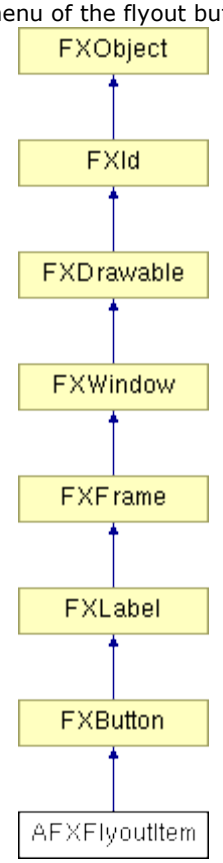
AFXFLYOUT_HORIZONTAL **Popup horizontal.**

AFXFLYOUT_VERTICAL **Popup vertical.**

AFXFLYOUT_RADIO **Current item is always active.**

AFXFlyoutItem

This class contains a button that is placed in the popup menu of the flyout button.



AFXFlyoutItem(p, text, ic=None, tgt=None, sel=0, opts=ICON_ABOVE_TEXT| BUTTON_TOOLBAR| FRAME_RAISED| FRAME_THICK, x=0, y=0, w=0, h=0, pl=0, pr=0, pt=0, pb=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
text	String		Label string.
ic	FXIcon	None	Icon.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	ICON_ABOVE_TEXT| BUTTON_TOOLBAR| FRAME_RAISED| FRAME_THICK	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y c coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	0	Left padding (margin).

pr	Int	0	Right padding (margin).
pt	Int	0	Top padding (margin).
pb	Int	0	Bottom padding (margin).

canFocus()

Returns True (because a flyout item can receive focus).

Reimplemented from FXButton.

hide()

Hides the flyout item.

Reimplemented from FXWindow.

setIcon(ic)

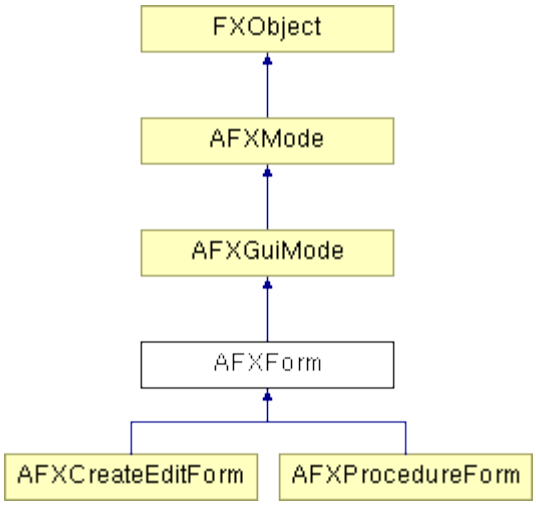
Sets the icon for the flyout item.

Reimplemented from FXLabel.

Argument	Type	Default	Description
ic	FXIcon		

AFXForm

This class is the abstract base class for forms.



AFXForm(owner)

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Owner (a module or a toolset) of the form.

activate()

Performs the necessary tasks when the form is activated.

Reimplemented from AFXGuiMode.

cancel(tgt=None, msg=0)

Requests a cancellation of the form. When the cancel operation completes, successfully or not, the target will be sent the given message. The message data pointer will be non-zero for successful cancellation and zero if the cancel operation was abandoned for some purpose.

Argument	Type	Default	Description
tgt	FXObject	None	Completion message target.
msg	Int	0	Completion message ID.

commit()

Performs the necessary tasks when the form is committed.

Implements AFXGuiMode.

continueMode()

Used to get the next dialog box in the mode.

Implements AFXGuiMode.

deactivateIfNeeded()

Deactivates the form if the user pressed the OK button of the currently posted dialog box. This method is called after the commands are processed successfully.

getFirstDialog()

Returns the first dialog box to be posted.

getNextDialog(previousDialog)

Returns the next dialog box to be posted.

Argument	Type	Default	Description
previousDialog	AFXDialog		Previous dialog box.

issueCommands(writeToReplay=True, writeToJournal=False)

Generates commands based on the current state, sends the commands, and deactivates the form if necessary. If the commands did not complete successfully, a dialog box will be posted with an error message.

Argument	Type	Default	Description
writeToReplay	Bool	True	True if commands should be written to the replay file; False if not.
writeToJournal	Bool	False	True if commands should be written to the journal file; False if not.

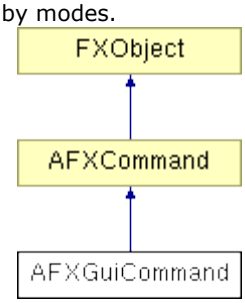
setModal(postModalDialogs)

Sets the modal state; if True, dialogs will be posted as modal. By default the form posts dialogs as non-modal.

Argument	Type	Default	Description
postModalDialogs	Bool		True if the form should post dialogs as modal.

AFXGuiCommand

This class is designed for the GUI commands processed by modes.



AFXGuiCommand(mode, method, objectName='', registerQuery=False)

Constructor.

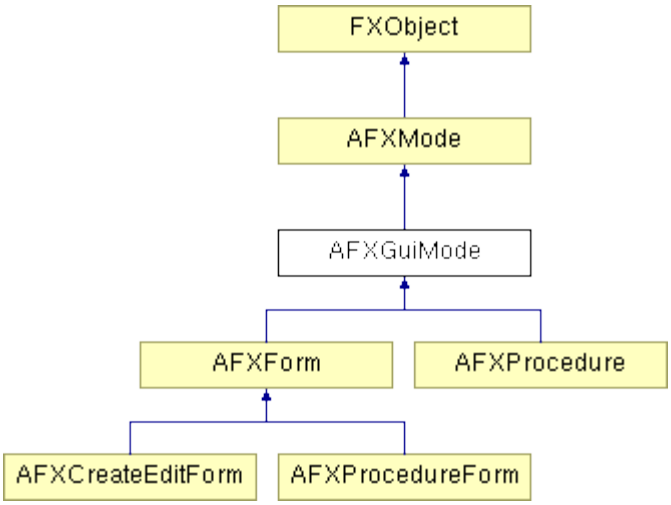
Argument	Type	Default	Description
mode	AFXGuiMode		Host mode.
method	String		Method.
objectName	String	''	Object name.
registerQuery	Bool	False	True if a query should be registered so that the command's keyword values can be updated based on the kernel state.

getMode()

Retrieves the command's mode.

AFXGuiMode

This class is the abstract base class for modes.



AFXGuiMode(owner)

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Owner (a module or a toolset) of the mode.

activate()

Performs the necessary tasks when the mode is activated.

Reimplemented in AFXForm, and AFXProcedure.

cancel(tgt=None, msg=0)

Requests a cancellation of the mode. When the cancel operation completes, successfully or not, the target will be sent the given message. The message data pointer will be non-zero for successful cancellation and zero if the cancel operation was abandoned for some purpose.

Argument	Type	Default	Description
tgt	FXObject	None	Completion message target.
msg	Int	0	Completion message ID.

commit()

Performs the necessary tasks when a dialog box of the mode commits.

Implemented in AFXForm, and AFXProcedure.

continueMode()

Virtual method that must be defined in the derived class--used to get the next dialog box or step in the mode.

Implemented in AFXForm, and AFXProcedure.

deactivate()

Performs the necessary tasks when the mode is deactivated.

Reimplemented in AFXProcedureForm, and AFXProcedure.

destroyDialogs()

Destroys the associated dialogs when the mode is deactivated. The exact behavior of this function is controlled by the return value of dialog box's getDeactivateAction() method.

doCustomChecks()

This class provides an empty implementation of this method; the derived class should redefine this method to perform its specific verifications on the user input.

doCustomTasks()

This class provides an empty implementation of this method, which is called after the commands are processed successfully. The derived class should redefine this method to perform its specific tasks such as writing a confirmation message to the main window.

getCommandString()

Returns a string containing a list of commands generated by each command associated with the mode.

getCurrentDialog()

Returns the dialog box currently posted by the mode (may be NULL).

getModeName()

Returns the mode's name.

getOwner()

Returns the mode's owner (a module or a tool set).

getPressedButtonId()

Returns the ID of the button that the user pressed in the currently posted dialog box.

handleException(exc)

Posts a dialog box with an error message for the given exception. The derived class should redefine this method if it needs to handle the exception differently.

Argument	Type	Default	Description
exc	nex_Exception		Exception.

handleKeywordException(exc)

Posts a dialog box with an error message for the given exception thrown by a keyword during data validation.

Argument	Type	Default	Description
exc	nex_Exception		Exception.

isKeyword(object)

Returns True if the object is one of the mode's keywords.

Argument	Type	Default	Description
object	FXObject		

issueCommands(writeToReplay=True, writeToJournal=False)

Generates commands based on the current state, sends the commands, and deactivates the mode if necessary. If the commands did not complete successfully, a dialog box will be posted with an error message.

Argument	Type	Default	Description
writeToReplay	Bool	True	True if commands should be written to the replay file; False if not.
writeToJournal	Bool	False	True if commands should be written to the journal file; False if not.

okToCancel()

Method indicating if the dialog box should be cancelled if the context changes (for example, if the user opens a new database); the base class implementation returns True indicating it is OK to cancel the mode, derived classes can overwrite this method and return False to prevent the mode from being canceled during a context change.

onKeywordsUpdated(command)

Hook method called whenever the command's object changes in Kernel. By default this function does nothing.

Argument	Type	Default	Description
command	AFXGuiCommand		Command for which the query fired.

registerDefaultsObject(command, objectName)

Registers the defaults object that will be queried when the Defaults button in the dialog box is pressed.

Argument	Type	Default	Description
command	AFXGuiCommand		Command whose keywords to be populated with the defaults object values.
objectName	String		Name of defaults object to be queried.

registerQueries()

Registers queries with the GUI infrastructure for the mode's active commands and unregisters queries for the inactive ones.

removeAllProxies()

Unregisters queries and deletes all the associated proxies.

sendCommandString(command, writeToReplay=True, writeToJournal=False)

Sends the given command string (which can contain multiple commands, separated by command delimiters).

Argument	Type	Default	Description
command	String		Command string.

writeToReplay	Bool	True	True if commands should be written to the replay file; False if not.
writeToJournal	Bool	False	True if commands should be written to the journal file; False if not.

setModeName(name)

Sets the mode's name.

Argument	Type	Default	Description
name	String		

unregisterAllQueries()

Unregisters queries with the GUI infrastructure for all the mode's commands.

verify()

Verifies each active command associated with the mode and throws an exception if any command contains keywords with invalid data.

verifyCurrentKeywordValues()

Checks whether keywords of active commands for the current dialog box contain valid data and, if not, posts a dialog box with an error message.

Reimplemented in AFXProcedure.

verifyKeywordValues()

Checks whether keywords of active commands contain valid data and, if not, posts a dialog box with an error message.

Class flags

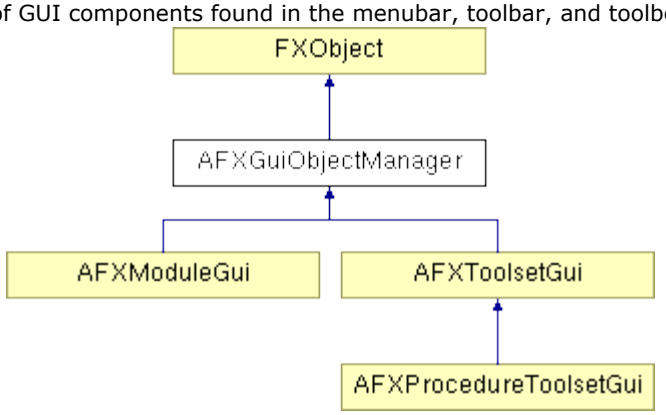
Message ID's.

ID_DESTROY_DIALOGS **Used to destroy dialogs.**

ID_HANDLE_BAILOUT **Used to handle bailout.**

AFXGuiObjectManager

This is a base class for management of GUI components found in the menubar, toolbar, and toolbox.



AFXGuiObjectManager()

Constructor.

AFXGuiObjectManager(source)

Undefined copy constructor (this class does not have copy semantics).

Argument	Type	Default	Description
source	AFXGuiObjectManager		Object to be copied.

getDialog(widgetAlias)

Returns the dialog box with the specified widget key.

Argument	Type	Default	Description
widgetAlias	String		Dialog box alias.

getKernelInitializationCommand()

Returns the command that should initialize the corresponding module or toolset in the kernel. Called by the module manager the first time the GUI module is switched into.

getToolbarGroup(name)

Returns the toolbar group specified by the given name argument.

Argument	Type	Default	Description
name	String		A String that specifies the toolbar to get.

hide(location)

Hides the GUI components in the menubar, toolbar, and toolbox.

Argument	Type	Default	Description
location	Int		Location where GUI components are placed.

registerAndShowDialog(dialog)

Registers the given dialog box and its widget key with the manager and posts the dialog box.

Argument	Type	Default	Description
dialog	AFXDialog		Dialog box.

registerAndShowModalDialog(dialog)

Registers the given dialog box and its widget key with the manager and posts the dialog box as a modal dialog box.

Argument	Type	Default	Description
dialog	AFXDialog		Dialog box.

registerShortcutFunction(text, tgt, sel, ic=None, tipText='', displayedName='', typesToDisplay=0)

Registers a shortcut function; this function will be available in the GUI so that users can assign it shortcut keys.

Argument	Type	Default	Description
text	String		Label used to identify the function in the GUI; To specify a shortcut, separate it from the label using "\t".
tgt	FXObject		Message target.
sel	Int		Messaged ID.
ic	FXIcon	None	Icon for the shortcut
tipText	String	''	Text used for buttom tooltip
displayedName	String	''	Name of the module to which this function belongs.
typesToDisplay	Int	0	Flags specifying the types of objects displayed in the module; see AFXModuleGui.

sendCommandString(command, writeToReplay=True, writeToJournal=False)

Sends the given command string (which can contain multiple commands, separated by command delimiters).

Argument	Type	Default	Description
command	String		Command string.
writeToReplay	Bool	True	True if commands should be written to the replay file; False if not.
writeToJournal	Bool	False	True if commands should be written to the journal file; False if not.

show(location)

Shows the GUI components in the menubar, toolbar, and toolbox.

Argument	Type	Default	Description
location	Int		Location where GUI components are placed.

unregisterDialog(widgetAlias)

Unregisters the dialog box of the given widget key from the manager.

Argument	Type	Default	Description

widgetAlias	String		Dialog box alias.

Class flags

Message ID's.

ID_DESTROY_DIALOGS **Used to destroy dialogs.**

Global flags

Flags for GUI locations.

GUI_IN_NONE **GUI has no components in standard locations.**

GUI_IN_MENUBAR **GUI has components in the menubar.**

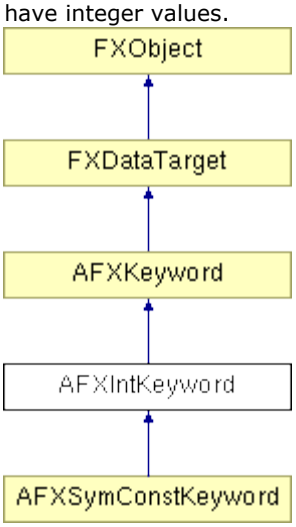
GUI_IN_TOOL_PANE **GUI has components in the Tools pull down pane.**

GUI_IN_TOOLBAR **GUI has components in the toolbar.**

GUI_IN_TOOLBOX **GUI has components in the toolbox.**

AFXIntKeyword

This class is designed for the command keywords that have integer values.



AFXIntKeyword(command, name, isRequired=False, defaultValue=INT_DEFAULT, evalExpression=True)

Constructor.

Argument	Type	Default	Description
command	AFXCommand		Host command.
name	String		Keyword name.
isRequired	Bool	False	True if the keyword is a required argument of the command.
defaultValue	Int	INT_DEFAULT	Default value.
evalExpression	Bool	True	True if the keyword supports expression evaluation.

getTypeName()

Returns the name of the keyword type.

Implements AFXKeyword.

Reimplemented in AFXSymConstKeyword.

getValue()

Returns the keyword's current value.

getValueAsString()

Returns the text string that represents the keyword's current value.

Implements AFXKeyword.

Reimplemented in AFXSymConstKeyword.

isValueChanged()

Returns True if the keyword value differs from its previous value.

Implements AFXKeyword.

setDefaultValue(defaultValue)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	Int		Default value.

setValue(newValue)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	Int		New value.

setValueToDefault(ignoreUnspecified=False)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Ignore setting the value if the default is unspecified.

setValueToPrevious()

Sets the keyword value to its previous value.

Implements AFXKeyword.

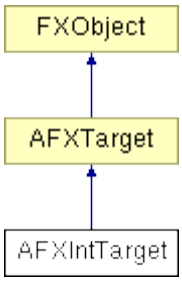
syncPreviousValue()

Sets the keyword's previous value to its current value.

Implements AFXKeyword.

AFXIntTarget

This class is designed for integer targets.



AFXIntTarget(initialValue=0)

Constructor.

Argument	Type	Default	Description
initialValue	Int	0	Initial value.

getTypeName()

Returns the name of the target type ("Int").

Implements AFXTarget.

getValue()

Returns the target's current value.

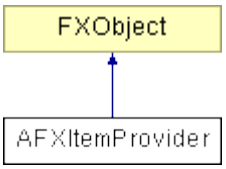
setValue(newValue)

Sets the target's current value.

Argument	Type	Default	Description
newValue	Int		New value.

AFXItemProvider

This class provides a way to supply items to widgets, such as AFXComboBox and AFXList.



AFXItemProvider(initialItems='')

Constructor.

Argument	Type	Default	Description
initialItems	String	''	Sequence string with initial items.

append(str)

Appends a string to the item string.

Argument	Type	Default	Description
str	String		

append(ch)

Appends a character to the item string.

Argument	Type	Default	Description
ch	String		

empty()

Returns True if the item string is empty.

getItems()

Returns a sequence string that contains all of the provider's items.

getVersion()

Returns the version of provider's items.

reset(sz=0)

Clears the contents of the item string and reallocates space.

Argument	Type	Default	Description
sz	Int	0	

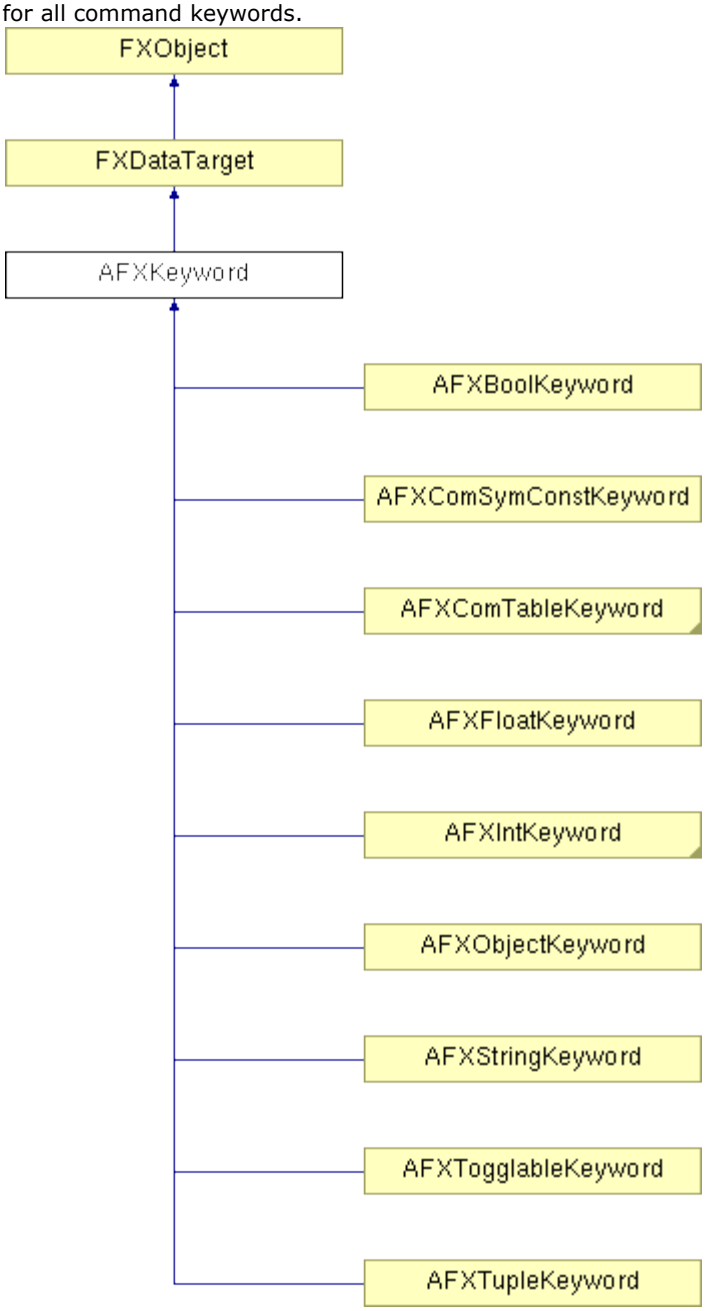
setItems(newItems)

Sets all of the providers's items, clearing any previous items first.

Argument	Type	Default	Description
newItems	String		Sequence string with new items.

AFXKeyword

This class is the abstract base class for all command keywords.



AFXKeyword(command, name, isRequired=False)

Constructor.

Argument	Type	Default	Description
command	AFXCommand		Host command, or NULL to create a keyword not associated with a command.
name	String		Keyword name.
isRequired	Bool	False	True if the keyword is a required argument of the command.

activate()

Activates the keyword; active keywords will be processed during command generation.

deactivate()

Deactivates the keyword; inactive keywords will not be processed during command generation.

getCommandSnippet()

Returns the command snippet of the keyword based on its current value.

getName()

Returns the keyword name.

getSetupCommands()

Returns the keyword's variable initilization commands (part of the generated command string).

getTypeName()

Returns the keyword type name.

Implemented in AFXBoolKeyword, AFXComSymConstKeyword, AFXComTableKeyword, AFXFloatKeyword, AFXIntKeyword, AFXObjectKeyword, AFXStringKeyword, AFXSymConstKeyword, AFXTogglableKeyword, AFXTupleKeyword, and AFXTableKeyword.

getValueAsString()

Returns the text string that represents the current keyword value.

Implemented in AFXBoolKeyword, AFXComSymConstKeyword, AFXComTableKeyword, AFXFloatKeyword, AFXIntKeyword, AFXObjectKeyword, AFXStringKeyword, AFXSymConstKeyword, AFXTogglableKeyword, and AFXTupleKeyword.

isActive()

Returns True if the keyword is active.

isRequired()

Returns True if the keyword is a required argument of the host command; or returns False if the keyword is optional.

isValueChanged()

Returns True if the keyword value differs from its previous value.

Implemented in AFXBoolKeyword, AFXComSymConstKeyword, AFXComTableKeyword, AFXFloatKeyword, AFXIntKeyword, AFXObjectKeyword, AFXStringKeyword, AFXTogglableKeyword, and AFXTupleKeyword.

setRequired(val)

Sets this object as a required keyword of the host command.

Argument	Type	Default	Description
val	Bool		

setSetupCommands(cmds)

Sets variable initialization commands needed by the keyword.

Argument	Type	Default	Description
cmds	String		

setValueToDefault(ignoreUnspecified=False)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Ignore setting the value if the default is unspecified.

setValueToPrevious()

Sets the keyword value to its previous value.

Implemented in AFXBoolKeyword, AFXComSymConstKeyword, AFXComTableKeyword, AFXFloatKeyword, AFXIntKeyword, AFXObjectKeyword, AFXStringKeyword, AFXTogglableKeyword, and AFXTupleKeyword.

syncPreviousValue()

Sets the keyword's previous value to its current value.

Implemented in AFXBoolKeyword, AFXComSymConstKeyword, AFXComTableKeyword, AFXFloatKeyword, AFXIntKeyword, AFXObjectKeyword, AFXStringKeyword, AFXTogglableKeyword, and AFXTupleKeyword.

Class flags

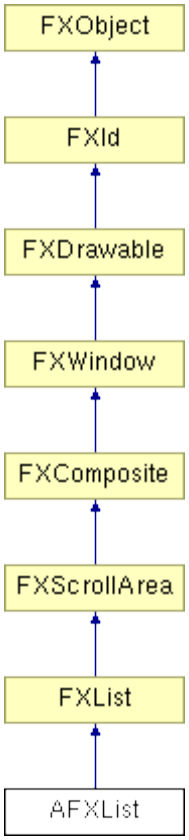
Message ID's.

ID_ACTIVATE **Used to activate the keyword.**

ID_DEACTIVATE **Used to deactivate the keyword.**

AFXList

This class is a list widget that allows displaying items in a scrollable window. Each item has associated data set as integers as items are added.



AFXList(p, nvis, tgt=None, sel=0, opts=0, x=0, y=0, w=0, h=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
nvis	Int		Number of visible items.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	0	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.

appendItem(text, icon=None, sel=0)

Adds a new item to the end of the list.

Argument	Type	Default	Description
text	String		Text

icon	FXIcon	None	Icon
sel	Int	0	Integer associated with this item

disable()

Disables the list.

Reimplemented from FXWindow.

enable()

Enables the list.

Reimplemented from FXWindow.

getAutoCommit()

Returns the auto-commit flag.

getDefaultHeight()

Returns the default height of the list.

Reimplemented from FXList.

getItemIndexForData(data)

Returns the index of the first item with the associated data or -1 if not found.

Argument	Type	Default	Description
data			

getItemProvider()

Returns the provider of the list's items.

getSingleSelection()

Returns the index of the uniquely selected item. If more than one item or no items are selected, returns -1.

insertItem(index, text, icon=None, sel=0)

Inserts a new item at the given index.

Argument	Type	Default	Description
index	Int		Index
text	String		Text
icon	FXIcon	None	Icon
sel	Int	0	Integer associated with this item

replaceItem(index, text, icon=None, sel=0)

Replaces the item at the given index.

Argument	Type	Default	Description
index	Int		Index
text	String		Text
icon	FXIcon	None	Icon
sel	Int	0	Integer associated with this item

setAutoCommit(commit)

Sets the auto-commit option for handling double clicks. This option is turned on by default.

Argument	Type	Default	Description
commit	Bool		

setItemProvider(cp)

Sets the provider of the list's items.

Argument	Type	Default	Description
cp	FXObject		

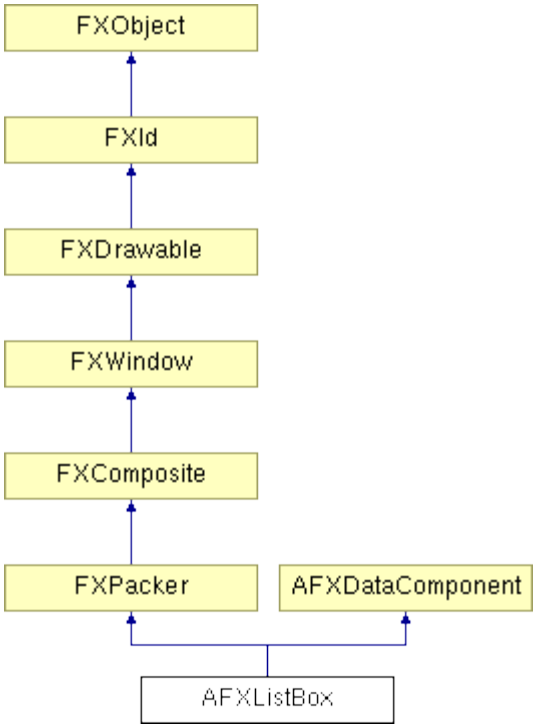
Global flags

Flags for AFX list options.

AFXLIST_NO_AUTOCOMMIT **Don't automatically commit on double click.**

AFXListBox

This class implements a labeled list box. It allows the user to select entries from a drop-down list. Each item has associated data set as integers initially as items are added.



AFXListBox(p, nvis, labelText, tgt=None, sel=0, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
nvis	Int		Number of visible items.
labelText	String		Label string.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	0	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	DEFAULT_PAD	Left padding (margin).
pr	Int	DEFAULT_PAD	Right padding (margin).
pt	Int	DEFAULT_PAD	Top padding (margin).
pb	Int	DEFAULT_PAD	Bottom padding (margin).

appendItem(text, icon=None, sel=0)

Adds an item to the end of the list.

Argument	Type	Default	Description
text	String		Text
icon	FXIcon	None	
sel	Int	0	Icon Integer associated with this item

clearItems()

Removes all items from the list.

create()

Creates the list box.

Reimplemented from FXComposite.

disable()

Disables the list box.

Reimplemented from FXWindow.

enable()

Enables the list box.

Reimplemented from FXWindow.

getCurrentItem()

Returns the index of the current item.

getHelpText()

Returns the status line help text.

getItemData(index)

Returns the data for the specified item.

Argument	Type	Default	Description
index	Int		

getItemIndexForData(data)

Returns the index of the first item with the associated data or -1 if not found.

Argument	Type	Default	Description
data			

getLabelFont()

Returns the label font.

getLabelText()

Returns the label string.

getNumItems()

Returns the number of items in the list.

getNumVisible()

Returns the number of visible items.

getTipText()

Returns the tool tip message.

insertItem(index, text, icon=None, sel=0)

Inserts a new item at the specified index position.

Argument	Type	Default	Description
index	Int		Index
text	String		Text
icon	FXIcon	None	Icon
sel	Int	0	Integer associated with this item

isItemCurrent(index)

Returns True if the item at the specified index position is the current item.

Argument	Type	Default	Description
index	Int		

isReadOnlyState()

Returns True if the list box is set to the read-only state.

removeItem(index)

Removes the item at the specified index position from the list.

Argument	Type	Default	Description
index	Int		

replaceItem(index, text, icon=None, sel=0)

Replaces the item at the specified index position.

Argument	Type	Default	Description
index	Int		Index
text	String		Text

icon	FXIcon	None	Icon
sel	Int	0	Integer associated with this item

setCurrentItem(index, notify=False)

Sets the current item. (The index is zero-based).

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

setHelpText(text)

Sets the status line help text.

Argument	Type	Default	Description
text	String		

setItemData(index, ptr)

Sets the data for the specified item.

Argument	Type	Default	Description
index	Int		
ptr	String		

setLabelFont(fnt)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		

setLabelText(txt)

Sets the label string.

Argument	Type	Default	Description
txt	String		

setNumVisible(nvis)

Sets the number of visible items.

Argument	Type	Default	Description
nvis	Int		

setReadOnlyState(readonly=True)

Sets the read-only state of the list box.

Argument	Type	Default	Description

readonly	Bool	True	

setTipText(text)

Sets the tool tip message.

Argument	Type	Default	Description
text	String		

Class flags

Message ID's.

ID_LIST **ID for the list box.**

ID_FIELD **ID for the text field.**

Global flags

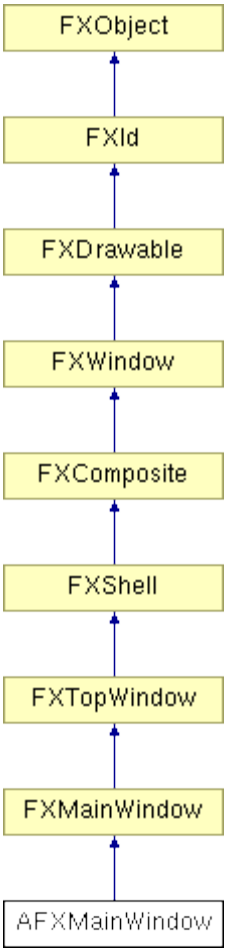
Flags for AFX list box options.

AFXLISTBOX_VERTICAL **Orient label above list box.**

AFXLISTBOX_READONLY **Configure list box to the read-only state.**

AFXMainWindow

This class is responsible for constructing the components of the main window. It also provides accessors for the various components constructed.



AFXMainWindow(app, title, icon=None, miniIcon=None, opts=DECOR_ALL, x=0, y=0, w=0, h=0)

Constructor.

Argument	Type	Default	Description
app	FXApp		FXApp object.
title	String		Main window title.
icon	FXIcon	None	Main window icon.
miniIcon	FXIcon	None	Minimized icon.
opts	Int	DECOR_ALL	Main window options.
x	Int	0	X location of the main window.
y	Int	0	Y location of the main window.
w	Int	0	Width of the main window.
h	Int	0	Height of the main window.

appendApplicableModuleForTreeTab(name, moduleName)

Appends a module name to the list of modules to which a tree tab is applicable.

Argument	Type	Default	Description
name	String		Name of the tab item.
moduleName	String		Module name to be appended to the list of the tab's applicable modules.

appendTreeTab(text, name)

Appends a new tab item to the tree toolset tab book and returns a vertical frame managed by the new tab item; you must call create() on the vertical frame after you construct all its child widgets.

Argument	Type	Default	Description
text	String		Text to be shown in the new tab item.
name	String		Name of the new tab item.

appendVisibleModuleForTreeTab(name, moduleName)

Appends a module to the list of modules in which a tree tab is visible.

Argument	Type	Default	Description
name	String		Name of the tab item.
moduleName	String		Module name to be appended to the list of the tab's modules in which it is visible.

create()

Virtual base class method for creating windowing system resources.

Reimplemented from FXTopWindow.

getContextBar()

Returns a pointer to the context bar container.

getCurrentTreeTab()

Returns the current tab item.

getDefaultHeight()

Returns the default main window height.

Reimplemented from FXTopWindow.

getDefaultWidth()

Returns the default main window width.

Reimplemented from FXTopWindow.

getDisplayedNameAtIndex(index)

Returns the displayed name at the given position in the list.

Argument	Type	Default	Description
index	Int		Position in the module list.

getDrawingAreaHeight()

Returns the height of the drawing area in pixels.

getDrawingAreaWidth()

Returns the width of the drawing area in pixels.

getHelpToolset()

Returns a pointer to the help toolset.

getMenubar()

Returns a pointer to the menubar.

getModule(name)

Returns the module specified by the given name argument.

Argument	Type	Default	Description
name	String		A String that specifies the module to get.

getModuleName(displayedName)

Returns the module name for the given displayed name.

Argument	Type	Default	Description
displayedName	String		Displayed module name (English).

getNumModules()

Returns the number of modules.

getPluginToolset()

Returns the Plugin toolset.

getSelectorFromFunction(function)

Returns the selector of the given shortcut function. Throws exception if not found.

Argument	Type	Default	Description
function	String		A String specifying the function as shown in the Customize dialog box.

getTargetFromFunction(function)

Returns the target of the given shortcut function. Throws exception if not found.

Argument	Type	Default	Description
function	String		A String specifying the function as shown in the Customize

			dialog box.

getToolbox()

Returns a pointer to the module toolbox container.

getToolMenuPane()

Returns a pointer to the tools menu pane.

getToolMenuTitle()

Returns a pointer to the Tools menu title.

getToolset(name)

Returns the toolset specified by the given name argument.

Argument	Type	Default	Description
name	String		A String in the local language that specifies to toolset to get.

getToolsetKernelInitializationCommands()

Returns the command string that should initialize the toolsets in the kernel that are corresponding to the toolsets registered with the main window.

getWorkDirectory()

Returns the current working directory.

hideCli()

Hides the command line interface.

hideMessageArea()

Hides the message area interface.

makeCustomToolsets()

This method has no base class implementation; it may be used by customizers to construct Abaqus/CAE toolsets or toolsets derived from Abaqus/CAE toolsets; constructing those toolsets in this method is necessary to insure that the toolset will be available to standard Abaqus/CAE modules that register that toolset, and to avoid creating duplicate widgets when the toolset is used by a custom toolset.

registerHelpToolset(tool, opts)

Registers the Help toolset.

Argument	Type	Default	Description
tool	AFXToolsetGui		Pointer to toolset being registered.
opts	Int		Options for creating toolset GUI components.

registerModule(displayedName, moduleImportName)

Registers a module to make it available in the module combo; Uses predefined initialization strings for Abaqus modules.

Argument	Type	Default	Description
displayedName	String		The name that will be displayed in the Module combo box in the context bar.
moduleImportName	String		The name that will be used to import this module.

registerModule(displayedName, moduleImportName, kernelInitializationCommand)

Registers a module to make it available in the module combo; also registers the initialization string to be sent to the kernel the first time the module is loaded.

Argument	Type	Default	Description
displayedName	String		The name that will be displayed in the Module combo box in the context bar.
moduleImportName	String		The name that will be used to import this module.
kernelInitializationCommand	String		The Python command sent to kernel when the module is loaded.

registerToolset(tool, opts)

Registers a toolset that is always available in the main window.

Argument	Type	Default	Description
tool	AFXGuiObjectManager		Pointer to the object being registered.
opts	Int		Options for creating toolset GUI components.

setApplicabilityForTreeTab(name, moduleNames)

Sets the modules that are applicable to the given tree tab. When switching modules, if the current tab is applicable to the new module, it will remain current. When a tree tab is created, it is applicable to all modules--use this method to set the applicability to only certain modules.

Argument	Type	Default	Description
name	String		Name of the tab item.
moduleNames	String		A String containing module names separated by commas.

setCurrentTreeTab(name)

Sets the tree toolset tab book's current tab item to the tab item specified by the given name.

Argument	Type	Default	Description
name	String		Name of the tab item to be set current.

setVisibilityForTreeTab(name, moduleNames)

Sets the modules in which a tree tab is visible. When switching modules, if the tab has not been specified to be visible in the new module, the tab will be hidden; otherwise it will be shown. When a tree tab is created it is visible in all modules--use this method to set the visibility to only certain modules.

Argument	Type	Default	Description
name	String		Name of the tab item.

moduleNames	String		A String containing module names separated by commas.

setWorkDirectory(directory)

Sets the current working directory.

Argument	Type	Default	Description
directory	String		A String specifying the new work directory.

showCli()

Shows the command line interface.

showMessageArea()

Shows the message area interface.

writeToMessageArea(message)

Writes a string to the message area.

Argument	Type	Default	Description
message	String		

Class flags

Message ID's.

ID_QUIT **Used to handle the quit message.**

ID_TAB **Used to handle tabbing.**

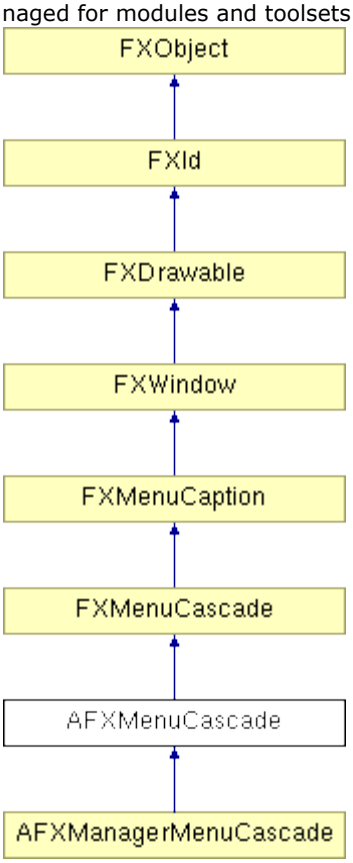
ID_TOGGLE_MODEL_TREE **Toggle the visibility of the model tree.**

ID_TOGGLE_FULL_SCREEN **Toggle full screen.**

ID_LAST **Do not use, do not delete; for use by derived classes.**

AFXMenuCascade

This class provides the interface for creating an FXMenuCascade and performing various management activities on it. It will use utility methods so the menu cascade is correctly managed for modules and toolsets.



AFXMenuCascade(owner, p, label, ic=None, popup=None)

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Creator of the menu cascade.
p	FXComposite		Parent widget.
label	String		Label for the menu button.
ic	FXIcon	None	Menu button icon.
popup	FXPopup	None	Menu cascade's pullright pane.

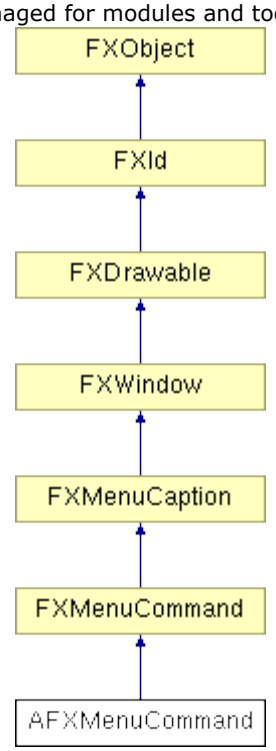
getOwner()

Returns the owner of the menu cascade.

Reimplemented from FXWindow.

AFXMenuCommand

This class provides the interface for creating an FXMenuCommand and performing various management activities on it. It will use utility methods so the menu command is correctly managed for modules and toolsets.



AFXMenuCommand(owner, p, label, ic=None, tgt=None, sel=0)

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Creator of the menu command.
p	FXComposite		Parent widget.
label	String		Label for the menu button.
ic	FXIcon	None	Menu button icon.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.

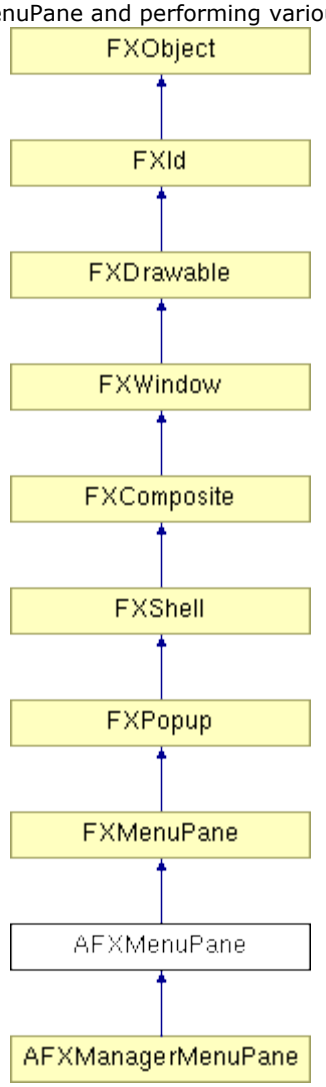
getOwner()

Returns the owner of the menu command.

Reimplemented from FXWindow.

AFXMenuPane

This class provides the interface for creating an FXMenuPane and performing various management activities on it.



AFXMenuPane(owner)

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Manager of the pane.

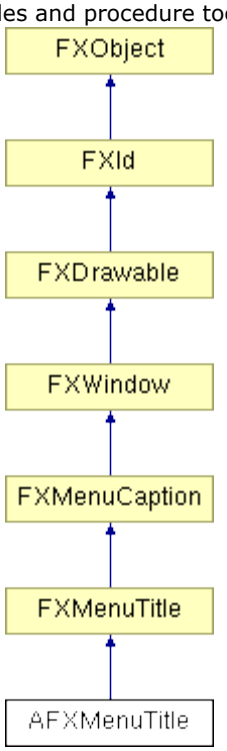
getOwner()

Returns the owner of the menu pane.

Reimplemented from FXWindow.

AFXMenuTitle

This class provides the interface for creating an FXMenuTitle and performing various management activities on it. It will use utility methods so the menu title is correctly managed for modules and procedure toolsets.



AFXMenuTitle(owner, label, ic=None, popup=None)

Constructor that takes an owner.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Owner (module or toolset GUI).
label	String		Label string.
ic	FXIcon	None	Icon.
popup	FXPopup	None	Pulldown menu.

AFXMenuTitle(parent, label, ic=None, popup=None)

Constructor that takes a parent.

Argument	Type	Default	Description
parent	FXComposite		Parent widget.
label	String		Label string.
ic	FXIcon	None	Icon.
popup	FXPopup	None	Pulldown menu.

getOwner()

Returns the owner of the menu title.

Reimplemented from FXWindow.

hide()

Hides the widget.

Reimplemented from FXWindow.

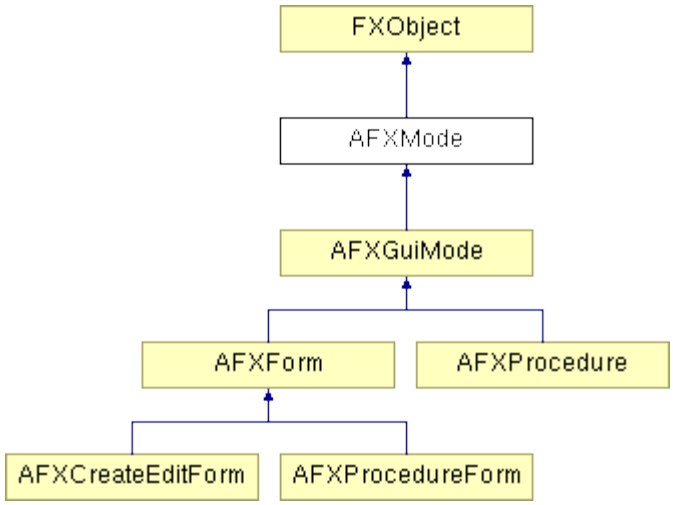
show()

Shows the widget.

Reimplemented from FXWindow.

AFXMode

This class is the base class for modes.



AFXMode()

Constructor.

getCommand(index)

Returns the command at the given index (returns 0 if the index is out-of-bounds).

Argument	Type	Default	Description
index	Int		Command index (0-based).

getNumCommands()

Returns the number of commands associated with the mode.

isActive()

Returns True if the mode is active.

Class flags

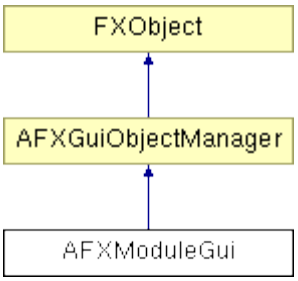
Message ID's.

ID_ACTIVATE	**Activates the mode.**
ID_COMMIT	**Commits the mode.**
ID_CANCEL	**Cancels the mode.**
ID_DEACTIVATE	**Deactivates the mode.**
ID_GET_NEXT	**Gets the next step/dialog box.**
ID_RESUME	**Resumes the mode.**
ID_SET_DEFAULTS	**Sets defaults.**
ID_SUSPEND	**Suspends the mode.**
ID_CMD_ACTIVATED	**Indicates that a command is activated.**

ID_CMD_DEACTIVATED	**Indicates that a command is deactivated.**
ID_CMD_MODIFIED	**Indicates that a command is modified.**

AFXModuleGui

This is the base class for module GUIs and provides an interface for module GUI management.



AFXModuleGui(moduleName, displayTypes)

Constructor.

Argument	Type	Default	Description
moduleName	String		Name used to identify this module.
displayTypes	Int		Types of primary objects that this module may display.

activate()

Activates the module during switch processing (allows for module specific activation requirements).

deactivate()

Deactivates the module when switching out (allows for module specific deactivation requirements).

getModuleName()

Returns the name of the module given on construction.

getToolsetKernelInitializationCommands()

Returns the command string to initialize the toolsets in the kernel that are corresponding to the toolsets registered with the module GUI.

getTypesToDisplay()

Returns the type of the primary objects which may be displayed when this module is active (this currently assumes a single type).

hide(location)

Deactivates and hides the module's GUI components in the menubar, toolbar and toolbox.

Argument	Type	Default	Description
location	Int		Location where gui components are placed.

registerProcedureToolset(tool)

Registers a procedure toolset (called during construction of derived modules).

Argument	Type	Default	Description
tool	AFXProcedureToolsetGui		Pointer to procedure toolset being registered.

registerToolset(tool, opts)

Registers a toolset (called during construction of derived modules).

Argument	Type	Default	Description
tool	AFXToolsetGui		Pointer to toolset being registered.
opts	Int		Options for creating toolset GUI components.

show(location)

Activates and shows the module's GUI components in the menubar, toolbar and toolbox.

Argument	Type	Default	Description
location	Int		Location where gui components are placed.

unregisterToolset(name)

Unregisters a toolset.

Argument	Type	Default	Description
name	String		Name of toolset to unregister.

Class flags

Message ID's.

MISSING ENUMERATOR **MISSING ENUMERATOR DESCRIPTION**

Flags for the object to display.

PART **Displays a part primary object.**

ASSEMBLY **Displays an assembly primary object.**

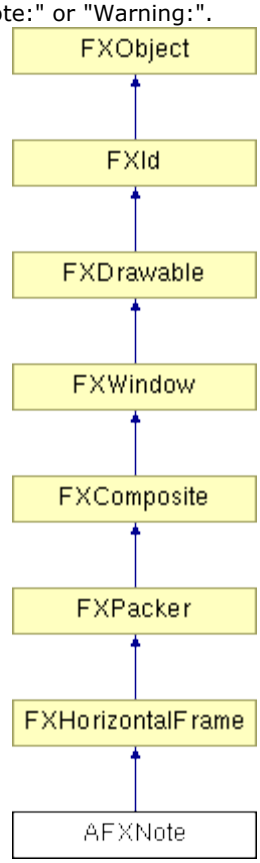
ODB **Displays an ODB primary object.**

XY_PLOT **Displays an XY plot primary object.**

SKETCH **Displays a sketch primary object.**

AFXNote

This class prefixes a given message string by either "Note:" or "Warning:".



AFXNote(p, message, opts=NOTE_INFORMATION, x=0, y=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
message	String		Note message string.
opts	Int	NOTE_INFORMATION	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.

create()

Creates the note.

Reimplemented from FXComposite.

detach()

Detaches the server-resources of the note.

Reimplemented from FXComposite.

disable()

Disables the note.

Reimplemented from FXWindow.

enable()

Enables the note.

Reimplemented from FXWindow.

getText()

Returns the note's message string.

setText(message)

Sets the note's message string.

Argument	Type	Default	Description
message	String		Message.

Global flags

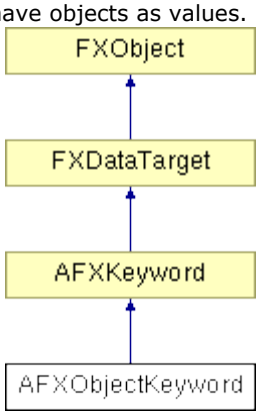
Flags for note styles.

NOTE_INFORMATION **Information note.**

NOTE_WARNING **Warning note.**

AFXObjectKeyword

This class is designed for the command keywords that have objects as values.



AFXObjectKeyword(command, name, isRequired=False, defaultValue='')

Constructor.

Argument	Type	Default	Description
command	AFXCommand		Host command.
name	String		Keyword name.
isRequired	Bool	False	True if the keyword is a required argument of the command.
defaultValue	String	''	Default value.

getTypeName()

Returns the name of the keyword type.

Implements AFXKeyword.

getValue()

Returns the keyword's current value.

getValueAsString()

Returns the text string that represents the keyword's current value.

Implements AFXKeyword.

isValueChanged()

Returns True if the keyword value differs from its previous value.

Implements AFXKeyword.

setDefaultValue(defaultValue)

Sets the keyword's default value.

Argument	Type	Default	Description

defaultValue	String		Default value.

setValue(newValue)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	String		New value.

setValueToDefault(ignoreUnspecified=False)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	If set to True, ignore setting the value if the default is unspecified.

setValueToPrevious()

Sets the keyword value to its previous value.

Implements AFXKeyword.

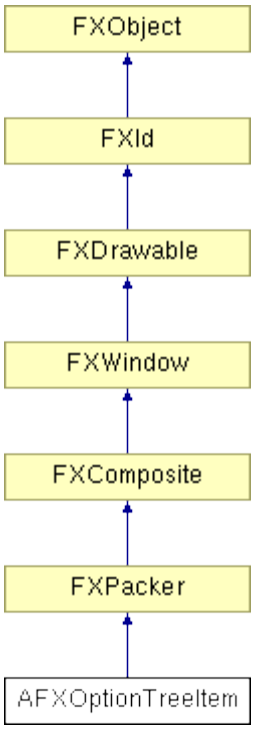
syncPreviousValue()

Sets the keyword's previous value to its current value.

Implements AFXKeyword.

AFXOptionTreeItem

This class is a tree widget with check buttons.



AFXOptionTreeItem(p, label, tgt=None, sel=0, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Constructor creating a top-level (root) tree item.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
label	String		Label text.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	0	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	DEFAULT_SPACING	Left padding.
pr	Int	DEFAULT_SPACING	Right padding.
pt	Int	DEFAULT_SPACING	Top padding.
pb	Int	DEFAULT_SPACING	Bottom padding.
hs	Int	DEFAULT_SPACING	Horizontal spacing.
vs	Int	DEFAULT_SPACING	Vertical spacing.

addItemAfter(label, tgt=None, sel=0)

Creates a new tree item as a sibling after (below) the tree item.

Argument	Type	Default	Description
label	String		Item label.
tgt	FXObject	None	Item target.
sel	Int	0	Item selector.

addItemBefore(label, tgt=None, sel=0)

Creates a new tree item as a sibling before (above) the tree item.

Argument	Type	Default	Description
label	String		Item label.
tgt	FXObject	None	Item target.
sel	Int	0	Item selector.

addItemFirst(label, tgt=None, sel=0)

Creates a new tree item as the first child of the tree item.

Argument	Type	Default	Description
label	String		Item label.
tgt	FXObject	None	Item target.
sel	Int	0	Item selector.

addItemLast(label, tgt=None, sel=0)

Creates a new tree item as the last child of the tree item.

Argument	Type	Default	Description
label	String		Item label.
tgt	FXObject	None	Item target.
sel	Int	0	Item selector.

childAtIndex(index)

Returns the child tree at the given index.

Argument	Type	Default	Description
index	Int		Index.

collapse()

Collapses (hides) the children.

computeDefaultArrowSize()

Computes default size of the arrow button.

containsChild(tree)

Checks whether the given tree is a child of this object.

Argument	Type	Default	Description
tree	AFXOptionTreeItem		Item.

create()

Creates the tree item.

Reimplemented from FXComposite.

disable()

Disables the tree item.

Reimplemented from FXWindow.

enable()

Enables the tree item.

Reimplemented from FXWindow.

expand()

Expands (shows) the children.

getArrowSize()

Returns the size of the arrow button.

getCheck()

Returns the check state of the tree item.

getDefaultWidth()

Returns the default width of the tree item.

Reimplemented from FXPacker.

getFirst()

Returns the first child tree.

Reimplemented from FXWindow.

getLast()

Returns the last child tree.

Reimplemented from FXWindow.

getNext()

Returns the next sibling tree.

Reimplemented from FXWindow.

getParent()

Returns the parent tree widget, or NULL if the tree item is the root.

Reimplemented from FXWindow.

getPrev()

Returns the previous sibling tree.

Reimplemented from FXWindow.

getText()

Returns the label text shown in the tree item's check button.

hasVisibleChildren()

Checks whether the tree item has any visible children.

hide()

Hides the tree item.

Reimplemented from FXWindow.

indexOfChild(tree)

Returns the index of an immediate child tree, or -1 if not found.

Argument	Type	Default	Description
tree	AFXOptionTreeItem		Item.

isChildOf(tree)

Checks whether this object is contained in the given tree.

Argument	Type	Default	Description
tree	AFXOptionTreeItem		Item.

isExpanded()

Checks whether the tree item shows its children.

numChildren()

Returns the number of child trees.

Reimplemented from FXWindow.

setArrowSize(size)

Sets the size of the arrow button for this object and all its children.

Argument	Type	Default	Description

size	Int		Size.

setCheck(check, notify, propagating=False)

Sets the check state of the tree item and its children, optionally notifying targets.

Argument	Type	Default	Description
check	Int		Check state.
notify	Bool		Notification flag.
propagating	Bool	False	Propagation flag.

setCheck(check=True)

Sets the check state of the tree item and its children.

Argument	Type	Default	Description
check	Int	True	Check state.

setText(txt)

Sets the label text shown in the tree item's check button.

Argument	Type	Default	Description
txt	String		Label text.

show()

Shows the tree item.

Reimplemented from FXWindow.

updateCheck(notify)

Updates the check state of the tree item and its ancestors.

Argument	Type	Default	Description
notify	Bool		Notification flag.

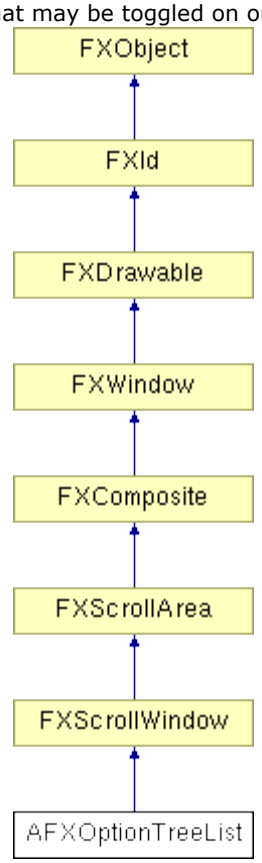
Class flags

Message ID's.

ID_TOGGLEEXPAND	**Toggles display of frame with children.**
ID_CHECKSTATE	**Represents checked state of this object.**
ID_SUBTREE	**Container for the subtree.**
ID_EXPAND	**Expands frame with children.**
ID_COLLAPSE	**Collapses frame with children.**

AFXOptionTreeList

This class provides a scrolled list of groups of options that may be toggled on or off as a group or individually.



AFXOptionTreeList(p, nvis, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
nvis	Int		Number of visible items of list.
opts	Int	0	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	DEFAULT_SPACING	Left padding.
pr	Int	DEFAULT_SPACING	Right padding.
pt	Int	DEFAULT_SPACING	Top padding.
pb	Int	DEFAULT_SPACING	Bottom padding.
hs	Int	DEFAULT_SPACING	Horizontal spacing.
vs	Int	DEFAULT_SPACING	Vertical spacing.

addItemFirst(text, tgt=None, msg=0)

Adds a new item with the given text as the first item of the list.

Argument	Type	Default	Description
text	String		Item text.
tgt	FXObject	None	Item target.
msg	Int	0	Item selector.

addItemLast(text, tgt=None, msg=0)

Adds a new item with the given text as the last item of the list.

Argument	Type	Default	Description
text	String		Item text.
tgt	FXObject	None	Item target.
msg	Int	0	Item selector.

clearItems()

Removes all items from the list.

computeItemHeight(p=None)

Computes the item size to be used as a base for default height computation.

Argument	Type	Default	Description
p	AFXOptionTreeItem	None	Item.

createItem(text, tgt, msg)

Creates a new tree item object.

Argument	Type	Default	Description
text	String		Item text.
tgt	FXObject		Item target.
msg	Int		Item selector.

getContentHeight()

Returns the content height.

Reimplemented from FXScrollWindow.

getContents()

Returns the content window.

getContentWidth()

Returns the content width.

Reimplemented from FXScrollWindow.

getDefaultHeight()

Returns the default height.

Reimplemented from FXScrollArea.

getDefaultWidth()

Returns the default width.

Reimplemented from FXScrollArea.

getFirstItem()

Returns the first root item.

getHSpacing()

Returns the horizontal inter-child spacing.

getLastItem()

Returns the last root item.

getNumItems()

Returns the number of top-level items.

getNumVisible()

Returns the number of visible items.

getPadBottom()

Returns the bottom padding.

getPadLeft()

Returns the left padding.

getPadRight()

Returns the right padding.

getPadTop()

Returns the top padding.

getVSpacing()

Returns the vertical inter-child spacing.

layout()

Recalculates layout.

Reimplemented from FXScrollWindow.

moveContents(x, y)

Moves contents to the specified position.

Argument	Type	Default	Description
x	Int		X location.
y	Int		Y location

removeItem(item)

Removes the given item from the list. This method does nothing if the given item does not exist.

Argument	Type	Default	Description
item	AFXOptionTreeItem		Item to be removed.

setHSpacing(hs)

Sets the horizontal inter-child spacing.

Argument	Type	Default	Description
hs	Int		Horizontal spacing.

setNumVisible(nvis)

Sets the number of visible items.

Argument	Type	Default	Description
nvis	Int		Number of visible items.

setPadBottom(pb)

Sets the bottom padding.

Argument	Type	Default	Description
pb	Int		Bottom padding.

setPadLeft(pl)

Sets the left padding.

Argument	Type	Default	Description
pl	Int		Left padding.

setPadRight(pr)

Sets the right padding.

Argument	Type	Default	Description
pr	Int		Right padding.

setPadTop(pt)

Sets the top padding.

Argument	Type	Default	Description
pt	Int		Top padding.

setVSpacing(vs)

Sets the vertical inter-child spacing.

Argument	Type	Default	Description
vs	Int		Vertical spacing.

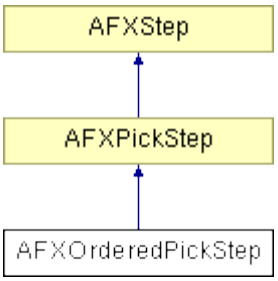
Class flags

Message ID's.

ID_CONTENTS **ID for the content window.**

AFXOrderedPickStep

This class is used to provide pick steps in GUI procedures.



AFXOrderedPickStep(owner, keyword, prompt, entitiesToPick, highlightLevel=1)

Constructor.

Argument	Type	Default	Description
owner	AFXProcedure		Procedure creating the step.
keyword	AFXObjectKeyword		Object kwd containing pick variable. Part of AFXGuiCommand.
prompt	String		Step's prompt displayed in prompt area.
entitiesToPick	Int		Type of entities to pick.
highlightLevel	Int	1	Highlight level.

onCancel()

Called when the step is cancelled.

Reimplemented from AFXPickStep.

onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.

Reimplemented from AFXPickStep.

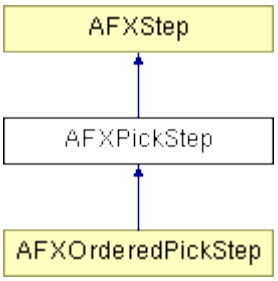
reset()

Allows a step to reset any of its data (if needed) when looping.

Reimplemented from AFXPickStep.

AFXPickStep

This class is used to provide pick steps in GUI procedures.



AFXPickStep(owner, keyword, prompt, entitiesToPick, numberToPick=ONE, highlightLevel=1, sequenceStyle=ARRAY)

Constructor.

Argument	Type	Default	Description
owner	AFXProcedure		Procedure creating the step.
keyword	AFXObjectKeyword		Object kwd containing pick variable. Part of AFXGuiCommand.
prompt	String		Step's prompt displayed in prompt area.
entitiesToPick	Int		Type of entities to pick.
numberToPick	pickAmountEnum	ONE	How many entities to pick.
highlightLevel	Int	1	Highlight level.
sequenceStyle	sequenceStyleEnum	ARRAY	Sequence style of picked variables in the command.

addElementSetSelection(name)

Adds an element set to the step's selections.

Argument	Type	Default	Description
name	String		Name of set.

addGeometrySetSelection(name)

Adds a geometry set to the step's selections.

Argument	Type	Default	Description
name	String		Name of set.

addNodeSetSelection(name)

Adds a node set to the step's selections.

Argument	Type	Default	Description
name	String		Name of set.

addPointKeyIn(keyword)

Creates a textfield on the prompt line as an alternative method of specifying a point.

Argument	Type	Default	Description
keyword	AFXTupleKeyword		Keyword

addSurfaceSelection(name)

Adds a surface to the step's selections.

Argument	Type	Default	Description
name	String		Name of surface.

allowRepeatedSelections(value=True)

Allows the picking of prior selections (from prior pick steps of the procedure).

Argument	Type	Default	Description
value	Bool	True	If True, allow repeated selections between steps.

onCancel()

Called when the step is cancelled.

Reimplemented from AFXStep.

Reimplemented in AFXOrderedPickStep.

onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.

Reimplemented from AFXStep.

Reimplemented in AFXOrderedPickStep.

onResume()

Called when the step is resumed.

Reimplemented from AFXStep.

onSuspend()

Called when the step is suspended.

Reimplemented from AFXStep.

reset()

Allows a step to reset any of its data (if needed) when looping.

Reimplemented from AFXStep.

Reimplemented in AFXOrderedPickStep.

setEdgeRefinements(refinement)

Sets the refinements to be used when picking edges.

Argument	Type	Default	Description

refinement	Int		Refinement flag.

setElementEdgeRefinements(refinement)

Sets the refinements to be used when picking element edges.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

setElementFaceRefinements(refinement)

Sets the refinements to be used when picking element faces.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

setElementRefinements(refinement)

Sets the refinements to be used when picking elements.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

setFaceRefinements(refinement)

Sets the refinements to be used when picking faces.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

setInstanceRefinements(refinement)

Sets the refinements to be used when picking instances.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

setNodeRefinements(refinement)

Sets the refinements to be used when picking nodes.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

setSketchRefinements(refinement)

Sets the refinements to be used when picking sketches.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

setXyRefinements(refinement)

Sets the refinements to be used when picking xy objects.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

Class flags

Flags for the number of entities to pick.

ONE **Allow only one entity to be picked.**

MANY **Allow one or more entities to be picked.**

Flags for refining pickable entities.

STRAIGHT **Allow only straight entities to be picked.**

WIRE **Allow only wire entities to be picked.**

PLANAR **Allow only planar entities to be picked.**

CONICAL **Allow only conical entities to be picked.**

SHELL **Allow only shell entities to be picked.**

ORPHAN_MESH **Allow only orphan mesh entities to be picked.**

SOLID **Allow only solid entities to be picked.**

GEOMETRY **Allow only geometry entities to be picked.**

POINT **Allow only point entities to be picked.**

BACKGROUND **Allow only background entities to be picked while sketching.**

FOREGROUND **Allow only sketch entities to be picked.**

VERTICAL **Allow only vertical geometric sketch entities to be picked.**

HORIZONTAL **Allow only horizontal geometric sketch entities to be picked.**

CONSTRUCTION **Allow only construction sketch entities to be picked.**

NO_CONSTRUCTION **Allow only non-construction geometric sketch entities to be picked.**

SPOT **Allow only spot sketch entities to be picked.**

CIRCULAR **Allow only circular sketch entities to be picked.**

INTERIOR **Allow only interior entities to be picked.**

EXTERIOR **Allow only exterior entities to be picked.**

Flags for pickable entities.

VERTICES **Allow vertices to be picked.**

EDGES **Allow edges to be picked.**

FACES **Allow faces to be picked.**

CELLS **Allow cells to be picked.**

STRINGERS **Allow stringers to be picked.**

SKINS **Allow skins to be picked.**

ELEMENT_EDGES **Allow element edges to be picked.**

ELEMENT_FACES **Allow element faces to be picked.**

NODES **Allow nodes to be picked.**

ELEMENTS **Allow elements to be picked.**

INSTANCES **Allow part instances in the model database to be picked.**

MAX_DIMENSION **Allow picking only objects of the highest dimension (1D, 2D, 3D).**

REFERENCE_POINTS **Allow reference points to be picked.**

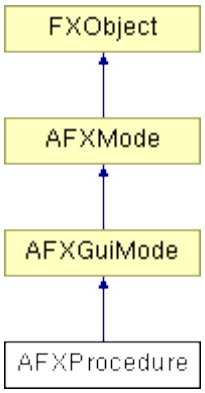
INTERESTING_POINTS	**Allow interesting points to be picked.**
DATUM_POINTS	**Allow datum points to be picked.**
DATUM_AXES	**Allow datum axes to be picked.**
DATUM_PLANES	**Allow datsum planes be picked.**
DATUM_CSYS	**Allow datum CSYS's to be picked.**
REMOVABLE_EDGES	**Allow edges to be removed from face selections.**
FEATURES	**Allow features to be picked.**
SKETCH_VERTICES	**Allow sketch vertices to be picked.**
SKETCH_GEOMETRIES	**Allow sketch geometries to be picked.**
SKETCH_DIMENSIONS	**Allow sketch dimensions to be picked.**
SKETCH_CONSTRAINTS	**Allow sketch constraints to be picked.**
SKETCH_COORDINATES	**Allow sketch coordinates to be picked must add keyin.**
SKIN_ELEMENTS	**Allow elements on skins to be picked.**
STRINGER_ELEMENTS	**Allow elements on stringers to be picked.**
POINTS	**Allow all types of points to be picked.**
LINES	**Allow all types of lines to be picked.**
PLANES	**Allow all types of planes to be picked.**
XY_POINT	**Allow picking of XY coordinates from an XY plot.**
XY_COORDINATE	**Allow picking of XY coordinates from an XY plot, with the option to type in XY values.**

Flags for the command sequence style of the picked items.

TUPLE	**Specify pick as a comma separated tuple of single items.**
ARRAY	**Specify pick as a plus separated sequence items.**

AFXProcedure

This class provides the basis for writing procedures.



AFXProcedure(owner, type=NORMAL)

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Owner (a module or a toolset) of the procedure.
type	typeEnum	NORMAL	

activate()

Activates the mode.

Reimplemented from AFXGuiMode.

cancel(tgt=None, msg=0)

Tries to cancel the procedure depending on checkCancel results.

Argument	Type	Default	Description
tgt	FXObject	None	Completion message target.
msg	Int	0	Completion message ID.

checkBackup()

Returns 1 if ok to backup else returns 0.

checkCancel()

Returns BAILOUT_NOTOK, BAILOUT_OK, BAILOUT_WIP (writes to the message area), or BAILOUT_SAVE (use the 3 button save dialog box).

commit()

Commits the procedure when the current dialog box calls either done or value changed.

Implements AFXGuiMode.

continueMode()

Used to get the next step in the mode.

Implements AFXGuiMode.

deactivate()

deactivates the mode.

Reimplemented from AFXGuiMode.

getCurrentStep()

Returns the current step.

getFirstStep()

Returns the first step to be executed in the procedure.

getLoopStep()

Returns the step to which the procedure should loop back after processing its commands; if zero is returned (the default behavior) the procedure will not loop.

getNextStep(previousStep)

Returns the next step to be executed; if zero is returned the procedure will process its commands.

Argument	Type	Default	Description
previousStep	AFXStep		Previous step.

getNumSteps()

Returns the number of steps in the step stack.

handleException(exc)

This method is called if an error occurs while issuing commands. It can be reimplemented in derived classes to perform special error handling.

Argument	Type	Default	Description
exc	nex_Exception		Exception.

onBackup()

Called when a procedure backs up a step.

onCancel()

Called when a procedure cancels.

onCmdBackup(sender, sel, ptr)

Message handler for handling backup button activation.

Argument	Type	Default	Description
sender	FXObject		Sender.
sel	Int		Selector.
ptr	String		Data.

onCmdHandle2BtnBailout(sender, sel, ptr)

Message handler for handling the user 2 button bailout choice.

Argument	Type	Default	Description
sender	FXObject		Sender.
sel	Int		Selector.
ptr	String		Data.

onCmdHandleBailout(sender, sel, ptr)

Message handler for handling the user 3 button bailout choice.

Argument	Type	Default	Description
sender	FXObject		Sender.
sel	Int		Selector.
ptr	String		Data.

onResume()

Called when a procedure resumes.

onSuspend()

Called when a procedure suspends.

onValueChanged()

Called when a procedure's step changes in value.

setCurrentDb(db)

Sets the current dialog box of the mode. Procedures will have this set by AFXDialogStep.

Argument	Type	Default	Description
db	AFXDialog		Dialog box.

setSelectionOptions(pickDepth=CLOSEST, pickScope=ALL, dragShape=RECTANGLE, dragScope=INSIDE_CROSSING, isoLines=True)

Sets the selection options to be used for picking.

Argument	Type	Default	Description
pickDepth	pickDepthEnum	CLOSEST	Depth into the screen of picking.
pickScope	pickScopeEnum	ALL	Entity type.

dragShape	dragShapeEnum	RECTANGLE	Drag-window shape.
dragScope	dragScopeEnum	INSIDE_CROSSING	Drag-window scope.
isoLines	Bool	True	If True, show isolines on surfaces.

verifyCurrentKeywordValues()

Checks whether keywords of active commands for the current dialog box contain valid data and, if not, posts a dialog box with an error message.

Reimplemented from AFXGuiMode.

Class flags

Message ID's.

ID_HANDLE_2BTN_BAILOUT **ID for handling bailout.**

ID_BACKUP **ID for the backup button.**

Flags for the drag scope.

INSIDE **Pick entities inside the drag shape only.**

INSIDE_CROSSING **Pick entities inside and crossing the drag shape.**

CROSSING **Pick entities crossing the drag shape only.**

OUTSIDE_CROSSING **Pick entities outside and crossing the drag shape.**

OUTSIDE **Pick entities outside the drag shape only.**

Flags for the drag shape.

RECTANGLE **Use rectangular drag shape.**

CIRCLE **Use circular drag shape.**

POLYGON **Use polygonal drag shape.**

Flags for the pick depth.

CLOSEST **Only pick the entity closest to the screen.**

INFINITE **Pick entities at any depth.**

Flags for the pick scope.

INTERIOR **Pick only interior entities.**

EXTERIOR **Pick only exterior entities.**

ALL **Pick all entities.**

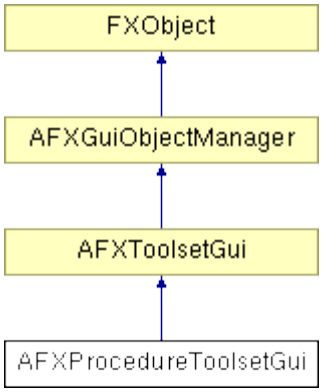
Flags for the activate action.

NORMAL **Cancel the currently running procedure (default).**

SUBPROCEDURE **Suspend the currently running procedure.**

AFXProcedureToolsetGui

This is the base class for toolset GUIs used in procedure steps (e.g. the Sketch toolset) and provides an interface for managing the toolset's GUI items. In conjuction with the AFXProcedureToolsetGuiData class, it provides a mechanism to overlay menubar, toolbar, and toolbox GUI items while the step executes.



AFXProcedureToolsetGui()

Deserialization.

AFXProcedureToolsetGui(toolsetName)

Constructor.

Argument	Type	Default	Description
toolsetName	String		Toolset name passed in from derived toolsets.

swapInToolsetItems()

Swaps in the toolset's GUI items.

swapOutToolsetItems()

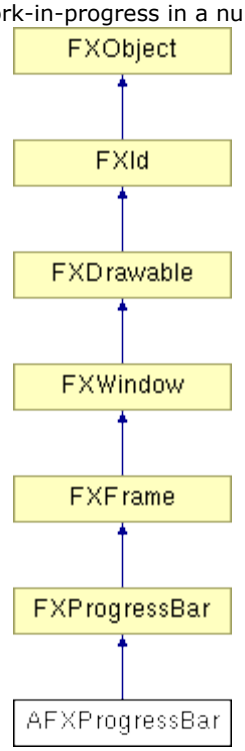
Swaps out the toolset's GUI items.

Class flags

ID_LAST **Do not use, do not delete; for use by derived classes.**

AFXProgressBar

This class contains a progress bar, which can present work-in-progress in a number of different styles.



AFXProgressBar(p, tgt=None, sel=0, opts=FRAME_SUNKEN| FRAME_THICK, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	FRAME_SUNKEN| FRAME_THICK	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	DEFAULT_PAD	Left padding (margin).
pr	Int	DEFAULT_PAD	Right padding (margin).
pt	Int	DEFAULT_PAD	Top padding (margin).
pb	Int	DEFAULT_PAD	Bottom padding (margin).

create()

Creates the progress bar.

Reimplemented from FXProgressBar.

getBarStyle()

Returns the progress bar style.

Reimplemented from FXProgressBar.

getDefaultHeight()

Returns the default height.

Reimplemented from FXProgressBar.

getDefaultWidth()

Returns the default width.

Reimplemented from FXProgressBar.

getNumCursorBoxes()

Returns the number of cursor boxes displayed.

getProgress()

Returns the current progress amount.

Reimplemented from FXProgressBar.

getTotal()

Returns the total progress amount.

Reimplemented from FXProgressBar.

hide()

Hides the progress bar.

Reimplemented from FXWindow.

hideNumber()

Hides the progress bar iteration or percentage text.

Reimplemented from FXProgressBar.

setBarStyle(style)

Sets the progress bar style.

Argument	Type	Default	Description
style	Int		Style flag.

setNumCursorBoxes(nb)

Sets the number of cursor boxes to display.

Argument	Type	Default	Description
nb	Int		Number of boxes.

setProgress(value)

Sets the current progress amount that is used by a progress bar in either iteration or percentage mode; the progress amount is ingored by a progress bar in scanner mode.

Reimplemented from FXProgressBar.

Argument	Type	Default	Description
value	Int		

setTotal(value)

Sets the total progress amount that is used by a progress bar in either iteration or percentage mode; the progress amount is ingored by a progress bar in scanner mode.

Reimplemented from FXProgressBar.

Argument	Type	Default	Description
value	Int		

show()

Shows the progress bar.

Reimplemented from FXWindow.

showNumber(style=AFXPROGRESSBAR_PERCENTAGE)

Shows the progress iteration or percentage text.

Argument	Type	Default	Description
style	Int	AFXPROGRESSBAR_PERCENTAGE	Style flag.

Class flags

Message ID's.

ID_TIMER **ID for timer.**

Global flags

Flags for progress bar styles.

AFXPROGRESSBAR_PERCENTAGE **Percentage complete mode.**

AFXPROGRESSBAR_HORIZONTAL **Horizontal display.**

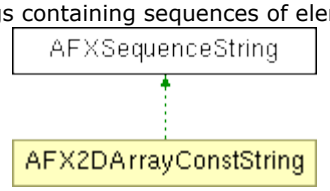
AFXPROGRESSBAR_VERTICAL **Vertical display.**

AFXPROGRESSBAR_SCANNER **Scanner mode.**

AFXPROGRESSBAR_ITERATOR **Iterator mode.**

AFXSequenceString

This class supports parsing and modification of strings containing sequences of elements separated with some separator character.



AFXSequenceString(value='', sep=',')

Constructor.

Argument	Type	Default	Description
value	String	''	String with initial sequence values.
sep	String	','	Separator character for sequence elements.

AFXSequenceString()

Undefined copy constructor (this class has no copy semantics).

forceNumElements(num, fill)

Forces the content string to contain a tuple with the given number of elements.

Argument	Type	Default	Description
num	Int		New number of elements.
fill	String		String to insert in empty spaces.

getContentString()

Returns a string containing values of the sequence elements.

Reimplemented in AFX2DArrayConstString.

getElementSeparator()

Returns the element separator character.

getLength(index)

Returns the length in characters of a sequence element.

Argument	Type	Default	Description
index	Int		Element index.

getNumElements()

Returns the number of elements in this sequence.

getPosition(index)

Returns the position in the content string of the beginning character of the sequence element.

Argument	Type	Default	Description
index	Int		Element index.

getValue(index)

Returns the value of a sequence element.

Argument	Type	Default	Description
index	Int		Element index.

insert(index, numElements, val)

Inserts many copies of an element.

Argument	Type	Default	Description
index	Int		Element index at which inserting begins.
numElements	Int		Number of elements to insert.
val	String		Value for the new elements.

isValidSequence()

Returns True if this object contains a valid sequence.

remove(index, numElements)

Removes elements starting at the given index.

Argument	Type	Default	Description
index	Int		Element index at which removal begins.
numElements	Int		Number of elements to remove.

setContentString(seqstr)

Resets all values for the sequence elements.

Argument	Type	Default	Description
seqstr	String		Sequence string with new values.

setElementSeparator(sep)

Sets the element separator character.

Argument	Type	Default	Description
sep	String		Separator character.

setLength(index, length)

Sets the length of the sequence element.

Argument	Type	Default	Description

index	Int		Element index.
length	Int		New length (in characters).

setPosition(index, position)

Sets the position of the sequence element.

Argument	Type	Default	Description
index	Int		Element index.
position	Int		New position within the string.

setValue(index, value, replaceAll=False)

Sets the value of a sequence element.

Argument	Type	Default	Description
index	Int		Element index.
value	String		New value.
replaceAll	Bool	False	If False (default), leading and trailing spaces will be preserved, otherwise, all space between separators will be replaced with the new value.

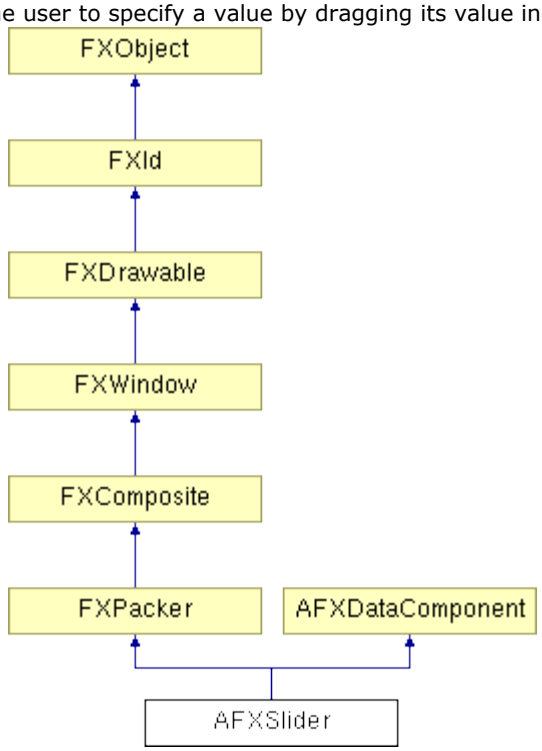
trimWhiteSpace(index)

Adjusts the position and length of the element to trim leading and trailing white spaces.

Argument	Type	Default	Description
index	Int		Element index.

AFXSlider

This class provides a slider, which allows the user to specify a value by dragging its value indicator.



AFXSlider(p, tgt=None, sel=0, opts=AFXSLIDER_NORMAL, x=0, y=0, w=0, h=0, pl=0, pr=0, pt=0, pb=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	AFXSLIDER_NORMAL	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	0	Left padding (margin).
pr	Int	0	Right padding (margin).
pt	Int	0	Top padding (margin).
pb	Int	0	Bottom padding (margin).

canFocus()

Returns True because a slider can receive focus.

Reimplemented from FXWindow.

disable()

Disables the slider.

Reimplemented from FXWindow.

enable()

Enables the slider.

Reimplemented from FXWindow.

getDecimalPlaces()

Returns the number of decimal points displayed.

getDefaultHeight()

Returns the default height.

Reimplemented from FXPacker.

getDefaultWidth()

Returns the default width.

Reimplemented from FXPacker.

getIncrement()

Returns the slider's auto-increment/decrement value.

getMaxLabelText()

Returns the maximum label's text.

getMinLabelText()

Returns the minimum label's text.

getRange()

Returns a sequence of ints (low, high) representing the widget's allowable minimum and maximum values.

getSliderStyle()

Returns the slider's style.

getTipText()

Returns the slider's tip text.

getTitleLabelJustify()

Returns the title label's justification mode.

getTitleLabelText()

Returns the title label's text.

getValue()

Returns the slider's value.

recalc()

Recalculates the slider. Redefined to handle slider movement.

Reimplemented from FXWindow.

setDecimalPlaces(dp)

Sets the number of decimal points displayed.

Argument	Type	Default	Description
dp	Int		Number of decimal places.

setIncrement(inc)

Sets the slider's auto-increment/decrement value.

Argument	Type	Default	Description
inc	Int		Increment.

setMaxLabelText(text)

Sets the maximum label's text.

Argument	Type	Default	Description
text	String		Max label text.

setMinLabelText(text)

Sets the minimum label's text.

Argument	Type	Default	Description
text	String		Min label text.

setRange(lo, hi)

Sets the slider's maximum and minimum values.

Argument	Type	Default	Description
lo	Int		Minimum value.
hi	Int		Maximum value.

setSliderStyle(style)

Sets the slider's style.

Argument	Type	Default	Description
style	Int		Style flag.

setTipText(text)

Sets the slider's tip text.

Argument	Type	Default	Description
text	String		Tip text.

setTitleLabelJustify(mode)

Sets the title label's justification mode.

Argument	Type	Default	Description
mode	Int		Justification mode.

setTitleLabelText(text)

Sets the title label's text.

Argument	Type	Default	Description
text	String		Title text.

setValue(value)

Sets the slider's value.

Argument	Type	Default	Description
value	Int		Value.

show()

Shows the slider.

Reimplemented from FXWindow.

Class flags

Message ID's.

ID_SLIDER **ID for the slider.**

ID_LAST **Last ID for this class.**

Global flags

options for tickmarks.

AFXSLIDER_HORIZONTAL **Slider shown horizontally.**

AFXSLIDER_VERTICAL **Slider shown vertically.**

AFXSLIDER_ARROW_UP **Slider has arrow head pointing up.**

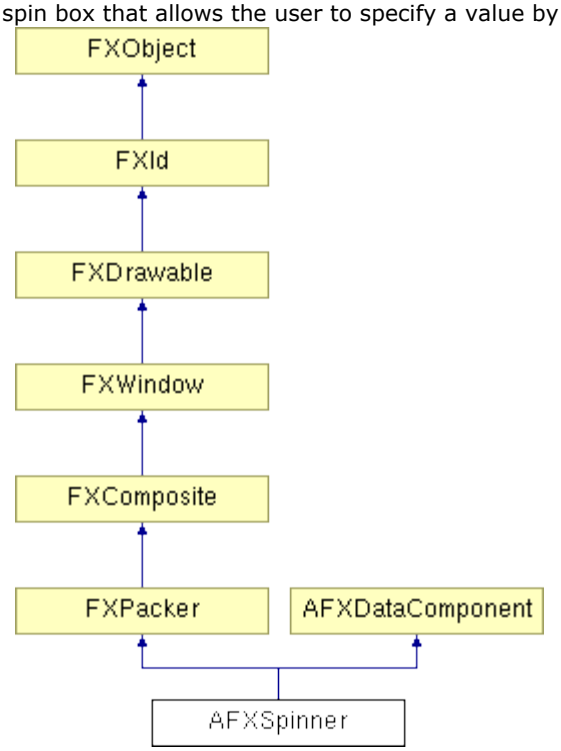
AFXSLIDER_ARROW_DOWN **Slider has arrow head pointing down.**

AFXSLIDER_ARROW_LEFT **Slider has arrow head pointing left.**

AFXSLIDER_ARROW_RIGHT	**Slider has arrow head pointing right.**
AFXSLIDER_INSIDE_BAR	**Slider is inside the slot rather than overhanging.**
AFXSLIDER_SHOW_VALUE	**Show slider value.**
AFXSLIDER_ABOVE_TITLE	**Show slider above its title.**
AFXSLIDER_AFTER_TITLE	**Show slider after its title.**
AFXSLIDER_NORMAL	**Default slider options--slider is horizontal, inside the slot, and shown above its title label.**

AFXSpinner

This class contains a label that precedes a spin box that allows the user to specify a value by clicking on its arrow buttons.



AFXSpinner(p, ncols, labelText, tgt=None, sel=0, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
ncols	Int		Number of columns.
labelText	String		Label string.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID
opts	Int	0	Options and hints.
x	Int	0	X coordinate of the origin.
y	Int	0	Y coordinate of the origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	DEFAULT_PAD	Left padding (margin).
pr	Int	DEFAULT_PAD	Right padding (margin).
pt	Int	DEFAULT_PAD	Top padding (margin).
pb	Int	DEFAULT_PAD	Bottom padding (margin).

create()

Creates the spinner.

Reimplemented from FXComposite.

disable()

Disables the spinner.

Reimplemented from FXWindow.

enable()

Enables the spinner.

Reimplemented from FXWindow.

getCheck()

Returns the state of the check button or the radio button.

getHelpText()

Returns the status line help text.

getIncrement()

Returns the spinner increment.

getLabelFont()

Returns the label font.

getLabelText()

Returns the label string.

getRange()

Returns a sequence of ints (low, high) representing the widget's allowable minimum and maximum values.

getTipText()

Returns the tool tip message.

getValue()

Returns the spinner value.

isEditable()

Returns True if the text in the text field may be edited.

isReadOnlyState()

Returns True if the spinner appears in the read-only state.

setCheck(state)

Sets the state of the check button or the radio button.

Argument	Type	Default	Description
state	Bool		State.

setCheckButtonSelector(sel)

Sets the message ID of the check button or the radio button.

Argument	Type	Default	Description
sel	Int		Selector.

setCheckButtonTarget(tgt)

Sets the message target of the check button or the radio button.

Argument	Type	Default	Description
tgt	FXObject		Target.

setEditable(edit=True)

Sets the editable state for the input field.

Argument	Type	Default	Description
edit	Bool	True	If True, input field is editable.

setHelpText(text)

Sets the status line help text.

Argument	Type	Default	Description
text	String		Help text.

setIncrement(incr)

Sets the spinner increment.

Argument	Type	Default	Description
incr	Int		Increment.

setLabelFont(fnt)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		Label font.

setLabelText(txt)

Sets the label string.

Argument	Type	Default	Description
txt	String		Label text.

setRange(low, high)

Sets the spinner range.

Argument	Type	Default	Description
low	Int		Minimum value.
high	Int		Maximum value.

setReadOnlyState(readonly=True)

Sets the read-only state of the spinner.

Argument	Type	Default	Description
readonly	Bool	True	State.

setTipText(text)

Sets the tool tip message.

Argument	Type	Default	Description
text	String		Tooltip text.

setValue(val, notify=False)

Sets the spinner's value.

Argument	Type	Default	Description
val	Int		Value.
notify	Bool	False	Notification flag.

Class flags

ID_BUTTON **ID for the check or radio button.**

ID_SPINNER **ID for the spinner.**

Global flags

Flags for AFX spinner options.

AFXSPINNER_CHECKBUTTON **Use a check button instead of a label.**

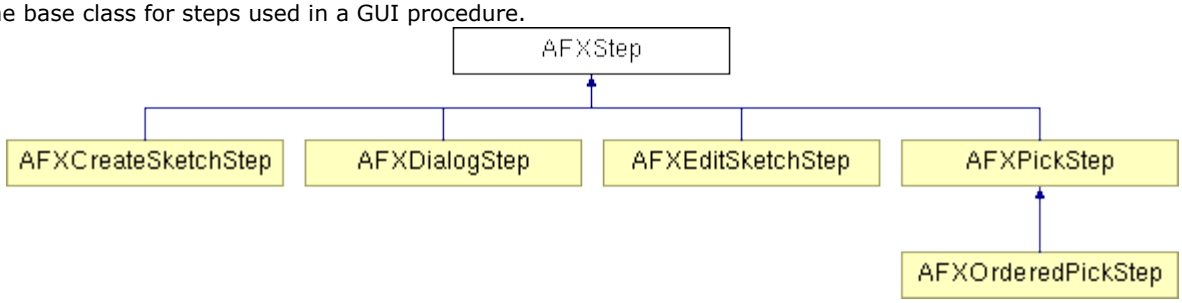
AFXSPINNER_RADIOBUTTON **Use a radio button instead of a label.**

AFXSPINNER_VERTICAL **Orient label or button above spinner.**

AFXSPINNER_READONLY **Configure spinner to the read-only state.**

AFXStep

This class is the base class for steps used in a GUI procedure.



AFXStep(prompt, owner)

Constructor.

Argument	Type	Default	Description
prompt	String		Prompt.
owner	AFXProcedure		Owner.

onCancel()

Called when the step is cancelled.

Reimplemented in AFXCreateSketchStep, AFXDialogStep, AFXEditSketchStep, AFXOrderedPickStep, and AFXPickStep.

onDone()

Called when the step completes.

onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.

Reimplemented in AFXCreateSketchStep, AFXDialogStep, AFXEditSketchStep, AFXOrderedPickStep, and AFXPickStep.

onResume()

Called when the step is resumed.

Reimplemented in AFXCreateSketchStep, AFXDialogStep, AFXEditSketchStep, and AFXPickStep.

onSuspend()

Called when the step is suspended.

Reimplemented in AFXCreateSketchStep, AFXDialogStep, AFXEditSketchStep, and AFXPickStep.

onValueChanged()

Called when the step's value changes.

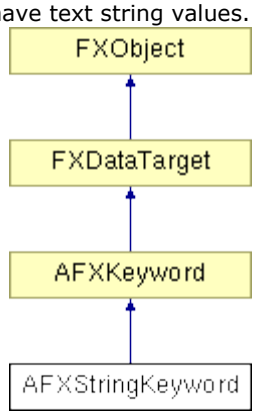
reset()

Allows a step to reset any of its data (if needed) when looping.

Reimplemented in AFXOrderedPickStep, and AFXPickStep.

AFXStringKeyword

This class is designed for the command keywords that have text string values.



AFXStringKeyword(command, name, isRequired=False, defaultValue='')

Constructor.

Argument	Type	Default	Description
command	AFXCommand		Host command.
name	String		Keyword name.
isRequired	Bool	False	True if the keyword is a required argument of the command.
defaultValue	String	''	Default value.

getTypeName()

Returns the name of the keyword type.

Implements AFXKeyword.

getValue()

Returns the keyword's current value.

getValueAsString()

Returns the text string that represents the keyword's current value.

Implements AFXKeyword.

isValueChanged()

Returns True if the keyword value differs from its previous value.

Implements AFXKeyword.

setDefaultValue(defaultValue)

Sets the keyword's default value.

Argument	Type	Default	Description

defaultValue	String		Default value.

setValue(newValue)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	String		New value.

setValueToDefault(ignoreUnspecified=False)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Ignore setting the value if the default is unspecified.

setValueToPrevious()

Sets the keyword value to its previous value.

Implements AFXKeyword.

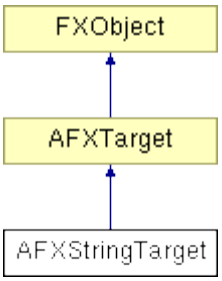
syncPreviousValue()

Sets the keyword's previous value to its current value.

Implements AFXKeyword.

AFXStringTarget

This class is designed for string targets.



AFXStringTarget(initialValue='')

Constructor.

Argument	Type	Default	Description
initialValue	String	''	Initial value.

getTypeName()

Returns the name of the target type ("String").

Implements AFXTarget.

getValue()

Returns the target's current value.

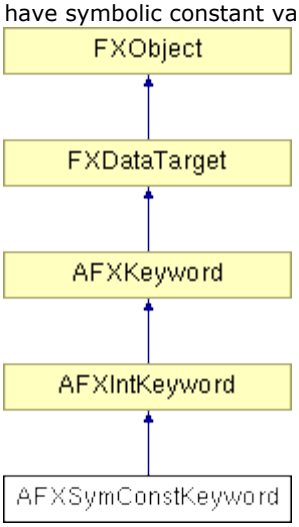
setValue(newValue)

Sets the target's current value.

Argument	Type	Default	Description
newValue	String		New value.

AFXSymConstKeyword

This class is designed for the command keywords that have symbolic constant values.



AFXSymConstKeyword(command, name, isRequired=False, defaultValue=0)

Constructor.

Argument	Type	Default	Description
command	AFXCommand		Host command.
name	String		Keyword name.
isRequired	Bool	False	True if the keyword is a required argument of the command.
defaultValue	Int	0	Default value.

getTypeName()

Returns the name of the keyword type.

Reimplemented from AFXIntKeyword.

getValueAsString()

Returns the text string that represents the keyword's current value.

Reimplemented from AFXIntKeyword.

setDefaultValue(defaultValue)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	Int		Default value.

setDefaultValueByString(defaultValueString)

Sets the keyword's default value (returns True if the given text string is valid).

Argument	Type	Default	Description

defaultValueString	String		Default value in text string form.

setDefaultValueByString(defaultValueString)

Sets the keyword's default value (returns True if the given text string is valid).

Argument	Type	Default	Description
defaultValueString	String		Default value in text string form.

setValue(newValue)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	Int		New value.

setValueByString(newValueString)

Sets the keyword's current value (returns True if the given text string is valid).

Argument	Type	Default	Description
newValueString	String		New value in text string form.

setValueByString(newValueString)

Sets the keyword's current value (returns True if the given text string is valid).

Argument	Type	Default	Description
newValueString	String		New value in text string form.

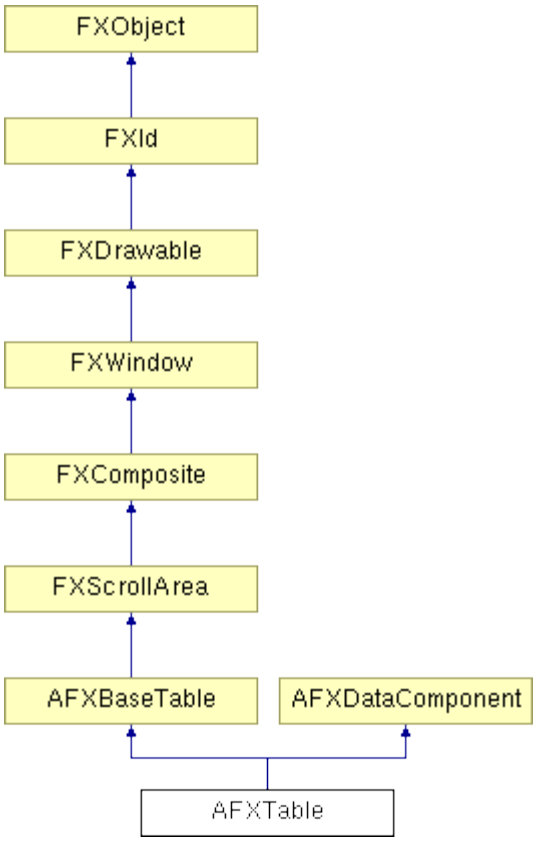
setValueToDefault(ignoreUnspecified=False)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Ignore setting the value if the default is unspecified.

AFXTable

This class implements an editable table.



AFXTable(p, numVisRows, numVisColumns, numRows, numColumns, tgt=None, sel=0, opts=AFXTABLE_NORMAL, x=0, y=0, w=0, h=0, pl=4, pr=4, pt=DEFAULT_MARGIN, pb=DEFAULT_MARGIN)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
numVisRows	Int		Number of rows to display.
numVisColumns	Int		Number of columns to display.
numRows	Int		Total number of rows including leading rows.
numColumns	Int		Total number of columns including leading columns.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	AFXTABLE_NORMAL	Options and hints.
x	Int	0	X coordinate of the origin.
y	Int	0	Y coordinate of the origin.
w	Int	0	Width of the table widget.
h	Int	0	Height of the table widget.

pl	Int	4	Left padding (margin).
pr	Int	4	Right padding (margin).
pt	Int	DEFAULT_MARGIN	Top padding (margin).
pb	Int	DEFAULT_MARGIN	Bottom padding (margin).

addList(text, opts=AFXTABLE_LIST_NORMAL)

Adds a list that have only text items to the table and returns the list ID. The text strings of the list items are delimited by tab "\t" characters in the given text. The list is used by items of type LIST.

Argument	Type	Default	Description
text	String		Tab "\t" delimited text string (e.g. "0\t50\t100\t150").
opts	Int	AFXTABLE_LIST_NORMAL	Options.

addList(opts=AFXTABLE_LIST_NORMAL)

Adds a list to the table and returns the list ID. The list is used by items of type LIST.

Argument	Type	Default	Description
opts	Int	AFXTABLE_LIST_NORMAL	List flag.

appendClientPopupItem(text, icon=None, tgt=None, sel=0, alias=None)

Appends a client item to the MB3 popup menu.

Argument	Type	Default	Description
text	String		Label, accelerator and help string (e.g. "Cu&t\tCtl+X\tCut selection to clipboard.").
icon	FXIcon	None	Icon to display.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
alias	String	None	Item alias name.

appendClientPopupSeparator()

Appends a client separator to the MB3 popup menu.

appendListItem(listId, text, icon=None)

Appends an item to the specified table list; returns the index of the new item.

Argument	Type	Default	Description
listId	Int		ID of the list to append to.
text	String		Item's text.
icon	FXIcon	None	Item's icon.

beginEdit(row, column)

Sets the specified item in edit mode if the item is editable.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

cancelEdit()

Cancels the edit mode.

clearClientPopupItems()

Removes all client items from the MB3 popup menu.

clearContents(startRow, startColumn, endRow, endColumn, clearEditableOnly=True)

Clears the text in the items in the specified range.

Argument	Type	Default	Description
startRow	Int		Row in which to start clearing.
startColumn	Int		Column in which to start clearing.
endRow	Int		Row in which to end clearing.
endColumn	Int		Column in which to end clearing.
clearEditableOnly	Bool	True	Specify True to clear the text of editable items only.

clearListItems(listId)

Removes all items from the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list to clear.

create()

Creates server-side resources.

Reimplemented from AFXBaseTable.

deleteColumns(startColumn, numColumns=1, notify=False)

Deletes columns starting at the specified column.

Argument	Type	Default	Description
startColumn	Int		Starting column.
numColumns	Int	1	Number of columns to delete.
notify	Bool	False	Specify True to notify target of the deletion.

deleteRows(startRow, numRows=1, notify=False)

Deletes rows starting at the specified row.

Argument	Type	Default	Description
startRow	Int		Starting row.

numRows	Int	1	Number of rows to delete.
notify	Bool	False	Specify True to notify target of the deletion.

deselectItem(row, column)

Deselects the specified item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

deselectRow(row)

Deselects all items in the row.

Argument	Type	Default	Description
row	Int		Row index.

destroy()

Destroys server-side resources.

Reimplemented from FXComposite.

detach()

Detaches server-side resources.

Reimplemented from AFXBaseTable.

disable()

Disables the table and the table items in the table.

Reimplemented from FXWindow.

disableItem(row, column)

Disables the specified item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

enable()

Enables the table and the table items in the table.

Reimplemented from FXWindow.

enableItem(row, column)

Enables the specified item.

Argument	Type	Default	Description
row	Int		Row index of item.

column	Int		Column index of item.

getColumnAtX(x)

Returns the column at the specified x coordinate; returns -1 if x is outside of the table.

Argument	Type	Default	Description
x	Int		X coordinate.

getColumnSortOrder(column)

Returns the sort order of the given column.

Argument	Type	Default	Description
column	Int		Column index.

getColumnWidth(column)

Returns the width, in pixels, of the specified column.

Argument	Type	Default	Description
column	Int		Column index.

getCurrentColumn()

Returns the column index of the current item.

Reimplemented from AFXBaseTable.

getCurrentRow()

Returns the row index of the current item.

Reimplemented from AFXBaseTable.

getCurrentSortColumn()

Returns the current sort column or -1 if none.

getDefaultColumnWidth()

Returns the default column width, in pixels, of the table.

getDefaultHeight()

Returns the default height of the table.

Reimplemented from AFXBaseTable.

getDefaultRowHeight()

Returns the default row height, in pixels, of the table.

getDefaultWidth()

Returns the default width of the table.

Reimplemented from AFXBaseTable.

getFont()

Gets the font for all text items in the table.

Reimplemented from AFXBaseTable.

getGridColor()

Gets the color of the gridlines in the table.

Reimplemented from AFXBaseTable.

getItemBackColor(row, column)

Gets the background color of an item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemBoolValue(row, column)

Returns the value of an item of type BOOL.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemColor(row, column)

Returns the color of an item of type COLOR. The color is "As is", "Default", or a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C").

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemFloatValue(row, column)

Returns the value of an item of type FLOAT.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemIcon(row, column)

Returns the icon of an item of type ICON.

Argument	Type	Default	Description
row	Int		Row index of item.

column	Int		Column index of item.

getItemIntValue(row, column)

Returns the value of an item of type INT.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemListId(row, column)

Returns the list ID of an item of type LIST.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemListIndex(row, column)

Returns the list index (selection) of an item of type LIST.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemProvider()

Returns the item provider of this object.

getItemSelector(row, column)

Returns the message ID for an item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemTarget(row, column)

Returns the target for an item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemText(row, column)

Returns the text of an item of type TEXT.

Argument	Type	Default	Description

row	Int		Row index of item.
column	Int		Column index of item.

getItemTextColor(row, column)

Returns the text color of an item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemType(row, column)

Returns the type of an item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getItemValue(row, column)

Returns the text-form value of an item of any type.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

getLeadingColumns()

Returns the number of leading columns in the table.

Reimplemented from AFXBaseTable.

getLeadingFont()

Returns the font of the leading rows and columns.

getLeadingRows()

Returns the number of leading rows in the table.

Reimplemented from AFXBaseTable.

getListItemIcon(listId, index)

Returns the icon of the item at the specified index of the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list.
index	Int		Index into the list of the item to return.

getListItemText(listId, index)

Returns the text of the item at the specified index of the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list.
index	Int		Index into the list of the item to return.

getNumColumns()

Returns the number of columns in the table (including leading columns).

Reimplemented from AFXBaseTable.

getNumEmptyRowsAtBottom()

Returns the number of empty (non-trailing) rows at the bottom of the table.

getNumListItems(listId)

Returns the number of items in the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list.

getNumRows()

Returns the number of rows in the table (including leading rows).

Reimplemented from AFXBaseTable.

getPopupOptions()

Returns the flags for the menu items to be displayed in the popup menu.

getPreferredColumnWidth(column, excludeTitle=True)

Returns the width required for a column.

Argument	Type	Default	Description
column	Int		Column index.
excludeTitle	Bool	True	Specify True to ignore the width of leading and trailing items of the column.

getPreferredRowHeight(row)

Returns the height required for a row (useful for multi-line labels).

Argument	Type	Default	Description
row	Int		Row index.

getRowAtY(y)

Returns the row at the specified y coordinate; returns -1 if y is outside of the table.

Argument	Type	Default	Description
y	Int		Y coordinate.

getRowHeight(row)

Returns the height, in pixels, of the specified row.

Argument	Type	Default	Description
row	Int		Row index.

getSelBackColor()

Gets the selection background color of the table.

Reimplemented from AFXBaseTable.

getSelTextColor()

Gets the selection text color of the table.

Reimplemented from AFXBaseTable.

getStretchableColumn()

Returns the index of the stretchable column.

getTableStyle()

Returns the options related only to the table.

getVisibleColumns()

Gets the number of visible columns in the table.

Reimplemented from AFXBaseTable.

getVisibleRows()

Gets the number of visible rows in the table.

Reimplemented from AFXBaseTable.

insertColumns(startColumn, numColumns=1, notify=False)

Inserts columns at the specified location.

Argument	Type	Default	Description
startColumn	Int		Starting column.
numColumns	Int	1	Number of columns to insert.
notify	Bool	False	Specify True to notify target of the insertion.

insertRows(startRow, numRows=1, notify=False)

Inserts rows at the specified location.

Argument	Type	Default	Description
startRow	Int		Starting row.

numRows	Int	1	Number of rows to insert.
notify	Bool	False	Specify True to notify target of the insertion.

isAnyItemInColumnSelected(column)

Returns True if any item in the column is selected.

Argument	Type	Default	Description
column	Int		Column index.

isAnyItemInRowSelected(row)

Returns True if any item in the row is selected.

Argument	Type	Default	Description
row	Int		Row index.

isColumnSelected(column)

Returns True if all items in the column are selected.

Argument	Type	Default	Description
column	Int		Column index.

isColumnSortable(column)

Returns True if the items of the given column can be sorted.

Argument	Type	Default	Description
column	Int		Column index.

isItemBool(row, column)

Returns True if the specified item is of type BOOL.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

isItemColor(row, column)

Returns True if the specified item is of type COLOR.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

isItemEditable(row, column)

Returns True if the item is editable.

Argument	Type	Default	Description

row	Int		Row index of item.
column	Int		Column index of item.

isItemEmpty(row, column)

Returns True if the specified item does not have a value. This method checks the actual contents of the specified item and does not account for the empty-item policy for the item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

isItemIcon(row, column)

Returns True if the specified item is of type ICON.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

isItemList(row, column)

Returns True if the specified item is of type LIST.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

isItemSelected(row, column)

Returns True if the specified item is selected.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

isItemText(row, column)

Returns True if the specified item is of type TEXT.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

isItemVisible(row, column)

Returns True if the specified item is visible.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

isRowSelected(row)

Returns True if all items in the row are selected.

Argument	Type	Default	Description
row	Int		Row index.

killFocus()

Removes the focus from this window.

Reimplemented from AFXBaseTable.

killSelection(notify=False)

Deselects all items of the table; returns True if this method deselects any items that were selected.

Argument	Type	Default	Description
notify	Bool	False	Specify True to notify target of the selection change (with a SEL_DESELECTED message).

layout()

Lays out the table contents.

Reimplemented from AFXBaseTable.

makePositionVisible(row, column)

Scrolls to make the specified row, column fully visible.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

makeRowVisible(row)

Scrolls vertically (only) to make the specified row fully visible.

Argument	Type	Default	Description
row	Int		Row index.

moveContents(x, y)

Scrolls the contents.

Argument	Type	Default	Description
x	Int		Distance scrolled in X direction.
y	Int		Distance scrolled in Y direction.

recalc()

Propagates size changes.

Reimplemented from AFXBaseTable.

removeListItem(listId, index)

Removes the item at the specified index from the specified table list; returns the number of items remaining in list.

Argument	Type	Default	Description
listId	Int		ID of the list to remove from.
index	Int		Index of the list item to remove.

selectItem(row, column)

Selects the specified item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

selectRow(row)

Selects all items in the row.

Argument	Type	Default	Description
row	Int		Row index.

setColumnBoolIcons(column, trueIcon=None, falseIcon=None)

Sets the True and False icons of all existing and future items in a column of type BOOL. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
trueIcon	FXIcon	None	Icon displayed when value is True; 0 = default icon.
falseIcon	FXIcon	None	Icon displayed when value is False; 0 = default icon.

setColumnBoolValue(column, value)

Sets the value of all existing and future items in a column of type BOOL. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
value	Bool		Specify True or False.

setColumnColor(column, color)

Sets the color of all existing and future items in a column of type COLOR. The color can be "As is", "Default", a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C"), or a pre-defined color name (e.g., "Red"). Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
color	String		Color.

setColumnColorItemDefault(column, color)

Sets the color of the color item in the flyout menu for all existing and future items that display "As is" or "Default" in a column of type COLOR. The color is either a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C") or a pre-defined color name (e.g., "Red"). Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
color	String		Color.

setColumnColorOptions(column, opts)

Sets the color flyout options for all existing and future items in a column of type COLOR. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
opts	Int		Options (see ColorFlyoutOptions).

setColumnEditable(column, editable)

Sets the editability of all existing and future items in a column. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
editable	Bool		Specify True for editable, False for read-only.

setColumnFloatValue(column, value)

Sets the value of all existing and future items in a column of type FLOAT. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
value	Float		Floating-point value.

setColumnIcon(column, icon=None)

Sets the icon of all existing and future items in a column of type ICON. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
icon	FXIcon	None	Icon.

setColumnIntValue(column, value)

Sets the value of all existing and future items in a column of type INT. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description

column	Int		Column index.
value	Int		Integer value.

setColumnJustify(column, justify)

Sets the justification of all existing and future items in a column. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
justify	Int		Justification (see ItemJustify).

setColumnListId(column, listId)

Sets the list ID of all existing and future items in a column of type LIST. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
listId	Int		List ID.

setColumnListIndex(column, index)

Sets the list index (selection) of all existing and future items in a column of type LIST. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
index	Int		Index of item to be selected.

setColumnSortable(column, sortable)

Sets a column to be sortable or not. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
sortable	Bool		Specify True for sortable, False for otherwise.

setColumnSortOrder(column, order)

Sets the sort order of the given column. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
order	Int		Sort order (see SortOrder).

setColumnText(column, text)

Sets the text of all existing and future items in a column of type TEXT. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
text	String		Text.

setColumnType(column, type)

Sets the type of a column. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
type	Int		Type (see Flags for item types).

setColumnWidth(column, width)

Sets the width, in pixels, of the specified column. Specifying -1 for the column will change all non-leading and non-trailing columns in the table and set the default for the table. Specify -1 for the width will resize each specified column to best fit the width of the title(s) currently shown in its leading and trailing items.

Argument	Type	Default	Description
column	Int		Column index.
width	Int		Width in pixels.

setColumnWidthInChars(column, numChars)

Sets the width, in number of characters, of the specified column. Specifying -1 for the column will change all non-leading and non-trailing columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
numChars	Int		Width in number of characters.

setCurrentItem(row, column)

Sets the specified item to be the current item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.

setCurrentSortColumn(column)

Sets the current sort column. The given column must be sortable; otherwise the current sort column will not change.

Argument	Type	Default	Description
column	Int		Column index.

setDefaultBoolIcons(trueIcon=None, falseIcon=None)

Sets the default True and False icons for the table (0 = default icon).

Argument	Type	Default	Description
trueIcon	FXIcon	None	Icon displayed when value is True; 0 = default icon.

falseIcon	FXIcon	None	Icon displayed when value is False; 0 = default icon.

setDefaultBoolValue(value)

Sets the default bool value for the table.

Argument	Type	Default	Description
value	Bool		Specify True or False.

setDefaultColor(color)

Sets the default color for all items of type COLOR in the table. The color can be "As is", "Default", a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C"), or a pre-defined color name (e.g., "Red").

Argument	Type	Default	Description
color	String		Color.

setDefaultColumnWidth(width)

Sets the default width, in pixels, for all columns.

Argument	Type	Default	Description
width	Int		Width in pixels.

setDefaultFloatValue(value)

Sets the default floating-point value for the table.

Argument	Type	Default	Description
value	Float		Floating-point value.

setDefaultIntValue(value)

Sets the default integer value for the table.

Argument	Type	Default	Description
value	Int		Integer value.

setDefaultJustify(justify)

Sets the default justification for the table.

Argument	Type	Default	Description
justify	Int		Justification (see ItemJustify).

setDefaultRowHeight(height)

Sets the default height, in pixels, for all rows.

Argument	Type	Default	Description
height	Int		Height in pixels.

setDefaultText(text)

Sets the default text for the table.

Argument	Type	Default	Description
text	String		Text.

setDefaultType(type)

Sets the default type for the table.

Argument	Type	Default	Description
type	Int		Type (see Flags for item types).

setEmptyItemDefault(defaultVal)

Sets the default value (in text form) used for empty items of the table if its empty-item policy includes DEFAULT_IF_EMPTY.

Argument	Type	Default	Description
defaultVal	String		Default value in text form.

setEmptyItemPolicy(policy)

Sets the policy for handling empty items of the table (see EmptyItemPolicy).

Argument	Type	Default	Description
policy	Int		Flags for handling empty items (see EmptyItemPolicy).

setFont(font)

Sets the font for all text items in the table.

Argument	Type	Default	Description
font	FXFont		Font.

setGridColor(color)

Sets the color for the gridlines in the table.

Argument	Type	Default	Description
color	FXColor		Color.

setItemBackColor(row, column, color)

Sets the background color of an item using an FXColor.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
color	FXColor		Color index.

setItemBackColor(row, column, color)

Sets the background color of an item using a string.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
color	String		Color name.

setItemBoolIcons(row, column, trueIcon=None, falseIcon=None)

Sets the True and False icons of an item of type BOOL.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
trueIcon	FXIcon	None	Icon displayed when value is True; 0 = default icon.
falseIcon	FXIcon	None	Icon displayed when value is False; 0 = default icon.

setItemBoolValue(row, column, value)

Sets the value of an item of type BOOL.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
value	Bool		Specify True or False.

setItemColor(row, column, color)

Sets the color of an item of type COLOR. The color can be "As is", "Default", a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C"), or a pre-defined color name (e.g., "Red").

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
color	String		Color.

setItemEditable(row, column, editable)

Sets the editability of an item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
editable	Bool		Specify True for editable, False for read-only.

setItemFloatValue(row, column, value)

Sets the value of an item of type FLOAT.

Argument	Type	Default	Description

row	Int		Row index of item.
column	Int		Column index of item.
value	Float		Floating-point value.

setItemIcon(row, column, icon=None)

Sets the icon of an item of type ICON.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
icon	FXIcon	None	Icon.

setItemIntValue(row, column, value)

Sets the value of an item of type INT.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
value	Int		Integer value.

setItemJustify(row, column, justify)

Sets the justification of an item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
justify	Int		Justification (see ItemJustify).

setItemListId(row, column, listId)

Sets the list ID of an item of type LIST.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
listId	Int		List ID.

setItemListIndex(row, column, index)

Sets the list index (selection) of an item of type LIST.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
index	Int		Index of item to be selected.

setItemProvider(ip)

Sets the item provider of this object.

Argument	Type	Default	Description
ip	FXObject		Item provideer.

setItemSpan(row, column, numRows, numColumns)

Sets a leading item to span more than one row or column.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
numRows	Int		Number of rows to span.
numColumns	Int		Number of columns to span.

setItemTarget(row, column, tgt, msg=0)

Sets the target and message ID for an item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
tgt	FXObject		Target.
msg	Int	0	Message ID.

setItemText(row, column, text)

Sets the text of an item of type TEXT.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
text	String		Text.

setItemTextColor(row, column, color)

Sets the text color of an item using an FXColor.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
color	FXColor		Color index.

setItemTextColor(row, column, color)

Sets the text color of an item using a string.

Argument	Type	Default	Description

row	Int		Row index of item.
column	Int		Column index of item.
color	String		Color name.

setItemType(row, column, type)

Sets the type of an item.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
type	Int		Type (see Flags for item types).

setItemValue(row, column, valueText)

Sets the value of an item of any type that can interpret a text string for its value. Returns True if the value of the specified item is set successfully.

Argument	Type	Default	Description
row	Int		Row index of item.
column	Int		Column index of item.
valueText	String		Text-form value of item.

setLeadingColumnLabels(str, column=0)

Sets the labels of a leading column. Note: this API must be used to set the header column labels, otherwise labels will be overwritten by auto-numbering.

Argument	Type	Default	Description
str	String		Tab "\t" delimited list, can also contain newline characters indicating that label contains multiple lines of text (e.g. "Young's\nModulus\tPoisson's\nRatio").
column	Int	0	Column, this column must have previously been specified as a leading column (see setLeadingColumns).

setLeadingColumns(numColumns)

Sets the number of leading columns.

Argument	Type	Default	Description
numColumns	Int		Number of columns.

setLeadingFont(font)

Sets the font of the leading rows and columns.

Argument	Type	Default	Description
font	FXFont		Font.

setLeadingRowLabels(str, row=0)

Set the labels of a leading row. Note: this API must be used to set the header row labels, otherwise labels will be overwritten by

auto-numbering.

Argument	Type	Default	Description
str	String		Tab "\t" delimited list, can also contain newline characters indicating that label contains multiple lines of text (e.g. "Young's\nModulus\tPoisson's\nRatio").
row	Int	0	Row, this row must have previously been specified as a leading row (see setLeadingRows).

setLeadingRows(numRows)

Sets the number of leading rows.

Argument	Type	Default	Description
numRows	Int		Number of rows.

setListMaxVisible(maxVisible)

Sets the maximum number of visible items for all table lists.

Argument	Type	Default	Description
maxVisible	Int		Maximum number of visible items.

setPopupOptions(opts)

Sets the menu items to be displayed in the popup menu.

Argument	Type	Default	Description
opts	Int		Options.

setRowHeight(row, height)

Sets the height, in pixels, of the specified row.

Argument	Type	Default	Description
row	Int		Row index.
height	Int		Height in pixels.

setSelBackColor(color)

Sets the selection background color of the table.

Argument	Type	Default	Description
color	FXColor		Color index.

setSelTextColor(color)

Sets the selection text color of the table.

Argument	Type	Default	Description
color	FXColor		Color index.

setStretchableColumn(column)

Sets the stretchable column. (This method only works for the last column.)

Argument	Type	Default	Description
column	Int		Column index.

setTableSize(numRows, numColumns, notify=False)

Sets the size of the table.

Argument	Type	Default	Description
numRows	Int		Number of rows.
numColumns	Int		Number of columns.
notify	Bool	False	Specify True to notify target of change.

setTableStyle(style)

Sets the table options.

Argument	Type	Default	Description
style	Int		Style flag (see Flags for AFX table options).

setVisibleColumns(visibleColumns)

Sets the number of visible columns in the table.

Argument	Type	Default	Description
visibleColumns	Int		Number of visible columns.

setVisibleRows(visibleRows)

Sets the number of visible rows in the table.

Argument	Type	Default	Description
visibleRows	Int		Number of visible rows.

shadeReadOnlyItems(shadeItems)

Makes the table to use a different, typically shaded, background color for read-only items if True is passed to the method. The table would use the same regular background color for both editable and read-only items if False is passed to the method.

Argument	Type	Default	Description
shadeItems	Bool		Specify True to use a different background color for read-only items.

showHorizontalGrid(on=True)

Controls the display of horizontal gridlines in the table.

Argument	Type	Default	Description
on	Bool	True	True if gridlines should be displayed.

showVerticalGrid(on=True)

Controls the display of vertical gridlines in the table.

Argument	Type	Default	Description
on	Bool	True	True if gridlines should be displayed.

Class flags

Flags for popup menu items.

POPUP_NONE	Popup not displayed.
POPUP_CUT	Display "Cut" menu item.
POPUP_COPY	Display "Copy" menu item.
POPUP_PASTE	Display "Paste" menu item.
POPUP_EDIT	Convenience flag for specifying multiple menu items.
POPUP_INSERT_ROW	Display "Insert Row Before/After" menu items.
POPUP_INSERT_COLUMN	Display "Insert Column Before/After" menu items.
POPUP_DELETE_ROW	Display "Delete Rows" menu item.
POPUP_DELETE_COLUMN	Display "Delete Columns" menu item.
POPUP_CLEAR_CONTENTS	Display "Clear Contents" and "Clear Table" menu items.
POPUP_MODIFY_ROW	Convenience flag for specifying multiple menu items.
POPUP_MODIFY_COLUMN	Convenience flag for specifying multiple menu items.
POPUP_MODIFY	Convenience flag for specifying multiple menu items.
POPUP_READ_FROM_FILE	Display "Read from File" menu item.
POPUP_WRITE_TO_FILE	Display "Write to File" menu item.
POPUP_FILE	Display "Read from file" and "Write to file" menu items.
POPUP_ALL	Display all menu items.

Message ID's.

ID_CUT_SEL	ID for the Cut button.
ID_COPY_SEL	ID for the Copy button.
ID_PASTE_SEL	ID for the Paste button.
ID_ADD_COLUMN	ID for the Insert Column buttons.
ID_ADD_ROW	ID for the Insert Row buttons.
ID_DELETE_COLUMNS	ID for the Delete Columns button.
ID_DELETE_ROWS	ID for the Delete Rows button.
ID_CLEAR_SEL	ID for the Clear Contents button.
ID_CLEAR_TABLE	ID for the Clear Table button.
ID_READ_SEL	ID for the Read from File button.
ID_WRITE	ID for the Write to File button.
ID_FILE_DB	ID for the Read Data from ASCII File dialog box used by the Read from File button.
ID_READ_WARNING	ID for the "Data were truncated" warning dialog box.
ID_WRITE_FILE_DB	ID for the file selection dialog box used by the Write to File button.
ID_CONFIRM_WRITE	ID for the "OK to overwrite?" warning dialog box.

IGNORE_BOTTOM_EMPTY_ROWS not exposed in python

Flags for color flyouts (used for items of type COLOR).

COLOR_INCLUDE_COLOR_ONLY	Color item has no As Is and Default in its flyout.
COLOR_INCLUDE_AS_IS	Color item has As Is in its flyout.
COLOR_INCLUDE_DEFAULT	Color item has Default in its flyout.
COLOR_INCLUDE_ALL	Color item has both As Is and Default in its flyout. This is the default option.

Flags for how empty items should be handled by the AFXTable value-retrieving and error-checking API's

DISALLOW_EMPTY	Disallow an item to be empty (default).
ALLOW_EMPTY	Allow an item to be empty.
DEFAULT_IF_EMPTY	Allow an item to be empty and use its default value for the item.
KEEP_BOTTOM_EMPTY_ROWS	Include empty rows at the bottom of the table.

Flags for item alignment.

LEFT	Left justified.
RIGHT	Right justified.
CENTER	Center justified (horizontal).
TOP	Top justified.
BOTTOM	Bottom justified.
MIDDLE	Middle justified (vertical).

Flags for item types.

TEXT	Item accepts a text string via a text field.
FLOAT	Item accepts a floating-point number via a text field.
INT	Item accepts an integer via a text field.
LIST	Item accepts input from a list.
BOOL	Item is a boolean; displayed as an icon.
ICON	Item displays an icon and does not accept input.
COLOR	Item accepts color selection via a color flyout.

Flags for real formats.

GENERAL	General.
SCIENTIFIC	Scientific.
AUTOMATIC	Automatic.

Flags for sorting items in a column.

SORT_INACTIVE	Sort currently not active for the column.
SORT_ASCENDING	Sort items of the column in the ascending order.
SORT_DESCENDING	Sort items of the column in the descending order.

Global flags

Flags for AFX table options.

AFXTABLE_COLUMN_RESIZABLE	Allow users to resize columns.
AFXTABLE_ROW_RESIZABLE	Allow users to resize rows.
AFXTABLE_RESIZE	Allow users to resize rows and columns.
AFXTABLE_NO_COLUMN_SELECT	Disallow column selections (selecting a header/footer item in a column selects the whole column).

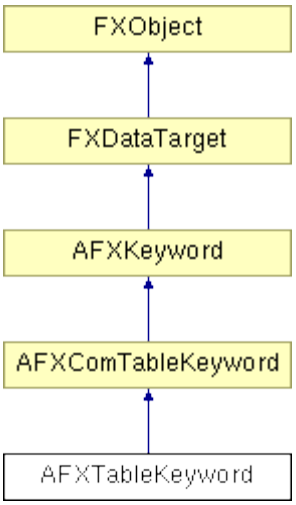
AFXTABLE_NO_ROW_SELECT	**Disallow row selections (selecting a header/footer item in a row selects the whole row).**
AFXTABLE_ROW_MODE	**Selecting any item in a row selects the whole row.**
AFXTABLE_EXTENDED_SELECT	**Use extended selection mode that allows multiple items to be selected and allows users to drag-select a range of items.**
AFXTABLE_SINGLE_SELECT	**Use single selection mode that allows up to one item to be selected.**
AFXTABLE_BROWSE_SELECT	**Use browse selection mode that enforces one single item to be selected at all times.**
AFXTABLE_EDITABLE	**Table is editable.**
AFXTABLE_NORMAL	**Default table options--use extended selection mode, columns are resizable, and layout fills both X and Y directions.**

Flags for the table's list (for items of type LIST).

AFXTABLE_LIST_PRESELECT_NONE	**Do not pre-select any list item.**

AFXTableKeyword

This class is designed for command keywords that have table values.



AFXTableKeyword(command, name, isRequired=False, minLength=0, maxLength=-1, opts=0)

Constructor.

Argument	Type	Default	Description
command	AFXGuiCommand		Host command.
name	String		Keyword name.
isRequired	Bool	False	True if this keyword is a required argument.
minLength	Int	0	Minimum (and default) row length.
maxLength	Int	-1	Maximum row length (-1 => unlimited).
opts	Int	0	Options.

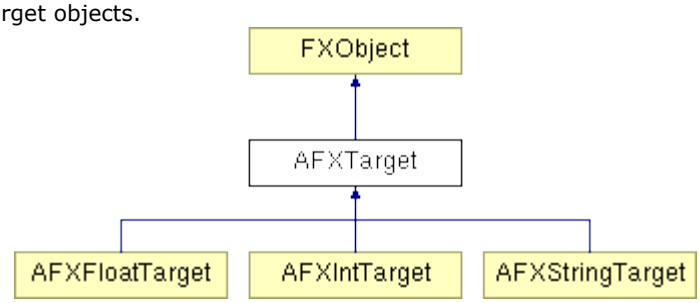
getTypeName()

Returns the name of the table keyword type.

Reimplemented from AFXComTableKeyword.

AFXTarget

This class is the base class for all target objects.



AFXTarget()

Constructor.

connect(value)

Associates the data with a string variable.

Argument	Type	Default	Description
value	String		Variable to be associated with.

connect(value)

Associates the data with a floating-point variable.

Argument	Type	Default	Description
value	Float		Variable to be associated with.

connect(value)

Associates the data with an integer variable.

Argument	Type	Default	Description
value	Int		Variable to be associated with.

getSelector()

Returns the message ID of this target object.

getTarget()

Returns the target of this target object.

getType()

Returns the target type; this method is deprecated in Abaqus 6.6, and its use should be replaced by getTypeName().

getTypeName()

Returns the name of the target type.

Implemented in AFXFloatTarget, AFXIntTarget, and AFXStringTarget.

setSelector(msgId)

Sets the message ID of this target object.

Argument	Type	Default	Description
msgId	Int		Message ID.

setTarget(target)

Sets the target of this target object.

Argument	Type	Default	Description
target	FXObject		Target.

Class flags

Message ID's.

ID_LAST **Last ID.**

Flags for the type of target.

UNSPECIFIED **Unspecified.**

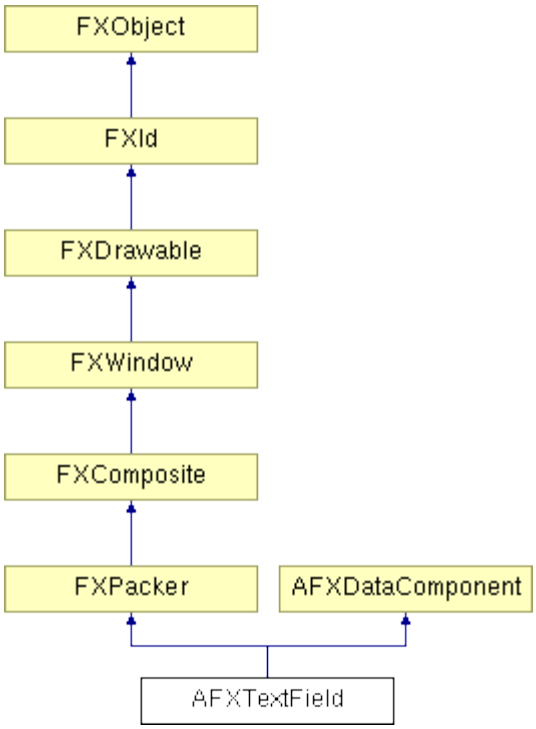
INT **Integer.**

FLOAT **Float.**

STRING **String.**

AFXTextField

This class contains a label that precedes a text field that allows the user to enter in text.



AFXTextField(p, ncols, labelText, tgt=None, sel=0, opts=AFXTEXTFIELD_STRING, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
ncols	Int		Number of columns.
labelText	String		Label string.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	AFXTEXTFIELD_STRING	Options and hints.
x	Int	0	X coordinate of origin.
y	Int	0	Y coordinate of origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	DEFAULT_PAD	Left padding (margin).
pr	Int	DEFAULT_PAD	Right padding (margin).
pt	Int	DEFAULT_PAD	Top padding (margin).
pb	Int	DEFAULT_PAD	Bottom padding (margin).

create()

Creates the text field.

Reimplemented from FXComposite.

disable()

Disables the text field.

Reimplemented from FXWindow.

enable()

Enables the text field.

Reimplemented from FXWindow.

getCheck()

Returns the state of the check button or the radio button.

getCursorPos()

Returns the cursor position.

getDefaultWidth()

Returns the default width of the text field.

Reimplemented from FXPacker.

getExponentType()

Returns the exponent type of the text field for real and complex types.

getImaginaryText()

Returns the imaginary text for the complex text field.

getJustify()

Returns the text justification mode.

getLabelFont()

Returns the label's font.

getLabelText()

Returns the label text.

getNumColumns()

Returns the number of columns.

getPrecision()

Returns the precision of the text field for real and complex types.

getText()

Returns the text.

getValueType()

Returns the value type (AFXTEXTFIELD_FLOAT, etc.) of the text field.

isEditable()

Returns True if the text in the input field may be edited.

isReadOnlyState()

Returns True if the text field appears in the read-only state.

isVerticalLayout()

Returns True if the layout orientation is vertical.

setCheck(state)

Sets the state of the check button or the radio button.

Argument	Type	Default	Description
state	Bool		Check state.

setCheckButtonSelector(sel)

Sets the message ID of the check button or the radio button.

Argument	Type	Default	Description
sel	Int		Selector.

setCheckButtonTarget(checkVal=False)

Sets the message target of the check button or the radio button.

Argument	Type	Default	Description
checkVal	Bool	False	Check state.

setCursorPos(pos)

Sets the cursor position.

Argument	Type	Default	Description
pos	Int		Position.

setEditable(edit=True)

Sets the editable state for the text field.

Argument	Type	Default	Description
edit	Bool	True	If True, text can be edited.

setExponentType(e)

Sets the exponent type of the text field for real and complex types.

Argument	Type	Default	Description
e	FXExponent		Exponent type.

setFocus()

Moves the focus to the text field.

Reimplemented from FXWindow.

setFocusAndSelection()

Sets the focus to the input field and selects its contents.

setFocusToCheckButton()

Moves the focus to the check button or the radio button (if existed) of the widget.

setFocusToImaginaryTextField()

Moves the focus to the input field for the imaginary part.

setFocusToTextField()

Moves the focus to the input field of the widget.

setImaginaryFocusAndSelection()

Sets the focus to the input field for the imaginary part and selects its contents.

setImaginaryText(text)

Sets the imaginary text for the complex text field.

Argument	Type	Default	Description
text	String		Imaginary text field text.

setJustify(mode)

Sets the text justification mode.

Argument	Type	Default	Description
mode	Int		Justification flag.

setLabelFont(fnt)

Sets the label's text font.

Argument	Type	Default	Description
fnt	FXFont		Label font.

setLabelText(txt)

Sets the label text.

Argument	Type	Default	Description
txt	String		Label text.

setNumColumns(cols)

Sets the number of columns. Note: The column width is based on the width of "m" of the font used.

Argument	Type	Default	Description
cols	Int		Number of columns.

setPrecision(p)

Sets the precision of the text field for real and complex types. Limitation: If an AFXTextField widget uses an AFXFloatKeyword object as its target, the widget must have AFXTEXTFIELD_FLOAT as one of its options for the precision setting to take effect.

Argument	Type	Default	Description
p	Int		Precision.

setReadOnlyState(readonly=True)

Sets the read-only state of the text field.

Argument	Type	Default	Description
readonly	Bool	True	Read-only state.

setSelection(pos, len)

Select the specified number of characters starting at given position.

Argument	Type	Default	Description
pos	Int		Position.
len	Int		Length.

setText(text)

Sets the text in the input field.

Argument	Type	Default	Description
text	String		Text field text.

setValueType(type)

Sets the value type (AFXTEXTFIELD_FLOAT, etc.) of the text field.

Argument	Type	Default	Description

type	Int		Value type.

setVerticalLayout(vertical)

Sets the layout orientation of the text field.

Argument	Type	Default	Description
vertical	Bool		Vertical flag.

Class flags

Message ID's.

ID_SETIMAGINARYVALUE **ID for setting imaginary values.**

ID_GETIMAGINARYVALUE **ID for getting imaginary values.**

ID_BUTTON **ID for the check/radio button.**

ID_TEXT **ID for the text field.**

ID_IMG_TEXT **ID for the text field with imaginary part.**

Global flags

Flags for AFX textfield options.

AFXTEXTFIELD_STRING **Value field is a string.**

AFXTEXTFIELD_INTEGER **Value field is an integer.**

AFXTEXTFIELD_FLOAT **Value field is a double.**

AFXTEXTFIELD_COMPLEX **Value fields consist of the real and imaginay components of a complex number.**

AFXTEXTFIELD_CHECKBUTTON **Use a check button instead of a label.**

AFXTEXTFIELD_RADIOBUTTON **Use a radio button instead of a label.**

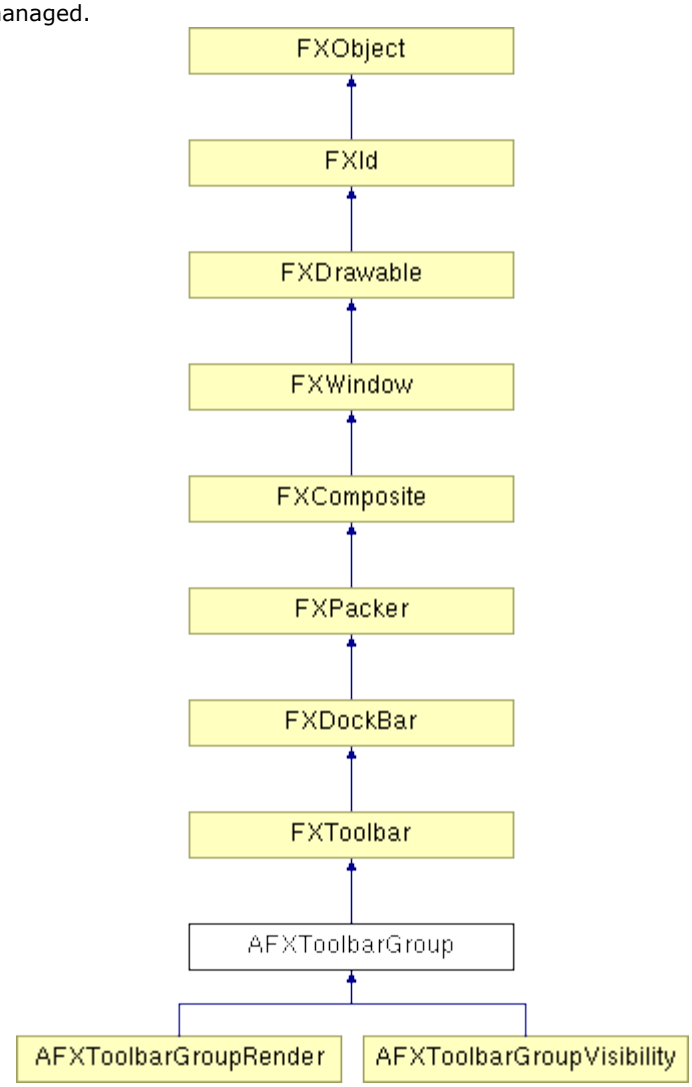
AFXTEXTFIELD_VERTICAL **Orient label or button above text field.**

AFXTEXTFIELD_READONLY **Configure text field to the read-only state.**

AFXTEXTFIELD_IME **Allow IME (Japanese etc.) input.**

AFXToolbarGroup

This class creates a container to be used for groups in the toolbar. It creates a vertical separator after the group. It will use utility methods so the group is correctly managed.



AFXToolbarGroup(owner, name='', title='')

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Creator of the group.
name	String	''	English toolset name.
title	String	''	Name appearing in the title bar when the group is floating.

getDefaultHeight()

Returns the default height.

Reimplemented from FXToolbar.

getDefaultWidth()

Returns the default width.

Reimplemented from FXToolbar.

getName()

Returns the English identifier for the group.

getOwner()

Returns the creator of the group.

Reimplemented from FXWindow.

getTitle()

Returns the name appearing in the title bar when the group is floating.

hide()

Hide this window.

Reimplemented from FXWindow.

Reimplemented in AFXToolbarGroupRender, and AFXToolbarGroupVisibility.

isActive()

Return True if the window is active.

Reimplemented from FXWindow.

layout()

Calculates layout.

Reimplemented from FXToolbar.

setName(name)

Sets the English identifier for the group.

Argument	Type	Default	Description
name	String		

setTitle(title)

Sets the name appearing in the title bar when the group is floating.

Argument	Type	Default	Description
title	String		

show()

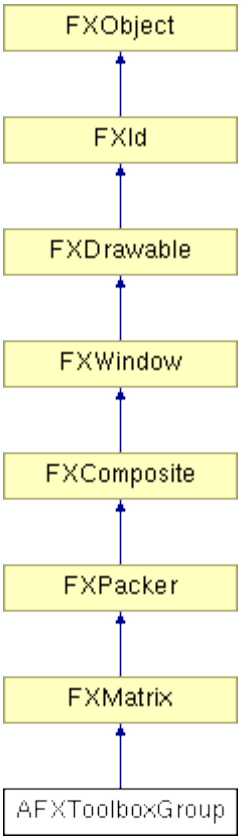
Show this window.

Reimplemented from FXWindow.

Reimplemented in AFXToolbarGroupRender, and AFXToolbarGroupVisibility.

AFXToolboxGroup

This class creates a container to be used for groups in the toolbox. It will use utility methods so the group is correctly managed for modules and toolsets.



AFXToolboxGroup(owner, parent=None)

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Creator of the toolbox group.
parent	FXComposite	None	Parent widget.

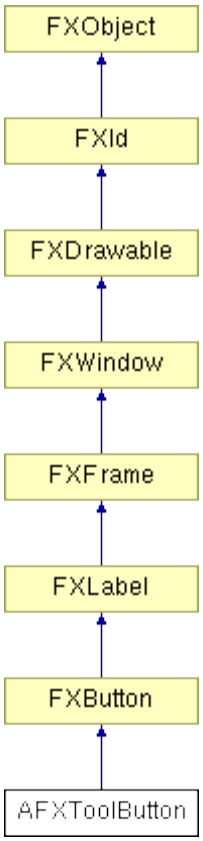
getOwner()

Returns the owner of the toolbox group.

Reimplemented from FXWindow.

AFXToolButton

This class contains a button for use in the tool bar or the toolbox.



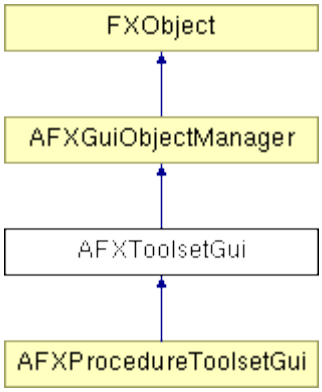
AFXToolButton(p, label, icon=None, tgt=None, sel=0, asToggle=True)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
label	String		Label for the button.
icon	FXIcon	None	Icon for the button.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
asToggle	Bool	True	Allow toggle off behavior.

AFXToolsetGui

This is the base class for toolset GUIs and provides an interface for managing the toolset's GUI items. It provides a mechanism to add in menubar, toolbar, and toolbox GUI items.



AFXToolsetGui(toolsetName)

Constructor.

Argument	Type	Default	Description
toolsetName	String		Toolset name passed in from derived toolsets.

activate()

Activates the toolset (if there is no mode factory, then this method need not be redefined).

deactivate()

Deactivates the toolset (if there is no mode factory, then this method need not be redefined).

getToolsetName()

Returns the name of the toolset given on construction.

hide(location)

Hides the GUI components in the menubar, toolbar, and toolbox.

Argument	Type	Default	Description
location	Int		Flags indicating the location where GUI components are placed. Possible values are GUI_IN_NONE, GUI_IN_MENUBAR, GUI_IN_TOOL_PANE, GUI_IN_TOOLBAR, and GUI_IN_TOOLBOX.

show(location)

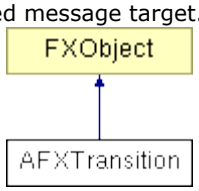
Shows the GUI components in the menubar, toolbar, and toolbox.

Argument	Type	Default	Description
location	Int		Flags indicating the location where GUI components are placed. Possible values are GUI_IN_NONE, GUI_IN_MENUBAR, GUI_IN_TOOL_PANE, GUI_IN_TOOLBAR, and

			GUI_IN_TOOLBOX.

AFXTransition

This class is designed for the finite state transition that the GUI (mostly the dialog boxes) can define to perform actions according to state changes. The first three arguments of the constructors (keyword, op, and refValue) define an expression (keyword.getValue() op refValue). The current value of the keyword is compared with the reference value. When the expression evaluates to True, a message with the given selector will be sent to the specified message target.



AFXTransition(boolKeyword, op, refValue, tgt, sel, ptr=None)

Constructor.

Argument	Type	Default	Description
boolKeyword	AFXBoolKeyword		Keyword.
op	Operator		Operator type.
refValue	Bool		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

AFXTransition(floatKeyword, op, refValue, tgt, sel, ptr=None)

Constructor.

Argument	Type	Default	Description
floatKeyword	AFXFloatKeyword		Keyword.
op	Operator		Operator type.
refValue	Float		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

AFXTransition(intKeyword, op, refValue, tgt, sel, ptr=None)

Constructor.

Argument	Type	Default	Description
intKeyword	AFXIntKeyword		Keyword.
op	Operator		Operator type.
refValue	Int		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

AFXTransition(togKeyword, op, refValue, tgt, sel, ptr=None)

Constructor.

Argument	Type	Default	Description
togKeyword	AFXTogglableKeyword		Keyword.
op	Operator		Operator type.
refValue	Bool		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

AFXTransition(floatTarget, op, refValue, tgt, sel, ptr=None)

Constructor.

Argument	Type	Default	Description
floatTarget	AFXFloatTarget		Target.
op	Operator		Operator type.
refValue	Float		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

AFXTransition(intTarget, op, refValue, tgt, sel, ptr=None)

Constructor.

Argument	Type	Default	Description
intTarget	AFXIntTarget		Target.
op	Operator		Operator type.
refValue	Int		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

process(sender)

Returns True and sends a message if the expression defined by the constructor arguments evaluates to True; returns False without performing any actions if otherwise.

Argument	Type	Default	Description
sender	FXObject		Message sender.

Class flags

Flags for specifying transition operators.

EQ **Equal to.**

NE **Not equal to.**

LT **Less than.**

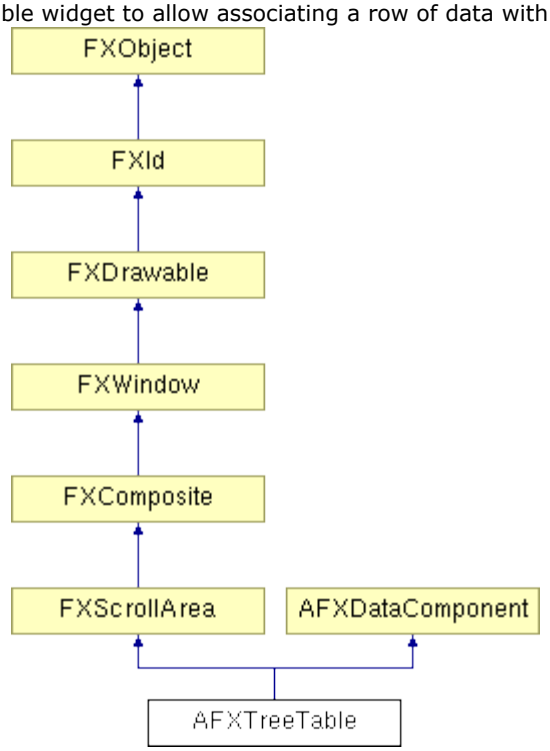
LE **Less than or equal to.**

GT **Greater than.**

GE **Greater than or equal to.**

AFXTreeTable

This class combines a tree widget with a table widget to allow associating a row of data with an item in a tree.



AFXTreeTable(p, numVisItems, numVisColumns, numColumns, tgt=None, sel=0, opts=AFXTREETABLE_NORMAL, x=0, y=0, w=0, h=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
numVisItems	Int		Number of items to display.
numVisColumns	Int		Number of columns in the table to display.
numColumns	Int		Number of columns in the table.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.
opts	Int	AFXTREETABLE_NORMAL	Options and hints.
x	Int	0	X coordinate of the origin.
y	Int	0	Y coordinate of the origin.
w	Int	0	Width of the table widget.
h	Int	0	Height of the table widget.

addItemAfter(other, text, oi=None, ci=None, notify=False)

Appends a new tree item with the given text and optional icon after the other tree item.

Argument	Type	Default	Description
other	FXTreeItem		

text	String		
oi	FXIcon	None	
ci	FXIcon	None	
notify	Bool	False	

addItemAfter(other, item, notify=False)

Appends the new tree item after the other tree item.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

addItemBefore(other, text, oi=None, ci=None, notify=False)

Prepends a new tree item with the given text and optional icon prior to the other item.

Argument	Type	Default	Description
other	FXTreeItem		
text	String		
oi	FXIcon	None	
ci	FXIcon	None	
notify	Bool	False	

addItemBefore(other, item, notify=False)

Prepends the new item prior to the other tree item.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

addItemFirst(p, text, oi=None, ci=None, notify=False)

Prepends a new tree item with the given text and optional icon as the first child of the parent.

Argument	Type	Default	Description
p	FXTreeItem		
text	String		
oi	FXIcon	None	
ci	FXIcon	None	
notify	Bool	False	

addItemFirst(p, item, notify=False)

Prepends the new tree item as first child of parent.

Argument	Type	Default	Description
p	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

addItemLast(p, text, oi=None, ci=None, notify=False)

Appends a new tree item with the given text and optional icon as the last child of the parent.

Argument	Type	Default	Description
p	FXTreeItem		
text	String		
oi	FXIcon	None	
ci	FXIcon	None	
notify	Bool	False	

addItemLast(p, item, notify=False)

Appends the new tree item as the last child of the parent.

Argument	Type	Default	Description
p	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

addList(text, opts=AFXTREETABLE_LIST_NORMAL)

Adds a list that have only text items to the table and returns the list ID. The text strings of the list items are delimited by tab "\t" characters in the given text. The list is used by items of type LIST.

Argument	Type	Default	Description
text	String		Tab "\t" delimited text string (e.g. "0\t50\t100\t150").
opts	Int	AFXTREETABLE_LIST_NORMAL	Options.

addList(opts=AFXTREETABLE_LIST_NORMAL)

Adds a list to the table and returns the list ID. The list is used by items of type LIST.

Argument	Type	Default	Description
opts	Int	AFXTREETABLE_LIST_NORMAL	List flag.

appendListItem(listId, text, icon=None)

Appends an item to the specified table list; returns the index of the new item.

Argument	Type	Default	Description
listId	Int		ID of the list to append to.
text	String		Item's text.

icon	FXIcon	None	Item's icon.

beginEdit(item, column)

Sets the specified item in edit mode if the item is editable.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.

cancelEdit()

Cancels the edit mode.

clearContents(startItem, startColumn, endItem, endColumn, clearEditableOnly=True)

Clears the text in the items in the specified range.

Argument	Type	Default	Description
startItem	FXTreeItem		Tree item in which to start clearing.
startColumn	Int		Column in which to start clearing.
endItem	FXTreeItem		Tree item in which to end clearing.
endColumn	Int		Column in which to end clearing.
clearEditableOnly	Bool	True	Specify True to clear the text of editable items only.

clearItems(notify=False)

Removes all tree items and table rows.

Argument	Type	Default	Description
notify	Bool	False	

clearListItems(listId)

Removes all items from the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list to clear.

closeItem(item, notify=False)

Closes the specified item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

collapseTree(item, notify=False)

Collapses the specified item to hide its children.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

deleteColumns(startColumn, numColumns=1, notify=False)

Deletes columns starting at the specified column.

Argument	Type	Default	Description
startColumn	Int		Starting column.
numColumns	Int	1	Number of columns to delete.
notify	Bool	False	Specify True to notify target of the deletion.

deselectItem(item, column, notify=False)

Deselects the specified item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.
notify	Bool	False	

deselectRow(item, notify=False)

Deselects all items in the row.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
notify	Bool	False	

expandTree(item, notify=False)

Expands the specified item to show its children.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

getColumnWidth(column)

Returns the width, in pixels, of the specified column.

Argument	Type	Default	Description
column	Int		Column index.

getCurrentColumn()

Returns the column index of the current item.

getCurrentItem()

Returns the current item, if any.

getDefaultColumnWidth()

Returns the default column width, in pixels, of the table.

getDefaultHeight()

Return default height.

Reimplemented from FXScrollArea.

getDefaultWidth()

Return default width.

Reimplemented from FXScrollArea.

getFirstItem()

Returns the first root tree item.

getItemBoolValue(item, column)

Returns the value of a table item of type BOOL.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemCheck(item)

Returns the item checked state.

Argument	Type	Default	Description
item	FXTreeItem		

getItemClosedIcon(item)

Returns the tree item's closed icon.

Argument	Type	Default	Description
item	FXTreeItem		

getItemColor(item, column)

Returns the color of a table item of type COLOR. The color is "As is", "Default", or a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C").

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemFloatValue(item, column)

Returns the value of a table item of type FLOAT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemFormat(item, column)

Returns the format of a table item of type REAL (see RealFormat).

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemIcon(item, column)

Returns the icon of a table item of type ICON.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemIntValue(item, column)

Returns the value of a table item of type INT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemListId(item, column)

Returns the list ID of a table item of type LIST.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemListIndex(item, column)

Returns the list index (selection) of a table item of type LIST.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemNumDigits(item, column)

Returns the number of digits to the left of the decimal point for a table item of type REAL.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemOpenIcon(item)

Returns the tree item's open icon.

Argument	Type	Default	Description
item	FXTreeItem		

getItemPrecision(item, column)

Returns the precision for a table item of type REAL.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemText(item, column)

Returns the text of an item of type TEXT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.

getItemType(item, column)

Returns the type of a table item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getItemValue(item, column)

Returns the text-form value of a table item of any type.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

getLastItem()

Returns the last root tree item.

getListItemIcon(listId, index)

Returns the icon of the item at the specified index of the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list.
index	Int		Index into the list of the item to return.

getListItemIndex(listId, icon)

Returns the index of the item of the specified table list that has the specified icon. Returns -1 if no such item exists.

Argument	Type	Default	Description
listId	Int		ID of the list.
icon	FXIcon		Icon.

getListItemIndex(listId, text)

Returns the index of the item of the specified table list that has the specified text. Returns -1 if no such item exists.

Argument	Type	Default	Description
listId	Int		ID of the list.
text	String		Text.

getListItemText(listId, index)

Returns the text of the item at the specified index of the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list.
index	Int		Index into the list of the item to return.

getNumColumns()

Returns the number of columns.

getNumItems()

Returns the number of items.

getNumListItems(listId)

Returns the number of items in the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list.

getTableStyle()

Returns the options related only to the table.

getTreeColumn()

Returns the column index of the tree.

getVisibleColumns()

Returns the number of visible columns.

getVisibleItems()

Returns the number of visible items.

insertColumns(startColumn, numColumns=1, notify=False)

Inserts columns at the specified location.

Argument	Type	Default	Description
startColumn	Int		Starting column.
numColumns	Int	1	Number of columns to insert.
notify	Bool	False	Specify True to notify target of the insertion.

isAnyItemInColumnSelected(column)

Returns True if any item in the column is selected.

Argument	Type	Default	Description
column	Int		Column index.

isAnyItemInRowSelected(item)

Returns True if any item in the row is selected.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.

isColumnSelected(column)

Returns True if all items in the column are selected.

Argument	Type	Default	Description
column	Int		Column index.

isItemBool(item, column)

Returns True if the specified table item is of type BOOL.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

isItemColor(item, column)

Returns True if the specified table item is of type COLOR.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.

column	Int		Column index of table item.

isItemEditable(item, column)

Returns True if the table item is editable.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

isItemEmpty(item, column)

Returns True if the specified table item does not have a value.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

isItemExpanded(item)

Returns True if the tree item is expanded, False otherwise.

Argument	Type	Default	Description
item	FXTreeItem		

isItemFloat(item, column)

Returns True if the specified table item is of type FLOAT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

isItemIcon(item, column)

Returns True if the specified table item is of type ICON.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

isItemInt(item, column)

Returns True if the specified table item is of type INT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

isItemLeaf(item)

Returns True if the tree item is a leaf-item (has no children), False otherwise.

Argument	Type	Default	Description
item	FXTreeItem		

isItemList(item, column)

Returns True if the specified tabl eitem is of type LIST.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

isItemOpened(item)

Returns True if the tree item is opened, False otherwise.

Argument	Type	Default	Description
item	FXTreeItem		

isItemSelected(item, column)

Returns True if the specified item is selected.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.

isItemText(item, column)

Returns True if the specified table item is of type TEXT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

isItemVisible(item, column)

Returns True if the specified item is visible.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.

isRowSelected(item)

Returns True if all items in the row are selected.

Argument	Type	Default	Description
item	FXTreeItem		

killSelection(notify=False)

Deselects all items.

Argument	Type	Default	Description
notify	Bool	False	

makePositionVisible(item, column)

Scrolls to make the specified row, column fully visible.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.

makeRowVisible(item)

Scrolls vertically (only) to make the specified row fully visible.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.

openItem(item, notify=False)

Opens the specified item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

removeItem(item, notify=False)

Removes the specified tree item and corresponding table row.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

removeItems(from, to, notify=False)

Removes the specified tree items and their corresponding table rows, inclusively.

Argument	Type	Default	Description
from	FXTreeItem		
to	FXTreeItem		
notify	Bool	False	

removeListItem(listId, index)

Removes the item at the specified index from the specified table list; returns the number of items remaining in list.

Argument	Type	Default	Description

listId	Int		ID of the list to remove from.
index	Int		Index of the list item to remove.

selectItem(item, column, notify=False)

Selects the specified item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.
notify	Bool	False	

selectRow(item, notify=False)

Selects all items in the row.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
notify	Bool	False	

setColumnBoolIcons(column, trueIcon=None, falseIcon=None)

Sets the True and False icons of all existing and future table items in a column of type BOOL. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
trueIcon	FXIcon	None	Icon displayed when value is True; 0 = default icon.
falseIcon	FXIcon	None	Icon displayed when value is False; 0 = default icon.

setColumnBoolValue(column, value)

Sets the value of all existing and future table items in a column of type BOOL. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
value	Bool		Specify True or False.

setColumnColor(column, color)

Sets the color of all existing and future table items in a column of type COLOR. The color can be "As is", "Default", a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C"), or a pre-defined color name (e.g., "Red"). Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
color	String		Color.

setColumnColorItemDefault(column, color)

Sets the color of the color item in the flyout menu for all existing and future table items that display "As is" or "Default" in a column of type COLOR. The color is either a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C") or a pre-defined color name (e.g., "Red"). Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
color	String		Color.

setColumnColorOptions(column, opts)

Sets the color flyout options for all existing and future table items in a column of type COLOR. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
opts	Int		Options (see ColorFlyoutOptions).

setColumnEditable(column, editable)

Sets the editability of all existing and future table items in a column. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
editable	Bool		Specify True for editable, False for read-only.

setColumnFloatValue(column, value)

Sets the value of all existing and future table items in a column of type FLOAT. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
value	Float		Floating-point value.

setColumnFormat(column, format)

Sets the real format for all existing and future table items in a column of type REAL.

Argument	Type	Default	Description
column	Int		Table column index.
format	Int		Default format of REAL values in column (see RealFormat).

setColumnIcon(column, icon=None)

Sets the icon of all existing and future table items in a column of type ICON. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
icon	FXIcon	None	Icon.

setColumnIntValue(column, value)

Sets the value of all existing and future table items in a column of type INT. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
value	Int		Integer value.

setColumnJustify(column, justify)

Sets the justification of all existing and future table items in a column. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
justify	Int		Justification (see ItemJustify).

setColumnListId(column, listId)

Sets the list ID of all existing and future table items in a column of type LIST. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
listId	Int		List ID.

setColumnListIndex(column, index)

Sets the list index (selection) of all existing and future table items in a column of type LIST. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
index	Int		Index of item to be selected.

setColumnNumDigits(column, numDigits)

Sets the number of digits to the left of the decimal point for all existing and future table items in a column of type REAL.

Argument	Type	Default	Description
column	Int		Table column index.
numDigits	Int		Default number of digits the left of the decimal point.

setColumnPrecision(column, precision)

Sets the precision for all existing and future table items in a column of type REAL.

Argument	Type	Default	Description
column	Int		Table column index.
precision	Int		Number of digits to the right of the decimal point.

setColumnText(column, text)

Sets the text of all existing and future table items in a column of type TEXT. Specifying -1 for the column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
text	String		Text.

setColumnType(column, type)

Sets the type of a column. Specifying -1 for the table column will change all columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Table column index.
type	Int		Type (see Flags for item types).

setColumnWidth(column, width)

Sets the width, in pixels, of the specified column. Specifying -1 for the column will change all non-leading and non-trailing columns in the table and set the default for the table. Specify -1 for the width will resize each specified column to best fit the width of the title(s) currently shown in its leading and trailing items.

Argument	Type	Default	Description
column	Int		Column index.
width	Int		Width in pixels.

setColumnWidthInChars(column, numChars)

Sets the width, in number of characters, of the specified column. Specifying -1 for the column will change all non-leading and non-trailing columns in the table and set the default for the table.

Argument	Type	Default	Description
column	Int		Column index.
numChars	Int		Width in number of characters.

setCurrentItem(item, column, notify=False)

Sets the current item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.
notify	Bool	False	

setDefaultBoolIcons(trueIcon=None, falseIcon=None)

Sets the default True and False icons for the table (0 = default icon).

Argument	Type	Default	Description
trueIcon	FXIcon	None	Icon displayed when value is True; 0 = default icon.
falseIcon	FXIcon	None	Icon displayed when value is False; 0 = default icon.

setDefaultBoolValue(value)

Sets the default bool value for the table.

Argument	Type	Default	Description
value	Bool		Specify True or False.

setDefaultColor(color)

Sets the default color for all items of type COLOR in the table. The color can be "As is", "Default", a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C"), or a pre-defined color name (e.g., "Red").

Argument	Type	Default	Description
color	String		Color.

setDefaultColumnWidth(width)

Sets the default width, in pixels, for all columns.

Argument	Type	Default	Description
width	Int		Width in pixels.

setDefaultFloatValue(value)

Sets the default floating-point value for the table.

Argument	Type	Default	Description
value	Float		Floating-point value.

setDefaultFormat(format)

Sets the default real format for the table.

Argument	Type	Default	Description
format	Int		Format flag.

setDefaultIntValue(value)

Sets the default integer value for the table.

Argument	Type	Default	Description
value	Int		Integer value.

setDefaultJustify(justify)

Sets the default justification for the table.

Argument	Type	Default	Description
justify	Int		Justification (see ItemJustify).

setDefaultNumDigits(numDigits)

Sets the default number of digits to the left of the decimal point for the table.

Argument	Type	Default	Description
numDigits	Int		Number of digits.

setDefaultPrecision(precision)

Sets the precision for the table.

Argument	Type	Default	Description
precision	Int		Precision.

setDefaultText(text)

Sets the default text for the table.

Argument	Type	Default	Description
text	String		Text.

setDefaultType(type)

Sets the default type for the table.

Argument	Type	Default	Description
type	Int		Type (see Flags for item types).

setItemBoolIcons(item, column, trueIcon=None, falseIcon=None)

Sets the True and False icons of a table item of type BOOL.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
trueIcon	FXIcon	None	Icon displayed when value is True; 0 = default icon.
falseIcon	FXIcon	None	Icon displayed when value is False; 0 = default icon.

setItemBoolValue(item, column, value)

Sets the value of a table item of type BOOL.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
value	Bool		Specify True or False.

setItemCheck(item, check, notify=False)

Sets the item checked state. Valid states are True, False and MAYBE. Returns True if the check value has changed, False otherwise.

Argument	Type	Default	Description
item	FXTreeItem		
check	Int		

notify	Bool	False	

setItemClosedIcon(item, icon)

Changes the tree item's closed icon.

Argument	Type	Default	Description
item	FXTreeItem		
icon	FXIcon		

setItemColor(item, column, color)

Sets the color of a table item of type COLOR. The color can be "As is", "Default", a color hex specification in the form of "RRGGBB" (e.g., "#0A1B2C"), or a pre-defined color name (e.g., "Red").

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
color	String		Color.

setItemEditable(item, column, editable)

Sets the editability of a table item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
editable	Bool		Specify True for editable, False for read-only.

setItemFloatValue(item, column, value)

Sets the value of a table item of type FLOAT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
value	Float		Floating-point value.

setItemFormat(item, column, format)

Sets the format for a table item of type REAL (see RealFormat).

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Table column index of item.
format	Int		Format of table item (see RealFormat).

setItemIcon(item, column, icon=None)

Sets the icon of a table item of type ICON.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
icon	FXIcon	None	Icon.

setItemIntValue(item, column, value)

Sets the value of a table item of type INT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
value	Int		Integer value.

setItemJustify(item, column, justify)

Sets the justification of an item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
justify	Int		Justification (see ItemJustify).

setItemListId(item, column, listId)

Sets the list ID of a table item of type LIST.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
listId	Int		List ID.

setItemListIndex(item, column, index)

Sets the list index (selection) of a table item of type LIST.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
index	Int		Index of item to be selected.

setItemNumDigits(item, column, numDigits)

Sets the number of digits to the left of the decimal point for a table item of type REAL.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
numDigits	Int		Number of digits.

setItemOpenIcon(item, icon)

Sets the tree item's open icon.

Argument	Type	Default	Description
item	FXTreeItem		
icon	FXIcon		

setItemPrecision(item, column, precision)

Sets the precision for a table item of type REAL.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
precision	Int		Number of digits to the right of the decimal point.

setItemText(item, column, text)

Sets the text of an item of type TEXT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.
text	String		Text.

setItemType(item, column, type)

Sets the type of a table item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
type	Int		Type (see Flags for item types).

setItemValue(item, column, valueText)

Sets the value of a table item of any type that can interpret a text string for its value. Returns True if the value of the specified item is set successfully.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.
valueText	String		Text-form value of table item.

setLeadingRowLabels(str)

Set the labels of the leading row. Note: this API must be used to set the header row labels, otherwise labels will be overwritten by auto-numbering.

Argument	Type	Default	Description
str	String		Tab "\t" delimited list, can also contain newline characters indicating that label contains multiple lines of text (e.g. "Young's\nModulus\tPoisson's\nRatio").

setListMaxVisible(maxVisible)

Sets the maximum number of visible items for all table lists.

Argument	Type	Default	Description
maxVisible	Int		Maximum number of visible items.

setTableStyle(style)

Sets the table options.

Argument	Type	Default	Description
style	Int		Style flag (see Flags for AFX table options).

setVisibleColumns(visibleColumns)

Sets the number of visible columns.

Argument	Type	Default	Description
visibleColumns	Int		Number of visible columns.

setVisibleItems(visibleItems)

Sets the number of visible items.

Argument	Type	Default	Description
visibleItems	Int		Number of visible items.

shadeReadOnlyItems(shadeItems)

Makes the table to use a different, typically shaded, background color for read-only items if True is passed to the method. The table would use the same regular background color for both editable and read-only items if False is passed to the method.

Argument	Type	Default	Description
shadeItems	Bool		Specify True to use a different background color for read-only items.

Class flags

Flags for color flyouts (used for items of type COLOR).

COLOR_INCLUDE_COLOR_ONLY **Color item has no As Is and Default in its flyout.**

COLOR_INCLUDE_AS_IS **Color item has As Is in its flyout.**

COLOR_INCLUDE_DEFAULT **Color item has Default in its flyout.**

COLOR_INCLUDE_ALL **Color item has both As Is and Default in its flyout. This is the default option.**

Flags for item alignment.

REAL **Not yet implemented (Real).**

LEFT	Left justified.
RIGHT	Right justified.
CENTER	Center justified (horizontal).
TOP	Top justified.
BOTTOM	Bottom justified.
MIDDLE	Middle justified (vertical).

Flags for item types.

TEXT	Item accepts a text string via a text field.
FLOAT	Item accepts a floating-point number via a text field.
INT	Item accepts an integer via a text field.
LIST	Item accepts input from a list.
BOOL	Item is a boolean; displayed as an icon.
ICON	Item displays an icon and does not accept input.
COLOR	Item accepts color selection via a color flyout.

Flags for real formats.

GENERAL	General.
SCIENTIFIC	Scientific.
AUTOMATIC	Automatic.

Global flags

Flags for AFX tree table options.

AFXTREETABLE_COLUMN_RESIZABLE	Allow users to resize columns.
AFXTREETABLE_NO_COLUMN_SELECT	Disallow column selections (selecting a header/footer item in a column selects the whole column).
AFXTREETABLE_ROW_MODE	Selecting any item in a row selects the whole row.
AFXTREETABLE_EXTENDED_SELECT	Use extended selection mode that allows multiple items to be selected and allows users to drag-select a range of items.
AFXTREETABLE_SINGLE_SELECT	Use single selection mode that allows up to one item to be selected.
AFXTREETABLE_BROWSE_SELECT	Use browse selection mode that enforces one single item to be selected at all times.
AFXTREETABLE_CHECK_BOXES	Show item check boxes.
AFXTREETABLE_PROPAGATE_CHECKS	Propagate checked state to children and parents.
AFXTREETABLE_NORMAL	Default table options--use extended selection mode, columns are resizable, and layout fills both X and Y directions.

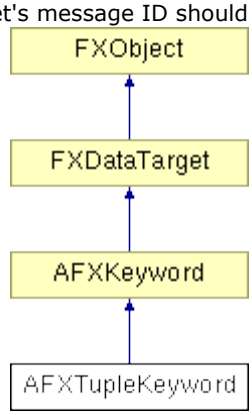
Flags for the table's list (for items of type LIST).

AFXTREETABLE_LIST_PRESELECT_NONE	Do not pre-select any list item.

AFXTupleKeyword

This class manages values which are sent as tuples in a command.

Widgets used to edit the whole tuple should have their message ID set to 0. It is also possible to edit individual elements of the tuple. In order to edit the N-th tuple element, the widget's message ID should be set to N (N is 1-based).



AFXTupleKeyword(command, name, isRequired=False, minLength=0, maxLength=-1, opts=0)

Constructor.

Argument	Type	Default	Description
command	AFXCommand		Host command.
name	String		Keyword name.
isRequired	Bool	False	True if this keyword is a required argument.
minLength	Int	0	Minimum (and default) tuple length.
maxLength	Int	-1	Maximum tuple length (-1 => unlimited).
opts	Int	0	Options.

equal(index, a, b)

Returns True if the two tuple element values compare equal (index is not used).

Argument	Type	Default	Description
index	Int		Element index (not used).
a	String		First value.
b	String		Second value.

getDefaultStyle()

Returns the default style for elements.

getDefaultType()

Returns the default type for elements.

getDefaultValues()

Returns the default values for this tuple.

getElementStyle(index)

Returns the style of one element. Will never return AFXTUPLE_STYLE_DEFAULT!

Argument	Type	Default	Description
index	Int		Element index.

getElementType(index)

Returns the type of one element. Will never return AFXTUPLE_TYPE_DEFAULT!

Argument	Type	Default	Description
index	Int		Element index.

getFormattedValue(index)

Returns the formatted value of the tuple element, suitable for placing in a command. If the element has AFXTUPLE_EVALUATE style and its contents are invalid, an exception will be thrown.

Argument	Type	Default	Description
index	Int		Element index.

getLength()

Returns the length of the tuple.

getMaxLength()

Returns the maximum length of this tuple, or -1 for unbounded length.

getMinLength()

Returns the minimum length of this tuple.

getTypeName()

Returns the name of the tuple keyword type.

Implements AFXKeyword.

getValue(index)

Returns the value of a tuple element.

Argument	Type	Default	Description
index	Int		Element index.

getValueAsDouble()

Returns the keyword's value as a float; returns False upon failure.

getValueAsInt()

Returns the keyword's value as an integer; returns False upon failure.

getValueAsString()

Returns the formatted string that represents the current keyword value in a command.

Implements AFXKeyword.

getValueForBlank(index)

Returns the value substituted for blank tuple element.

Argument	Type	Default	Description
index	Int		Element index.

getValues()

Returns a string containing values (separated by commas) of the tuple elements.

getValuesForBlanks()

Returns a string containing values substituted for blanks for the tuple elements.

insertElements(index, numCols)

Inserts elements starting at the given index.

Argument	Type	Default	Description
index	Int		Starting index.
numCols	Int		Number of elements to insert.

isValueChanged()

Returns True if the keyword value differs from its previous value.

Implements AFXKeyword.

removeElements(index, numCols)

Removes elements starting at the given index.

Argument	Type	Default	Description
index	Int		Starting index.
numCols	Int		Number of elements to remove.

setDefaultStyle(style)

Sets the default style for elements.

Argument	Type	Default	Description
style	Int		New default element style.

setDefaultType(type)

Sets the default type for elements.

Argument	Type	Default	Description
type	Int		New default type.

setDefaultValues(values)

Sets the default values for this tuple.

Argument	Type	Default	Description
values	String		Sequence string with default values.

setElementStyle(index, style)

Sets the style of one element.

Argument	Type	Default	Description
index	Int		Element index.
style	Int		New element style.

setElementType(index, type)

Sets the type of one element.

Argument	Type	Default	Description
index	Int		Element index.
type	Int		New element type.

setLengthRange(minLength, maxLength)

Sets the range of allowable tuple lengths.

Argument	Type	Default	Description
minLength	Int		New minimum length.
maxLength	Int		New maximum length, or -1 for unbounded length.

setMaxLength(length)

Sets the maximum length of this tuple.

Argument	Type	Default	Description
length	Int		New maximum length, or -1 for unbounded length.

setMinLength(length)

Sets the minimum length of this tuple.

Argument	Type	Default	Description
length	Int		New minimum length.

setValue(index, value)

Sets the value of the tuple element; returns False upon failure.

Argument	Type	Default	Description
index	Int		Element index.
value	String		New value.

setValueForBlank(index, value)

Sets the value substituted for a blank tuple element.

Argument	Type	Default	Description
index	Int		Element index.
value	String		New value.

setValues(values)

Sets values for all tuple elements (use commas to separate values within the string).

Argument	Type	Default	Description
values	String		Tuple string with new values.

setValuesForBlanks(values)

Sets all values substituted for blanks for the tuple elements.

Argument	Type	Default	Description
values	String		Tuple string with values.

setValueToDefault(ignoreUnspecified=False)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Should ignore if default is an unspecified value.

setValueToPrevious()

Sets the keyword value to its previous value.

Implements AFXKeyword.

syncPreviousValue()

Sets the keyword's previous value to its current value.

Implements AFXKeyword.

Class flags

Message ID's.

`ID_PRINTSNIPPET` **For debugging.**

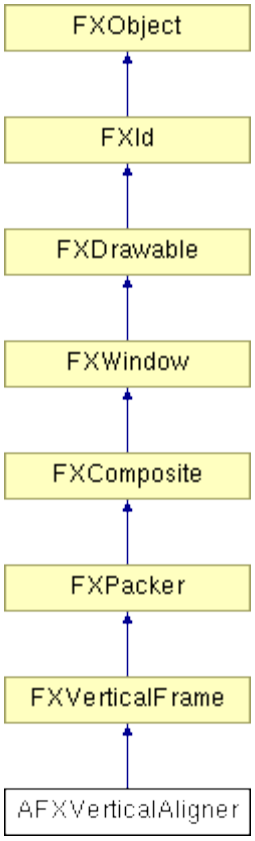
Global flags

Flags for tuple keyword options.

AFXTUPLE_TYPE_ANY	Any type is accepted.
AFXTUPLE_TYPE_DEFAULT	Element type is the same as the tuple default type.
AFXTUPLE_TYPE_INT	Element is an integer number.
AFXTUPLE_TYPE_FLOAT	Element is a floating-point number.
AFXTUPLE_TYPE_STRING	Element is a string.
AFXTUPLE_TYPE_BOOL	Element is True or False.
AFXTUPLE_TYPE_MASK	Mask for element types.
AFXTUPLE_ALLOW_EMPTY	Allow empty values for the element.
AFXTUPLE_DEFAULT_IF_EMPTY	Always substitute the default for an empty value.
AFXTUPLE_EVALUATE	Evaluate integer and float elements.
AFXTUPLE_STYLE_DEFAULT	Use tuple default element style.
AFXTUPLE_STYLE_MASK	Mask for element styles.

AFXVerticalAligner

This class is used to automatically vertically align its children "container" widgets (e.g. AFXTextField or AFXComboBox). The width of the first widget in the container of each child of the vertical aligner is set to the width of the widest first widget of all the vertical aligner's children.



AFXVerticalAligner(p, opts=0, x=0, y=0, w=0, h=0, pl=0, pr=0, pt=0, pb=0, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
opts	Int	0	Options and hints.
x	Int	0	X coordinate of the origin.
y	Int	0	Y coordinate of the origin.
w	Int	0	Width of the widget.
h	Int	0	Height of the widget.
pl	Int	0	Left padding (margin).
pr	Int	0	Right padding (margin).
pt	Int	0	Top padding (margin).
pb	Int	0	Bottom padding (margin).
hs	Int	DEFAULT_SPACING	Horizontal spacing.
vs	Int	DEFAULT_SPACING	Vertical spacing.

create()

Creates the aligner.

Reimplemented from FXComposite.

getDefaultHeight()

Returns the default height.

Reimplemented from FXVerticalFrame.

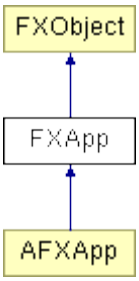
getDefaultWidth()

Returns the default width.

Reimplemented from FXVerticalFrame.

FXApp

Application Object



FXApp(name=Application, vendor=FoxDefault)

Copyright notice of library.

Construct application object; the name and vendor strings are used as keys into the registry database for this application's settings

Argument	Type	Default	Description
name	String	Application	
vendor	String	FoxDefault	

addChore(tgt, sel)

Add a idle processing message to be sent to target object when the system becomes idle, i.e. there are no events to be processed.

Argument	Type	Default	Description
tgt	FXObject		
sel	Int		

addInput(fd, mode, tgt, sel)

Add a file descriptor fd to be watched for activity as determined by mode, where mode is a bitwise OR (INPUT_READ, INPUT_WRITE, INPUT_EXCEPT). A message of type SEL_IO_READ, SEL_IO_WRITE, or SEL_IO_EXCEPT will be sent to the target when the specified activity is detected on the file descriptor.

On Windows, a Win32 event, not a file descriptor, must be specified. The client code for this interface must be platform-dependent. See addSocket below for a portable interface. CAE

Argument	Type	Default	Description
fd	FXInputHandle		
mode	Int		
tgt	FXObject		
sel	Int		

addSocket(sd, mode, tgt, sel)

CAE Add a socket descriptor sd to be watched for activity as determined by mode, where mode is a bitwise OR (INPUT_READ, INPUT_WRITE, INPUT_EXCEPT). A message of type SEL_IO_READ, SEL_IO_WRITE, or SEL_IO_EXCEPT will be sent to the target when the specified activity is detected on the socket descriptor.

This is identical to addInput on Unix. It behaves the same on Windows.

Argument	Type	Default	Description

sd	SOCKET		
mode	Int		
tgt	FXObject		
sel	Int		

addTimeout(ms, tgt, sel)

Add timeout message to be sent to target object in ms milliseconds; the timer fires only once after the interval expires.

Argument	Type	Default	Description
ms	Int		
tgt	FXObject		
sel	Int		

beep()

Beep.

beginWaitCursor()

Begin of wait-cursor block; wait-cursor blocks may be nested.

create()

Create application's windows.

Reimplemented in AFXApp.

endWaitCursor()

End of wait-cursor block.

forceRefresh()

Force GUI refresh.

getAppName()

Get application name.

getBorderColor()

Obtain default colors.

getMainWindow()

Get main window, if any.

getMonoVisual()

Get monochrome visual.

getNormalFont()

Return default font.

getRoot()

Get root Window.

getTypingSpeed()

Obtain application-wide settings.

init(argc, argv, connect=True)

Initialize application. Parses and removes common command line arguments, reads the registry. Finally, if connect is True, it opens the display.

Argument	Type	Default	Description
argc	Int		
argv	String		
connect	Bool	True	

peekEvent()

Peek to determine if there's an event.

refresh()

Schedule a refresh.

removeChore(c)

Remove idle processing message.

Argument	Type	Default	Description
c	FXChore		

removeInput(fd, mode)

Remove input message and target object for the specified file descriptor and mode, which is a bitwise OR of (INPUT_READ, INPUT_WRITE, INPUT_EXCEPT).

Argument	Type	Default	Description
fd	FXInputHandle		
mode	Int		

removeSocket(sd, mode)

CAE Remove input message and target object for the specified socket descriptor and mode, which is a bitwise OR of (INPUT_READ, INPUT_WRITE, INPUT_EXCEPT).

Argument	Type	Default	Description
sd	SOCKET		

mode	Int		

removeTimeout(t)

Remove timeout, returns NULL.

Argument	Type	Default	Description
t	FXTimer		

repaint()

Paint all windows marked for repainting. On return all the applications windows have been painted.

run()

Run the main application event loop until stop() is called, and return the exit code passed as argument to stop().

Reimplemented in AFXApp.

runOneEvent()

Perform one event dispatch.

runUntil(condition)

Run an event loop till some flag becomes non-zero.

Reimplemented in AFXApp.

Argument	Type	Default	Description
condition	Int		

runWhileEvents(window=None)

Run event loop while there are events are available in the queue. Returns 1 when all events in the queue have been handled, and 0 when the event loop was terminated due to stop() or stopModal(). Except for the modal window and its children, user input to all windows is blocked; if the modal window is NULL all user input is blocked.

Argument	Type	Default	Description
window	FXWindow	None	

setBorderColor(color)

Change default colors.

Argument	Type	Default	Description
color	FXColor		

setNormalFont(font)

Change default font.

Argument	Type	Default	Description
font	FXFont		

setTypingSpeed(speed)

Change application-wide settings.

Argument	Type	Default	Description
speed	Int		

stop(value=0)

Terminate the outermost event loop, and all inner modal loops; All more deeper nested event loops will be terminated with code equal to 0, while the outermost event loop will return code equal to value.

Argument	Type	Default	Description
value	Int	0	

useWidgetBackColor(state)

CAE Begin.

On Windows only, a widget's background color is normally obtained from the desktop theme, even if the widget's background color has been set using its setBackColor method. The useWidgetBackColor method allows customizers to override that behavior and have the application use the color set by a widget's setBackColor method instead. It is not necessary to use this method if the Windows Classic theme is used.

Argument	Type	Default	Description
state	Bool		

Class flags

Messages applications understand.

ID_QUIT **Terminate the application normally.**

ID_DUMP **Dump the current widget tree.**

Global flags

File input modes for addInput

INPUT_NONE **Inactive.**

INPUT_READ **Read input fd.**

INPUT_WRITE **Write input fd.**

INPUT_EXCEPT **Except input fd.**

All ways of being modal

MODAL_FOR_NONE **Non modal event loop (dispatch normally).**

MODAL_FOR_WINDOW **Modal dialog (beep if outside of modal dialog).**

MODAL_FOR_POPUP **Modal for popup (always dispatch to popup).**

Default cursors provided by the application

DEF_ARROW_CURSOR **Arrow cursor.**

DEF_RARROW_CURSOR **Reverse arrow cursor.**

DEF_TEXT_CURSOR **Text cursor.**

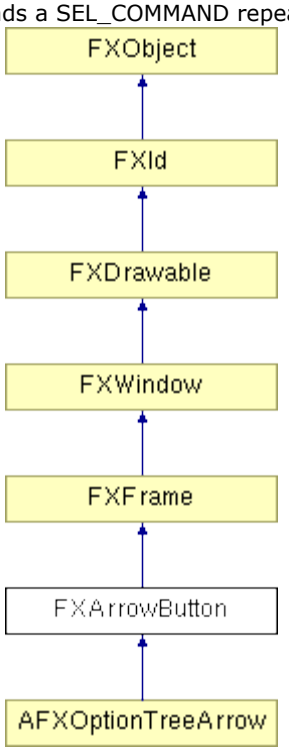
DEF_HSPLIT_CURSOR **Horizontal split cursor.**

DEF_VSPLIT_CURSOR **Vertical split cursor.**

DEF_XSPLIT_CURSOR	**Cross split cursor.**
DEF_SWATCH_CURSOR	**Color swatch drag cursor.**
DEF_MOVE_CURSOR	**Move cursor.**
DEF_DRAGH_CURSOR	**Resize horizontal edge.**
DEF_DRAGV_CURSOR	**Resize vertical edge.**
DEF_DRAGTL_CURSOR	**Resize upper-leftcorner.**
DEF_DRAGBR_CURSOR	**Resize bottom-right corner.**
DEF_DRAGTR_CURSOR	**Resize upper-right corner.**
DEF_DRAGBL_CURSOR	**Resize bottom-left corner.**
DEF_DNDSTOP_CURSOR	**Drag and drop stop.**
DEF_DNDCOPY_CURSOR	**Drag and drop copy.**
DEF_DNDMOVE_CURSOR	**Drag and drop move.**
DEF_DNDLINK_CURSOR	**Drag and drop link.**
DEF_CROSSHAIR_CURSOR	**Cross hair cursor.**
DEF_CORNERNE_CURSOR	**North-east cursor.**
DEF_CORNERNW_CURSOR	**North-west cursor.**
DEF_CORNERSE_CURSOR	**South-east cursor.**
DEF_CORNERSW_CURSOR	**South-west cursor.**
DEF_ROTATE_CURSOR	**Rotate cursor.**

FXArrowButton

Button with an arrow; the arrow can point in any direction. When clicked, the arrow button sends a SEL_COMMAND to its target. When ARROW_REPEAT is passed, the arrow button sends a SEL_COMMAND repeatedly while the button is pressed.



FXArrowButton(p, tgt=None, sel=0, opts=ARROW_NORMAL, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Construct arrow button.

Argument	Type	Default	Description
p	FXComposite		
tgt	FXObject	None	
sel	Int	0	
opts	Int	ARROW_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_PAD	
pr	Int	DEFAULT_PAD	
pt	Int	DEFAULT_PAD	
pb	Int	DEFAULT_PAD	

canFocus()

Returns True because a button can receive focus.

Reimplemented from FXWindow.

disable()

Disable the button.

Reimplemented from FXWindow.

enable()

Enable the button.

Reimplemented from FXWindow.

getArrowColor()

Get the fill color for the arrow.

getArrowSize()

Get the default arrow size.

getArrowStyle()

Get the arrow style flags.

getDefaultHeight()

Get default height.

Reimplemented from FXFrame.

getDefaultWidth()

Get default width.

Reimplemented from FXFrame.

getJustify()

Get the current justification mode.

getState()

Get the button state (where True means the button is down).

getTipText()

Get tool tip message for this arrow button.

setArrowColor(clr)

Set the fill color for the arrow.

Argument	Type	Default	Description
clr	FXColor		

setArrowSize(size)

Set the default arrow size.

Argument	Type	Default	Description
size	Int		

setArrowStyle(style)

Set the arrow style flags.

Argument	Type	Default	Description
style	Int		

setJustify(mode)

Set the current justification mode.

Argument	Type	Default	Description
mode	Int		

setState(s)

Set the button state (where True means the button is down).

Argument	Type	Default	Description
s	Bool		

setTipText(text)

Set tool tip message for this arrow button.

Argument	Type	Default	Description
text	String		

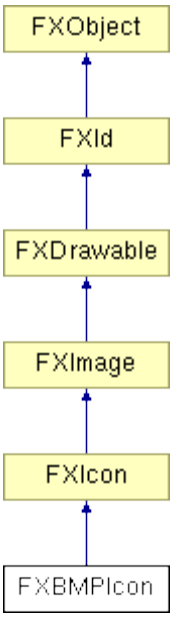
Global flags

Arrow style options

ARROW_NONE	**No arrow.**
ARROW_UP	**Arrow points up.**
ARROW_DOWN	**Arrow points down.**
ARROW_LEFT	**Arrow points left.**
ARROW_RIGHT	**Arrow points right.**
ARROW_REPEAT	**Button repeats if held down.**
ARROW_AUTOGRAY	**Automatically gray out when not updated.**
ARROW_AUTOHIDE	**Automatically hide button when not updated.**
ARROW_TOOLBAR	**Button is toolbar-style.**
ARROW_SPINNER	**Button is spinner-style.**

FXBMPIcon

Microsoft Bitmap icon.



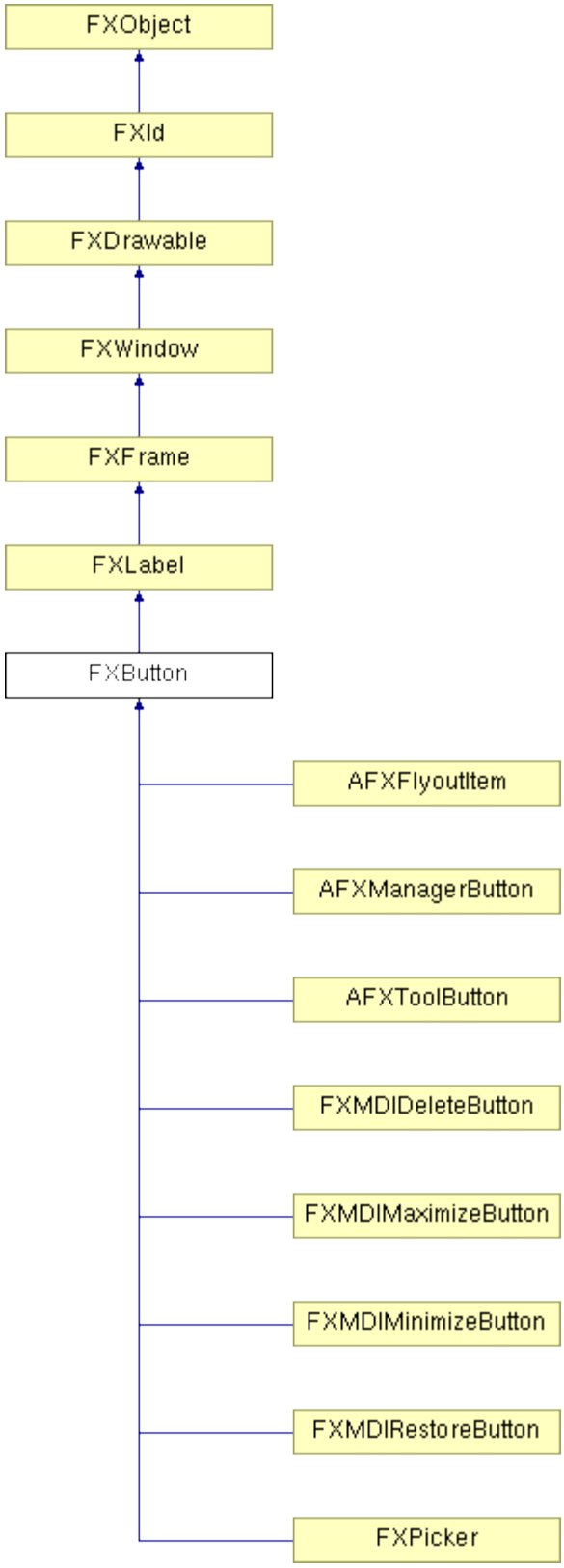
FXBMPIcon(a, pix=None, clr=FXRGB(192, 192, 192), opts=0, w=1, h=1)

Construct icon from memory stream formatted in Microsoft BMP format.

Argument	Type	Default	Description
a	FXApp		
pix		None	
clr	FXColor	FXRGB(192, 192, 192)	
opts	Int	0	
w	Int	1	
h	Int	1	

FXButton

A button provides a push button, with optional icon and/or text label. When pressed, the button widget sends a SEL_COMMAND to its target.



FXButton(p, text, ic=None, tgt=None, sel=0, opts=BUTTON_NORMAL, x=0, y=0, w=0, h=0,

pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Construct button with text and icon.

Argument	Type	Default	Description
p	FXComposite		
text	String		
ic	FXIcon	None	
tgt	FXObject	None	
sel	Int	0	
opts	Int	BUTTON_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_PAD	
pr	Int	DEFAULT_PAD	
pt	Int	DEFAULT_PAD	
pb	Int	DEFAULT_PAD	

canFocus()

Returns True because a button can receive focus.

Reimplemented from FXWindow.

Reimplemented in AFXFlyoutItem.

getButtonStyle()

Get the button style flags.

getState()

Get the button state.

setButtonStyle(style)

Set the button style flags.

Argument	Type	Default	Description
style	Int		

setDefault(enable=True)

Set as default button.

Reimplemented from FXWindow.

Argument	Type	Default	Description
enable	Bool	True	

setState(s)

Set the button state.

Argument	Type	Default	Description
s	Int		

Global flags

Button state bits

STATE_UP **Button is up.**

STATE_DOWN **Button is down.**

STATE_ENGAGED **Button is engaged.**

STATE_UNCHECKED **Same as STATE_UP (used for check buttons or radio buttons).**

STATE_CHECKED **Same as STATE_ENGAGED (used for check buttons or radio buttons).**

Button flags

BUTTON_AUTOGRAY **Automatically gray out when not updated.**

BUTTON_AUTOHIDE **Automatically hide button when not updated.**

BUTTON_TOOLBAR **Toolbar style button [flat look].**

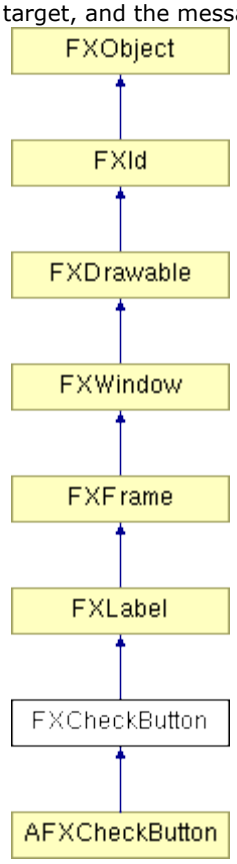
BUTTON_DEFAULT **May become default button when receiving focus.**

BUTTON_INITIAL **This button is the initial default button.**

BUTTON_NORMAL **Normal button style.**

FXCheckButton

A check button is a tri-state button. Normally, it is either True or False, and toggles between True or False whenever it is pressed. A third state MAYBE may be set to indicate that no selection has been made yet by the user, or that the state is ambiguous. When pressed, the check button sends a SEL_COMMAND to its target, and the message data represents the state of the check button.



FXCheckButton(p, text, tgt=None, sel=0, opts=CHECKBUTTON_NORMAL, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Construct new check button.

Argument	Type	Default	Description
p	FXComposite		
text	String		
tgt	FXObject	None	
sel	Int	0	
opts	Int	CHECKBUTTON_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_PAD	
pr	Int	DEFAULT_PAD	
pt	Int	DEFAULT_PAD	
pb	Int	DEFAULT_PAD	

canFocus()

Returns True because a check button can receive focus.

Reimplemented from FXWindow.

getCheck()

Get check button state (True, False or MAYBE).

Reimplemented in AFXCheckButton.

getDefaultHeight()

Get default height.

Reimplemented from FXLabel.

getDefaultWidth()

Get default width.

Reimplemented from FXLabel.

setCheck(state=True)

Set check button state (True, False or MAYBE).

Argument	Type	Default	Description
state	Bool	True	

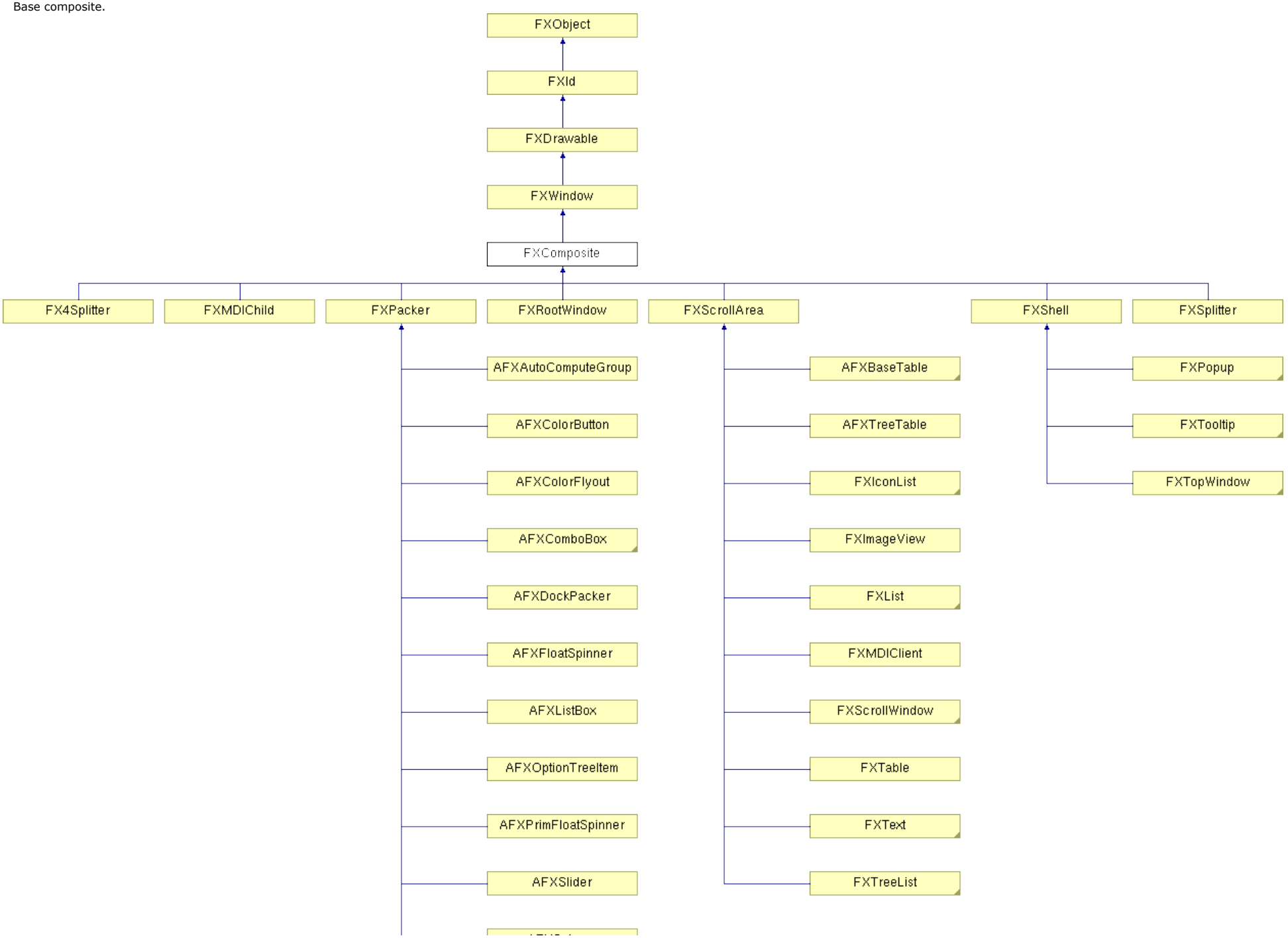
Global flags

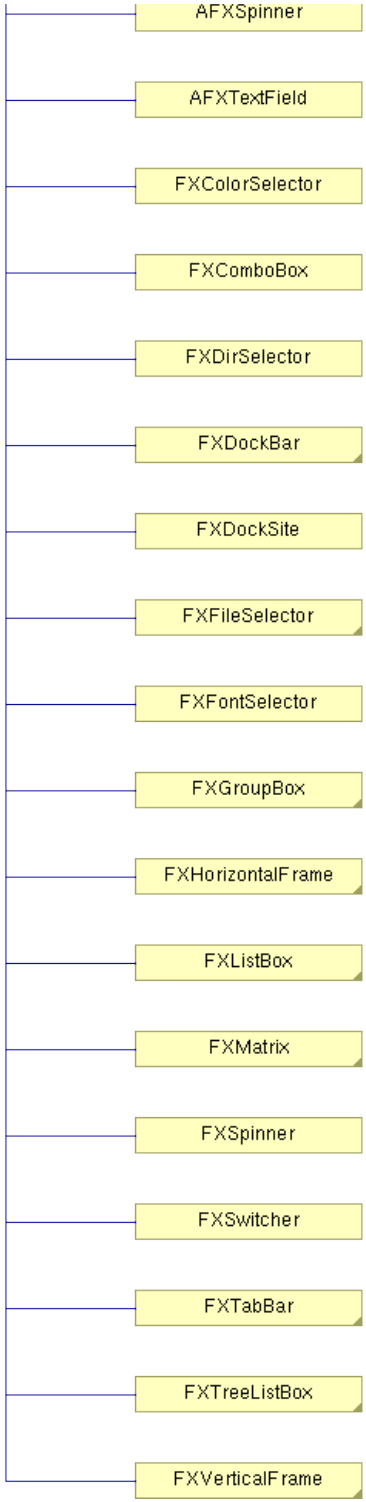
CheckButton styles

CHECKBUTTON_AUTOGRAY	**Automatically gray out when not updated.**
CHECKBUTTON_AUTOHIDE	**Automatically hide when not updated.**

FXComposite

Base composite.




AFXSpinner
AFXTextField
FXColorSelector
FXComboBox
FXDirSelector
FXDockBar
FXDockSite
FXFileSelector
FXFontSelector
FXGroupBox
FXHorizontalFrame
FXListBox
FXMatrix
FXSpinner
FXSwitcher
FXTabBar
FXTreeListBox
FXVerticalFrame

FXComposite(p, opts=0, x=0, y=0, w=0, h=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

create()

Create server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXColorSelector, FXComboBox, FXDirBox, FXDirList, FXDriveBox, FXFileList, FXFontSelector, FXGroupBox, FXIconList, FXImageView, FXList, FXListBox, FXMDIChild, FXPrintDialog, FXRootWindow, FXScrollWindow, FXShell, FXSpinner, FXTabBar, FXTable, FXText, FXToolbarShell, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, AFXManagerMenuPane, AFXMainWindow, AFXPromptArea, AFXBaseTable, AFXColorButton, AFXColorFlyout, AFXComboBox, AFXDialog, AFXFloatSpinner, AFXListBox, AFXNote, AFXOptionTreeItem, AFXPrimFloatSpinner, AFXSpinner, AFXTable, AFXTextField, and AFXVerticalAligner.

destroy()

Destroy server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXComboBox, FXDirBox, FXDirList, FXDriveBox, FXFileList, FXListBox, FXRootWindow, FXTreeList, FXTreeListBox, AFXColorFlyout, and AFXTable.

detach()

Detach server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXComboBox, FXDirBox, FXDirList, FXDriveBox, FXFileList, FXGroupBox, FXIconList, FXImageView, FXList, FXListBox, FXMDIChild, FXRootWindow, FXTable, FXText, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, AFXBaseTable, AFXColorFlyout, AFXNote, and AFXTable.

getDefaultHeight()

Return default height.

Reimplemented from FXWindow.

Reimplemented in FX4Splitter, FXComboBox, FXDockSite, FXGroupBox, FXHorizontalFrame, FXList, FXListBox, FXMDIChild, FXMatrix, FXPacker, FXPopup, FXRootWindow, FXScrollArea, FXSpinner, FXSplitter, FXStatusbar, FXSwitcher, FXTabBar, FXTabBook, FXTable, FXText, FXToolbar, FXToolbarShell, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, FXVerticalFrame, AFXMainWindow, AFXToolbarGroup, AFXBaseTable, AFXList, AFXOptionTreeList, AFXPrimFloatSpinner, AFXSlider, AFXTable, AFXTreeTable, and AFXVerticalAligner.

getDefaultWidth()

Return default width.

Reimplemented from FXWindow.

Reimplemented in FX4Splitter, FXComboBox, FXDockSite, FXGroupBox, FXHorizontalFrame, FXList, FXListBox, FXMDIChild, FXMatrix, FXPacker, FXPopup, FXRootWindow, FXScrollArea, FXSpinner, FXSplitter, FXStatusbar, FXSwitcher, FXTabBar, FXTabBook, FXTable, FXText, FXToolbar, FXToolbarShell, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, FXVerticalFrame, AFXMainWindow, AFXToolbarGroup, AFXBaseTable, AFXOptionTreeItem, AFXOptionTreeList, AFXPrimFloatSpinner, AFXSlider, AFXTable, AFXTextField, AFXTreeTable, and AFXVerticalAligner.

maxChildHeight()

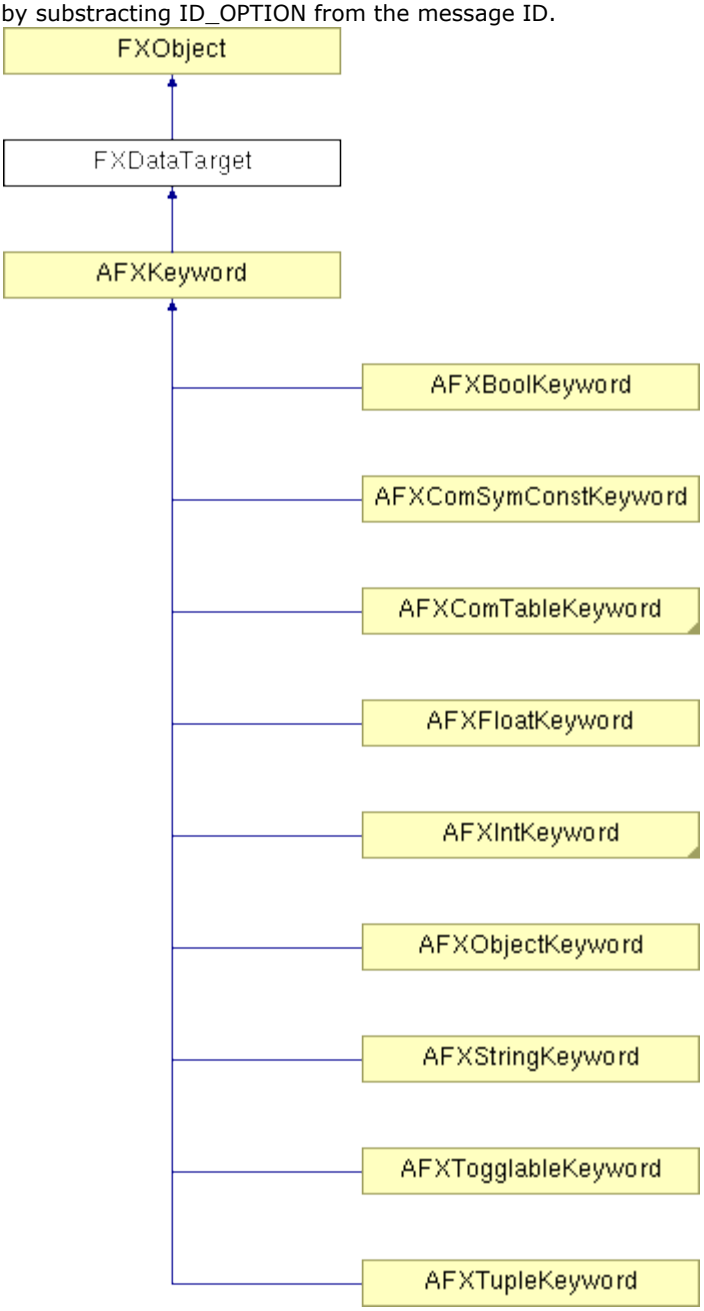
Return the height of the tallest child window.

maxChildWidth()

Return the width of the widest child window.

FXDataTarget

A Data Target allows a valuator widget such as a Slider or Text Field to be directly connected with a variable in the program. Whenever the valuator control changes, the variable connected through the data target is automatically updated; conversely, whenever the program changes a variable, all the connected valuator widgets will be updated to reflect this new value on the display. Data Targets also allow connecting Radio Buttons, Menu Commands, and so on to a variable. In this case, the new value of the connected variable is computed by substracting ID_OPTION from the message ID.



FXDataTarget(tgt=None, sel=0)

Associate with nothing.

Argument	Type	Default	Description
tgt	FXObject	None	
sel	Int	0	

FXDataTarget(value, tgt=None, sel=0)

Associate with character variable.

Argument	Type	Default	Description
value	String		
tgt	FXObject	None	
sel	Int	0	

FXDataTarget(value, tgt=None, sel=0)

Associate with unsigned character variable.

Argument	Type	Default	Description
value	Int		
tgt	FXObject	None	
sel	Int	0	

FXDataTarget(value, tgt=None, sel=0)

Associate with signed short variable.

Argument	Type	Default	Description
value	Int		
tgt	FXObject	None	
sel	Int	0	

FXDataTarget(value, tgt=None, sel=0)

Associate with unsigned short variable.

Argument	Type	Default	Description
value	Int		
tgt	FXObject	None	
sel	Int	0	

FXDataTarget(value, tgt=None, sel=0)

Associate with int variable.

Argument	Type	Default	Description
value	Int		
tgt	FXObject	None	
sel	Int	0	

FXDataTarget(value, tgt=None, sel=0)

Associate with unsigned int variable.

Argument	Type	Default	Description
value	Int		

tgt	FXObject	None	
sel	Int	0	

FXDataTarget(value, tgt=None, sel=0)

Associate with float variable.

Argument	Type	Default	Description
value	Float		
tgt	FXObject	None	
sel	Int	0	

FXDataTarget(value, tgt=None, sel=0)

Associate with double variable.

Argument	Type	Default	Description
value	Float		
tgt	FXObject	None	
sel	Int	0	

FXDataTarget(value, tgt=None, sel=0)

Associate with string variable.

Argument	Type	Default	Description
value	String		
tgt	FXObject	None	
sel	Int	0	

connect(value)

Associate with string variable.

Argument	Type	Default	Description
value	String		

connect(value)

Associate with double variable.

Argument	Type	Default	Description
value	Float		

connect(value)

Associate with float variable.

Argument	Type	Default	Description
value	Float		

connect(value)

Associate with unsigned int variable.

Argument	Type	Default	Description
value	Int		

connect(value)

Associate with int variable.

Argument	Type	Default	Description
value	Int		

connect(value)

Associate with unsigned short variable.

Argument	Type	Default	Description
value	Int		

connect(value)

Associate with signed short variable.

Argument	Type	Default	Description
value	Int		

connect(value)

Associate with unsigned character variable.

Argument	Type	Default	Description
value	Int		

connect(value)

Associate with character variable.

Argument	Type	Default	Description
value	String		

connect()

Associate with nothing.

getData()

Return pointer to data its connected to.

getSelector()

Get the message identifier for this data target.

getTarget()

Get the message target object for this data target, if any.

getType()

Return type of data its connected to.

setSelector(sel)

Set the message identifier for this data target.

Argument	Type	Default	Description
sel	Int		

setTarget(t)

Set the message target object for this data target.

Argument	Type	Default	Description
t	FXObject		

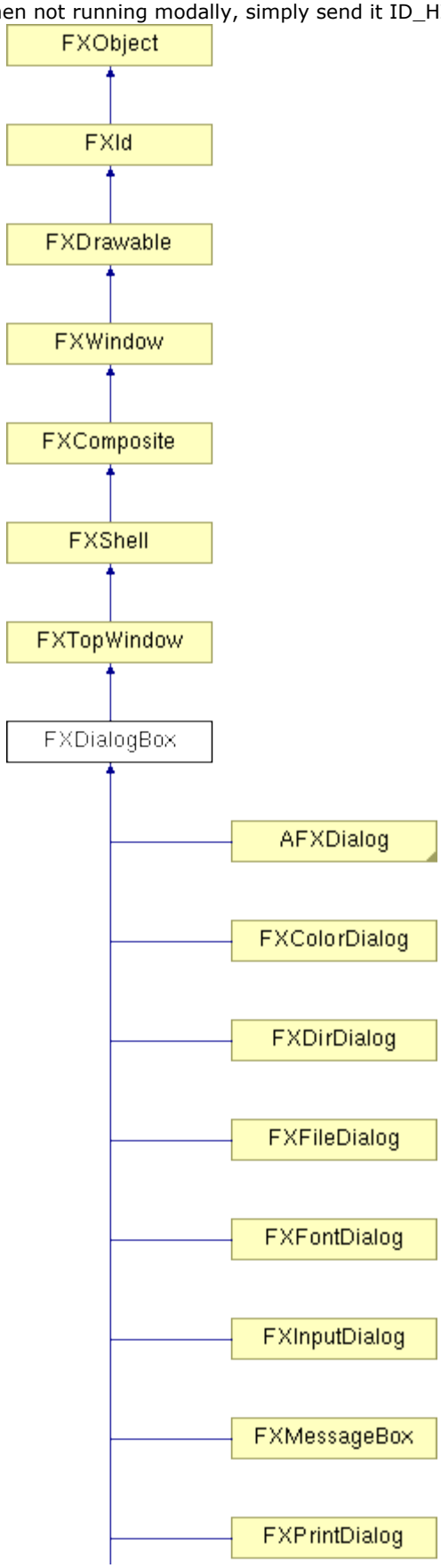
Class flags

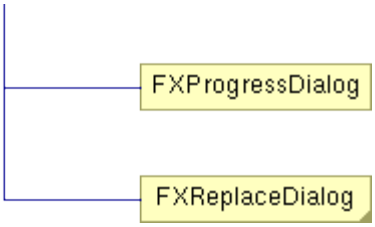
ID_VALUE **Will cause the FXDataTarget to ask sender for value.**

ID_OPTION **ID_OPTION+i will set the value to i where -10000<=i<=10000.**

FXDialogBox

DialogBox window. When receiving ID_CANCEL or ID_ACCEPT, the DialogBox breaks out of the modal loop and returns False or True, respectively. To close the DialogBox when not running modally, simply send it ID_HIDE.





FXDialogBox(a, name, opts=DECOR_TITLE| DECOR_BORDER, x=0, y=0, w=0, h=0, pl=10, pr=10, pt=10, pb=10, hs=4, vs=4)

Construct free-floating dialog.

Argument	Type	Default	Description
a	FXApp		
name	String		
opts	Int	DECOR_TITLE| DECOR_BORDER	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	10	
pr	Int	10	
pt	Int	10	
pb	Int	10	
hs	Int	4	
vs	Int	4	

FXDialogBox(owner, name, opts=DECOR_TITLE| DECOR_BORDER, x=0, y=0, w=0, h=0, pl=10, pr=10, pt=10, pb=10, hs=4, vs=4)

Construct dialog which will always float over the owner window.

Argument	Type	Default	Description
owner	FXWindow		
name	String		
opts	Int	DECOR_TITLE| DECOR_BORDER	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	10	
pr	Int	10	
pt	Int	10	
pb	Int	10	
hs	Int	4	
vs	Int	4	

execute(placement=PLACEMENT_CURSOR)

Run modal invocation of the dialog.

Reimplemented in FXInputDialog, and FXReplaceDialog.

Argument	Type	Default	Description
placement	Int	PLACEMENT_CURSOR	

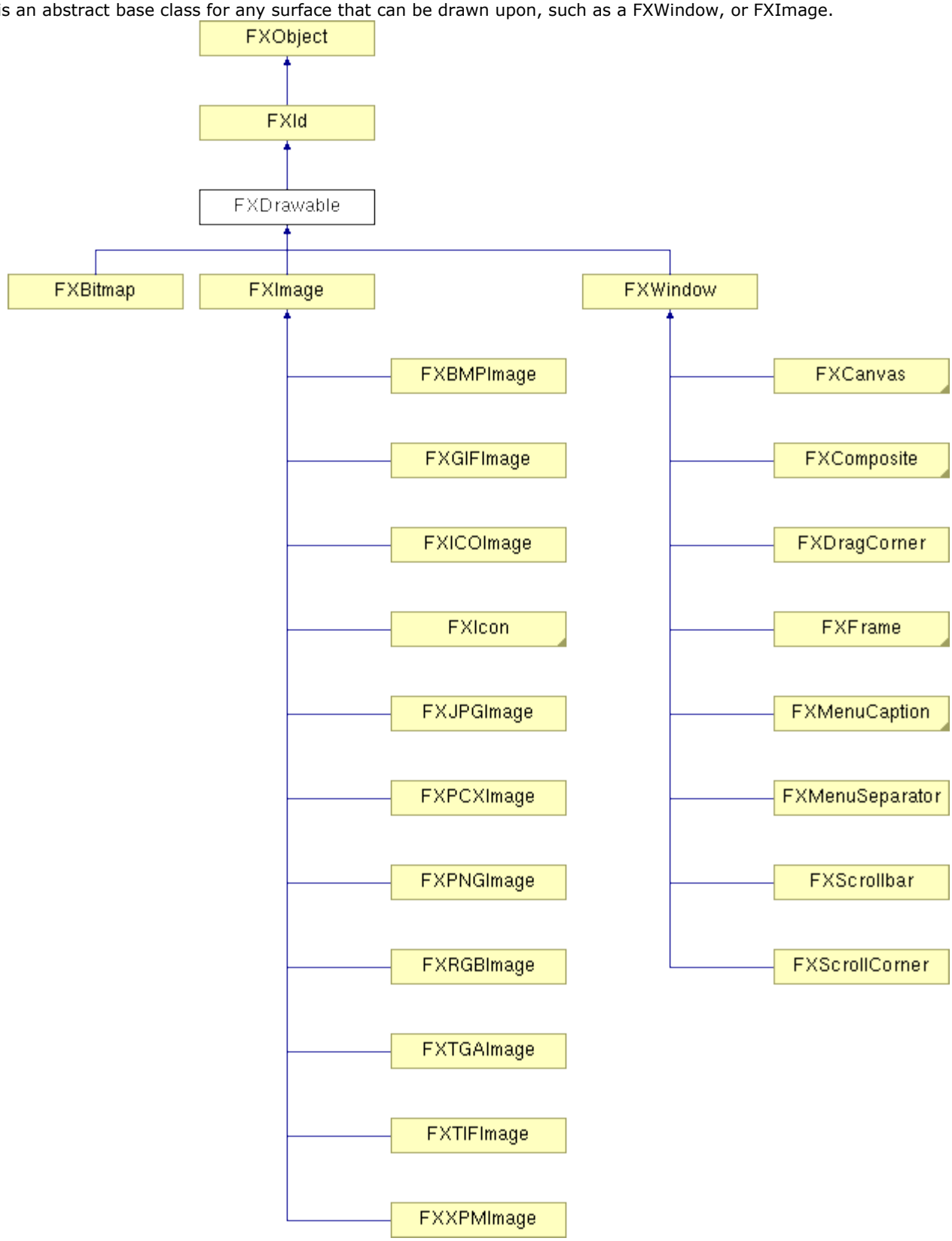
Class flags

ID_CANCEL **Closes the dialog, cancel the entry.**

ID_ACCEPT **Closes the dialog, accept the entry.**

FXDrawable

Drawable is an abstract base class for any surface that can be drawn upon, such as a FXWindow, or FXImage.



getHeight()

Height of drawable.

getVisual()

Get the visual.

getWidth()

Width of drawable.

resize(w, h)

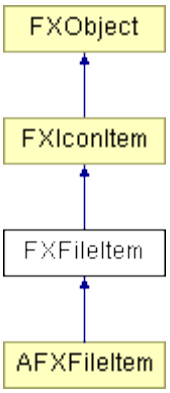
Resize drawable to the specified width and height.

Reimplemented in FXBitmap, FXIcon, FXIconList, FXImage, FXMDIChild, FXRootWindow, FXText, FXTopWindow, and FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

FXFileItem

File item



FXFileItem()

File time.

FXFileItem(text, bi=None, mi=None, ptr=None)

Constructor.

Argument	Type	Default	Description
text	String		
bi	FXIcon	None	
mi	FXIcon	None	
ptr	String	None	

getDate()

Return the date for this item.

getSize()

Return the file size for this item.

isBlockdev()

Return True if this is a block device item.

isChardev()

Return True if this is a character device item.

isDirectory()

Return True if this is a directory item.

Reimplemented in AFXFileItem.

isExecutable()

Return True if this is an executable item.

isFifo()

Return True if this is an FIFO item.

isFile()

Return True if this is a file item.

isSocket()

Return True if this is a socket.

isSymlink()

Return True if this is a symbolic link item.

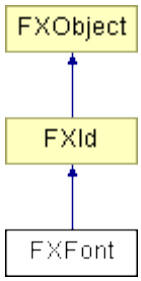
Global flags

File List options

FILELIST_SHOWHIDDEN	**Show hidden files or directories.**
FILELIST_SHOWDIRS	**Show only directories.**
FILELIST_NO_OWN_ASSOC	**Do not create associations for files.**

FXFont

Font class



FXFont(a, fontdesc)

Construct font from font description.

Argument	Type	Default	Description
a	FXApp		
fontdesc	FXFontDesc		

FXFont(a, face, sz, wt=FONTWEIGHT_NORMAL, sl=FONTSLANT_REGULAR, enc=FONTENCODING_DEFAULT, setw=FONTSETWIDTH_DONTCARE, h=0)

Construct a font with given face name, size in points(pixels), weight, slant, character set encoding, setwidth, and hints.

Argument	Type	Default	Description
a	FXApp		
face	String		
sz	Int		
wt	Int	FONTWEIGHT_NORMAL	
sl	Int	FONTSLANT_REGULAR	
enc	Int	FONTENCODING_DEFAULT	
setw	Int	FONTSETWIDTH_DONTCARE	
h	Int	0	

FXFont(a, nm)

Construct a font with given X11 font string.

Argument	Type	Default	Description
a	FXApp		
nm	String		

create()

Create the font.

Reimplemented from FXId.

destroy()

Destroy the font.

Reimplemented from FXId.

detach()

Detach the font.

Reimplemented from FXId.

getEncoding()

Get character set encoding.

getFontAscent()

Ascent from baseline.

getFontDesc(fontdesc)

Get font description.

Argument	Type	Default	Description
fontdesc	FXFontDesc		

getFontDescent()

Descent from baseline.

getFontHeight()

Height of highest character in font.

getFontLeading()

Get font leading [that is lead-ing as in Pb!].

getFontSpacing()

Get font line spacing.

getFontWidth()

Width of widest character in font.

getHints()

Get hints.

getMaxChar()

Get last character glyph in font.

getMinChar()

Get first character glyph in font.

getName()

Get face name.

getSetWidth()

Get setwidth.

getSize()

Get size in deci-points.

getSlant()

Get slant.

getTextHeight(text, n)

Calculate height of given text in this font.

Argument	Type	Default	Description
text	String		The string whose height is being evaluated.
n	Int		The number of characters in 'text,' starting from the left end, for which the height will be returned.

getTextWidth(text, n)

Calculate width of given text in this font.

Argument	Type	Default	Description
text	String		The string whose width is being evaluated.
n	Int		The number of characters in 'text,' starting from the left end, for which the width will be returned.

getWeight()

Get font weight.

hasChar(ch)

See if font has glyph for ch.

Argument	Type	Default	Description
ch	Int		

isFontMono()

Find out if the font is monotype or proportional.

leftBearing(ch)

Left bearing.

Argument	Type	Default	Description
ch	String		

rightBearing(ch)

Right bearing.

Argument	Type	Default	Description
ch	String		

setFontDesc(fontdesc)

Change font description.

Argument	Type	Default	Description
fontdesc	FXFontDesc		

Global flags

Font style hints which influence the matcher

FONTPITCH_DEFAULT Default pitch.

FONTPITCH_FIXED Fixed pitch, mono-spaced.

FONTPITCH_VARIABLE Variable pitch, proportional spacing.

FONTHINT_DONTCARE Don't care which font.

FONTHINT_DECORATIVE Fancy fonts.

FONTHINT_MODERN Monospace typewriter font.

FONTHINT_ROMAN Variable width times-like font, serif.

FONTHINT_SCRIPT Script or cursive.

FONTHINT_SWISS Helvetica/swiss type font, sans-serif.

FONTHINT_SYSTEM System font.

FONTHINT_X11 X11 Font string.

FONTHINT_SCALABLE Scalable fonts.

FONTHINT_POLYMORPHIC Polymorphic fonts.

Font slant

FONTSLANT_DONTCARE Don't care about slant.

FONTSLANT_REGULAR Regular straight up.

FONTSLANT_ITALIC Italics.

FONTSLANT_OBLIQUE Oblique slant.

FONTSLANT_REVERSE_ITALIC Reversed italic.

FONTSLANT_REVERSE_OBLIQUE Reversed oblique.

Font character set encoding

FONTENCODING_DEFAULT Don't care character encoding.

FONTENCODING_ISO_8859_5

	Cyrillic (almost obsolete).
FONTENCODING_KOI8_R	**Russian.**
FONTENCODING_KOI8_U	**Ukrainian.**
FONTENCODING_LATIN1	**Latin 1 (West European).**
FONTENCODING_LATIN2	**Latin 2 (East European).**
FONTENCODING_LATIN3	**Latin 3 (South European).**
FONTENCODING_LATIN4	**Latin 4 (North European).**
FONTENCODING_LATIN5	**Latin 5 (Turkish).**
FONTENCODING_LATIN6	**Latin 6 (Nordic).**
FONTENCODING_LATIN7	**Latin 7 (Baltic Rim).**
FONTENCODING_LATIN8	**Latin 8 (Celtic).**
FONTENCODING_LATIN9	**Latin 9 AKA Latin 0.**
FONTENCODING_LATIN10	**Latin 10.**
FONTENCODING_USASCII	**Latin 1.**
FONTENCODING_WESTEUROPE	**Latin 1 (West European).**
FONTENCODING_EASTEUROPE	**Latin 2 (East European).**
FONTENCODING_SOUTHEUROPE	**Latin 3 (South European).**
FONTENCODING_NORTHEUROPE	**Latin 4 (North European).**
FONTENCODING_CYRILLIC	**Cyrillic.**
FONTENCODING_RUSSIAN	**Cyrillic.**
FONTENCODING_ARABIC	**Arabic.**
FONTENCODING_GREEK	**Greek.**
FONTENCODING_HEBREW	**Hebrew.**
FONTENCODING_TURKISH	**Latin 5 (Turkish).**
FONTENCODING_NORDIC	**Latin 6 (Nordic).**
FONTENCODING_THAI	**Thai.**
FONTENCODING_BALTIC	**Latin 7 (Baltic Rim).**
FONTENCODING_CELTIC	**Latin 8 (Celtic).**

Font weight

FONTWEIGHT_DONTCARE	**Don't care about weight.**
FONTWEIGHT_THIN	**Thin.**
FONTWEIGHT_EXTRALIGHT	**Extra light.**
FONTWEIGHT_LIGHT	**Light.**
FONTWEIGHT_NORMAL	**Normal or regular weight.**
FONTWEIGHT_REGULAR	**Normal or regular weight.**
FONTWEIGHT_MEDIUM	**Medium bold face.**
FONTWEIGHT_DEMIBOLD	**Demi bold face.**
FONTWEIGHT_BOLD	**Bold face.**
FONTWEIGHT_EXTRABOLD	**Extra.**
FONTWEIGHT_HEAVY	**Heavy.**
FONTWEIGHT_BLACK	**Black.**

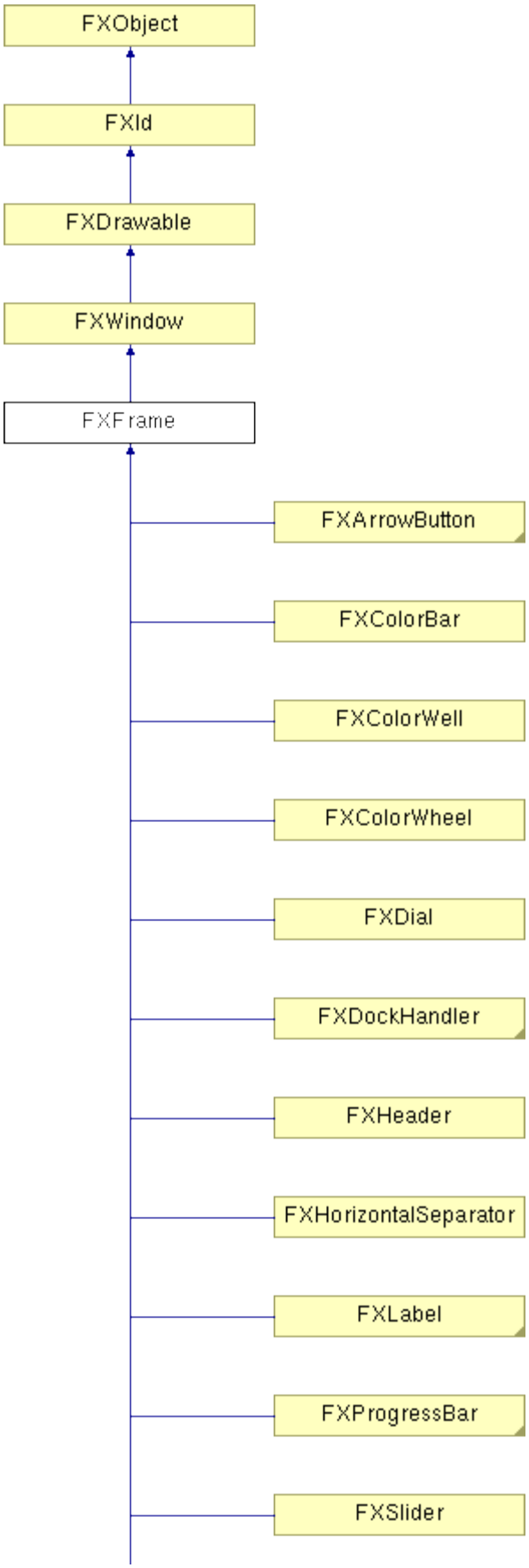
Font relative setwidth

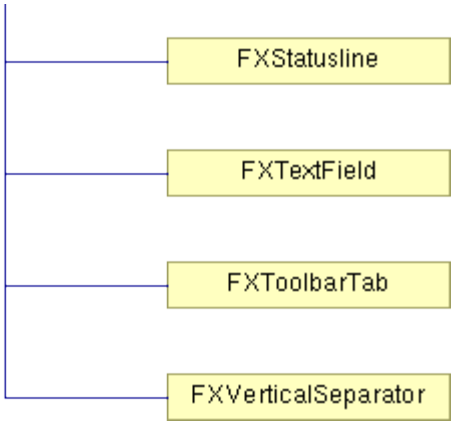
FONTSETWIDTH_DONTCARE	**Don't care about set width.**

FONTSETWIDTH_ULTRACONDENSED	**Ultra condensed printing.**
FONTSETWIDTH_EXTRACONDENSED	**Extra condensed.**
FONTSETWIDTH_CONDENSED	**Condensed.**
FONTSETWIDTH_NARROW	**Narrow.**
FONTSETWIDTH_COMPRESSED	**Compressed.**
FONTSETWIDTH_SEMICONDENSED	**Semi-condensed.**
FONTSETWIDTH_MEDIUM	**Medium printing.**
FONTSETWIDTH_NORMAL	**Normal printing.**
FONTSETWIDTH_REGULAR	**Regular printing.**
FONTSETWIDTH_SEMIEXPANDED	**Semi expanded.**
FONTSETWIDTH_EXPANDED	**Expanded.**
FONTSETWIDTH_WIDE	**Wide.**
FONTSETWIDTH_EXTRAEXPANDED	**Extra expanded.**
FONTSETWIDTH_ULTRAEXPANDED	**Ultra expanded.**

FXFrame

Base Frame





FXFrame(p, opts=FRAME_NORMAL, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Construct frame window.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	FRAME_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_PAD	
pr	Int	DEFAULT_PAD	
pt	Int	DEFAULT_PAD	
pb	Int	DEFAULT_PAD	

getBaseColor()

Get base gui color.

getBorderColor()

Get border color.

getBorderWidth()

Get border width.

getDefaultHeight()

Return default height.

Reimplemented from FXWindow.

Reimplemented in FXArrowButton, FXCheckButton, FXColorBar, FXColorWell, FXColorWheel, FXDial, FXDockTitle, FXHeader, FXLabel, FXMDIDeleteButton, FXMDIRestoreButton, FXMDIMaximizeButton, FXMDIMinimizeButton, FXMDIWindowButton, FXMenuButton, FXProgressBar, FXOption, FXOptionMenu, FXRadioButton, FXHorizontalSeparator, FXVerticalSeparator, FXSlider, FXStatusline, FXTextField, FXToggleButton, FXToolbarGrip, FXToolbarTab, and AFXProgressBar.

getDefaultWidth()

Return default width.

Reimplemented from FXWindow.

Reimplemented in FXArrowButton, FXCheckButton, FXColorBar, FXColorWell, FXColorWheel, FXDial, FXDockTitle, FXHeader, FXLabel, FXMDIDeleteButton, FXMDIRestoreButton, FXMDIMaximizeButton, FXMDIMinimizeButton, FXMDIWindowButton, FXMenuButton, FXProgressBar, FXOption, FXOptionMenu, FXRadioButton, FXHorizontalSeparator, FXVerticalSeparator, FXSlider, FXStatusline, FXTextField, FXToggleButton, FXToolbarGrip, FXToolbarTab, and AFXProgressBar.

getFrameStyle()

Get current frame style.

getHiliteColor()

Get highlight color.

getPadBottom()

Get bottom interior padding.

getPadLeft()

Get left interior padding.

getPadRight()

Get right interior padding.

getPadTop()

Get top interior padding.

getShadowColor()

Get shadow color.

setBaseColor(clr)

Change base gui color.

Argument	Type	Default	Description
clr	FXColor		

setBorderColor(clr)

Change border color.

Argument	Type	Default	Description
clr	FXColor		

setFrameStyle(style)

Change frame style.

Argument	Type	Default	Description
style	Int		

setHiliteColor(clr)

Change highlight color.

Argument	Type	Default	Description
clr	FXColor		

setPadBottom(pb)

Change bottom padding.

Argument	Type	Default	Description
pb	Int		

setPadLeft(pl)

Change left padding.

Argument	Type	Default	Description
pl	Int		

setPadRight(pr)

Change right padding.

Argument	Type	Default	Description
pr	Int		

setPadTop(pt)

Change top padding.

Argument	Type	Default	Description
pt	Int		

setShadowColor(clr)

Change shadow color.

Argument	Type	Default	Description
clr	FXColor		

Global flags

Justification modes used by certain subclasses

JUSTIFY_NORMAL **Default justification is centered text.**

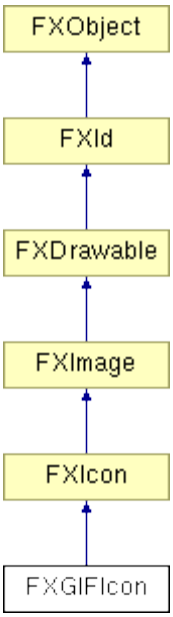
JUSTIFY_CENTER_X **Contents centered horizontally.**

JUSTIFY_LEFT **Contents left-justified.**

JUSTIFY_RIGHT	**Contents right-justified.**
JUSTIFY_HZ_APART	**Combination of JUSTIFY_LEFT & JUSTIFY_RIGHT.**
JUSTIFY_CENTER_Y	**Contents centered vertically.**
JUSTIFY_TOP	**Contents aligned with label top.**
JUSTIFY_BOTTOM	**Contents aligned with label bottom.**
JUSTIFY_VT_APART	**Combination of JUSTIFY_TOP & JUSTIFY_BOTTOM.**

FXGIFIcon

GIF Icon class.



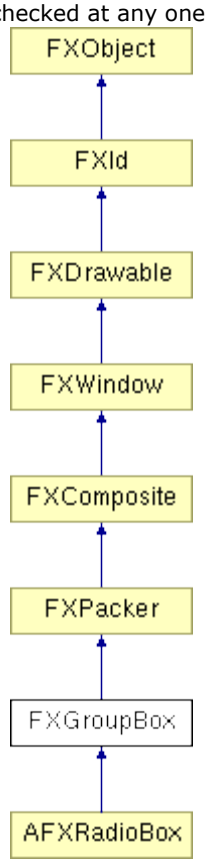
FXGIFIcon(a, pix=None, clr=FXRGB(192, 192, 192), opts=0, w=1, h=1)

Construct an icon from memory stream formatted as CompuServe GIF format.

Argument	Type	Default	Description
a	FXApp		
pix		None	
clr	FXColor	FXRGB(192, 192, 192)	
opts	Int	0	
w	Int	1	
h	Int	1	

FXGroupBox

A group box widget provides a nice raised or sunken border around a group of widgets, providing a visual delineation. Typically, a title is placed over the border to provide some clarification. Radio buttons placed inside a group box automatically assume mutually exclusive behaviour, i.e. at most one radio button will be checked at any one time.



FXGroupBox(p, text, opts=GROUPBOX_NORMAL, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Construct group box layout manager.

Argument	Type	Default	Description
p	FXComposite		
text	String		
opts	Int	GROUPBOX_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_SPACING	
pr	Int	DEFAULT_SPACING	
pt	Int	DEFAULT_SPACING	
pb	Int	DEFAULT_SPACING	
hs	Int	DEFAULT_SPACING	

vs	Int	DEFAULT_SPACING	

create()

Create server-side resources.

Reimplemented from FXComposite.

detach()

Detach server-side resources.

Reimplemented from FXComposite.

disable()

Disable the window.

Reimplemented from FXWindow.

enable()

Enable the window.

Reimplemented from FXWindow.

getDefaultHeight()

Return default height.

Reimplemented from FXPacker.

getDefaultWidth()

Return default width.

Reimplemented from FXPacker.

getText()

Return current groupbox title text.

setText(text)

Change group box title text.

Argument	Type	Default	Description
text	String		

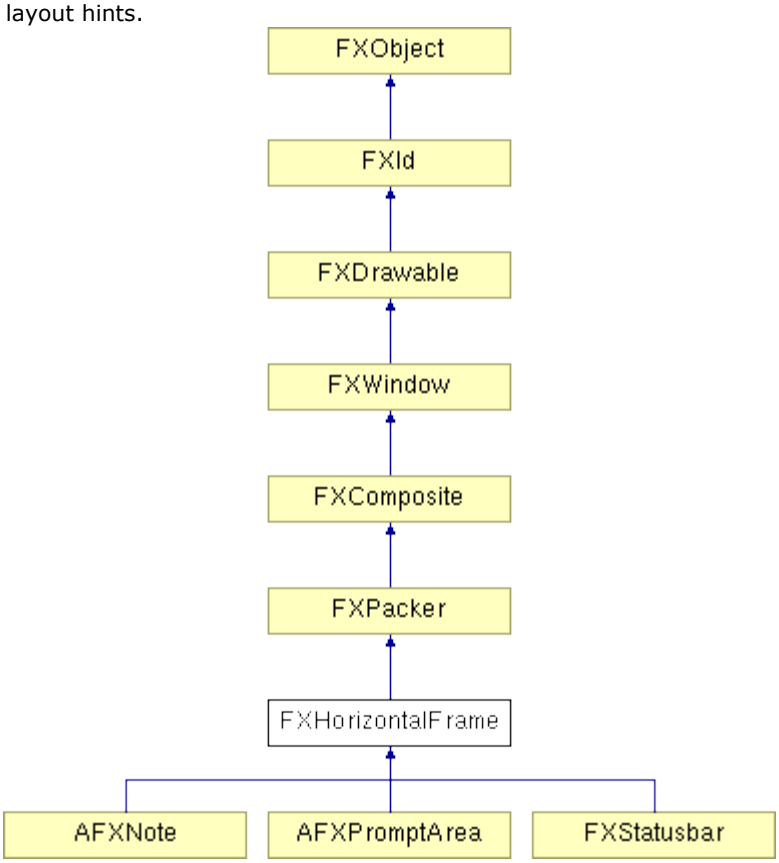
Global flags

Group box options

GROUPBOX_TITLE_LEFT	**Title is left-justified.**
GROUPBOX_TITLE_CENTER	**Title is centered.**
GROUPBOX_TITLE_RIGHT	**Title is right-justified.**

FXHorizontalFrame

Horizontal frame layout manager widget is used to automatically place child-windows horizontally from left-to-right, or right-to-left, depending on the child window's layout hints.



FXHorizontalFrame(p, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Construct a horizontal frame layout manager.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_SPACING	
pr	Int	DEFAULT_SPACING	
pt	Int	DEFAULT_SPACING	
pb	Int	DEFAULT_SPACING	
hs	Int	DEFAULT_SPACING	
vs	Int	DEFAULT_SPACING	

getDefaultHeight()

Return default height.

Reimplemented from FXPacker.

Reimplemented in FXStatusbar.

getDefaultWidth()

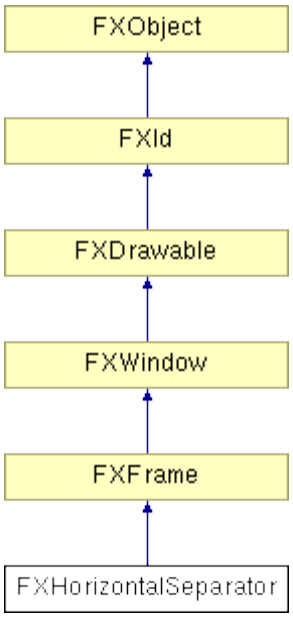
Return default width.

Reimplemented from FXPacker.

Reimplemented in FXStatusbar.

FXHorizontalSeparator

Horizontal separator



FXHorizontalSeparator(p, opts=SEPARATOR_GROOVE| LAYOUT_FILL_X, x=0, y=0, w=0, h=0, pl=1, pr=1, pt=0, pb=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	SEPARATOR_GROOVE| LAYOUT_FILL_X	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	1	
pr	Int	1	
pt	Int	0	
pb	Int	0	

getDefaultHeight()

Return default height.

Reimplemented from FXFrame.

getDefaultWidth()

Return default width.

Reimplemented from FXFrame.

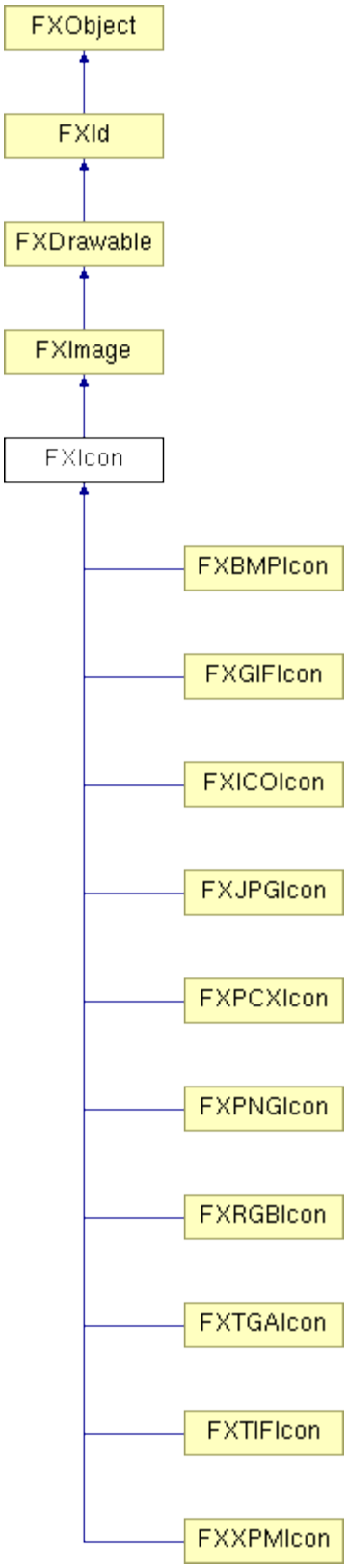
Global flags

Separator Options

SEPARATOR_NONE	**Nothing visible.**
SEPARATOR_GROOVE	**Etched-in looking groove.**
SEPARATOR_RIDGE	**Embossed looking ridge.**
SEPARATOR_LINE	**Simple line.**

FXIcon

Icon class.



FXIcon(a, pix=None, clr=0, opts=0, w=1, h=1)

Create an icon with an initial pixel buffer pix, a transparent color clr, and options as in FXImage.

Argument	Type	Default	Description
a	FXApp		
pix		None	
clr	FXColor	0	
opts	Int	0	
w	Int	1	
h	Int	1	

create()

Create the icon resource.

Reimplemented from FXImage.

destroy()

Destroy the icon resource.

Reimplemented from FXImage.

detach()

Detach the icon resource.

Reimplemented from FXImage.

render()

Render the image from client-side pixel buffer.

Reimplemented from FXImage.

resize(w, h)

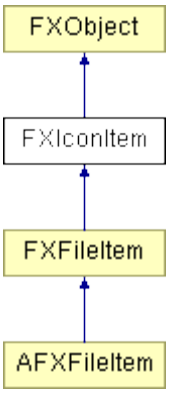
Resize pixmap to the specified width and height.

Reimplemented from FXImage.

Argument	Type	Default	Description
w	Int		
h	Int		

FXIconItem

Icon item



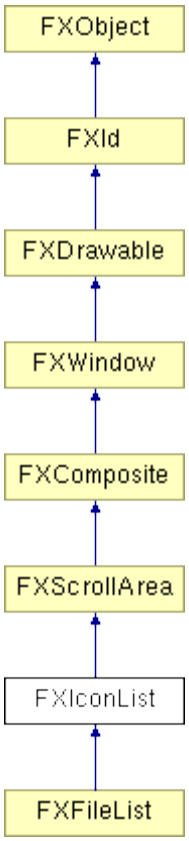
Global flags

Icon list styles

ICONLIST_EXTENDEDSELECT	**Extended selection mode.**
ICONLIST_SINGLESELECT	**At most one selected item.**
ICONLIST_BROWSESELECT	**Always exactly one selected item.**
ICONLIST_MULTIPLESELECT	**Multiple selection mode.**
ICONLIST_AUTOSIZE	**Automatically size item spacing.**
ICONLIST_DETAILED	**List mode.**
ICONLIST_MINI_ICONS	**Mini Icon mode.**
ICONLIST_BIG_ICONS	**Big Icon mode.**
ICONLIST_ROWS	**Row-wise mode.**
ICONLIST_COLUMNS	**Column-wise mode.**

FXIconList

Icon List Widget



FXIconList(p, tgt=None, sel=0, opts=ICONLIST_NORMAL, x=0, y=0, w=0, h=0)

Construct icon list.

Argument	Type	Default	Description
p	FXComposite		
tgt	FXObject	None	
sel	Int	0	
opts	Int	ICONLIST_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

appendHeader(text, icon=None, size=1)

Append header with given text and optional icon.

Argument	Type	Default	Description
text	String		
icon	FXIcon	None	

size	Int	1	

appendItem(text, big=None, mini=None, ptr=None, notify=False)

Append new item with given text and optional icons, and user-data pointer.

Argument	Type	Default	Description
text	String		
big	FXIcon	None	
mini	FXIcon	None	
ptr	String	None	
notify	Bool	False	

appendItem(item, notify=False)

Append a [possibly subclassed] item to the end of the list.

Argument	Type	Default	Description
item	FXIconItem		
notify	Bool	False	

canFocus()

Icon list can receive focus.

Reimplemented from FXWindow.

clearItems(notify=False)

Remove all items from list.

Argument	Type	Default	Description
notify	Bool	False	

create()

Create server-side resources.

Reimplemented from FXComposite.

Reimplemented in FXFileList.

deselectItem(index, notify=False)

Deselect item at index.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

detach()

Detach server-side resources.

Reimplemented from FXComposite.

Reimplemented in FXFileList.

disableItem(index)

Disable item at index.

Argument	Type	Default	Description
index	Int		

enableItem(index)

Enable item at index.

Argument	Type	Default	Description
index	Int		

extendSelection(index, notify=False)

Extend selection from anchor index to index.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

findItem(text, start=-1, flags=SEARCH_FORWARD| SEARCH_WRAP)

Search items for item by name, starting from start item; the flags argument controls the search direction, and case sensitivity.

Argument	Type	Default	Description
text	String		
start	Int	-1	
flags	Int	SEARCH_FORWARD| SEARCH_WRAP	

getAnchorItem()

Return anchor item index, or -1 if none.

getContentHeight()

Return content height.

Reimplemented from FXScrollArea.

getContentWidth()

Compute and return content width.

Reimplemented from FXScrollArea.

getCurrentItem()

Return current item index, or -1 if none.

getCursorItem()

Return index of item under cursor, or -1 if none.

getFont()

Return text font.

getHeader()

Return header control.

getHeaderIcon(index)

Return icon of header at index.

Argument	Type	Default	Description
index	Int		

getHeaderSize(index)

Return width of header at index.

Argument	Type	Default	Description
index	Int		

getHeaderText(index)

Return text of header at index.

Argument	Type	Default	Description
index	Int		

getHelpText()

Get the status line help text for this widget.

getItemAt(x, y)

Return index of item at x,y, or -1 if none.

Argument	Type	Default	Description
x	Int		
y	Int		

getItemBigIcon(index)

Return big icon of item at index.

Argument	Type	Default	Description
index	Int		

getItemData(index)

Return item user-data pointer.

Argument	Type	Default	Description
index	Int		

getItemHeight()

Return item height.

getItemMiniIcon(index)

Return mini icon of item at index.

Argument	Type	Default	Description
index	Int		

getItemSpace()

Return maximum item space.

getItemText(index)

Return item text.

Argument	Type	Default	Description
index	Int		

getItemWidth()

Return item width.

getListStyle()

Get the current icon list style.

getNumCols()

Return number of columns.

getNumHeaders()

Return number of headers.

getNumItems()

Return number of items.

getNumRows()

Return number of rows.

getSelBackColor()

Return selected text background.

getSelTextColor()

Return selected text color.

getSortFunc()

Return sort function.

getTextColor()

Return normal text color.

getViewportHeight()

Return viewport size.

Reimplemented from FXScrollArea.

hitItem(index, x, y, ww=1, hh=1)

Return item hit code: 0 outside, 1 icon, 2 text.

Argument	Type	Default	Description
index	Int		
x	Int		
y	Int		
ww	Int	1	
hh	Int	1	

insertItem(index, text, big=None, mini=None, ptr=None, notify=False)

Insert item at index with given text, icons, and user-data pointer.

Argument	Type	Default	Description
index	Int		
text	String		
big	FXIcon	None	
mini	FXIcon	None	
ptr	String	None	
notify	Bool	False	

insertItem(index, item, notify=False)

Insert a new [possibly subclassed] item at the give index.

Argument	Type	Default	Description
index	Int		

item	FXIconItem		
notify	Bool	False	

isItemCurrent(index)

Return True if item at index is current.

Argument	Type	Default	Description
index	Int		

isItemEnabled(index)

Return True if item at index is enabled.

Argument	Type	Default	Description
index	Int		

isItemSelected(index)

Return True if item at index is selected.

Argument	Type	Default	Description
index	Int		

isItemVisible(index)

Return True if item at index is visible.

Argument	Type	Default	Description
index	Int		

killFocus()

Remove the focus from this window.

Reimplemented from FXWindow.

killSelection(notify=False)

Deselect all items.

Argument	Type	Default	Description
notify	Bool	False	

makeItemVisible(index)

Scroll to make item at index visible.

Argument	Type	Default	Description
index	Int		

moveContents(x, y)

Move contents to the specified position.

Reimplemented from FXScrollArea.

Argument	Type	Default	Description
x	Int		
y	Int		

position(x, y, w, h)

Move and resize this window in the parent's coordinates.

Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		

prependItem(text, big=None, mini=None, ptr=None, notify=False)

Append new item with given text and optional icons, and user-data pointer.

Argument	Type	Default	Description
text	String		
big	FXIcon	None	
mini	FXIcon	None	
ptr	String	None	
notify	Bool	False	

prependItem(item, notify=False)

Append a [possibly subclassed] item to the end of the list.

Argument	Type	Default	Description
item	FXIconItem		
notify	Bool	False	

recalc()

Recalculate layout.

Reimplemented from FXWindow.

removeHeader(index)

Remove header at index.

Argument	Type	Default	Description
index	Int		

removeItem(index, notify=False)

Remove item from list.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

replaceItem(index, text, big=None, mini=None, ptr=None, notify=False)

Replace items text, icons, and user-data pointer.

Argument	Type	Default	Description
index	Int		
text	String		
big	FXIcon	None	
mini	FXIcon	None	
ptr	String	None	
notify	Bool	False	

replaceItem(index, item, notify=False)

Replace the item with a [possibly subclassed] item.

Argument	Type	Default	Description
index	Int		
item	FXIconItem		
notify	Bool	False	

resize(w, h)

Resize this window to the specified width and height.

Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

retrieveItem(index)

Return the item at the given index.

Argument	Type	Default	Description
index	Int		

selectInRectangle(x, y, w, h, notify=False)

Select items in rectangle.

Argument	Type	Default	Description
x	Int		

y	Int		
w	Int		
h	Int		
notify	Bool	False	

selectItem(index, notify=False)

Select item at index.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

setAnchorItem(index)

Change anchor item index.

Argument	Type	Default	Description
index	Int		

setCurrentItem(index, notify=False)

Change current item index.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

setFocus()

Move the focus to this window.

Reimplemented from FXWindow.

setFont(fnt)

Change text font.

Argument	Type	Default	Description
fnt	FXFont		

setHeaderIcon(index, icon)

Change icon of header at index.

Argument	Type	Default	Description
index	Int		
icon	FXIcon		

setHeaderSize(index, size)

Change size of header at index.

Argument	Type	Default	Description
index	Int		
size	Int		

setHeaderText(index, text)

Change text of header at index.

Argument	Type	Default	Description
index	Int		
text	String		

setHelpText(text)

Set the status line help text for this widget.

Argument	Type	Default	Description
text	String		

setItemBigIcon(index, icon)

Change item big icon.

Argument	Type	Default	Description
index	Int		
icon	FXIcon		

setItemData(index, ptr)

Change item user-data pointer.

Argument	Type	Default	Description
index	Int		
ptr	String		

setItemMiniIcon(index, icon)

Change item mini icon.

Argument	Type	Default	Description
index	Int		
icon	FXIcon		

setItemSpace(s)

Change maximum item space for each item.

Argument	Type	Default	Description
s	Int		

setItemText(index, text)

Change item text.

Argument	Type	Default	Description
index	Int		
text	String		

setListStyle(style)

Set the current icon list style.

Argument	Type	Default	Description
style	Int		

setSelBackColor(clr)

Change selected text background.

Argument	Type	Default	Description
clr	FXColor		

setSelTextColor(clr)

Change selected text color.

Argument	Type	Default	Description
clr	FXColor		

setSortFunc(func)

Change sort function.

Argument	Type	Default	Description
func	FXIconListSortFunc		

setTextColor(clr)

Change normal text color.

Argument	Type	Default	Description
clr	FXColor		

sortItems()

Sort items.

toggleItem(index, notify=False)

Toggle item at index.

Argument	Type	Default	Description
index	Int		

notify	Bool	False	

updateItem(index)

Repaint item at index.

Argument	Type	Default	Description
index	Int		

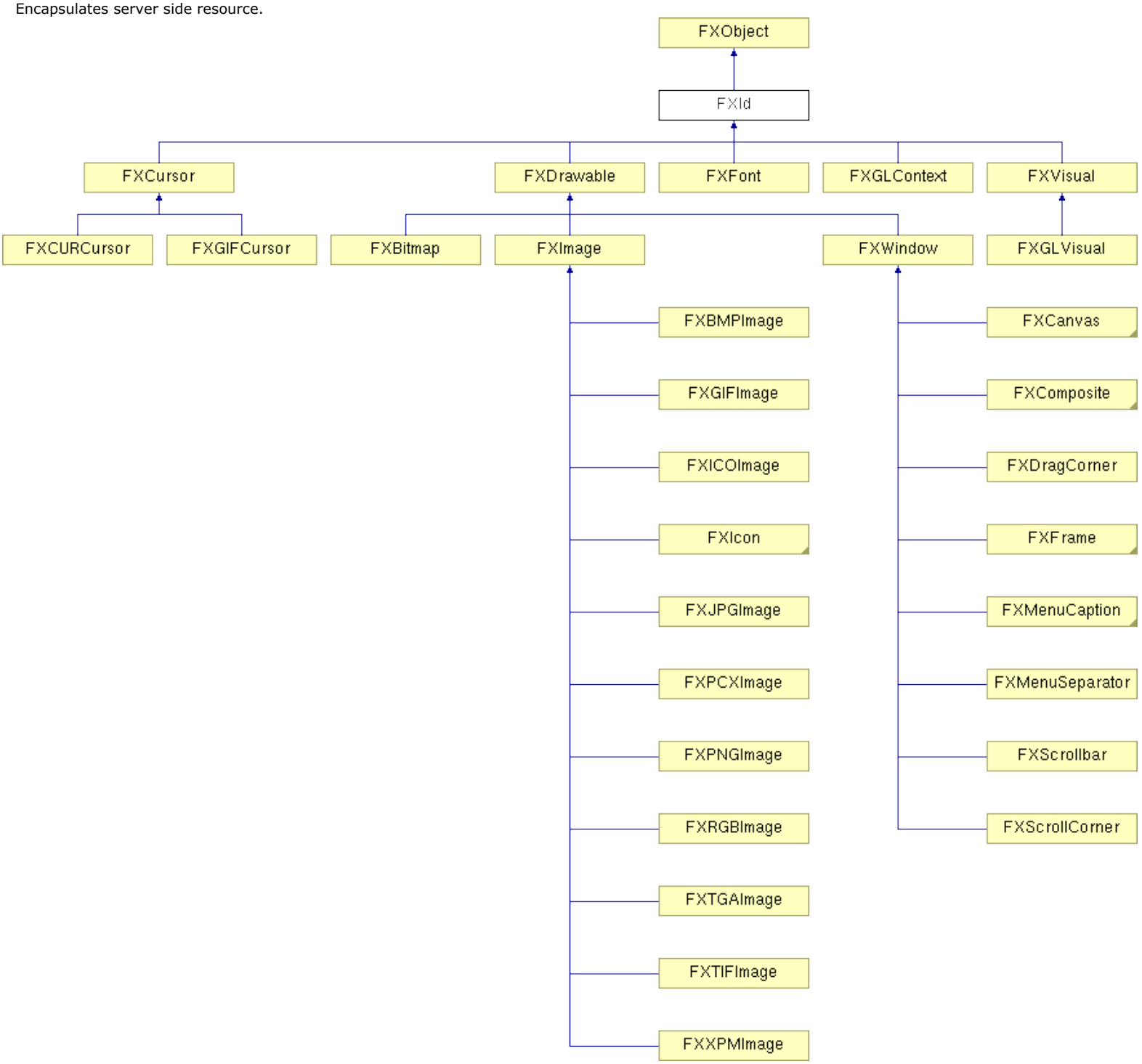
Global flags

Icon list styles

ICONLIST_EXTENDEDSELECT	**Extended selection mode.**
ICONLIST_SINGLESELECT	**At most one selected item.**
ICONLIST_BROWSESELECT	**Always exactly one selected item.**
ICONLIST_MULTIPLESELECT	**Multiple selection mode.**
ICONLIST_AUTOSIZE	**Automatically size item spacing.**
ICONLIST_DETAILED	**List mode.**
ICONLIST_MINI_ICONS	**Mini Icon mode.**
ICONLIST_BIG_ICONS	**Big Icon mode.**
ICONLIST_ROWS	**Row-wise mode.**
ICONLIST_COLUMNS	**Column-wise mode.**

FXId

Encapsulates server side resource.



create()

Create resource.

Reimplemented in FXBitmap, FXColorBar, FXColorSelector, FXColorWell, FXColorWheel, FXComboBox, FXComposite, FXCursor, FXDirBox, FXDirList, FXDockTitle, FXDriveBox, FXFileList, FXFont, FXFontSelector, FXGLCanvas, FXGLContext, FXGLViewer, FXGLVisual, FXGroupBox, FXHeader, FXIcon, FXIconList, FXImage, FXImageView, FXLabel, FXList, FXListBox, FXMDIChild, FXMenuButton, FXMenuCaption, FXMenuCascade, FXProgressBar, FXMenuTitle, FXOptionMenu, FXPrintDialog, FXRootWindow, FXScrollWindow, FXShell, FXSpinner, FXStatusline, FXTabBar, FXTable, FXText, FXTextField, FXToggleButton, FXToolbarShell, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, FXVisual, FXWindow, AFXManagerMenuPane, AFXMainWindow, AFXPromptArea, AFXBaseTable, AFXColorButton, AFXColorFlyout, AFXComboBox, AFXDialog, AFXFloatSpinner, AFXFlyoutButton, AFXListBox, AFXNote, AFXOptionTreeItem, AFXPrimFloatSpinner, AFXProgressBar, AFXSpinner, AFXTable, AFXTextField, and AFXVerticalAligner.

destroy()

Destroy resource.

Reimplemented in FXBitmap, FXComboBox, FXComposite, FXCursor, FXDirBox, FXDirList, FXDriveBox, FXFileList, FXFont, FXGLCanvas, FXGLContext, FXGLVisual, FXIcon, FXImage, FXListBox, FXMenuCascade, FXOptionMenu, FXRootWindow, FXTreeList, FXTreeListBox, FXVisual, FXWindow, AFXManagerMenuCascade, AFXColorFlyout, and AFXTable.

detach()

Detach resource.

Reimplemented in FXBitmap, FXColorBar, FXColorWell, FXColorWheel, FXComboBox, FXComposite, FXCursor, FXDirBox, FXDirList, FXDockTitle, FXDriveBox, FXFileList, FXFont, FXGLCanvas, FXGLContext, FXGLViewer, FXGLVisual, FXGroupBox, FXHeader, FXIcon, FXIconList, FXImage, FXImageView, FXLabel, FXList, FXListBox, FXMDIChild, FXMenuButton, FXMenuCaption, FXMenuCascade, FXProgressBar, FXMenuTitle, FXOptionMenu, FXRootWindow, FXStatusline, FXTable, FXText, FXToggleButton, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, FXVisual, FXWindow, AFXBaseTable, AFXColorFlyout, AFXFlyoutButton, AFXNote, and AFXTable.

getApp()

Get application.

getUserData()

Get user data pointer.

id()

Get XID handle.

Reimplemented in FXFont.

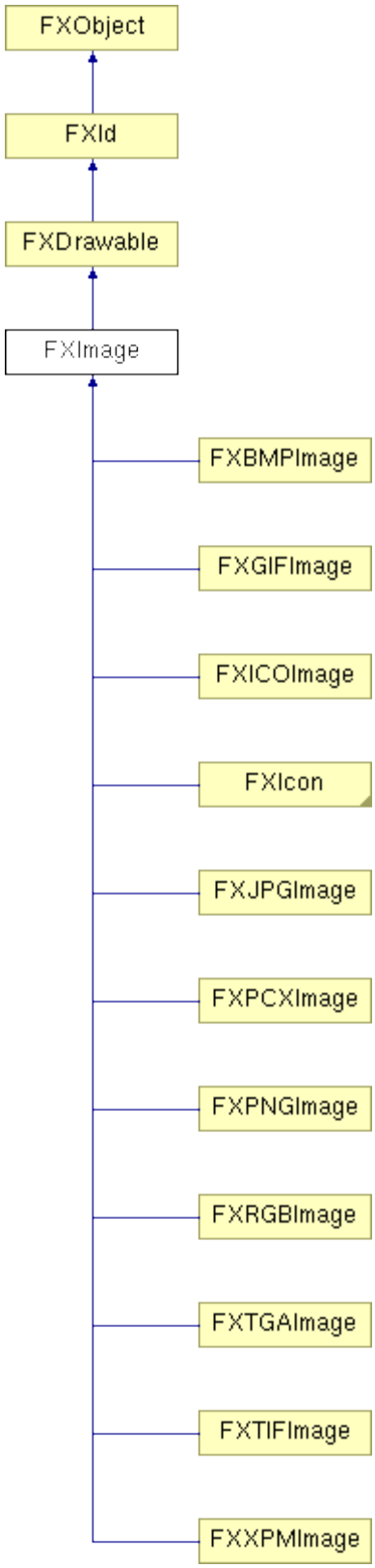
setUserData(ptr)

Set user data pointer.

Argument	Type	Default	Description
ptr	String		

FXImage

Image class



FXImage(a, pix=None, opts=0, w=1, h=1)

Create an image.

Argument	Type	Default	Description
a	FXApp		
pix		None	
opts	Int	0	
w	Int	1	
h	Int	1	

blend(color, sharpen=True)

Blends the icon with the specified color; should only be used on icons that support an alpha channel, for example, PNG.

Argument	Type	Default	Description
color	FXColor		
sharpen	Bool	True	

create()

Create image resource.

Reimplemented from FXId.

Reimplemented in FXIcon.

destroy()

Destroy image resource.

Reimplemented from FXId.

Reimplemented in FXIcon.

detach()

Detach image resource.

Reimplemented from FXId.

Reimplemented in FXIcon.

getOptions()

To get to the option flags.

getPixel(x, y)

Get pixel at x,y.

Argument	Type	Default	Description
x	Int		
y	Int		

render()

Render the image from client-side pixel buffer.

Reimplemented in FXIcon.

resize(w, h)

Resize pixmap to the specified width and height.

Reimplemented from FXDrawable.

Reimplemented in FXIcon.

Argument	Type	Default	Description
w	Int		
h	Int		

scale(w, h)

Rescale pixels image to the specified width and height.

Argument	Type	Default	Description
w	Int		
h	Int		

setPixel(x, y, color)

Change pixel at x,y.

Argument	Type	Default	Description
x	Int		
y	Int		
color	FXColor		

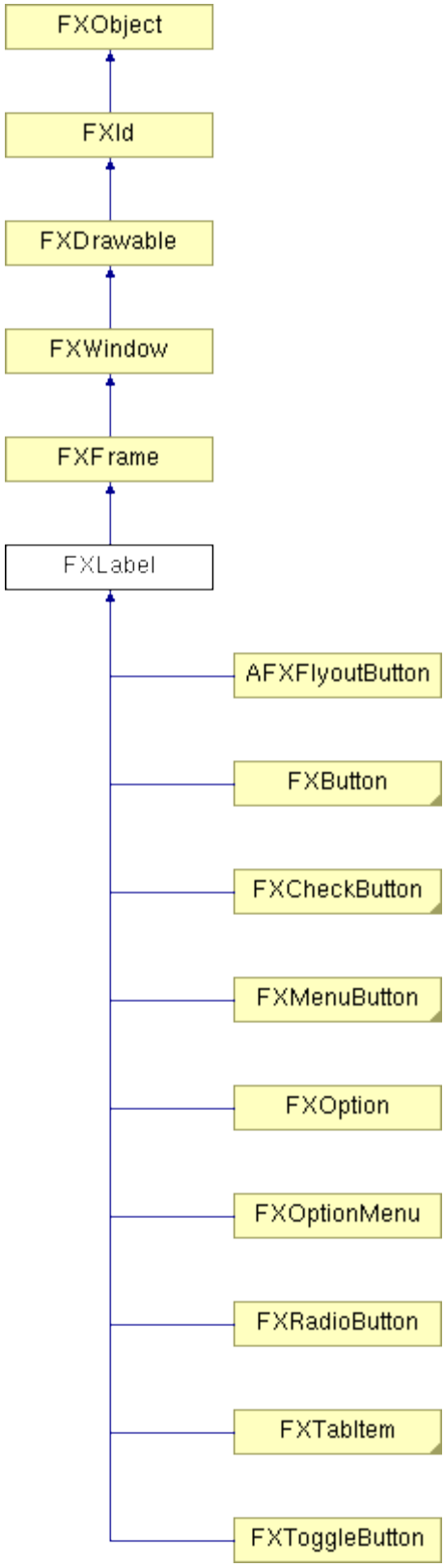
Global flags

Image rendering hints

IMAGE_KEEP	**Keep pixel data in client.**
IMAGE_OWNED	**Pixel data is owned by image.**
IMAGE_DITHER	**Dither image to look better.**
IMAGE_NEAREST	**Turn off dithering and map to nearest color.**
IMAGE_ALPHA	**Data has alpha channel.**
IMAGE_OPAQUE	**Force opaque background.**
IMAGE_ALPHACOLOR	**Override transparancy color.**
IMAGE_SHMI	**Using shared memory image.**
IMAGE_SHMP	**Using shared memory pixmap.**
IMAGE_ALPHAGUESS	**Guess transparency color from corners.**

FXLabel

A label widget can be used to place a text and/or icon for explanation purposes. The text label may have an optional tooltip and/or help string.



FXLabel(p, text, ic=None, opts=LABEL_NORMAL, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD,

pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Construct label with given text and icon.

Argument	Type	Default	Description
p	FXComposite		
text	String		
ic	FXIcon	None	
opts	Int	LABEL_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_PAD	
pr	Int	DEFAULT_PAD	
pt	Int	DEFAULT_PAD	
pb	Int	DEFAULT_PAD	

create()

Create server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXMenuButton, FXOptionMenu, FXToggleButton, and AFXFlyoutButton.

detach()

Detach server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXMenuButton, FXOptionMenu, FXToggleButton, and AFXFlyoutButton.

disable()

Disable the window.

Reimplemented from FXWindow.

Reimplemented in AFXFlyoutButton.

enable()

Enable the window.

Reimplemented from FXWindow.

Reimplemented in AFXFlyoutButton.

getDefaultHeight()

Return default height.

Reimplemented from FXFrame.

Reimplemented in FXCheckButton, FXMDIDeleteButton, FXMDIRestoreButton, FXMDIMaximizeButton, FXMDIMinimizeButton, FXMDIWindowButton, FXMenuButton, FXOption, FXOptionMenu, FXRadioButton, and FXToggleButton.

getDefaultWidth()

Return default width.

Reimplemented from FXFrame.

Reimplemented in FXCheckButton, FXMDIDeleteButton, FXMDIRestoreButton, FXMDIMaximizeButton, FXMDIMinimizeButton, FXMDIWindowButton, FXMenuButton, FXOption, FXOptionMenu, FXRadioButton, and FXToggleButton.

getFont()

Get the text font.

getHelpText()

Get the status line help text for this label.

getIcon()

Get the icon for this label.

getIconPosition()

Get the current icon position.

getJustify()

Get the current text-justification mode.

getText()

Get the text for this label.

getTextColor()

Get the current text color.

getTipText()

Get the tool tip message for this label.

setFont(fnt)

Set the text font.

Argument	Type	Default	Description
fnt	FXFont		

setHelpText(text)

Set the status line help text for this label.

Reimplemented in AFXFlyoutItem.

Argument	Type	Default	Description
text	String		

setIcon(ic)

Set the icon for this label.

Reimplemented in AFXFlyoutItem.

Argument	Type	Default	Description
ic	FXIcon		

setIconPosition(mode)

Set the current icon position.

Argument	Type	Default	Description
mode	Int		

setJustify(mode)

Set the current text-justification mode.

Argument	Type	Default	Description
mode	Int		

setText(text)

Set the text for this label.

Reimplemented in AFXFlyoutItem.

Argument	Type	Default	Description
text	String		

setTextColor(clr)

Set the current text color.

Argument	Type	Default	Description
clr	FXColor		

setTipText(text)

Set the tool tip message for this label.

Reimplemented in AFXFlyoutItem.

Argument	Type	Default	Description
text	String		

Global flags

Relationship options for icon-labels

ICON_UNDER_TEXT **Icon appears under text.**

ICON_AFTER_TEXT **Icon appears after text (to its right).**

ICON_BEFORE_TEXT **Icon appears before text (to its left).**

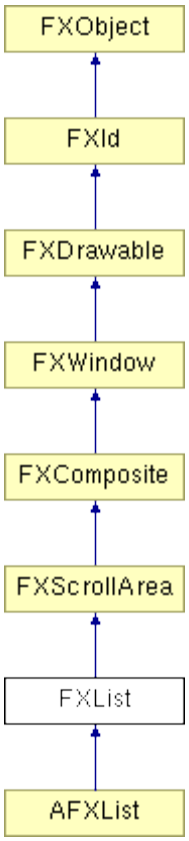
ICON_ABOVE_TEXT	**Icon appears above text.**
ICON_BELOW_TEXT	**Icon appears below text.**
TEXT_OVER_ICON	**Same as ICON_UNDER_TEXT.**
TEXT_AFTER_ICON	**Same as ICON_BEFORE_TEXT.**
TEXT_BEFORE_ICON	**Same as ICON_AFTER_TEXT.**
TEXT_ABOVE_ICON	**Same as ICON_BELOW_TEXT.**
TEXT_BELOW_ICON	**Same as ICON_ABOVE_TEXT.**

Normal way to show label

LABEL_NORMAL	**Combination of JUSTIFY_NORMAL & ICON_BEFORE_TEXT.**

FXList

List Widget



FXList(p, nvis, tgt=None, sel=0, opts=LIST_NORMAL, x=0, y=0, w=0, h=0)

Construct a list with nvis visible items; the list is initially empty.

Argument	Type	Default	Description
p	FXComposite		
nvis	Int		
tgt	FXObject	None	
sel	Int	0	
opts	Int	LIST_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

appendItem(text, icon=None, ptr=None, notify=False)

Append new item with given text and optional icon, and user-data pointer.

Argument	Type	Default	Description
text	String		

icon	FXIcon	None	
ptr	String	None	
notify	Bool	False	

appendItem(item, notify=False)

Append a [possibly subclassed] item to the list.

Argument	Type	Default	Description
item	FXListItem		
notify	Bool	False	

canFocus()

List widget can receive focus.

Reimplemented from FXWindow.

clearItems(notify=False)

Remove all items from list.

Argument	Type	Default	Description
notify	Bool	False	

create()

Create server-side resources.

Reimplemented from FXComposite.

deselectItem(index, notify=False)

Deselect item.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

detach()

Detach server-side resources.

Reimplemented from FXComposite.

findItem(text, start=-1, flags=SEARCH_FORWARD| SEARCH_WRAP)

Search items for item by name, starting from start item; the flags argument controls the search direction, and case sensitivity.

Argument	Type	Default	Description
text	String		
start	Int	-1	
flags	Int	SEARCH_FORWARD| SEARCH_WRAP	

getContentHeight()

Return content height.

Reimplemented from FXScrollArea.

getContentWidth()

Compute and return content width.

Reimplemented from FXScrollArea.

getCurrentItem()

Return current item, if any.

getDefaultHeight()

Return default height.

Reimplemented from FXScrollArea.

Reimplemented in AFXList.

getDefaultWidth()

Return default width.

Reimplemented from FXScrollArea.

getItemData(index)

Return item user-data pointer.

Argument	Type	Default	Description
index	Int		

getItemHeight(index)

Return item height.

Argument	Type	Default	Description
index	Int		

getItemIcon(index)

Return item icon, if any.

Argument	Type	Default	Description
index	Int		

getItemText(index)

Return item text.

Argument	Type	Default	Description
index	Int		

getItemWidth(index)

Return item width.

Argument	Type	Default	Description
index	Int		

getListStyle()

Return list style.

getNumItems()

Return the number of items in the list.

getNumVisible()

Return number of visible items.

insertItem(index, text, icon=None, ptr=None, notify=False)

Insert item at index with given text, icon, and user-data pointer.

Argument	Type	Default	Description
index	Int		
text	String		
icon	FXIcon	None	
ptr	String	None	
notify	Bool	False	

insertItem(index, item, notify=False)

Insert a new [possibly subclassed] item at the give index.

Argument	Type	Default	Description
index	Int		
item	FXListItem		
notify	Bool	False	

isItemSelected(index)

Return True if item is selected.

Argument	Type	Default	Description
index	Int		

isItemVisible(index)

Return True if item is visible.

Argument	Type	Default	Description

index	Int		

killSelection(notify=False)

Deselect all items.

Argument	Type	Default	Description
notify	Bool	False	

makeItemVisible(index)

Scroll to bring item into view.

Argument	Type	Default	Description
index	Int		

recalc()

Recalculate layout.

Reimplemented from FXWindow.

removeItem(index, notify=False)

Remove item from list.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

replaceItem(index, text, icon=None, ptr=None, notify=False)

Replace items text, icon, and user-data pointer.

Argument	Type	Default	Description
index	Int		
text	String		
icon	FXIcon	None	
ptr	String	None	
notify	Bool	False	

replaceItem(index, item, notify=False)

Replace the item with a [possibly subclassed] item.

Argument	Type	Default	Description
index	Int		
item	FXListItem		
notify	Bool	False	

retrieveItem(index)

Return the item at the given index.

Argument	Type	Default	Description
index	Int		

selectItem(index, notify=False)

Select item.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

setCurrentItem(index, notify=False)

Change current item.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

setItemData(index, ptr)

Change item user-data pointer.

Argument	Type	Default	Description
index	Int		
ptr	String		

setItemIcon(index, icon)

Change item icon.

Argument	Type	Default	Description
index	Int		
icon	FXIcon		

setItemText(index, text)

Change item text.

Argument	Type	Default	Description
index	Int		
text	String		

setListStyle(style)

Change list style.

Argument	Type	Default	Description

style	Int		

setNumVisible(nvis)

Change the number of visible items.

Argument	Type	Default	Description
nvis	Int		

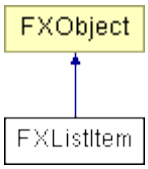
Global flags

List styles

LIST_EXTENDEDSELECT	**Extended selection mode allows for drag-selection of ranges of items.**
LIST_SINGLESELECT	**Single selection mode allows up to one item to be selected.**
LIST_BROWSESELECT	**Browse selection mode enforces one single item to be selected at all times.**
LIST_MULTIPLESELECT	**Multiple selection mode is used for selection of individual items.**
LIST_AUTOSELECT	**Automatically select under cursor.**

FXListItem

List item



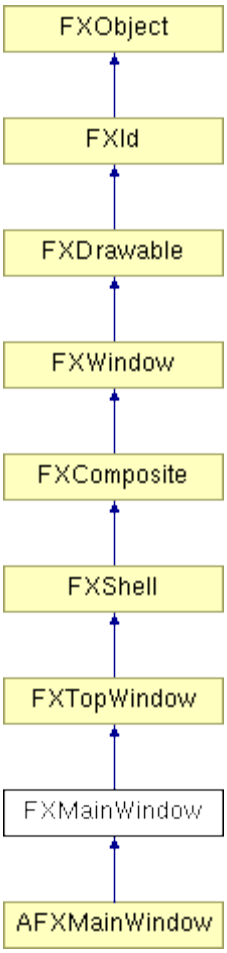
Global flags

List styles

LIST_EXTENDEDSELECT	**Extended selection mode allows for drag-selection of ranges of items.**
LIST_SINGLESELECT	**Single selection mode allows up to one item to be selected.**
LIST_BROWSESELECT	**Browse selection mode enforces one single item to be selected at all times.**
LIST_MULTIPLESELECT	**Multiple selection mode is used for selection of individual items.**
LIST_AUTOSELECT	**Automatically select under cursor.**

FXMainWindow

Main application window.



FXMainWindow(a, name, ic=None, mi=None, opts=DECOR_ALL, x=0, y=0, w=0, h=0, pl=0, pr=0, pt=0, pb=0, hs=0, vs=0)

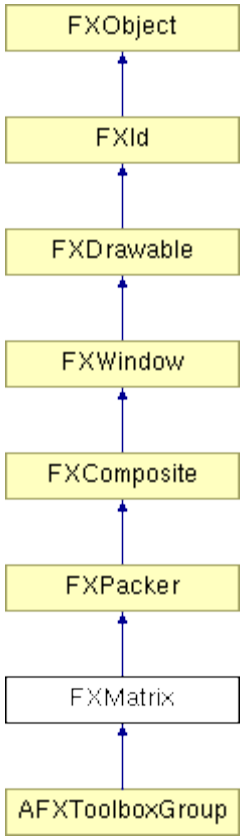
Construct a main window.

Argument	Type	Default	Description
a	FXApp		
name	String		
ic	FXIcon	None	
mi	FXIcon	None	
opts	Int	DECOR_ALL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	0	
pr	Int	0	
pt	Int	0	

pb	Int	0	
hs	Int	0	
vs	Int	0	

FXMatrix

The Matrix layout manager automatically arranges its child windows in rows and columns. If the matrix style is MATRIX_BY_ROWS, then the matrix will have the given number of rows and the number of columns grows as more child windows are added; if the matrix style is MATRIX_BY_COLUMNS, then the number of columns is fixed and the number of rows grows as more children are added. If all children in a row (column) have the LAYOUT_FILL_ROW (LAYOUT_FILL_COLUMN) hint set, then the row (column) will be stretchable as the matrix layout manager itself is resized. If more than one row (column) is stretchable, the space is apportioned to each stretchable row (column) proportionally. Within each cell of the matrix, all other layout hints are observed. For example, a child having LAYOUT_CENTER_Y and LAYOUT_FILL_X hints will be centered in the Y-direction, while being stretched in the X-direction. Empty cells can be obtained by simply placing a borderless FXFrame widget as a space-holder.



FXMatrix(p, n=1, opts=MATRIX_BY_ROWS, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Construct a matrix layout manager with n rows or columns.

Argument	Type	Default	Description
p	FXComposite		
n	Int	1	
opts	Int	MATRIX_BY_ROWS	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_SPACING	
pr	Int	DEFAULT_SPACING	
pt	Int	DEFAULT_SPACING	

pb	Int	DEFAULT_SPACING	
hs	Int	DEFAULT_SPACING	
vs	Int	DEFAULT_SPACING	

getDefaultHeight()

Return default height.

Reimplemented from FXPacker.

getDefaultWidth()

Return default width.

Reimplemented from FXPacker.

getNumColumns()

Return the number of columns.

getNumRows()

Return the number of rows.

setNumColumns(nc)

Change the number of columns.

Argument	Type	Default	Description
nc	Int		

setNumRows(nr)

Change the number of rows.

Argument	Type	Default	Description
nr	Int		

Global flags

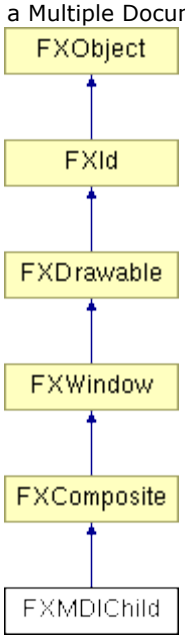
Matrix packing options

MATRIX_BY_ROWS **Fixed number of rows, add columns as needed.**

MATRIX_BY_COLUMNS **Fixed number of columns, adding rows as needed.**

FXMDIChild

The MDI child window contains the application work area in a Multiple Document Interface application.



FXMDIChild(p, name, ic=None, mn=None, opts=0, x=0, y=0, w=0, h=0)

Construct MDI Child window with given name and icon.

Argument	Type	Default	Description
p	FXMDIClient		
name	String		
ic	FXIcon	None	
mn	FXMenuPane	None	
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

canFocus()

MDI Child can receive focus.

Reimplemented from FXWindow.

contentWindow()

Return content window.

create()

Create window.

Reimplemented from FXComposite.

detach()

Detach window.

Reimplemented from FXComposite.

getDefaultHeight()

Return default height.

Reimplemented from FXComposite.

getDefaultWidth()

Compute default size.

Reimplemented from FXComposite.

getFont()

Get title font.

getHiliteColor()

Get colors.

getIconX()

Return iconified position.

getMDINext()

Get next MDI Child.

getMDIPrev()

Get previous MDI Child.

getNormalX()

Return normal (restored) position.

getTitle()

Get current title.

getWindowIcon()

Get window icon.

getWindowMenu()

Get window menu.

isMaximized()

Return True if maximized.

isMinimized()

Return True if minimized.

maximize(notify=False)

Maximize MDI Child.

Argument	Type	Default	Description
notify	Bool	False	

minimize(notify=False)

Minimize/iconify MDI Child.

Argument	Type	Default	Description
notify	Bool	False	

move(x, y)

Move this window to the specified position in the parent's coordinates.

Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		

position(x, y, w, h)

Move and resize this window in the parent's coordinates.

Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		

resize(w, h)

Resize this window to the specified width and height.

Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

restore(notify=False)

Restore MDI Child to normal.

Argument	Type	Default	Description
notify	Bool	False	

setFont(fnt)

Set title font.

Argument	Type	Default	Description
fnt	FXFont		

setHiliteColor(clr)

Change colors.

Argument	Type	Default	Description
clr	FXColor		

setIconX(x)

Change iconified position.

Argument	Type	Default	Description
x	Int		

setNormalX(x)

Change normal (restored) position.

Argument	Type	Default	Description
x	Int		

setTitle(name)

Change MDI Child's title.

Argument	Type	Default	Description
name	String		

setWindowIcon(icon)

Set window icon.

Argument	Type	Default	Description
icon	FXIcon		

setWindowMenu(menu)

Set window menu.

Argument	Type	Default	Description
menu	FXPopup		

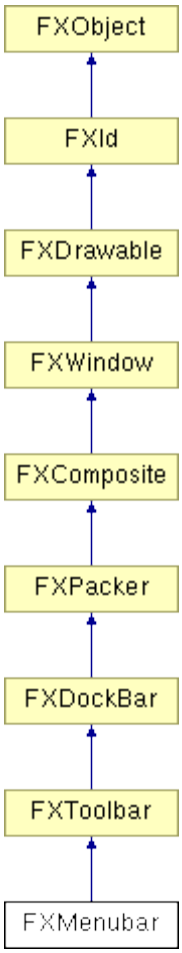
Global flags

MDI Child Window styles

MDI_NORMAL	**Normal display mode.**
MDI_MAXIMIZED	**Window appears maximized.**
MDI_MINIMIZED	**Window is iconified or minimized.**

FXMenubar

Menu bar.



FXMenubar(p, q, opts=LAYOUT_TOP| LAYOUT_LEFT| LAYOUT_FILL_X, x=0, y=0, w=0, h=0, pl=3, pr=3, pt=2, pb=2, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Construct a floatable menubar Normally, the menubar is docked under window p. When floated, the menubar can be docked under window q, which is typically an FXToolbarShell window.

Argument	Type	Default	Description
p	FXComposite		
q	FXComposite		
opts	Int	LAYOUT_TOP| LAYOUT_LEFT| LAYOUT_FILL_X	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	3	
pr	Int	3	
pt	Int	2	
pb	Int	2	

hs	Int	DEFAULT_SPACING	
vs	Int	DEFAULT_SPACING	

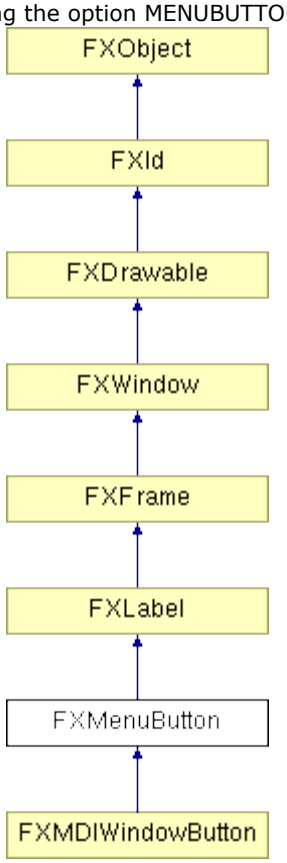
FXMenubar(p, opts=LAYOUT_TOP| LAYOUT_LEFT| LAYOUT_FILL_X, x=0, y=0, w=0, h=0, pl=3, pr=3, pt=2, pb=2, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Construct a non-floatable menubar. The menubar can not be undocked.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	LAYOUT_TOP| LAYOUT_LEFT| LAYOUT_FILL_X	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	3	
pr	Int	3	
pt	Int	2	
pb	Int	2	
hs	Int	DEFAULT_SPACING	
vs	Int	DEFAULT_SPACING	

FXMenuButton

A menu button posts a popup menu when clicked. There are many ways to control the placement where the popup will appear; first, the popup may be placed on either of the four sides relative to the menu button; this is controlled by the flags MENUBUTTON_DOWN, etc. Next, there are several attachment modes; the popup's left/bottom edge may attach to the menu button's left/top edge, or the popup's right/top edge may attach to the menu button's right/bottom edge, or both. Also, the popup may apear centered relative to the menu button. Finally, a small offset may be specified to displace the location of the popup by a few pixels so as to account for borders and so on. Normally, the menu button shows an arrow pointing to the direction where the popup is set to appear; this can be turned off by passing the option MENUBUTTON_NOARROWS.



FXMenuButton(p, text, ic=None, pup=None, opts=JUSTIFY_NORMAL| ICON_BEFORE_TEXT| MENUBUTTON_DOWN, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Constructor.

Argument	Type	Default	Description
p	FXComposite		
text	String		
ic	FXIcon	None	
pup	FXPopup	None	
opts	Int	JUSTIFY_NORMAL| ICON_BEFORE_TEXT| MENUBUTTON_DOWN	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_PAD	

pr	Int	DEFAULT_PAD	
pt	Int	DEFAULT_PAD	
pb	Int	DEFAULT_PAD	

canFocus()

Returns True because a menu button can receive focus.

Reimplemented from FXWindow.

create()

Create server-side resources.

Reimplemented from FXLabel.

detach()

Detach server-side resources.

Reimplemented from FXLabel.

getAttachment()

Get attachment.

getButtonStyle()

Get menu button style.

getDefaultHeight()

Return default height.

Reimplemented from FXLabel.

Reimplemented in FXMDIWindowButton.

getDefaultWidth()

Return default width.

Reimplemented from FXLabel.

Reimplemented in FXMDIWindowButton.

getMenu()

Return current popup menu.

getPopupStyle()

Get popup style.

getXOffset()

Return current X offset.

getYOffset()

Return current Y offset.

setAttachment(att)

Change attachment.

Argument	Type	Default	Description
att	Int		

setButtonStyle(style)

Change menu button style.

Argument	Type	Default	Description
style	Int		

setMenu(pup)

Change the popup menu.

Argument	Type	Default	Description
pup	FXPopup		

setPopupStyle(style)

Change popup style.

Argument	Type	Default	Description
style	Int		

setXOffset(offx)

Set X offset where menu pops up relative to button.

Argument	Type	Default	Description
offx	Int		

setYOffset(offy)

Set Y offset where menu pops up relative to button.

Argument	Type	Default	Description
offy	Int		

Global flags

Menu button options

MENUBUTTON_AUTOGRAY **Automatically gray out when no target.**

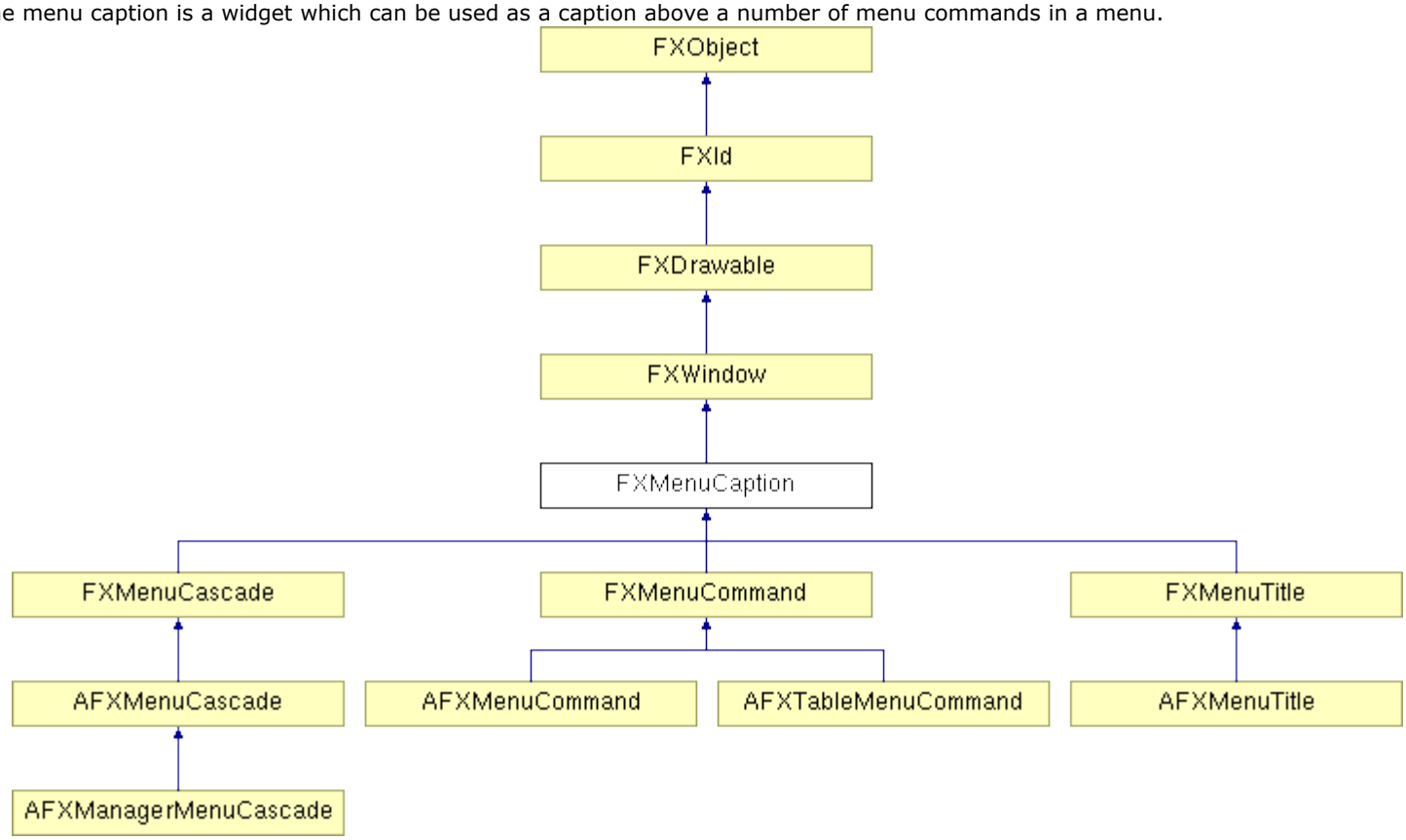
MENUBUTTON_AUTOHIDE **Automatically hide when no target.**

MENUBUTTON_TOOLBAR **Toolbar style.**

MENUBUTTON_COMBOBOX	**CAE - Combobox style.**
MENUBUTTON_DOWN	**Popup window appears below menu button.**
MENUBUTTON_UP	**Popup window appears above menu button.**
MENUBUTTON_LEFT	**Popup window to the left of the menu button.**
MENUBUTTON_RIGHT	**Popup window to the right of the menu button.**
MENUBUTTON_NOARROWS	**Do not show arrows.**
MENUBUTTON_ATTACH_LEFT	**Popup attaches to the left side of the menu button.**
MENUBUTTON_ATTACH_TOP	**Popup attaches to the top of the menu button.**
MENUBUTTON_ATTACH_RIGHT	**Popup attaches to the right side of the menu button.**
MENUBUTTON_ATTACH_BOTTOM	**Popup attaches to the bottom of the menu button.**
MENUBUTTON_ATTACH_CENTER	**Popup attaches to the center of the menu button.**
MENUBUTTON_ATTACH_BOTH	**Popup attaches to both sides of the menu button.**

FXMenuCaption

The menu caption is a widget which can be used as a caption above a number of menu commands in a menu.



FXMenuCaption(p, text, ic=None, opts=0)

Construct a menu caption.

Argument	Type	Default	Description
p	FXComposite		
text	String		
ic	FXIcon	None	
opts	Int	0	

create()

Create server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXMenuCascade, and FXMenuTitle.

detach()

Detach server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXMenuCascade, and FXMenuTitle.

disable()

Disable the menu.

Reimplemented from FXWindow.

enable()

Enable the menu.

Reimplemented from FXWindow.

getDefaultHeight()

Return default height.

Reimplemented from FXWindow.

Reimplemented in FXMenuCommand, and FXMenuTitle.

getDefaultWidth()

Return default width.

Reimplemented from FXWindow.

Reimplemented in FXMenuCommand, and FXMenuTitle.

getFont()

Return the text font.

getIcon()

Get the icon for this menu.

getText()

Get the text for this menu.

getTextColor()

Get the current text color.

setFont(fnt)

Set the text font.

Argument	Type	Default	Description
fnt	FXFont		

setIcon(ic)

Set the icon for this menu.

Argument	Type	Default	Description
ic	FXIcon		

setText(text)

Set the text for this menu.

Argument	Type	Default	Description
text	String		

setTextColor(clr)

Return the current text color.

Argument	Type	Default	Description
clr	FXColor		

Global flags

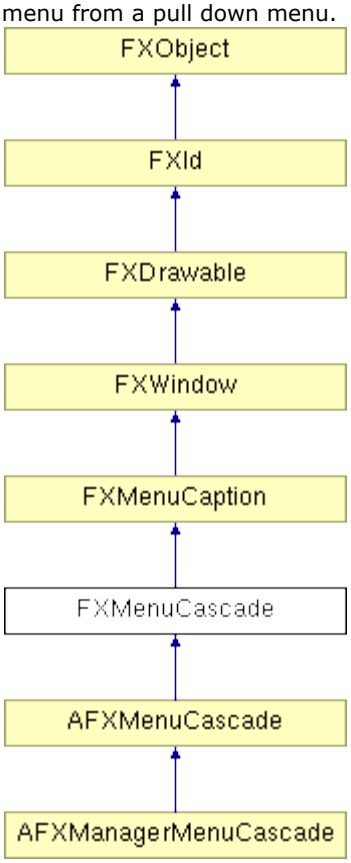
Menu Caption options

MENU_AUTOGRAY **Automatically gray out when not updated.**

MENU_AUTOHIDE **Automatically hide button when not updated.**

FXMenuCascade

The cascade menu widget is used to bring up a sub menu from a pull down menu.



FXMenuCascade(p, text, ic=None, pup=None, opts=0)

Construct a menu cascade responsible for the given popup menu.

Argument	Type	Default	Description
p	FXComposite		
text	String		
ic	FXIcon	None	
pup	FXPopup	None	
opts	Int	0	

canFocus()

Yes it can receive the focus.

Reimplemented from FXWindow.

create()

Create server-side resources.

Reimplemented from FXMenuCaption.

destroy()

Destroy server-side resources.

Reimplemented from FXWindow.

Reimplemented in AFXManagerMenuCascade.

detach()

Detach server-side resources.

Reimplemented from FXMenuCaption.

getMenu()

Return popup menu.

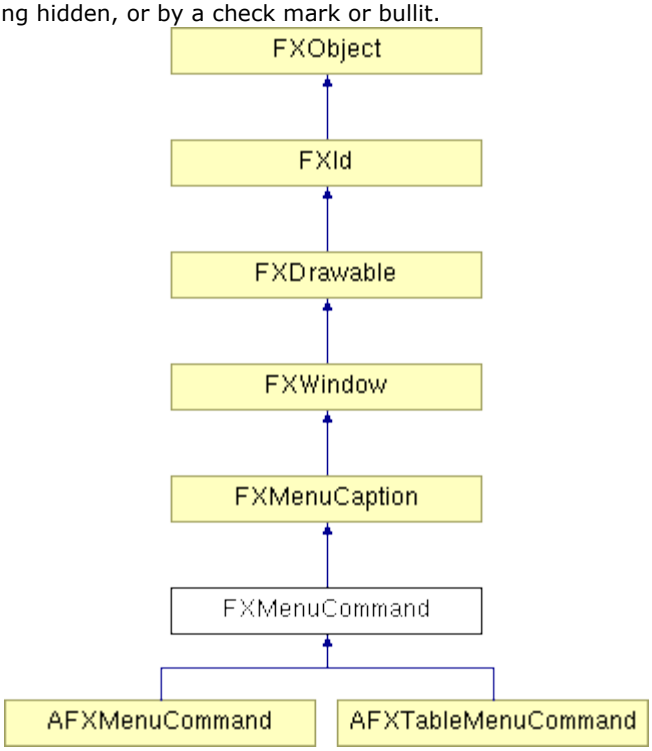
setMenu(pup)

Set popup menu to pop up.

Argument	Type	Default	Description
pup	FXPopup		

FXMenuCommand

The menu command widget is used to invoke a command in the application from a menu. Menu commands may reflect the state of the application by graying out, becoming hidden, or by a check mark or bullit.



FXMenuCommand(p, text, ic=None, tgt=None, sel=0, opts=0)

Construct a menu command.

Argument	Type	Default	Description
p	FXComposite		
text	String		
ic	FXIcon	None	
tgt	FXObject	None	
sel	Int	0	
opts	Int	0	

canFocus()

Yes it can receive the focus.

Reimplemented from FXWindow.

check()

Place checkmark next to text.

checkRadio()

Place radio bullit next to text.

getAccelText()

Return accelarator text.

getDefaultHeight()

Return default height.

Reimplemented from FXMenuCaption.

getDefaultWidth()

Return default width.

Reimplemented from FXMenuCaption.

isChecked()

Return True if checked.

isRadioChecked()

Return True if radio-checked.

setAccelText(text)

Set accelerator text.

Argument	Type	Default	Description
text	String		

uncheck()

Uncheck the item.

uncheckRadio()

Uncheck radio bullit.

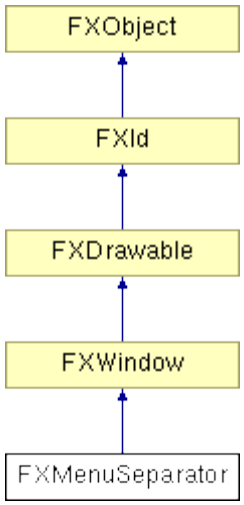
Global flags

States the menu command can be in

MENUSTATE_NORMAL	**Normal, unchecked state.**
MENUSTATE_CHECKED	**Checked with a checkmark.**
MENUSTATE_RCHECKED	**Checked with a bullet.**

FXMenuSeparator

The menu separator is a simple decorative groove used to delineate items in a popup menu.



FXMenuSeparator(p, opts=0)

Construct a menu separator.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	

getDefaultHeight()

Return default height.

Reimplemented from FXWindow.

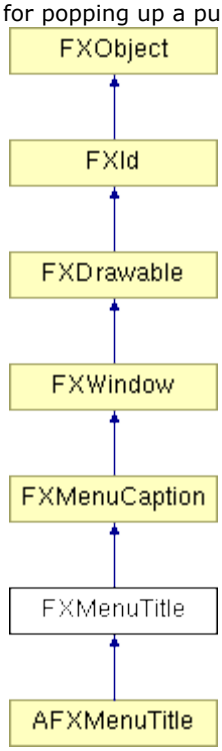
getDefaultWidth()

Return default width.

Reimplemented from FXWindow.

FXMenuTitle

A menu title is a child of a menu bar which is responsible for popping up a pulldown menu.



FXMenuTitle(p, text, ic=None, pup=None, opts=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		
text	String		
ic	FXIcon	None	
pup	FXPopup	None	
opts	Int	0	

canFocus()

Yes it can receive the focus.

Reimplemented from FXWindow.

create()

Create server-side resources.

Reimplemented from FXMenuCaption.

detach()

Detach server-side resources.

Reimplemented from FXMenuCaption.

getDefaultHeight()

Return default height.

Reimplemented from FXMenuCaption.

getDefaultWidth()

Return default width.

Reimplemented from FXMenuCaption.

getMenu()

Return popup menu.

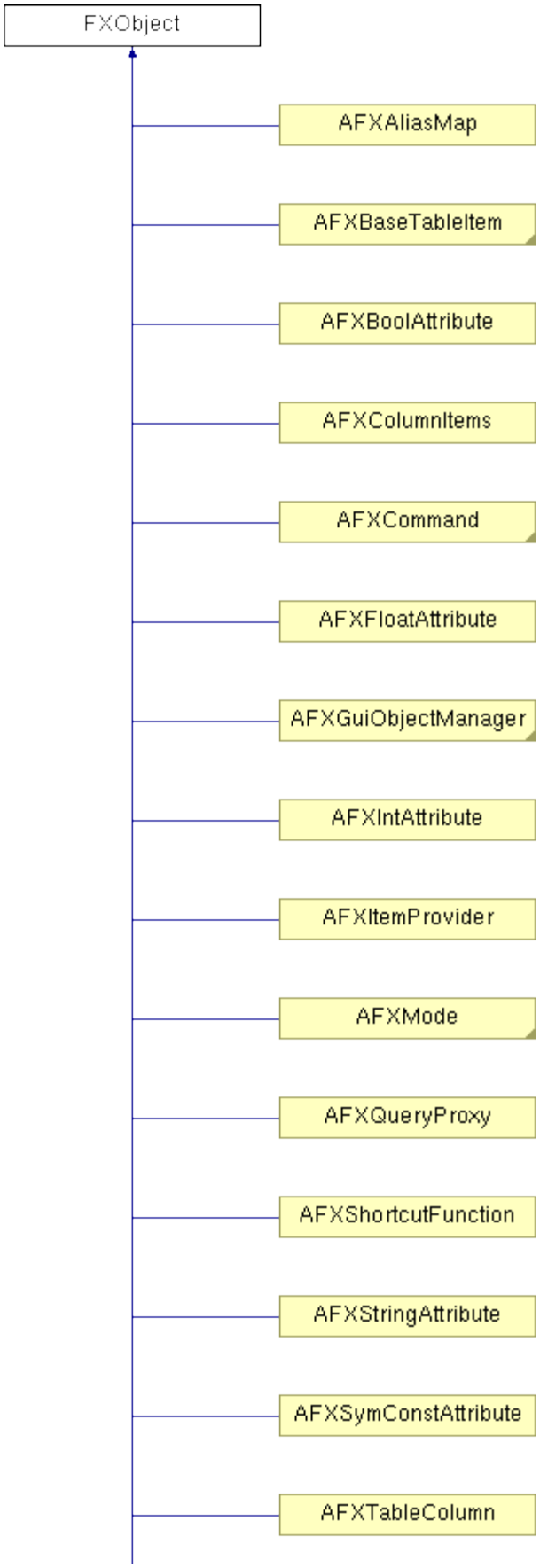
setMenu(menu)

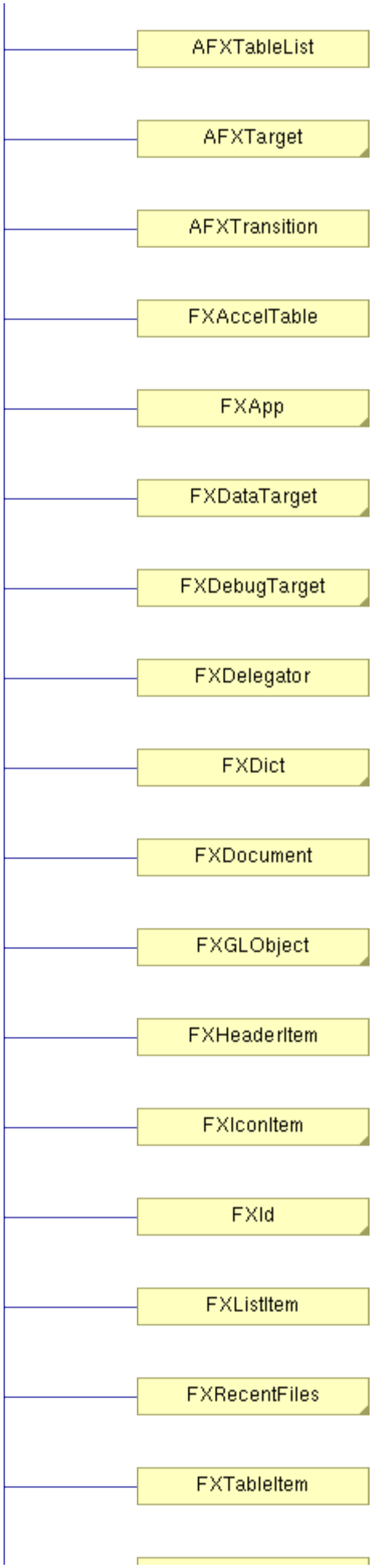
Set popup menu to pop up.

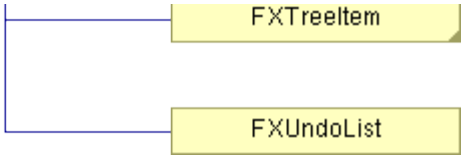
Argument	Type	Default	Description
menu	FXPopup		

FXObject

Base of all FOX object.




AFXTableList
AFXTarget
AFXTransition
FXAccelTable
FXApp
FXDataTarget
FXDebugTarget
FXDelegator
FXDict
FXDocument
FXGLObject
FXHeaderItem
FXIconItem
FXId
FXListItem
FXRecentFiles
FXTableItem



getClassName()

Get class name of some object.

isMemberOf(metaclass)

Check if object is member of metaclass.

Argument	Type	Default	Description
metaclass	FXMetaClass		

onDefault()

Called for unhandled messages.

Reimplemented in FXDelegator, FXGLViewer, FXMDIChild, and FXMDIClient.

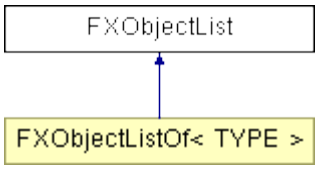
handle(sender, sel, ptr)

Handles messages sent to this class.

Argument	Type	Default	Description
sender	FXObject		The sender of the message.
sel	FXSelector		The selector of the message.
ptr	void*		Associated data.

FXObjectList

List of pointers to objects.



FXObjectList()

Default constructor.

FXObjectList(orig)

Copy constructor.

Argument	Type	Default	Description
orig	FXObjectList		

append(p)

Append element.

Argument	Type	Default	Description
p	FXObject		

clear()

Remove all elements.

findb(p, pos=2147483647)

Find object in list, searching backward; return position or -1.

Argument	Type	Default	Description
p	FXObject		
pos	Int	2147483647	

findf(p, pos=0)

Find object in list, searching forward; return position or -1.

Argument	Type	Default	Description
p	FXObject		
pos	Int	0	

insert(pos, p)

Insert element at certain position.

Argument	Type	Default	Description
pos	Int		
p	FXObject		

no(n)

Set number of elements.

Argument	Type	Default	Description
n	Int		

no()

Return number of elements.

remove(p)

Remove element p.

Argument	Type	Default	Description
p	FXObject		

remove(pos)

Remove element at pos.

Argument	Type	Default	Description
pos	Int		

size(m)

Set max number of elements.

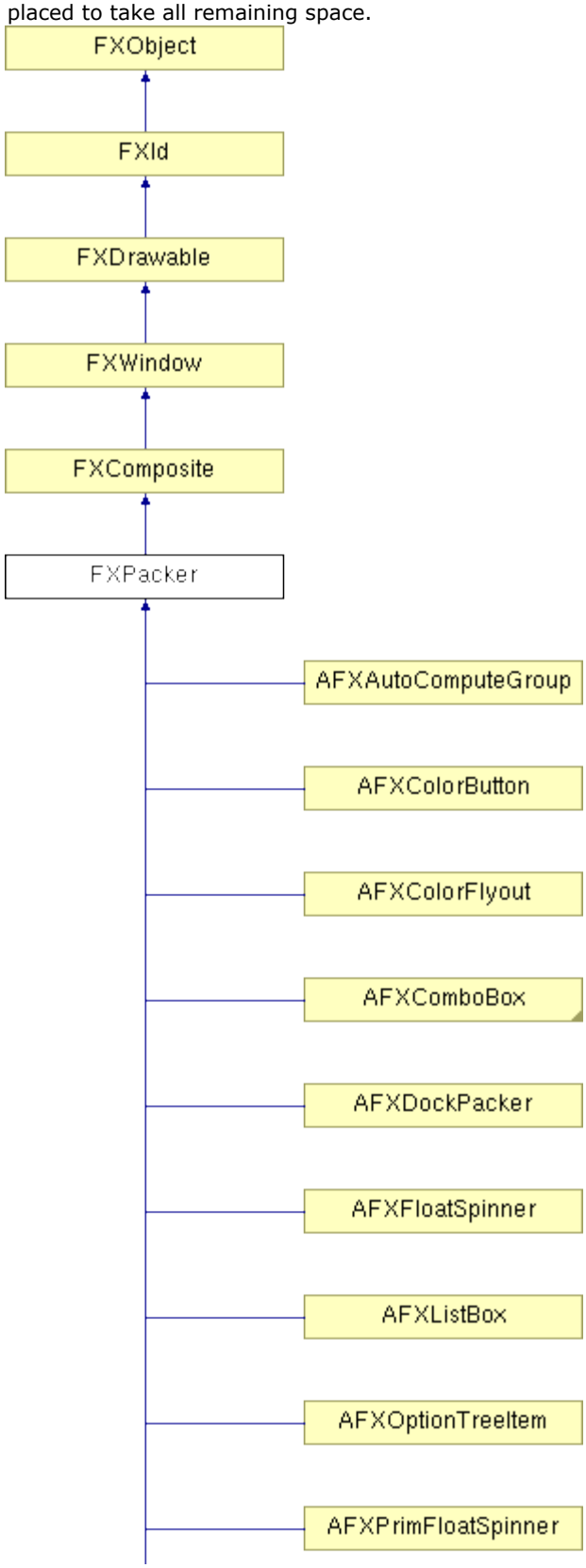
Argument	Type	Default	Description
m	Int		

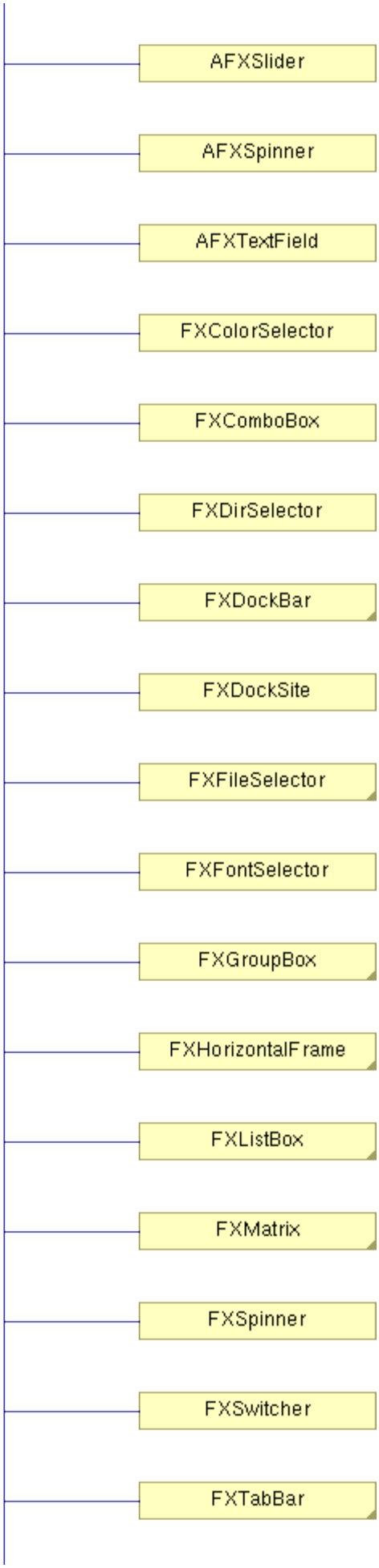
size()

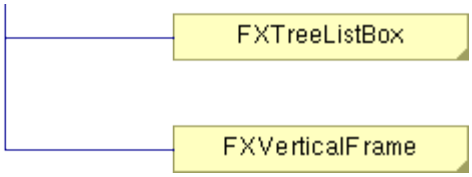
Return size of list.

FXPacker

Packer is a layout manager which automatically places child windows inside its area against the left, right, top, or bottom side. Each time a child is placed, the remaining space is decreased by the amount of space taken by the child window. The side against which a child is placed is determined by the LAYOUT_SIDE_TOP, LAYOUT_SIDE_BOTTOM, LAYOUT_SIDE_LEFT, and LAYOUT_SIDE_RIGHT hints given by the child window. Other layout hints from the child are observed as far as sensible. So for example, a child placed against the right edge can still have LAYOUT_FILL_Y or LAYOUT_TOP, and so on. The last child may have both LAYOUT_FILL_X and LAYOUT_FILL_Y, in which case it will be placed to take all remaining space.




AFXSlider
AFXSpinner
AFXTextField
FXColorSelector
FXComboBox
FXDirSelector
FXDockBar
FXDockSite
FXFileSelector
FXFontSelector
FXGroupBox
FXHorizontalFrame
FXListBox
FXMatrix
FXSpinner
FXSwitcher
FXTabBar



FXPacker(p, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Construct packer layout manager.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_SPACING	
pr	Int	DEFAULT_SPACING	
pt	Int	DEFAULT_SPACING	
pb	Int	DEFAULT_SPACING	
hs	Int	DEFAULT_SPACING	
vs	Int	DEFAULT_SPACING	

getBaseColor()

Get base gui color.

getBorderColor()

Get border color.

getBorderWidth()

Get border width.

getDefaultHeight()

Return default height.

Reimplemented from FXComposite.

Reimplemented in FXComboBox, FXDockSite, FXGroupBox, FXHorizontalFrame, FXListBox, FXMatrix, FXSpinner, FXStatusbar, FXSwitcher, FXTabBar, FXTabBook, FXToolbar, FXTreeListBox, FXVerticalFrame, AFXToolbarGroup, AFXPrimFloatSpinner, AFXSlider, and AFXVerticalAligner.

getDefaultWidth()

Return default width.

Reimplemented from FXComposite.

Reimplemented in FXComboBox, FXDockSite, FXGroupBox, FXHorizontalFrame, FXListBox, FXMatrix, FXSpinner, FXStatusbar,

FXSwitcher, FXTabBar, FXTabBook, FXToolbar, FXTreeListBox, FXVerticalFrame, AFXToolbarGroup, AFXOptionTreeItem, AFXPrimFloatSpinner, AFXSlider, AFXTextField, and AFXVerticalAligner.

getFrameStyle()

Get current frame style.

getHiliteColor()

Get highlight color.

getHSpacing()

Return current horizontal inter-child spacing.

getPackingHints()

Return packing hints.

getPadBottom()

Get bottom interior padding.

getPadLeft()

Get left interior padding.

getPadRight()

Get right interior padding.

getPadTop()

Get top interior padding.

getShadowColor()

Get shadow color.

getVSpacing()

Return current vertical inter-child spacing.

setBaseColor(clr)

Change base gui color.

Argument	Type	Default	Description
clr	FXColor		

setBorderColor(clr)

Change border color.

Argument	Type	Default	Description
clr	FXColor		

setFrameStyle(style)

Change frame style.

Argument	Type	Default	Description
style	Int		

setHiliteColor(clr)

Change highlight color.

Argument	Type	Default	Description
clr	FXColor		

setHSpacing(hs)

Change horizontal inter-child spacing.

Argument	Type	Default	Description
hs	Int		

setPackingHints(ph)

Change packing hints.

Argument	Type	Default	Description
ph	Int		

setPadBottom(pb)

Change bottom padding.

Argument	Type	Default	Description
pb	Int		

setPadLeft(pl)

Change left padding.

Argument	Type	Default	Description
pl	Int		

setPadRight(pr)

Change right padding.

Argument	Type	Default	Description
pr	Int		

setPadTop(pt)

Change top padding.

Argument	Type	Default	Description
pt	Int		

setShadowColor(clr)

Change shadow color.

Argument	Type	Default	Description
clr	FXColor		

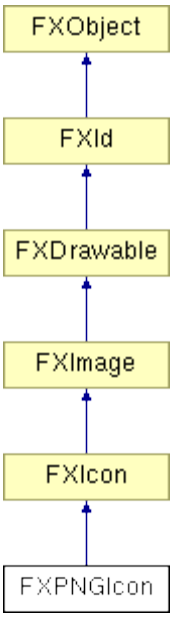
setVSpacing(vs)

Change vertical inter-child spacing.

Argument	Type	Default	Description
vs	Int		

FXPNGIcon

Portable Network Graphics (PNG) Icon class.



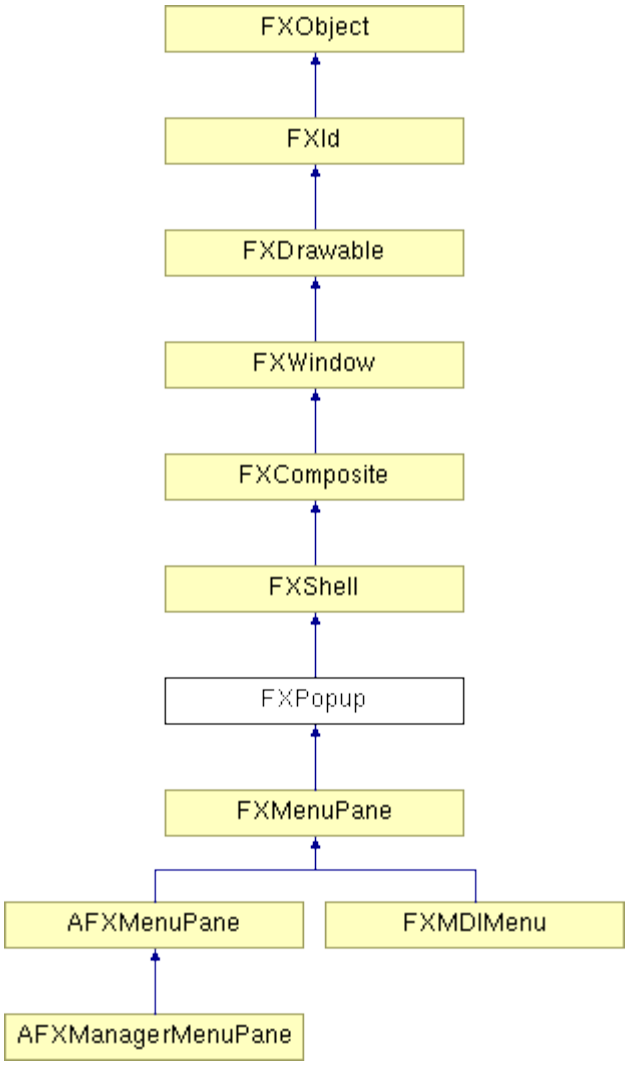
FXPNGIcon(a, pix=None, clr=FXRGB(192, 192, 192), opts=0, w=1, h=1)

Construct an icon from memory stream formatted in PNG format.

Argument	Type	Default	Description
a	FXApp		
pix		None	
clr	FXColor	FXRGB(192, 192, 192)	
opts	Int	0	
w	Int	1	
h	Int	1	

FXPopup

Popup window



FXPopup(owner, opts=POPUP_VERTICAL| FRAME_RAISED| FRAME_THICK, x=0, y=0, w=0, h=0)

Construct popup pane.

Argument	Type	Default	Description
owner	FXWindow		
opts	Int	POPUP_VERTICAL| FRAME_RAISED| FRAME_THICK	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

getBaseColor()

Return base color.

getBorderColor()

Return border color.

getBorderWidth()

Return border width.

getDefaultHeight()

Return the default height of this window.

Reimplemented from FXComposite.

getDefaultWidth()

Return the default width of this window.

Reimplemented from FXComposite.

getFrameStyle()

Return frame style.

getGrabOwner()

Return current grab owner.

getHiliteColor()

Return highlight color.

getOrientation()

Return popup orientation.

getShadowColor()

Return shadow color.

getShrinkWrap()

Return shrinkwrap mode.

popdown()

Pop down the menu.

popup(grabto, x, y, w=0, h=0)

Popup the menu and grab to the given owner.

Argument	Type	Default	Description
grabto	FXWindow		
x	Int		

y	Int		
w	Int	0	
h	Int	0	

setBaseColor(clr)

Change base color.

Argument	Type	Default	Description
clr	FXColor		

setBorderColor(clr)

Change border color.

Argument	Type	Default	Description
clr	FXColor		

setFrameStyle(style)

Change frame style.

Argument	Type	Default	Description
style	Int		

setHiliteColor(clr)

Change highlight color.

Argument	Type	Default	Description
clr	FXColor		

setOrientation(orient)

Change popup orientation.

Argument	Type	Default	Description
orient	Int		

setShadowColor(clr)

Change shadow color.

Argument	Type	Default	Description
clr	FXColor		

setShrinkWrap(sw)

Change shrinkwrap mode.

Argument	Type	Default	Description
sw	Bool		

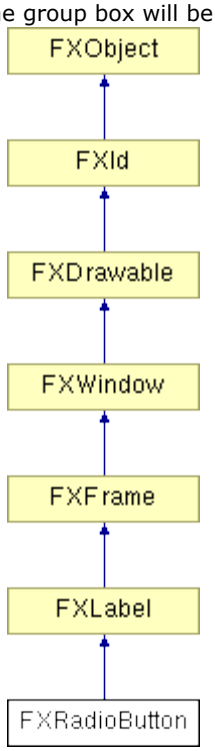
Global flags

Popup internal orientation

POPUP_VERTICAL	**Vertical orientation.**
POPUP_HORIZONTAL	**Horizontal orientation.**
POPUP_SHRINKWRAP	**Shrinkwrap to content.**

FXRadioButton

A radio button is a tri-state button. Normally, it is either True or False; a third state MAYBE may be set to indicate that no selection has been made yet by the user, or that the state is ambiguous. When pressed, the radio button sets its state to True and sends a SEL_COMMAND to its target, and the message data set to the state of the radio button, of the type FXbool. If the radio button is contained inside a group box, the other radio buttons in the group box will be set to False and will send a message as well.



FXRadioButton(p, text, tgt=None, sel=0, opts=RADIOBUTTON_NORMAL, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Construct new radio button.

Argument	Type	Default	Description
p	FXComposite		
text	String		
tgt	FXObject	None	
sel	Int	0	
opts	Int	RADIOBUTTON_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_PAD	
pr	Int	DEFAULT_PAD	
pt	Int	DEFAULT_PAD	
pb	Int	DEFAULT_PAD	

canFocus()

Returns True because a radio button can receive focus.

Reimplemented from FXWindow.

getCheck()

Get radio button state (True, False or MAYBE).

getDefaultHeight()

Get default height.

Reimplemented from FXLabel.

getDefaultWidth()

Get default width.

Reimplemented from FXLabel.

setCheck(s=True)

Set radio button state (True, False or MAYBE).

Argument	Type	Default	Description
s	Bool	True	

Global flags

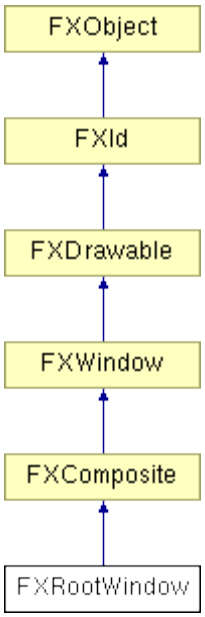
RadioButton flags

RADIOBUTTON_AUTOGRAY **Automatically gray out when not updated.**

RADIOBUTTON_AUTOHIDE **Automatically hide when not updated.**

FXRootWindow

Root window.



FXRootWindow(a, vis)

Construct root window.

Argument	Type	Default	Description
a	FXApp		
vis	FXVisual		

create()

Root window need not be created.

Reimplemented from FXComposite.

destroy()

Root window can not be destroyed.

Reimplemented from FXComposite.

detach()

Root window may not be detached.

Reimplemented from FXComposite.

getDefaultHeight()

Return height of the root window.

Reimplemented from FXComposite.

getDefaultWidth()

Return width of the root window.

Reimplemented from FXComposite.

move(x, y)

Root window can not be moved.

Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		

position(x, y, w, h)

Root window can not be positioned.

Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		

recalc()

No op.

Reimplemented from FXWindow.

resize(w, h)

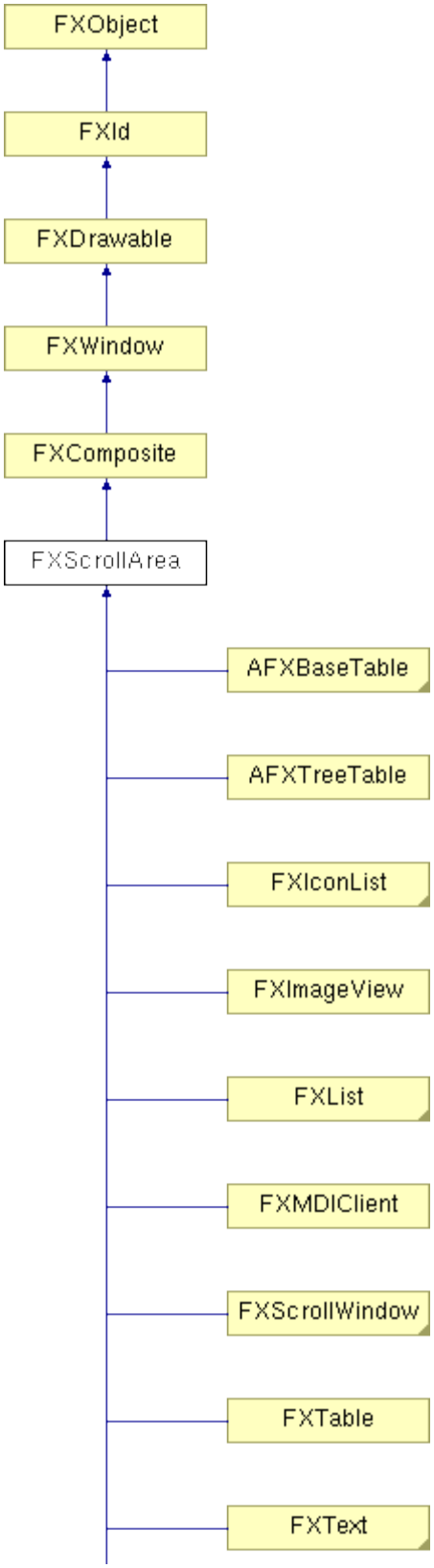
Root window can not be resized.

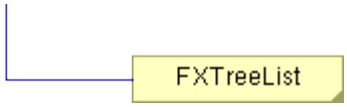
Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

FXScrollArea

The scroll area widget manages a content area and a viewport area through which the content is viewed. When the content area becomes larger than the viewport area, scrollbars are placed to permit viewing of the entire content by scrolling the content. Depending on the mode, scrollbars may be displayed on an as-needed basis, always, or never. Normally, the scroll area's size and the content's size are independent; however, it is possible to disable scrolling in the horizontal (vertical) direction. In this case, the content width (height) will influence the width (height) of the scroll area widget. For content which is time-consuming to repaint, continuous scrolling may be turned off.





getContentWidth()

Return content size.

Reimplemented in FXIconList, FXImageView, FXList, FXMDIClient, FXScrollWindow, FXTable, FXText, FXTreeList, AFXBaseTable, and AFXOptionTreeList.

getDefaultHeight()

Return default height.

Reimplemented from FXComposite.

Reimplemented in FXList, FXTable, FXText, FXTreeList, AFXBaseTable, AFXList, AFXOptionTreeList, AFXTable, and AFXTreeTable.

getDefaultWidth()

Return default width.

Reimplemented from FXComposite.

Reimplemented in FXList, FXTable, FXText, FXTreeList, AFXBaseTable, AFXOptionTreeList, AFXTable, and AFXTreeTable.

getPosition(x, y)

Get the current position.

Argument	Type	Default	Description
x	Int		
y	Int		

getScrollStyle()

Return scroll style.

getViewportHeight()

Return viewport size.

Reimplemented in FXIconList.

getXPosition()

Return the current x-position.

getYPosition()

Return the current y-position.

horizontalScrollbar()

Return a pointer to the horizontal scrollbar.

isHorizontalScrollable()

Return True if horizontally scrollable.

isVerticalScrollable()

Return True if vertically scrollable.

moveContents(x, y)

Move contents to the specified position.

Reimplemented in FXIconList, FXMDIClient, FXScrollWindow, FXTable, FXText, AFXBaseTable, AFXOptionTreeList, and AFXTable.

Argument	Type	Default	Description
x	Int		
y	Int		

setPosition(x, y)

Set the current position.

Argument	Type	Default	Description
x	Int		
y	Int		

setScrollStyle(style)

Change scroll style.

Argument	Type	Default	Description
style	Int		

verticalScrollbar()

Return a pointer to the vertical scrollbar.

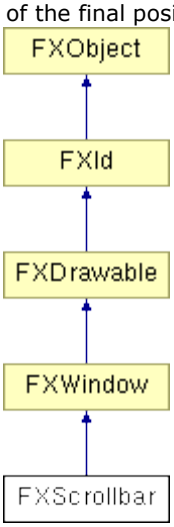
Global flags

Scrollbar options

SCROLLERS_NORMAL	**Show the scrollbars when needed.**
HSCROLLER_ALWAYS	**Always show horizontal scrollers.**
HSCROLLER_NEVER	**Never show horizontal scrollers.**
VSCROLLER_ALWAYS	**Always show vertical scrollers.**
VSCROLLER_NEVER	**Never show vertical scrollers.**
HSCROLLING_ON	**Horizontal scrolling turned on (default).**
HSCROLLING_OFF	**Horizontal scrolling turned off.**
VSCROLLING_ON	**Vertical scrolling turned on (default).**
VSCROLLING_OFF	**Vertical scrolling turned off.**
SCROLLERS_TRACK	**Scrollers track continuously for smooth scrolling.**
SCROLLERS_DONT_TRACK	**Scrollers don't track continuously.**

FXScrollbar

The scroll bar is used when a document has a larger content than may be made visible. The range is the total size of the document, the page is the part of the document which is visible. The size of the scrollbar thumb is adjusted to give feedback of the relative sizes of each. The scroll bar may be manipulated by the left mouse (normal scrolling), right mouse (vernier or fine-scrolling), or middle mouse (same as the left mouse only the scroll position can hop to the place where the click is made). Finally, if the mouse sports a wheel, the scroll bar can be manipulated by means of the mouse wheel as well. Holding down the Control-key during wheel motion will cause the scrolling to go faster than normal. While moving the scroll bar, a message of type SEL_CHANGED will be sent to the target, and the message data will reflect the current position of type FXint. At the end of the interaction, the scroll bar will send a message of type SEL_COMMAND to notify the target of the final position.



FXScrollbar(p, tgt=None, sel=0, opts=SCROLLBAR_VERTICAL, x=0, y=0, w=0, h=0)

Construct scroll bar.

Argument	Type	Default	Description
p	FXComposite		
tgt	FXObject	None	
sel	Int	0	
opts	Int	SCROLLBAR_VERTICAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

getBorderColor()

Change border color.

getDefaultHeight()

Return default height.

Reimplemented from FXWindow.

getDefaultWidth()

Return default width.

Reimplemented from FXWindow.

getHiliteColor()

Return highlight color.

getLine()

Return line increment.

getPage()

Return page size.

getPosition()

return scroll position

getRange()

Return content size range.

getScrollbarStyle()

Change the scrollbar style.

getShadowColor()

Return shadow color.

setBorderColor(clr)

Return border color.

Argument	Type	Default	Description
clr	FXColor		

setHiliteColor(clr)

Change highlight color.

Argument	Type	Default	Description
clr	FXColor		

setLine(l)

Set scoll increment for line.

Argument	Type	Default	Description
l	Int		

setPage(p)

Set viewport page size.

Argument	Type	Default	Description
p	Int		

setPosition(p, notifyTgt=False)

Change current scroll position.

Argument	Type	Default	Description
p	Int		
notifyTgt	Bool	False	

setRange(r)

Set content size range.

Argument	Type	Default	Description
r	Int		

setScrollbarStyle(style)

Get the current scrollbar style.

Argument	Type	Default	Description
style	Int		

setShadowColor(clr)

Change shadow color.

Argument	Type	Default	Description
clr	FXColor		

Global flags

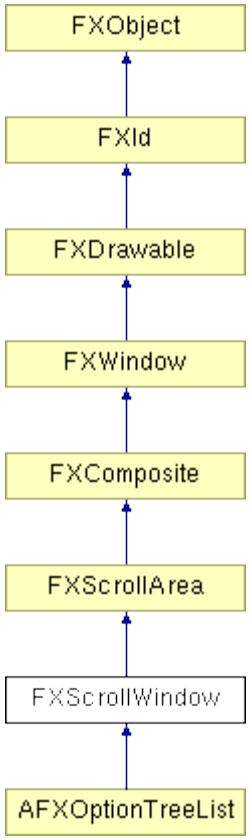
Scrollbar styles

SCROLLBAR_HORIZONTAL **Horizontally oriented.**

SCROLLBAR_VERTICAL **Vertically oriented.**

FXScrollWindow

The scroll window widget scrolls an arbitrary child window. Use the scroll window when parts of the user interface itself need to be scrolled, for example when applications need to run on small screens.



FXScrollWindow(p, opts=0, x=0, y=0, w=0, h=0)

Construct a scroll window.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

contentWindow()

Return a pointer to the contents window.

create()

Create server-side resources.

Reimplemented from FXComposite.

getContentHeight()

Return the height of the contents.

Reimplemented from FXScrollArea.

Reimplemented in AFXOptionTreeList.

getContentWidth()

Return the width of the contents.

Reimplemented from FXScrollArea.

Reimplemented in AFXOptionTreeList.

moveContents(x, y)

Move contents to the specified position.

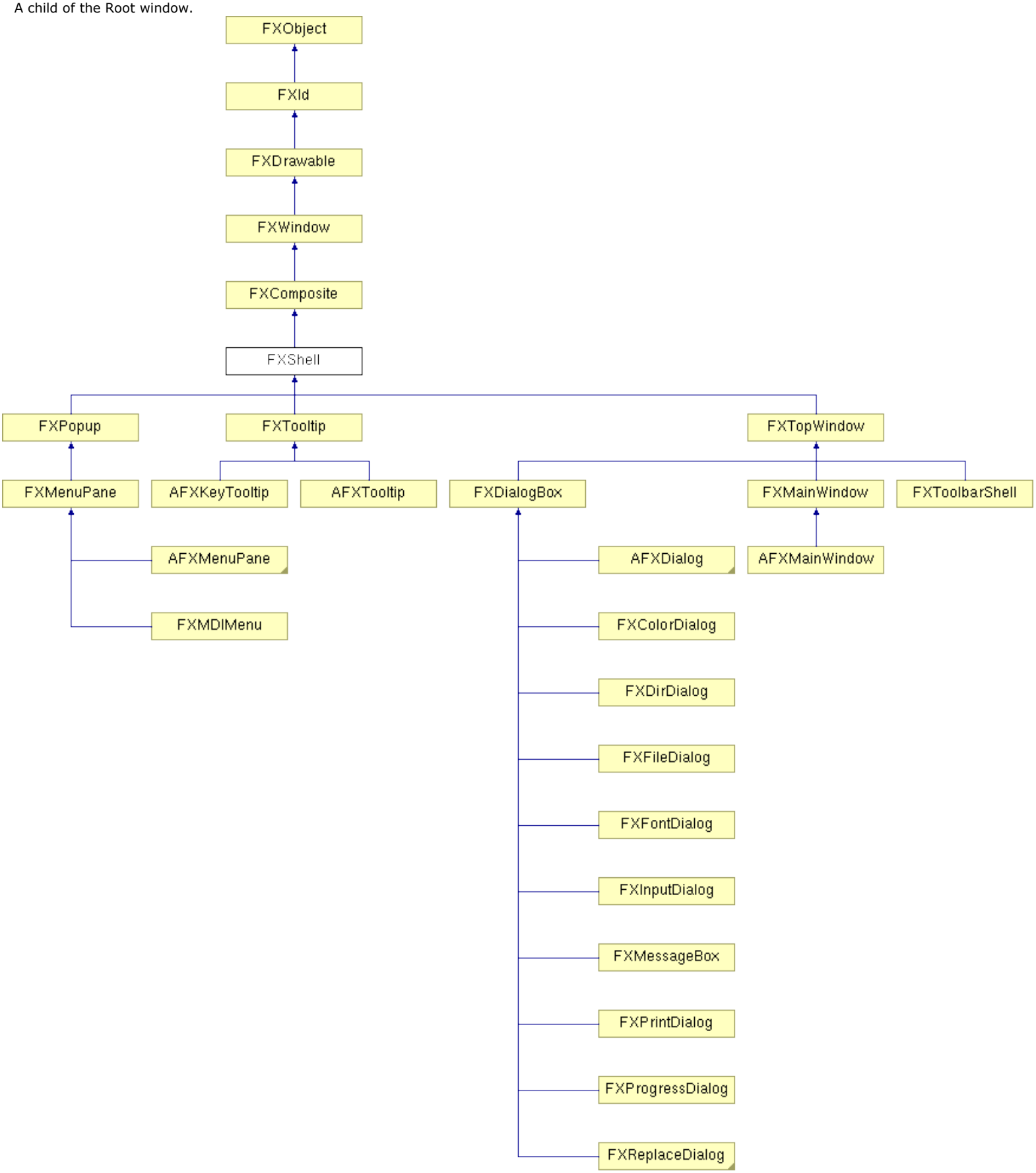
Reimplemented from FXScrollArea.

Reimplemented in AFXOptionTreeList.

Argument	Type	Default	Description
x	Int		
y	Int		

FXShell

A child of the Root window.



create()

Create server-side resources.

Reimplemented from FXComposite.

Reimplemented in FXPrintDialog, FXToolbarShell, FXTooltip, FXTopWindow, AFXManagerMenuPane, AFXMainWindow, and AFXDialog.

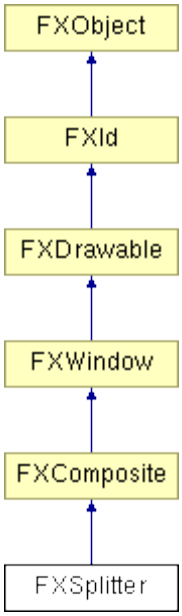
recalc()

Mark this window's layout as dirty.

Reimplemented from FXWindow.

FXSplitter

Splitter window is used to interactively repartition two or more subpanes. Space may be subdivided horizontally or vertically. When the splitter is itself resized, the right-most (bottom-most) child window will be resized unless the splitter window is reversed; if the splitter is reversed, the left-most (top-most) child window will be resized instead. The splitter widget sends a SEL_CHANGED to its target during the resizing of the panes; at the end of the resize interaction, it sends a SEL_COMMAND to signify that the resize operation is complete. Normally, children are resizable from 0 upwards; however, if the child in a horizontally oriented splitter has LAYOUT_FILL_X in combination with LAYOUT_FIX_WIDTH, it will not be made smaller than its default width, except when the child is the last visible widget (or first when the option SPLITTER_REVERSED has been passed to the splitter). In a vertically oriented splitter, children with LAYOUT_FILL_Y and LAYOUT_FIX_HEIGHT behave analogously. These options only affect interactive resizing.



FXSplitter(p, opts=SPLITTER_NORMAL, x=0, y=0, w=0, h=0)

Construct new splitter widget.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	SPLITTER_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

FXSplitter(p, tgt, sel, opts=SPLITTER_NORMAL, x=0, y=0, w=0, h=0)

Construct new splitter widget, which will notify target about size changes.

Argument	Type	Default	Description
p	FXComposite		
tgt	FXObject		
sel	Int		
opts	Int	SPLITTER_NORMAL	
x	Int	0	

y	Int	0	
w	Int	0	
h	Int	0	

getDefaultHeight()

Get default height.

Reimplemented from FXComposite.

getDefaultWidth()

Get default width.

Reimplemented from FXComposite.

getSplitterStyle()

Return current splitter style.

setSplitterStyle(style)

Change splitter style.

Argument	Type	Default	Description
style	Int		

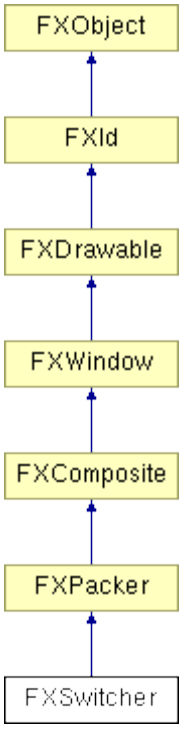
Global flags

Splitter options

SPLITTER_HORIZONTAL	**Split horizontally.**
SPLITTER_VERTICAL	**Split vertically.**
SPLITTER_REVERSED	**Reverse-anchored.**
SPLITTER_TRACKING	**Track continuous during split.**

FXSwitcher

The Switcher layout manager automatically arranges its child windows such that one of them is placed on top; all other child windows are hidden. Switcher provides a convenient method to conserve screen real-estate by arranging several GUI panels to appear in the same space, depending on context. Switcher ignores all layout hints from its children:- all children are stretched according to the switcher layout managers own size. When the SWITCHER_HCOLLAPSE or SWITCHER_VCOLLAPSE options are used, Switcher's default size is based on the width or height of the current child, instead of the maximum width or height of all of the children.



FXSwitcher(p, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING)

Construct a switcher layout manager.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_SPACING	
pr	Int	DEFAULT_SPACING	
pt	Int	DEFAULT_SPACING	
pb	Int	DEFAULT_SPACING	

getCurrent()

Return the index of the child window currently on top.

getDefaultHeight()

Return default height.

Reimplemented from FXPacker.

getDefaultWidth()

Return default width.

Reimplemented from FXPacker.

setCurrent(index, notify=False)

Bring the child window at index to the top.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

Class flags

ID's that identify children of the switcher; these ID's can be used to set the current child by sending the switcher a message using MKUINT(id, SEL_COMMAND).

ID_OPEN_FIRST	**ID for the 1st child.**
ID_OPEN_SECOND	**ID for the 2nd child.**
ID_OPEN_THIRD	**ID for the 3rd child.**
ID_OPEN_FOURTH	**ID for the 4th child.**
ID_OPEN_FIFTH	**ID for the 5th child.**
ID_OPEN_SIXTH	**ID for the 6th child.**
ID_OPEN_SEVENTH	**ID for the 7th child.**
ID_OPEN_EIGHTH	**ID for the 8th child.**
ID_OPEN_NINETH	**ID for the 9th child.**
ID_OPEN_TENTH	**ID for the 10th child.**

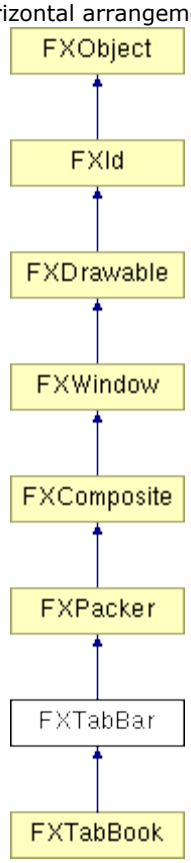
Global flags

Switcher options

SWITCHER_HCOLLAPSE	**Collapse horizontally to width of current child.**
SWITCHER_VCOLLAPSE	**Collapse vertically to height of current child.**

FXTabBar

The tab bar layout manager arranges tab items side by side, and raises the active tab item above the neighboring tab items. The tab bar can be have the tab items on the top or bottom for horizontal arrangement, or on the left or right for vertical arrangement.



FXTabBar(p, tgt=None, sel=0, opts=TABBOOK_NORMAL, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING)

Construct a tab bar.

Argument	Type	Default	Description
p	FXComposite		
tgt	FXObject	None	
sel	Int	0	
opts	Int	TABBOOK_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_SPACING	
pr	Int	DEFAULT_SPACING	
pt	Int	DEFAULT_SPACING	
pb	Int	DEFAULT_SPACING	

create()

Create all of the server-side resources for this window // CAE.

Reimplemented from FXComposite.

getCurrent()

Return the currently active tab item.

getDefaultHeight()

Return default height.

Reimplemented from FXPacker.

Reimplemented in FXTabBook.

getDefaultWidth()

Return default width.

Reimplemented from FXPacker.

Reimplemented in FXTabBook.

getTabStyle()

Return tab bar style.

setCurrent(panel, notify=False)

Change currently active tab item; this raises the active tab item slightly above the neighboring tab items.

Argument	Type	Default	Description
panel	Int		
notify	Bool	False	

setTabStyle(style)

Change tab tab style.

Argument	Type	Default	Description
style	Int		

Class flags

ID_OPEN_ITEM **Sent from one of the FXTabItems.**

ID_OPEN_FIRST **Switch to panel ID_OPEN_FIRST+i.**

Global flags

Tab Book options

TABBOOK_TOPTABS **Tabs on top (default).**

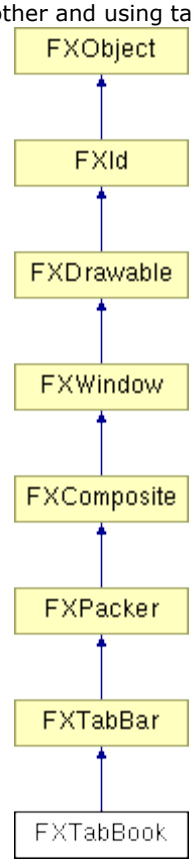
TABBOOK_BOTTOMTABS **Tabs on bottom.**

TABBOOK_SIDEWAYS **Tabs on left.**

TABBOOK_LEFTTABS	**Tabs on left.**
TABBOOK_RIGHTTABS	**Tabs on right.**

FXTabBook

The tab book layout manager arranges pairs of children; the even numbered children (0,2,4,...) are usually tab items, and are placed on the top. The odd numbered children are usually layout managers, and are placed below; all the odd numbered children are placed on top of each other, similar to the switcher widget. When the user presses one of the tab items, the tab item is raised above the neighboring tabs, and the corresponding panel is raised to the top. Thus, a tab book can be used to present many GUI controls in a small space by placing several panels on top of each other and using tab items to select the desired panel.



FXTabBook(p, tgt=None, sel=0, opts=TABBOOK_NORMAL, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING)

Construct tab book.

Argument	Type	Default	Description
p	FXComposite		
tgt	FXObject	None	
sel	Int	0	
opts	Int	TABBOOK_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_SPACING	
pr	Int	DEFAULT_SPACING	
pt	Int	DEFAULT_SPACING	
pb	Int	DEFAULT_SPACING	

getDefaultHeight()

Return default height.

Reimplemented from FXTabBar.

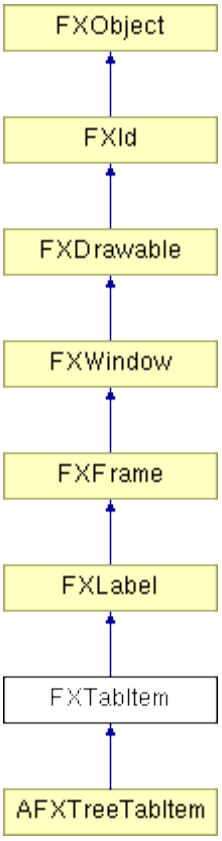
getDefaultWidth()

Return default width.

Reimplemented from FXTabBar.

FXTabItem

A tab item is placed in a tab bar or tab book. When selected, the tab item sends a message to its parent, and causes itself to become the active tab, and raised slightly above the other tabs. In the tab book, activating a tab item also causes the corresponding panel to be raised to the top.



FXTabItem(p, text, ic=None, opts=TAB_TOP_NORMAL, x=0, y=0, w=0, h=0, pl=6, pr=6, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

Construct a tab item.

Argument	Type	Default	Description
p	FXTabBar		
text	String		
ic	FXIcon	None	
opts	Int	TAB_TOP_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	6	
pr	Int	6	
pt	Int	DEFAULT_PAD	
pb	Int	DEFAULT_PAD	

canFocus()

Returns True because a tab item can receive focus.

Reimplemented from FXWindow.

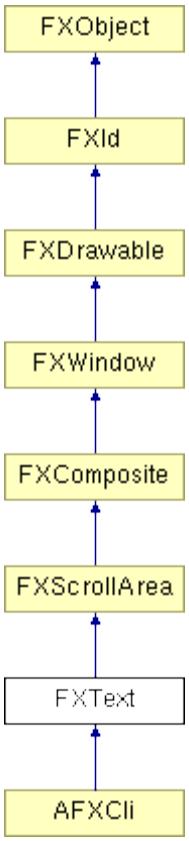
Global flags

Tab Item orientations which affect border

TAB_TOP	**Top side tabs.**
TAB_LEFT	**Left side tabs.**
TAB_RIGHT	**Right side tabs.**
TAB_BOTTOM	**Bottom side tabs.**

FXText

Multiline text widget



FXText(p, tgt=None, sel=0, opts=0, x=0, y=0, w=0, h=0)

Construct multi-line text widget.

Argument	Type	Default	Description
p	FXComposite		
tgt	FXObject	None	
sel	Int	0	
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

appendText(text, n, notify=False)

Append n characters of text at the end of the buffer.

Argument	Type	Default	Description
text	String		
n	Int		

notify	Bool	False	

canFocus()

Returns True because a text widget can receive focus.

Reimplemented from FXWindow.

create()

Create server-side resources.

Reimplemented from FXComposite.

detach()

Detach server-side resources.

Reimplemented from FXComposite.

disable()

Disable the text widget.

Reimplemented from FXWindow.

enable()

Enable the text widget.

Reimplemented from FXWindow.

extractText(text, pos, n)

Extract n characters of text from position pos.

Argument	Type	Default	Description
text	String		
pos	Int		
n	Int		

getBarColor()

Return bar color.

getBarColumns()

Return number of columns used for line numbers.

getChar(pos)

Get character at position in text buffer.

Argument	Type	Default	Description
pos	Int		

getContentHeight()

Get default height.

Reimplemented from FXScrollArea.

getContentWidth()

Get default width.

Reimplemented from FXScrollArea.

getCursorCol()

Return cursor row, i.e. indent position.

getCursorColor()

Return cursor color.

getCursorPos()

Return the cursor position.

getCursorRow()

Return cursor row.

getDefaultHeight()

Return default height.

Reimplemented from FXScrollArea.

getDefaultWidth()

Return default width.

Reimplemented from FXScrollArea.

getFont()

Return text font.

getLength()

Return length of buffer.

getMarginBottom()

Return bottom margin.

getMarginLeft()

Return left margin.

getMarginRight()

Return right margin.

getMarginTop()

Return top margin.

getNumberColor()

Return line number color.

getPosAt(x, y)

Return text position at given visible x,y coordinate.

Argument	Type	Default	Description
x	Int		
y	Int		

getSelBackColor()

Return selected background color.

getSelEndPos()

Return selendpos.

getSelStartPos()

Return selstartpos.

getSelTextColor()

Return selected text color.

getText()

Return text in the widget.

getText(text, n)

Retrieve text into buffer.

Argument	Type	Default	Description
text	String		
n	Int		

getTextColor()

Return text color.

getTopLine()

Return position of top line.

getVisCols()

Return number of visible columns.

getVisRows()

Return number of visible rows.

insertText(pos, text, n, notify=False)

Insert n characters of text at position pos into the buffer.

Argument	Type	Default	Description
pos	Int		
text	String		
n	Int		
notify	Bool	False	

isEditable()

Return True if text is editable.

isModified()

Return True if text was modified.

isPosSelected(pos)

Return True if position pos is selected.

Argument	Type	Default	Description
pos	Int		

killSelection(notify=False)

Unselect the text.

Argument	Type	Default	Description
notify	Bool	False	

lineEnd(pos)

Return position of end of line containing position pos.

Argument	Type	Default	Description
pos	Int		

lineStart(pos)

Return position of begin of line containing position pos.

Argument	Type	Default	Description
pos	Int		

makePositionVisible(pos)

Scroll text to make the given position visible.

Argument	Type	Default	Description
pos	Int		

moveContents(x, y)

Scroll the contents.

Reimplemented from FXScrollArea.

Argument	Type	Default	Description
x	Int		
y	Int		

nextLine(pos, nl=1)

Return start of next line.

Argument	Type	Default	Description
pos	Int		
nl	Int	1	

nextRow(pos, nr=1)

Return start of next row.

Argument	Type	Default	Description
pos	Int		
nr	Int	1	

position(x, y, w, h)

Move and resize this window in the parent's coordinates.

Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		

prevLine(pos, nl=1)

Return start of previous line.

Argument	Type	Default	Description
pos	Int		
nl	Int	1	

prevRow(pos, nr=1)

Return start of previous row.

Argument	Type	Default	Description
pos	Int		
nr	Int	1	

recalc()

Need to recalculate size.

Reimplemented from FXWindow.

removeText(pos, n, notify=False)

Remove n characters of text at position pos from the buffer.

Argument	Type	Default	Description
pos	Int		
n	Int		
notify	Bool	False	

replaceText(pos, m, text, n, notify=False)

Replace m characters at pos by n characters.

Argument	Type	Default	Description
pos	Int		
m	Int		
text	String		
n	Int		
notify	Bool	False	

resize(w, h)

Resize this window to the specified width and height.

Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

rowEnd(pos)

Return row end.

Argument	Type	Default	Description
pos	Int		

rowStart(pos)

Return row start.

Argument	Type	Default	Description
pos	Int		

setBarColor(clr)

Change bar color.

Argument	Type	Default	Description
clr	FXColor		

setBarColumns(cols)

Change number of columns used for line numbers.

Argument	Type	Default	Description
cols	Int		

setBottomLine(pos)

Make line containing pos the bottom line.

Argument	Type	Default	Description
pos	Int		

setCursorCol(col, notify=False)

Set cursor column.

Argument	Type	Default	Description
col	Int		
notify	Bool	False	

setCursorColor(clr)

Change cursor color.

Argument	Type	Default	Description
clr	FXColor		

setCursorPos(pos, notify=False)

Set the cursor position.

Argument	Type	Default	Description

pos	Int		
notify	Bool	False	

setCursorRow(row, notify=False)

Set cursor row.

Argument	Type	Default	Description
row	Int		
notify	Bool	False	

setEditable(edit=True)

Set editable flag.

Argument	Type	Default	Description
edit	Bool	True	

setFont(fnt)

Change text font.

Argument	Type	Default	Description
fnt	FXFont		

setMarginBottom(pb)

Change bottom margin.

Argument	Type	Default	Description
pb	Int		

setMarginLeft(pl)

Change left margin.

Argument	Type	Default	Description
pl	Int		

setMarginRight(pr)

Change right margin.

Argument	Type	Default	Description
pr	Int		

setMarginTop(pt)

Change top margin.

Argument	Type	Default	Description
pt	Int		

setModified(mod=True)

Set modified flag.

Argument	Type	Default	Description
mod	Bool	True	

setNumberColor(clr)

Change line number color.

Argument	Type	Default	Description
clr	FXColor		

setSelBackColor(clr)

Change selected background color.

Argument	Type	Default	Description
clr	FXColor		

setSelection(pos, len, notify=False)

Select len characters starting at given position pos.

Argument	Type	Default	Description
pos	Int		
len	Int		
notify	Bool	False	

setSelTextColor(clr)

Change selected text color.

Argument	Type	Default	Description
clr	FXColor		

setText(text, notify=False)

Change the text.

Argument	Type	Default	Description
text	String		
notify	Bool	False	

setText(text, n, notify=False)

Change the text in the buffer to new text.

Argument	Type	Default	Description
text	String		

n	Int		
notify	Bool	False	

setTextColor(clr)

Change text color.

Argument	Type	Default	Description
clr	FXColor		

setTopLine(pos)

Make line containing pos the top line.

Argument	Type	Default	Description
pos	Int		

setVisCols(cols)

Change number of visible columns.

Argument	Type	Default	Description
cols	Int		

setVisRows(rows)

Change number of visible rows.

Argument	Type	Default	Description
rows	Int		

Global flags

Text widget options

TEXT_READONLY **Text is NOT editable.**

TEXT_WORDWRAP **Wrap at word breaks.**

TEXT_OVERSTRIKE **Overstrike mode.**

TEXT_FIXEDWRAP **Fixed wrap columns.**

TEXT_NO_TABS **Insert spaces for tabs.**

TEXT_AUTOINDENT **Autoindent.**

TEXT_SHOWACTIVE **Show active line.**

Selection modes

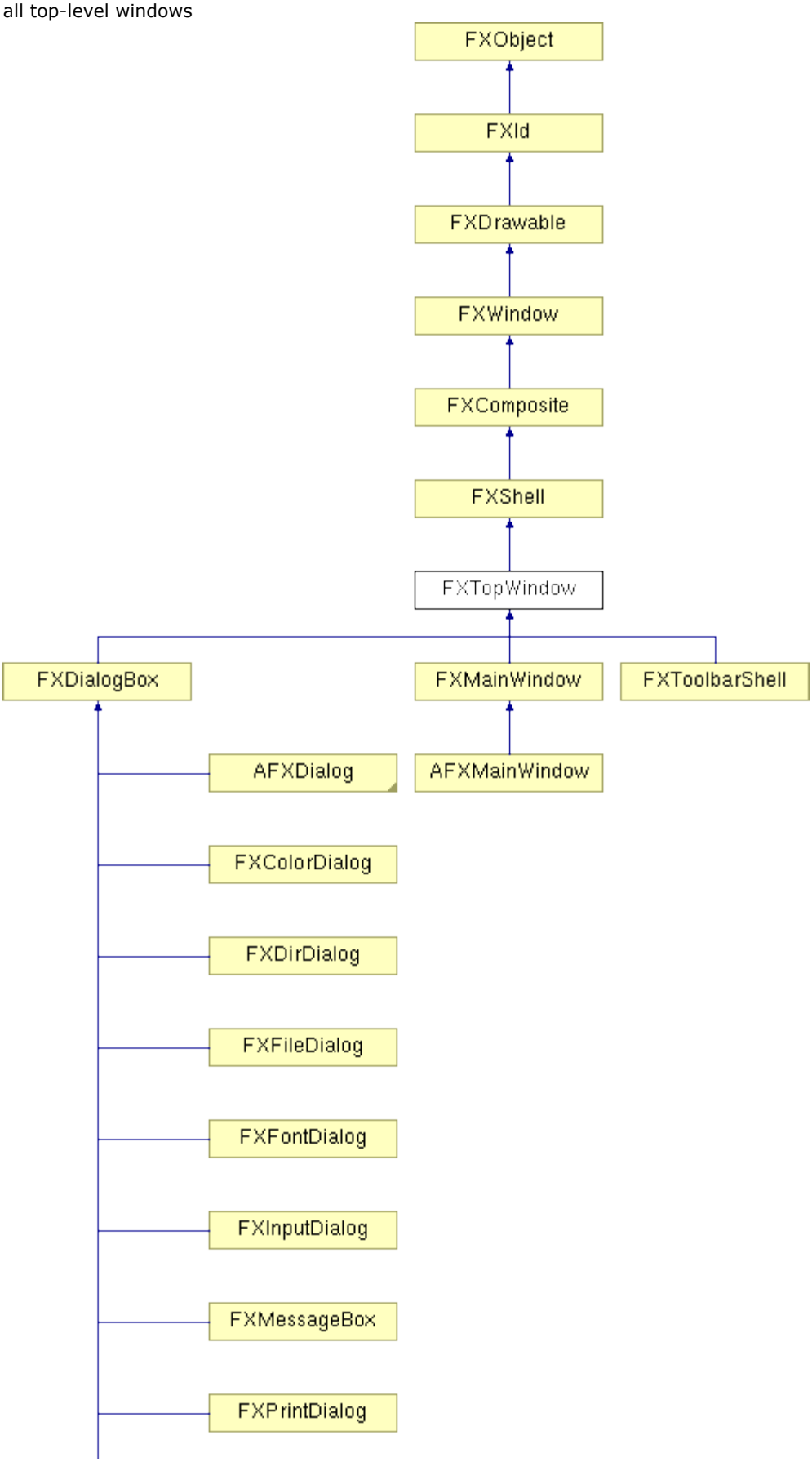
SELECT_CHARS **Select characters.**

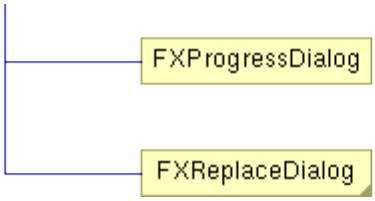
SELECT_WORDS **Select words.**

SELECT_LINES **Select lines.**

FXTopWindow

Abstract base class for all top-level windows





create()

Create server-side resources.
Reimplemented from FXShell.
Reimplemented in FXPrintDialog, FXToolbarShell, AFXMainWindow, and AFXDialog.

deiconify()

Deiconify window.

detach()

Detach the server-side resources for this window.
Reimplemented from FXComposite.

getDecorations()

Return current title and border decorations.

getDefaultHeight()

Return the default height of this window.
Reimplemented from FXComposite.
Reimplemented in FXToolbarShell, and AFXMainWindow.

getDefaultWidth()

Return the default width of this window.
Reimplemented from FXComposite.
Reimplemented in FXToolbarShell, and AFXMainWindow.

getHSpacing()

Return horizontal spacing between children.

getIcon()

Return window icon.

getMiniIcon()

Return window mini (title) icon.

getPackingHints()

Return packing hints for children.

getPadBottom()

Get bottom interior padding.

getPadLeft()

Get left interior padding.

getPadRight()

Get right interior padding.

getPadTop()

Get top interior padding.

getTitle()

Return window title.

getVSpacing()

Return vertical spacing between children.

hide()

Hide this window.

Reimplemented from FXWindow.

Reimplemented in AFXManagerMenuDB, AFXDialog, and AFXMessageDialog.

iconify()

Iconify window.

isIconified()

Return True if window has been iconified.

killFocus()

Remove the focus from this window.

Reimplemented from FXShell.

move(x, y)

Move this window to the specified position in the parent's coordinates.

Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		

y	Int		

place(placement)

Position the window based on placement.

Argument	Type	Default	Description
placement	Int		

position(x, y, w, h)

Move and resize this window in the parent's coordinates.

Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		

resize(w, h)

Resize this window to the specified width and height.

Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

setDecorations(decorations)

Change title and border decorations.

Argument	Type	Default	Description
decorations	Int		

setFocus()

Move the focus to this window.

Reimplemented from FXShell.

setHSpacing(hs)

Change horizontal spacing between children.

Argument	Type	Default	Description
hs	Int		

setIcon(ic)

Change window icon.

Argument	Type	Default	Description
ic	FXIcon		

setMiniIcon(ic)

Change window mini (title) icon.

Argument	Type	Default	Description
ic	FXIcon		

setPackingHints(ph)

Change packing hints for children.

Argument	Type	Default	Description
ph	Int		

setPadBottom(pb)

Change bottom padding.

Argument	Type	Default	Description
pb	Int		

setPadLeft(pl)

Change left padding.

Argument	Type	Default	Description
pl	Int		

setPadRight(pr)

Change right padding.

Argument	Type	Default	Description
pr	Int		

setPadTop(pt)

Change top padding.

Argument	Type	Default	Description
pt	Int		

setTitle(name)

Change window title.

Argument	Type	Default	Description
name	String		

setVSpacing(vs)

Change vertical spacing between children.

Argument	Type	Default	Description
vs	Int		

show(placement)

Show this window with given placement.

Argument	Type	Default	Description
placement	Int		

show()

Show this window.

Reimplemented from FXWindow.

Reimplemented in AFXDialog, AFXFileDialog, and AFXMessageDialog.

Class flags

ID_ICONIFY **Iconify the window.**

ID_DEICONIFY **Deiconify the window.**

ID_QUERY_DOCK **Toolbar asks to dock.**

Global flags

Title and border decorations

DECOR_NONE **Borderless window.**

DECOR_TITLE **Window title.**

DECOR_MINIMIZE **Minimize button.**

DECOR_MAXIMIZE **Maximize button.**

DECOR_CLOSE **Close button.**

DECOR_BORDER **Border.**

DECOR_RESIZE **Resize handles.**

DECOR_MENU **Window menu.**

Initial window placement

PLACEMENT_DEFAULT **Place it at the default size and location.**

PLACEMENT_VISIBLE **Place window to be fully visible.**

PLACEMENT_CURSOR **Place it under the cursor position.**

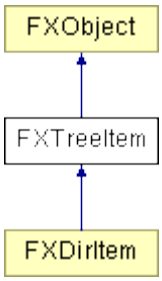
PLACEMENT_OWNER **Place it centered on its owner.**

PLACEMENT_SCREEN **Place it centered on the screen.**

PLACEMENT_MAXIMIZED **Place it maximized to the screen size.**

FXTreeItem

Tree list Item



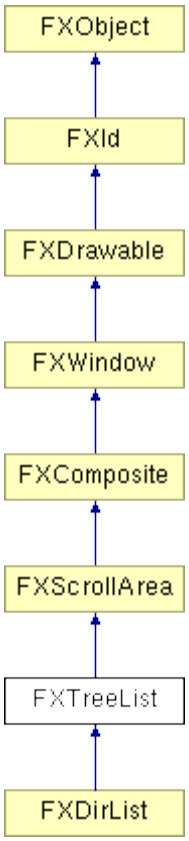
Global flags

Tree list styles

TREELIST_EXTENDEDSELECT	**Extended selection mode allows for drag-selection of ranges of items.**
TREELIST_SINGLESELECT	**Single selection mode allows up to one item to be selected.**
TREELIST_BROWSESELECT	**Browse selection mode enforces one single item to be selected at all times.**
TREELIST_MULTIPLESELECT	**Multiple selection mode is used for selection of individual items.**
TREELIST_AUTOSELECT	**Automatically select under cursor.**
TREELIST_SHOWS_LINES	**Lines shown.**
TREELIST_SHOWS_BOXES	**Boxes to expand shown.**
TREELIST_ROOT_BOXES	**Display root boxes also.**
TREELIST_CHECK_BOXES	**Display check boxes.**
TREELIST_PROPAGATE_CHECKS	**Propagate checked state to children and parents.**

FXTreeList

Tree list Widget



FXTreeList(p, nvis, tgt=None, sel=0, opts=TREELIST_NORMAL, x=0, y=0, w=0, h=0)

Construct a tree list with nvis visible items; the tree list is initially empty.

Argument	Type	Default	Description
p	FXComposite		
nvis	Int		
tgt	FXObject	None	
sel	Int	0	
opts	Int	TREELIST_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

addItemAfter(other, text, oi=None, ci=None, ptr=None, notify=False)

Append new item with given text and optional icon, and user-data pointer after to other item.

Argument	Type	Default	Description
other	FXTreeItem		

text	String		
oi	FXIcon	None	
ci	FXIcon	None	
ptr	String	None	
notify	Bool	False	

addItemAfter(other, text, oi=None, ci=None, ptr=None, notify=False)

Append new item with given text and optional icon, and user-data pointer after to other item.

Argument	Type	Default	Description
other	FXTreeItem		
text	String		
oi	FXIcon	None	
ci	FXIcon	None	
ptr	String	None	
notify	Bool	False	

addItemAfter(other, item, notify=False)

Append new [possibly subclassed] item after to other item.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

addItemBefore(other, text, oi=None, ci=None, ptr=None, notify=False)

Prepend new item with given text and optional icon, and user-data pointer prior to other item.

Argument	Type	Default	Description
other	FXTreeItem		
text	String		
oi	FXIcon	None	
ci	FXIcon	None	
ptr	String	None	
notify	Bool	False	

addItemBefore(other, text, oi=None, ci=None, ptr=None, notify=False)

Prepend new item with given text and optional icon, and user-data pointer prior to other item.

Argument	Type	Default	Description
other	FXTreeItem		
text	String		
oi	FXIcon	None	

ci	FXIcon	None	
ptr	String	None	
notify	Bool	False	

addItemBefore(other, item, notify=False)

Prepend new [possibly subclassed] item prior to other item.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

addItemFirst(p, text, oi=None, ci=None, ptr=None, notify=False)

Prepend new item with given text and optional icon, and user-data pointer as first child of p.

Argument	Type	Default	Description
p	FXTreeItem		
text	String		
oi	FXIcon	None	
ci	FXIcon	None	
ptr	String	None	
notify	Bool	False	

addItemFirst(p, text, oi=None, ci=None, ptr=None, notify=False)

Prepend new item with given text and optional icon, and user-data pointer as first child of p.

Argument	Type	Default	Description
p	FXTreeItem		
text	String		
oi	FXIcon	None	
ci	FXIcon	None	
ptr	String	None	
notify	Bool	False	

addItemFirst(p, item, notify=False)

Prepend new [possibly subclassed] item as first child of p.

Argument	Type	Default	Description
p	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

addItemLast(p, text, oi=None, ci=None, ptr=None, notify=False)

Append new item with given text and optional icon, and user-data pointer as last child of p.

Argument	Type	Default	Description
p	FXTreeItem		
text	String		
oi	FXIcon	None	
ci	FXIcon	None	
ptr	String	None	
notify	Bool	False	

addItemLast(p, text, oi=None, ci=None, ptr=None, notify=False)

Append new item with given text and optional icon, and user-data pointer as last child of p.

Argument	Type	Default	Description
p	FXTreeItem		
text	String		
oi	FXIcon	None	
ci	FXIcon	None	
ptr	String	None	
notify	Bool	False	

addItemLast(p, item, notify=False)

Append new [possibly subclassed] item as last child of p.

Argument	Type	Default	Description
p	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

canFocus()

Tree list can receive focus.

Reimplemented from FXWindow.

clearItems(notify=False)

Remove all items from list.

Argument	Type	Default	Description
notify	Bool	False	

closeItem(item, notify=False)

Close item.

Argument	Type	Default	Description
item	FXTreeItem		

notify	Bool	False	

collapseTree(tree, notify=False)

Collapse tree.

Argument	Type	Default	Description
tree	FXTreeItem		
notify	Bool	False	

create()

Create server-side resources.

Reimplemented from FXComposite.

Reimplemented in FXDirList.

deselectItem(item, notify=False)

Deselect item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

destroy()

Destroy server-side resources.

Reimplemented from FXComposite.

Reimplemented in FXDirList.

detach()

Detach server-side resources.

Reimplemented from FXComposite.

Reimplemented in FXDirList.

disableItem(item)

Disable item.

Argument	Type	Default	Description
item	FXTreeItem		

enableItem(item)

Enable item.

Argument	Type	Default	Description
item	FXTreeItem		

expandTree(tree, notify=False)

Expand tree.

Argument	Type	Default	Description
tree	FXTreeItem		
notify	Bool	False	

extendSelection(item, notify=False)

Extend selection from anchor item to item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

findItem(text, start=None, flags=SEARCH_FORWARD| SEARCH_WRAP)

Search items for item by name, starting from start item; the flags argument controls the search direction, and case sensitivity.

Argument	Type	Default	Description
text	String		
start	FXTreeItem	None	
flags	Int	SEARCH_FORWARD| SEARCH_WRAP	

getAnchorItem()

Return anchor item, if any.

getContentHeight()

Return content height.

Reimplemented from FXScrollArea.

getContentWidth()

Compute and return content width.

Reimplemented from FXScrollArea.

getCurrentItem()

Return current item, if any.

getCursorItem()

Return item under cursor, if any.

getDefaultHeight()

Return default height.

Reimplemented from FXScrollArea.

getDefaultWidth()

Return default width.

Reimplemented from FXScrollArea.

getFirstItem()

REturn first root item.

getFont()

Return text font.

getHelpText()

Get the status line help text for this list.

getIndent()

Return parent-child indent amount.

getItemAt(x, y)

Get item at x,y, if any.

Argument	Type	Default	Description
x	Int		
y	Int		

getItemCheck(item)

Returns the item checked state.

Argument	Type	Default	Description
item	FXTreeItem		

getItemClosedIcon(item)

Return item's closed icon.

Argument	Type	Default	Description
item	FXTreeItem		

getItemData(item)

Return item user-data pointer.

Argument	Type	Default	Description
item	FXTreeItem		

getItemHeight(item)

Return item height.

Argument	Type	Default	Description
item	FXTreeItem		

getItemOpenIcon(item)

Return item's open icon.

Argument	Type	Default	Description
item	FXTreeItem		

getItemText(item)

Return item's text.

Argument	Type	Default	Description
item	FXTreeItem		

getItemWidth(item)

Return item width.

Argument	Type	Default	Description
item	FXTreeItem		

getLastItem()

Return last root item.

getLineColor()

Return line color.

getListStyle()

Return list style.

getNumItems()

Return number of items.

getNumVisible()

Return number of visible items.

getSelBackColor()

Return selected text background.

getSelTextColor()

Return selected text color.

getTextColor()

Return normal text color.

hitItem(item, x, y)

Return item hit code: 0 outside, 1 icon, 2 text, 3 box.

Argument	Type	Default	Description
item	FXTreeItem		
x	Int		
y	Int		

isItemCurrent(item)

Return True if item is current.

Argument	Type	Default	Description
item	FXTreeItem		

isItemEnabled(item)

Return True if item is enabled.

Argument	Type	Default	Description
item	FXTreeItem		

isItemExpanded(item)

Return True if item expanded.

Argument	Type	Default	Description
item	FXTreeItem		

isItemLeaf(item)

Return True if item is a leaf-item, i.e. has no children.

Argument	Type	Default	Description
item	FXTreeItem		

isItemOpened(item)

Return True if item opened.

Argument	Type	Default	Description
item	FXTreeItem		

isItemSelected(item)

Return True if item is selected.

Argument	Type	Default	Description

item	FXTreeItem		

isItemVisible(item)

Return True if item is visible.

Argument	Type	Default	Description
item	FXTreeItem		

killFocus()

Remove the focus from this window.

Reimplemented from FXWindow.

killSelection(notify=False)

Deselect all items.

Argument	Type	Default	Description
notify	Bool	False	

makeItemVisible(item)

Scroll to make item visible.

Argument	Type	Default	Description
item	FXTreeItem		

moveItemAfter(other, item)

Move item after other.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		

moveItemBefore(other, item)

Move item before other.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		

openItem(item, notify=False)

Open item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

recalc()

Recalculate layout.

Reimplemented from FXWindow.

removeItem(item, notify=False)

Remove item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

removeItems(fm, to, notify=False)

Remove items in range [fm, to] inclusively.

Argument	Type	Default	Description
fm	FXTreeItem		
to	FXTreeItem		
notify	Bool	False	

reparentItem(item, p)

Reparent item under parent p.

Argument	Type	Default	Description
item	FXTreeItem		
p	FXTreeItem		

selectItem(item, notify=False)

Select item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

setAnchorItem(item)

Change anchor item.

Argument	Type	Default	Description
item	FXTreeItem		

setCurrentItem(item, notify=False)

Change current item.

Argument	Type	Default	Description
item	FXTreeItem		

notify	Bool	False	

setFocus()

Move the focus to this window.

Reimplemented from FXWindow.

setFont(fnt)

Change text font.

Argument	Type	Default	Description
fnt	FXFont		

setHelpText(text)

Set the status line help text for this list.

Argument	Type	Default	Description
text	String		

setIndent(in)

Change parent-child indent amount.

Argument	Type	Default	Description
in	Int		

setItemCheck(item, check, notify=False)

Sets the item checked state. Valid states are True, False and MAYBE. Returns True if the check value has changed, False otherwise.

Argument	Type	Default	Description
item	FXTreeItem		
check	Int		
notify	Bool	False	

setItemClosedIcon(item, icon)

Chance item's closed icon.

Argument	Type	Default	Description
item	FXTreeItem		
icon	FXIcon		

setItemData(item, ptr)

Change item user-data pointer.

Argument	Type	Default	Description
item	FXTreeItem		
ptr	String		

setItemOpenIcon(item, icon)

Change item's open icon.

Argument	Type	Default	Description
item	FXTreeItem		
icon	FXIcon		

setItemText(item, text)

Change item's text.

Argument	Type	Default	Description
item	FXTreeItem		
text	String		

setLineColor(clr)

Change line color.

Argument	Type	Default	Description
clr	FXColor		

setListStyle(style)

Change list style.

Argument	Type	Default	Description
style	Int		

setNumVisible(nvis)

Change number of visible items.

Argument	Type	Default	Description
nvis	Int		

setSelBackColor(clr)

Change selected text background.

Argument	Type	Default	Description
clr	FXColor		

setSelTextColor(clr)

Change selected text color.

Argument	Type	Default	Description
clr	FXColor		

setTextColor(clr)

Change normal text color.

Argument	Type	Default	Description
clr	FXColor		

toggleItem(item, notify=False)

Toggle item selection.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

updateItem(item)

Repaint item.

Argument	Type	Default	Description
item	FXTreeItem		

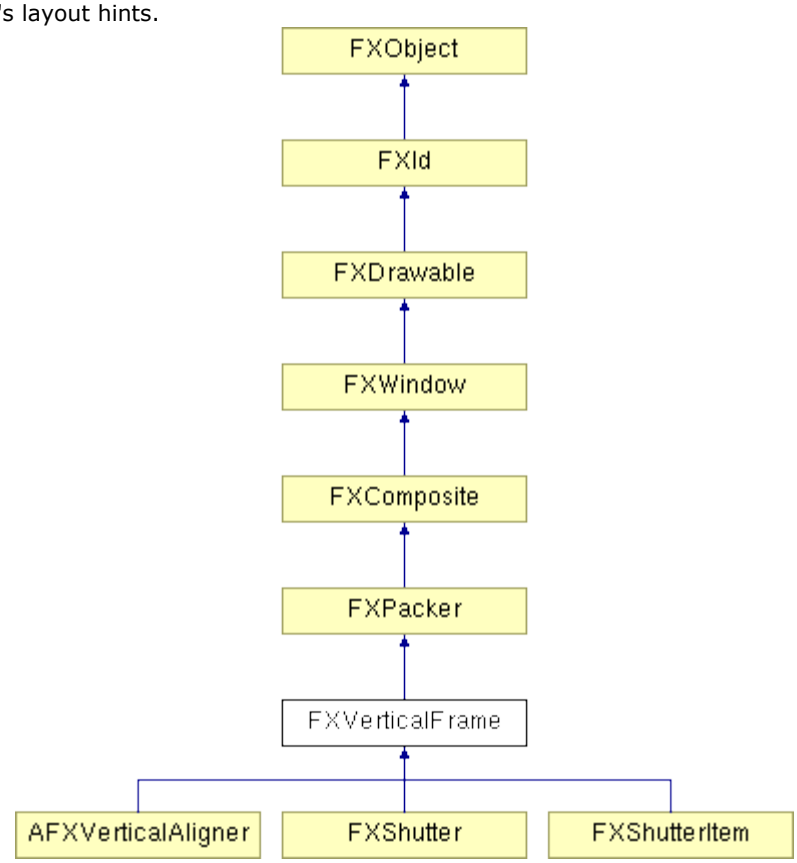
Global flags

Tree list styles

TREELIST_EXTENDEDSELECT	**Extended selection mode allows for drag-selection of ranges of items.**
TREELIST_SINGLESELECT	**Single selection mode allows up to one item to be selected.**
TREELIST_BROWSESELECT	**Browse selection mode enforces one single item to be selected at all times.**
TREELIST_MULTIPLESELECT	**Multiple selection mode is used for selection of individual items.**
TREELIST_AUTOSELECT	**Automatically select under cursor.**
TREELIST_SHOWS_LINES	**Lines shown.**
TREELIST_SHOWS_BOXES	**Boxes to expand shown.**
TREELIST_ROOT_BOXES	**Display root boxes also.**
TREELIST_CHECK_BOXES	**Display check boxes.**
TREELIST_PROPAGATE_CHECKS	**Propagate checked state to children and parents.**

FXVerticalFrame

Vertical frame layout manager widget is used to automatically place child-windows vertically from top-to-bottom, or bottom-to-top, depending on the child window's layout hints.



FXVerticalFrame(p, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

Construct a vertical frame layout manager.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	DEFAULT_SPACING	
pr	Int	DEFAULT_SPACING	
pt	Int	DEFAULT_SPACING	
pb	Int	DEFAULT_SPACING	
hs	Int	DEFAULT_SPACING	
vs	Int	DEFAULT_SPACING	

getDefaultHeight()

Return default height.

Reimplemented from FXPacker.

Reimplemented in AFXVerticalAligner.

getDefaultWidth()

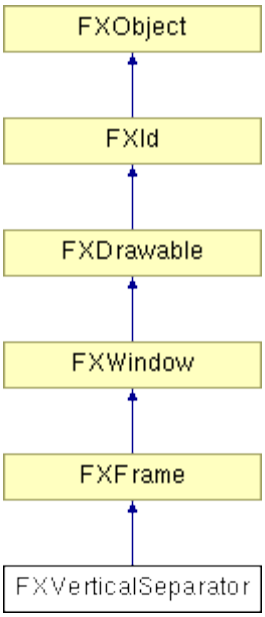
Return default width.

Reimplemented from FXPacker.

Reimplemented in AFXVerticalAligner.

FXVerticalSeparator

Vertical separator



FXVerticalSeparator(p, opts=SEPARATOR_GROOVE| LAYOUT_FILL_Y, x=0, y=0, w=0, h=0, pl=0, pr=0, pt=1, pb=1)

Constructor.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	SEPARATOR_GROOVE| LAYOUT_FILL_Y	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	
pl	Int	0	
pr	Int	0	
pt	Int	1	
pb	Int	1	

getDefaultHeight()

Return default height.

Reimplemented from FXFrame.

getDefaultWidth()

Return default width.

Reimplemented from FXFrame.

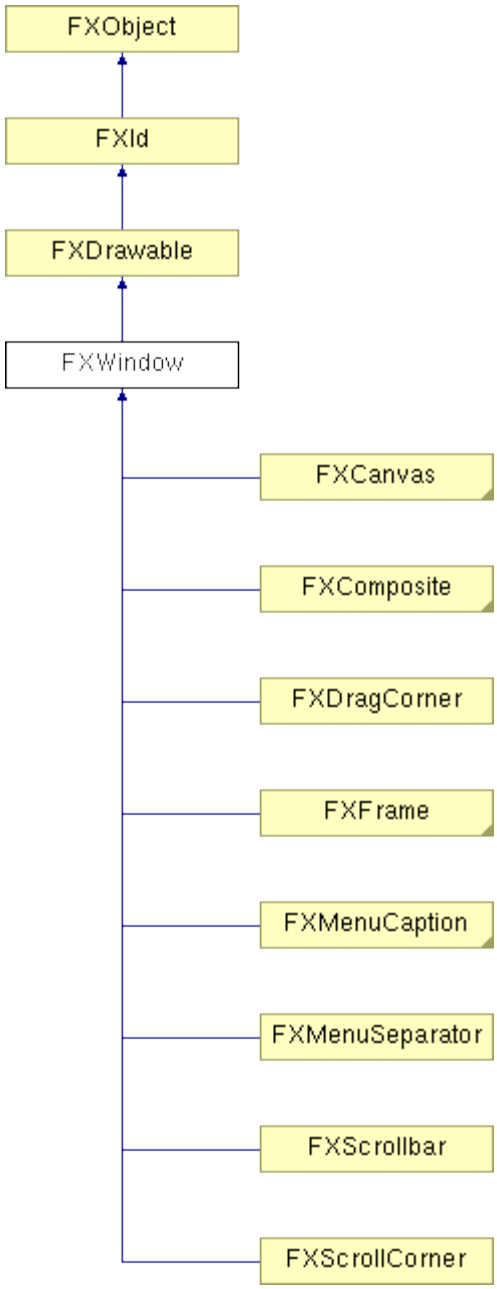
Global flags

Separator Options

SEPARATOR_NONE	**Nothing visible.**
SEPARATOR_GROOVE	**Etched-in looking groove.**
SEPARATOR_RIDGE	**Embossed looking ridge.**
SEPARATOR_LINE	**Simple line.**

FXWindow

Base class for all windows



FXWindow(p, opts=0, x=0, y=0, w=0, h=0)

Constructor.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	

h	Int	0	

canFocus()

Return True if this window is a control capable of receiving the focus.

Reimplemented in FXArrowButton, FXButton, FXCanvas, FXCheckButton, FXColorWell, FXDockHandler, FXIconList, FXImageView, FXList, FXMDIChild, FXMenuButton, FXMenuCascade, FXMenuCommand, FXMenuTitle, FXOption, FXOptionMenu, FXRadioButton, FXSlider, FXTabItem, FXTable, FXText, FXTextField, FXToggleButton, FXToolbarTab, FXTreeList, AFXBaseTable, AFXFloatSpinner, AFXFlyoutButton, AFXFlyoutItem, and AFXSlider.

childAtIndex(index)

Return the child window at specified index, or NULL if the index is negative or out of range

Reimplemented in AFXOptionTreeItem.

Argument	Type	Default	Description
index	Int		

containsChild(child)

Return True if specified window is a child of this window.

Argument	Type	Default	Description
child	FXWindow		

create()

Create all of the server-side resources for this window.

Reimplemented from FXId.

Reimplemented in FXColorBar, FXColorSelector, FXColorWell, FXColorWheel, FXComboBox, FXComposite, FXDirBox, FXDirList, FXDockTitle, FXDriveBox, FXFileList, FXFontSelector, FXGLCanvas, FXGLViewer, FXGroupBox, FXHeader, FXIconList, FXImageView, FXLabel, FXList, FXListBox, FXMDIChild, FXMenuButton, FXMenuCaption, FXMenuCascade, FXProgressBar, FXMenuTitle, FXOptionMenu, FXPrintDialog, FXRootWindow, FXScrollWindow, FXShell, FXSpinner, FXStatusline, FXTabBar, FXTable, FXText, FXTextField, FXToggleButton, FXToolbarShell, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, AFXManagerMenuPane, AFXMainWindow, AFXPromptArea, AFXBaseTable, AFXColorButton, AFXColorFlyout, AFXComboBox, AFXDialog, AFXFloatSpinner, AFXFlyoutButton, AFXListBox, AFXNote, AFXOptionTreeItem, AFXPrimFloatSpinner, AFXProgressBar, AFXSpinner, AFXTable, AFXTextField, and AFXVerticalAligner.

destroy()

Destroy the server-side resources for this window.

Reimplemented from FXId.

Reimplemented in FXComboBox, FXComposite, FXDirBox, FXDirList, FXDriveBox, FXFileList, FXGLCanvas, FXListBox, FXMenuCascade, FXOptionMenu, FXRootWindow, FXTreeList, FXTreeListBox, AFXManagerMenuCascade, AFXColorFlyout, and AFXTable.

detach()

Detach the server-side resources for this window.

Reimplemented from FXId.

Reimplemented in FXColorBar, FXColorWell, FXColorWheel, FXComboBox, FXComposite, FXDirBox, FXDirList, FXDockTitle, FXDriveBox, FXFileList, FXGLCanvas, FXGLViewer, FXGroupBox, FXHeader, FXIconList, FXImageView, FXLabel, FXList, FXListBox, FXMDIChild, FXMenuButton, FXMenuCaption, FXMenuCascade, FXProgressBar, FXMenuTitle, FXOptionMenu, FXRootWindow, FXStatusline, FXTable, FXText, FXToggleButton, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, AFXBaseTable, AFXColorFlyout, AFXFlyoutButton, AFXNote, and AFXTable.

disable()

Disable the window from receiving mouse and keyboard events.

Reimplemented in FXArrowButton, FXComboBox, FXGroupBox, FXLabel, FXListBox, FXMenuCaption, FXScrollCorner, FXSlider, FXSpinner, FXText, FXTextField, FXToolbarTab, FXTreeListBox, AFXAutoComputeGroup, AFXManagerMenuDB, AFXColorButton, AFXColorFlyout, AFXComboBox, AFXFloatSpinner, AFXFlyoutButton, AFXList, AFXListBox, AFXNote, AFXOptionTreeItem, AFXPrimFloatSpinner, AFXSlider, AFXSpinner, AFXTable, and AFXTextField.

enable()

Enable the window to receive mouse and keyboard events.

Reimplemented in FXArrowButton, FXComboBox, FXGroupBox, FXLabel, FXListBox, FXMenuCaption, FXScrollCorner, FXSlider, FXSpinner, FXText, FXTextField, FXToolbarTab, FXTreeListBox, AFXAutoComputeGroup, AFXManagerMenuDB, AFXColorButton, AFXColorFlyout, AFXComboBox, AFXFloatSpinner, AFXFlyoutButton, AFXList, AFXListBox, AFXNote, AFXOptionTreeItem, AFXPrimFloatSpinner, AFXSlider, AFXSpinner, AFXTable, and AFXTextField.

forceRefresh()

Force a GUI update of this window and its children.

getBackColor()

Get background color.

Reimplemented in FXComboBox, and FXListBox.

getCursorPosition()

Returns a sequence of (status, x, y, mouseButtonState) representing the relative location of the cursor in the widget.

getDefaultHeight()

Return the default height of this window.

Reimplemented in FX4Splitter, FXArrowButton, FXCheckButton, FXColorBar, FXColorWell, FXColorWheel, FXComboBox, FXComposite, FXDial, FXDockSite, FXDockTitle, FXDragCorner, FXFrame, FXGroupBox, FXHeader, FXHorizontalFrame, FXLabel, FXList, FXListBox, FXMDIDeleteButton, FXMDIRestoreButton, FXMDIMaximizeButton, FXMDIMinimizeButton, FXMDIWindowButton, FXMDIChild, FXMatrix, FXMenuButton, FXMenuCaption, FXMenuCommand, FXProgressBar, FXMenuSeparator, FXMenuTitle, FXOption, FXOptionMenu, FXPacker, FXPopup, FXRadioButton, FXRootWindow, FXScrollArea, FXScrollbar, FXHorizontalSeparator, FXVerticalSeparator, FXSlider, FXSpinner, FXSplitter, FXStatusbar, FXStatusline, FXSwitcher, FXTabBar, FXTabBook, FXTable, FXText, FXTextField, FXToggleButton, FXToolbar, FXToolbarGrip, FXToolbarShell, FXToolbarTab, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, FXVerticalFrame, AFXMainWindow, AFXToolbarGroup, AFXBaseTable, AFXList, AFXOptionTreeList, AFXPrimFloatSpinner, AFXProgressBar, AFXSlider, AFXTable, AFXTreeTable, and AFXVerticalAligner.

getDefaultWidth()

Return the default width of this window.

Reimplemented in FX4Splitter, FXArrowButton, FXCheckButton, FXColorBar, FXColorWell, FXColorWheel, FXComboBox, FXComposite, FXDial, FXDockSite, FXDockTitle, FXDragCorner, FXFrame, FXGroupBox, FXHeader, FXHorizontalFrame, FXLabel, FXList, FXListBox, FXMDIDeleteButton, FXMDIRestoreButton, FXMDIMaximizeButton, FXMDIMinimizeButton, FXMDIWindowButton, FXMDIChild, FXMatrix, FXMenuButton, FXMenuCaption, FXMenuCommand, FXProgressBar, FXMenuSeparator, FXMenuTitle, FXOption, FXOptionMenu, FXPacker, FXPopup, FXRadioButton, FXRootWindow, FXScrollArea, FXScrollbar, FXHorizontalSeparator, FXVerticalSeparator, FXSlider, FXSpinner, FXSplitter, FXStatusbar, FXStatusline, FXSwitcher, FXTabBar, FXTabBook, FXTable, FXText, FXTextField, FXToggleButton, FXToolbar, FXToolbarGrip, FXToolbarShell, FXToolbarTab, FXTooltip, FXTopWindow, FXTreeList, FXTreeListBox, FXVerticalFrame, AFXMainWindow, AFXToolbarGroup, AFXBaseTable, AFXOptionTreeItem, AFXOptionTreeList, AFXPrimFloatSpinner, AFXProgressBar, AFXSlider, AFXTable, AFXTextField, AFXTreeTable, and AFXVerticalAligner.

getFirst()

Return a pointer to this window's first child window , if any.

Reimplemented in AFXOptionTreeItem.

getHeightForWidth(givenwidth)

Return height for given width.

Reimplemented in FXDockSite.

Argument	Type	Default	Description
givenwidth	Int		

getKey()

Return window key.

getLast()

Return a pointer to this window's last child window, if any.

Reimplemented in AFXOptionTreeItem.

getLayoutHints()

Get layout hints for this window.

getNext()

Return a pointer to the next (sibling) window, if any.

Reimplemented in AFXOptionTreeItem.

getOwner()

Return a pointer to the owner window.

Reimplemented in AFXMenuCascade, AFXMenuCommand, AFXMenuPane, AFXMenuTitle, AFXToolbarGroup, and AFXToolboxGroup.

getParent()

Return a pointer to the parent window.

Reimplemented in AFXOptionTreeItem.

getPrev()

Return a pointer to the previous (sibling) window , if any.

Reimplemented in AFXOptionTreeItem.

getRoot()

Return a pointer to the root window.

getSelector()

Get the message identifier for this window.

getShell()

Return a pointer to the shell window.

getTarget()

Get the message target object for this window, if any.

getWidthForHeight(givenheight)

Return width for given height.

Reimplemented in FXDockSite.

Argument	Type	Default	Description
givenheight	Int		

getX()

Get this window's x-coordinate, in the parent's coordinate system.

getY()

Get this window's y-coordinate, in the parent's coordinate system.

grab(confineTo=None)

Grab the mouse to this window; future mouse events will be reported to this window even while the cursor goes outside of this window

Argument	Type	Default	Description
confineTo	FXWindow	None	

grabbed()

Return True if the window has been grabbed.

hasFocus()

Return True if this window has the focus.

hide()

Hide this window.

Reimplemented in FXTopWindow, AFXManagerMenuDB, AFXMenuTitle, AFXToolbarGroup, AFXToolbarGroupRender, AFXToolbarGroupVisibility, AFXDialog, AFXFlyoutItem, AFXMessageDialog, AFXOptionTreeItem, and AFXProgressBar.

indexOfChild(window)

Return the index (starting from zero) of the specified child window, or -1 if the window is not a child or NULL

Argument	Type	Default	Description
window	FXWindow		

isActive()

Return True if the window is active.

Reimplemented in AFXToolbarGroup.

isChildOf(window)

Return True if specified window is this window's parent.

Argument	Type	Default	Description
window	FXWindow		

isDefault()

Return True if this is the default window.

isEnabled()

Return True if this window is able to receive mouse and keyboard events.

isInitial()

Return True if this is the initial default window.

linkAfter(sibling)

Relink this window after sibling in the window list.

Argument	Type	Default	Description
sibling	FXWindow		

linkBefore(sibling)

Relink this window before sibling in the window list.

Argument	Type	Default	Description
sibling	FXWindow		

move(x, y)

Move this window to the specified position in the parent's coordinates.

Reimplemented in FXMDIChild, FXRootWindow, and FXTopWindow.

Argument	Type	Default	Description
x	Int		
y	Int		

numChildren()

Return the number of child windows for this window.

Reimplemented in AFXOptionTreeItem.

position(x, y, w, h)

Move and resize this window in the parent's coordinates.

Reimplemented in FXIconList, FXMDIChild, FXRootWindow, FXText, and FXTopWindow.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		

recalc()

Mark this window's layout as dirty.

Reimplemented in FXIconList, FXList, FXMDIClient, FXRootWindow, FXShell, FXTable, FXText, FXTreeList, AFXBaseTable, AFXSlider, and AFXTable.

repaint()

If marked but not yet painted, paint the entire window.

repaint(x, y, w, h)

If marked but not yet painted, paint the given area.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		

resize(w, h)

Resize this window to the specified width and height.

Reimplemented from FXDrawable.

Reimplemented in FXIconList, FXMDIChild, FXRootWindow, FXText, and FXTopWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

setBackColor(clr)

Set window background color.

Reimplemented in FXComboBox, and FXListBox.

Argument	Type	Default	Description
clr	FXColor		

setCursorPosition(x, y)

Warp the cursor to the new position.

Argument	Type	Default	Description
x	Int		
y	Int		

setFocus()

Move the focus to this window.

Reimplemented in FXButton, FXColorWell, FXIconList, FXList, FXMenuCascade, FXMenuCommand, FXMenuTitle, FXOption, FXPopup, FXRootWindow, FXShell, FXTable, FXText, FXTextField, FXTopWindow, FXTreeList, AFXBaseTable, AFXFlyoutItem, and AFXTextField.

setHeight(h)

Set the window height.

Argument	Type	Default	Description
h	Int		

setInitial(enable=True)

Make this window the initial default window.

Argument	Type	Default	Description
enable	Bool	True	

setKey(k)

Change window key.

Argument	Type	Default	Description
k	Int		

setLayoutHints(lout)

Set layout hints for this window.

Argument	Type	Default	Description
lout	Int		

setSelector(sel)

Set the message identifier for this window.

Argument	Type	Default	Description
sel	Int		

setTarget(t)

Set the message target object for this window.

Argument	Type	Default	Description
t	FXObject		

setWidth(w)

Set the window width.

Argument	Type	Default	Description
w	Int		

setX(x)

Set this window's x-coordinate, in the parent's coordinate system.

Argument	Type	Default	Description
x	Int		

setY(y)

Set this window's y-coordinate, in the parent's coordinate system.

Argument	Type	Default	Description
y	Int		

show()

Show this window.

Reimplemented in FXTooltip, FXTopWindow, AFXMenuTitle, AFXToolbarGroup, AFXToolbarGroupRender, AFXToolbarGroupVisibility, AFXDialog, AFXFileDialog, AFXMessageDialog, AFXOptionTreeItem, AFXProgressBar, and AFXSlider.

shown()

Return True if the window is shown.

translateCoordinatesTo(tox, toy, towindow, fromx, fromy)

Translate coordinates from this window's coordinate space to towindow's coordinate space.

Argument	Type	Default	Description
tox	Int		
toy	Int		
towindow	FXWindow		
fromx	Int		
fromy	Int		

ungrab()

Release the mouse grab.

update()

Mark the entire window client area dirty.

update(x, y, w, h)

Mark the specified rectangle dirty, i.e. to be repainted.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		

Global flags

Layout hints for child widgets

LAYOUT_NORMAL	**Default layout mode.**
LAYOUT_SIDE_TOP	**Pack on top side (default).**
LAYOUT_SIDE_BOTTOM	**Pack on bottom side.**
LAYOUT_SIDE_LEFT	**Pack on left side.**
LAYOUT_SIDE_RIGHT	**Pack on right side.**
LAYOUT_FILL_COLUMN	**Matrix column is stretchable.**
LAYOUT_FILL_ROW	**Matrix row is stretchable.**
LAYOUT_LEFT	**Stick on left (default).**
LAYOUT_RIGHT	**Stick on right.**
LAYOUT_CENTER_X	**Center horizontally.**
LAYOUT_FIX_X	**X fixed.**
LAYOUT_TOP	**Stick on top (default).**
LAYOUT_BOTTOM	**Stick on bottom.**
LAYOUT_CENTER_Y	**Center vertically.**
LAYOUT_FIX_Y	**Y fixed CAE: Copied from FOX 1.4.34 to support dockable tool bars.**
LAYOUT_DOCK_SAME	**Dock on same galley if it fits.**
LAYOUT_DOCK_NEXT	**Dock on next galley LAYOUT_RESERVED_1 = 0x00000040, LAYOUT_RESERVED_2 = 0x00000080,**
LAYOUT_RESERVED_1	**CAE End.**
LAYOUT_FIX_WIDTH	**Width fixed.**
LAYOUT_FIX_HEIGHT	**height fixed**
LAYOUT_MIN_WIDTH	**Minimum width is the default.**
LAYOUT_MIN_HEIGHT	**Minimum height is the default.**
LAYOUT_FILL_X	**Stretch or shrink horizontally.**
LAYOUT_FILL_Y	**Stretch or shrink vertically.**
LAYOUT_EXPLICIT	**Explicit placement.**

Frame border appearance styles (for subclasses)

FRAME_NONE	**Default is no frame.**
FRAME_SUNKEN	**Sunken border.**
FRAME_RAISED	**Raised border.**
FRAME_THICK	**Thick border.**
FRAME_GROOVE	**A groove or etched-in border.**
FRAME_RIDGE	**A ridge or embossed border.**
FRAME_LINE	**Simple line border.**

FRAME_NORMAL	Regular raised/thick border.

Packing style (for packers)

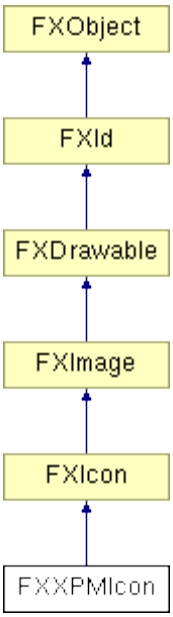
PACK_NORMAL	Default is each its own size.
PACK_UNIFORM_HEIGHT	Uniform height.
PACK_UNIFORM_WIDTH	Uniform width.

BackgroundStyle

BACKGROUND_NORMAL	Default.
BACKGROUND_H_GRADIENT	Horizontal gradient background.
BACKGROUND_V_GRADIENT	Vertical gradient background.
BACKGROUND_PLAIN	plain background

FXXPMIcon

X Pixmap icon.



FXXPMIcon(a, pix=None, clr=FXRGB(192, 192, 192), opts=0, w=1, h=1)

Construct icon from compiled-in X Pixmap format.

Argument	Type	Default	Description
a	FXApp		
pix	String	None	
clr	FXColor	FXRGB(192, 192, 192)	
opts	Int	0	
w	Int	1	
h	Int	1	

Message maps

Message maps are used by classes derived from FXObject (which includes most GUI classes) to route messages sent to the object to a message handling method. A typical use might be to route a message from a button in a dialog to one of the dialog's methods so that some action is taken when the button is pressed.

A message consists of two parts, a type and an ID. The type indicates what kind of action generated the message, while the ID identifies who sent the message. These two numbers are combined into a single value using the MKUINT function.

When the message handler method is called, it gets passed three arguments: the sender, the selector, and some optional data. Your message handler prototype must contain these three arguments, which are typically specified as *sender*, *sel*, *ptr*. The optional data is sometimes used by the sender to include extra data that might be useful to the message handler (for example the coordinates of the cursor when a mouse button was clicked). However, due to the language differences between C++ and Python, this optional data is not available for use in Python scripts.

Refer to the Abaqus GUI Toolkit User's Guide for more details.

FXMAPFUNC(object, messageType, messageId, method)

Creates an entry in the object's message map that will route a message to a method.

Argument	Type	Default	Description
object	FXObject		An instance of the class in which the message map entry is to be made. Typically this is "self".
messageType	Int		An integer flag specifying the message type (e.g. SEL_COMMAND).
messageId	Int		An integer specifying the message ID.
method	Function		The method to which the message is to be routed. This method must be specified by including the class name (e.g. MyDB.myMethod).

FXMAPFUNCS(object, messageType, startMessageId, endMessageId, method)

Creates multiple entries in the object's message map that will route messages to a method.

Argument	Type	Default	Description
object	FXObject		An instance of the class in which the message map entry is to be made. Typically this is "self".
messageType	Int		An integer flag specifying the message type (e.g. SEL_COMMAND).
startMessageId	Int		An integer specifying the starting message ID.
endMessageId	Int		An integer specifying the ending message ID.
method	Function		The method to which the message is to be routed. This method must be specified by including the class name (e.g. MyDB.myMethod).

MKUINT(messageId, messageType)

Creates a message selector by combining a message ID and a message type.

Argument	Type	Default	Description
messageId	Int		An integer specifying the message ID.
messageType	Int		An integer flag specifying the message type (e.g. SEL_COMMAND).

SELID(selector)

Returns the message ID from a message selector.

Argument	Type	Default	Description
selector	Int		A message selector.

SELTYPE(selector)

Returns the message type from a message selector.

Argument	Type	Default	Description
selector	Int		A message selector.

Auxiliary functions

This section lists various auxiliary GUI toolkit functions.

addExitCallback(callback)

Registers a callback function to be called when the application is about to exit.

Argument	Type	Default	Description
callback	Function		The function to be called when the application is about to exit.

afxCreateIcon(fileName)

Returns an icon created by reading the specified file, which can be in one of these formats: BMP, GIF, PNG, XPM. The file format is assumed from the file extension (which is not case sensitive). Returns 0 if the file cannot be opened.

Argument	Type	Default	Description
fileName	String		File name.

afxCreateBMPIcon(fileName)

Returns an icon created by reading the specified file in BMP format. Returns 0 if the file cannot be opened.

Argument	Type	Default	Description
fileName	String		File name.

afxCreateGIFIcon(fileName)

Returns an icon created by reading the specified file in GIF format. Returns 0 if the file cannot be opened.

Argument	Type	Default	Description
fileName	String		File name.

afxCreatePNGIcon(fileName)

Returns an icon created by reading the specified file in PNG format. Returns 0 if the file cannot be opened.

Argument	Type	Default	Description
fileName	String		File name.

afxCreateXPMIcon(fileName)

Returns an icon created by reading the specified file in XPM format. Returns 0 if the file cannot be opened.

Argument	Type	Default	Description
fileName	String		File name.

afxDisableWidgetTree(widget)

Disables the specified widget and all its children.

Argument	Type	Default	Description
widget	Widget		The parent widget, which will be disabled along with all its children.

afxEnableWidgetTree(widget)

Enables the specified widget and all its children.

Argument	Type	Default	Description
widget	Widget		The parent widget, which will be enabled along with all its children.

afxGetIcon(fileName, size)

Enables you to use Abaqus/CAE icons in your customization. Set the *size* argument to 1 for normal-sized icons or to 0 for small icons.

Argument	Type	Default	Description
fileName	String		File name.
size	String		Icon size.

displayURL(url)

Displays the specified URL in a web browser. Returns the status of the call. This call will use an open web browser if there is one. This method can be accessed via webBrowser.displayURL from module uti. See also openWithURL.

Argument	Type	Default	Description
url	String		The URL of the page to be displayed.

getAFXApp()

Returns the application object.

getAFXAliasMap().setPrefix(widget, prefix)

Sets the prefix key —usually the dialog box class name—of a widget.

Argument	Type	Default	Description
widget	FXWindow		The widget to which the prefix is being set.
prefix	String		The prefix key for the widget.

getAFXAliasMap().setName(widget, name)

Sets the name key of a widget.

Argument	Type	Default	Description
widget	FXWindow		The widget for which the name is being set. The name key is used, along with a prefix, to identify the widget.
name	String		The name of the widget.

getAFXFont(opts=FONT_PROPORTIONAL)

Returns the specified font.

Argument	Type	Default	Description
opts	Int	FONT_PROPORTIONAL	Type of font to get. Possible choices are FONT_PROPORTIONAL, FONT_MONOSPACE, FONT_REGULAR, FONT_BOLD, FONT_ITALIC, FONT_BASE, or FONT_SMALL.

afxGetColorHexSpecFromID(colorId)

Returns the equivalent hex string for the specified color index.

Argument	Type	Default	Description
colorId	Integer		The index of the color to be converted to a hex string. For example, green would be FXRGB(0, 255,0). See the Colors appendix of the Abaqus GUI Toolkit User's Guide for details.

afxGetColorHexSpecFromName(colorName)

Returns the equivalent hex string for the specified color name.

Argument	Type	Default	Description
colorName	String		The name of the color to be converted to a hex string. For example, 'Red'. See the Colors appendix of the Abaqus GUI Toolkit User's Guide for details.

getCurrentContext()

Returns the current GUI context dictionary, which contains the following keys: *mdbName*, *viewportName*, *objectPath*, *objectType*, *modelName*, and *moduleName*. You can be notified of context changes by using the registerCurrentContext function.

getCurrentModuleGui()

Returns the current moduleGui object. You can call that object's getModuleName method to check its name.

getCursorPosition()

Returns a tuple of status,x,y,buttonState. The status (TRUE or FALSE) indicates the success of the call. The x and y values represent the position of the cursor in the window's coordinate system (origin is in the upper left, positive Y points downward). This method is defined in FXWindow, so it may be called on any object derived from FXWindow.

getDisplayedObjectType()

Returns the type of the object displayed in the current viewport. Possible return values are: PART, ASSEMBLY, ODB, XY_PLOT, SKETCH, or None.

getSeparator(parent, count)

Returns the nth (specified by the count argument) separator widget of the parent.

Argument	Type	Default	Description
parent	Widget		The widget to be searched (children are not traversed).
count	Int		The number of the separator (e.g. specify 2 to get the second separator).

getWidgetFromText(parent, text)

Returns a widget whose label or tip text matches the specified text and is also a child of the specified widget.

Argument	Type	Default	Description
parent	Widget		The widget at which the search begins (children are traversed too).
text	String		The text to be matched (must be an exact match).

openWithURL(url)

Displays the specified URL in a web browser. Returns the status of the call. This call will always open a new web browser. This method can be accessed via webBrowser.openWithURL from module uti. See also displayURL.

Argument	Type	Default	Description
url	String		The URL of the page to be displayed.

registerCurrentContext(callbackFunction)

Registers a query on the current context. The specified callback function will be invoked when the current viewport or displayed object changes. Use the getCurrentContext method to get the values of the current objects. Note that unregisterCurrentContext should be called with the same argument to unregister the query when it is no longer needed.

Argument	Type	Default	Description
callbackFunction	Function		Function to be called when the current context changes.

registerPluginToolset()

Registers the plug-in toolset with the application.

objectPath.registerQuery(callbackFunction, callOnRegister=TRUE)

Registers a query on the object. The specified callback function will be invoked when the object changes. For more information, see "Receiving notification of kernel data changes," in the Abaqus GUI Toolkit User's Guide. Note that unregisterQuery should be called with the same argument to unregister the query when it is no longer needed.

Argument	Type	Default	Description
callbackFunction	Function		Function to be called when the object changes.
callOnRegister	Bool	TRUE	If TRUE, the callback function will be called when the query is registered.

objectPath.registerInclusive(callbackFunction, callOnRegister=TRUE)

Registers a query on the object and its children. The specified callback function will be invoked when the object or its children change. Note that unregisterQuery should be called with the same argument to unregister the query when it is no longer needed.

Argument	Type	Default	Description
callbackFunction	Function		Function to be called when the object or its children change.
callOnRegister	Bool	TRUE	If TRUE, the callback function will be called when the query is registered.

removeExitCallback(callback)

Unregisters a callback function to be called when the application is about to exit. The function must have been previously registered using addExitCallback.

Argument	Type	Default	Description
callback	Function		The function to be removed from the callback list.

sendCommand(writeToJournal, command, writeToReplay=True, writeToJournal=False)

Sends a command string to the kernel.

Note: The *writeToJournal* argument should not usually be necessary. If the command is a built-in Abaqus Scripting Interface command, it will be written to the journal file automatically. If the command is not a built-in Abaqus Scripting Interface command but it changes the mdb using built-in Abaqus Scripting Interface commands, the changes to the mdb will be written to the journal file automatically. However, if the command is not a built-in Abaqus Scripting Interface command and it makes changes to the mdb that would not otherwise be written to the journal file, the command itself should use the `journalMethodCall` function to write itself to the journal file.

Argument	Type	Default	Description
writeToJournal			
command	String		Command string.
writeToReplay	Bool	True	If True, the command will be written to the replay file.
writeToJournal	Bool	False	If True, the command will be written to the journal file.

setCurrentModel(modelName)

Sets the current model to the specified model name.

Argument	Type	Default	Description
modelName	String		The name of the model to be made current.

setSwitchModuleHook(callbackFunction)

Sets a function that will be called whenever the user switches into a GUI module. When the user switches into a GUI module, the specified function will be called, passing it the name (shown in the Module combo box) of the new module. Note that the setSwitchModuleHook function does not take keyword arguments and, if necessary, it may be called multiple times.

Argument	Type	Default	Description
callbackFunction	Function		The function to be called when a GUI module is switched into. Note that you cannot issue a kernel command from within this function or it will cause the application to lock up.

shutdown()

Exits Abaqus/CAE. This is equivalent to selecting File->Exit.

switchModule(moduleName)

Switches into a GUI module.

Argument	Type	Default	Description
moduleName	String		Module to switch into.

unregisterCurrentContext(callbackFunction)

Unregisters a query on the current context. The specified callback function should be the same argument that was used to register the query.

Argument	Type	Default	Description

callbackFunction	Function		Function to be called when the current context changes.

objectPath.unregisterQuery(callbackFunction)

Unregisters a query on the object. The specified callback function should be the same argument that was specified to register the query.

Argument	Type	Default	Description
callbackFunction	Function		Function to be called when the object changes.

Message dialogs

The following methods provide access to the standard message dialogs.

showAFXDismissableWarningDialog(owner, message, tgt=None, sel=0)

Posts a modal dismissable warning dialog box that has a "Show this dialog next time" check button, via which the user can request not to see a specific type of messages in the future. The dialog box has a getCheckButtonState method that can be called to get the state of the check button in the dialog box.

Argument	Type	Default	Description
owner	FXWindow		Window over which the dialog box is to be centered.
message	String		Text to be displayed in the dialog box.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.

showAFXErrorDialog(owner, message, tgt=None, sel=0)

Shows an error dialog box.

Argument	Type	Default	Description
owner	FXWindow		Window over which the dialog box is to be centered.
message	String		Text to be displayed in the dialog box.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.

showAFXItemsWarningDialog(owner, topMessage, items, bottomMessage, buttonIds=YES|NO|CANCEL, tgt=None, sel=0)

Shows a modal warning dialog box that has a scrollable list of items and two messages: one placed above and one below the list. .

Argument	Type	Default	Description
owner	FXWindow		Window over which the dialog box is to be centered.
topMessage	String		Text to be displayed on top of the list of items.
items	String		A String containing a comma-separated list of items to be displayed between the topMessage and bottomMessage.
bottomMessage	String		Text to be displayed below the list of items.
buttonIds	Int	YES|NO|CANCEL	ID's of the action area buttons to be created.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.

showAFXWarningDialog(owner, message, buttonIds=YES|NO|CANCEL, tgt=None, sel=0)

Shows a warning dialog box.

Argument	Type	Default	Description

owner	FXWindow		Window over which the dialog box is to be centered.
message	String		Text to be displayed in the dialog box.
buttonIds	Int	YES|NO|CANCEL	ID's of the action area buttons to be created.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.

showAFXInformationDialog(owner, message, tgt=None, sel=0)

Shows an information dialog box.

Argument	Type	Default	Description
owner	FXWindow		Window over which the dialog box is to be centered.
message	String		Text to be displayed in the dialog box.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.

Plug-in registration commands

Plug-in commands register kernel and GUI plug-ins in either the **Plug-ins** menu or in a toolbox.

Kernel plug-in registration commands

Kernel plug-in commands register kernel plug-ins in either the **Plug-ins** menu or in a toolbox.

Access

```
from abaqusGui import getAFXApp
toolset=getAFXApp().getAFXMainWindow().getPluginToolset()
```

---

registerKernelMenuButton(...)

This method registers a kernel plug-in in the **Plug-ins** menu.

Path

`toolset.registerKernelMenuButton`

Required arguments

moduleName

A String specifying the name of the module to be imported. The module must contain the function to be executed.

functionName

A String specifying the name of the function to be executed. The function must be located in *moduleName*.

buttonText

A String specifying the text to be displayed in the **Plug-ins** menu. Use a pipe (|) between words to specify submenus. The default value is the empty string.

Optional arguments

icon

A `FXIcon` object specifying an icon to be displayed to the left of the text in the menu. For more information, see `afxCreateIcon` in the "Auxiliary functions" section of this guide. The default value is `None`.

applicableModules

The SymbolicConstant ALL or a sequence of one or more Strings specifying the list of modules to which this plug-in applies. If a plug-in is not applicable to a module, it will be hidden when the user switches into that module. Possible values of the Strings in the sequence are "Part", "Property", "Assembly", "Step", "Interaction", "Load", "Mesh", "Job", "Visualization", and "Sketch". The default value is ALL.

version

A String specifying the version of the plug-in. The version is displayed in the About Plug-ins dialog box. The default value is "N/A".

author

A String specifying the author of the plug-in. The author is displayed in the About Plug-ins dialog box. The default value is "N/A".

description

A String specifying the description of the plug-in. The description is displayed in the About Plug-ins dialog box. The default value is "N/A".

helpUrl

A String specifying the universal resource locator (URL) that points to the help for this plug-in. This URL can be loaded in a web browser from the View button in the About Plug-ins dialog box. The default value is "N/A".

Return value

`None`

Exceptions

None.

---

registerKernelToolButton(...)

Registers a kernel plug-in in a toolbox.

Path

`toolset.registerKernelToolButton`

Required arguments

toolboxName

A String specifying the name of the toolbox in which the button will be shown. The name appears in the toolbox title bar.

moduleName

A String specifying the name of the module to be imported. The module must contain the function to be executed.

functionName

A String specifying the name of the function to be executed. The function must be located in *moduleName*.

Optional arguments

buttonText

A String specifying the text to be displayed in the **Plug-ins** menu. The default value is the empty string.

icon

A `FXIcon` object specifying an icon to be displayed to the left of the text in the menu. For more information, see `afxCreateIcon` in the "Auxiliary functions" section of this guide. The default value is `None`.

applicableModules

The SymbolicConstant ALL or a sequence of one or more Strings specifying the list of modules to which this plug-in applies. If a plug-in is not applicable to a module, it will be hidden when the user switches into that module. Possible values of the Strings in the sequence are "Part", "Property", "Assembly", "Step", "Interaction", "Load", "Mesh", "Job", "Visualization", and "Sketch". The default value is ALL.

version

A String specifying the version of the plug-in. The version is displayed in the About Plug-ins dialog box. The default value is "N/A".

author

A String specifying the author of the plug-in. The author is displayed in the About Plug-ins dialog box. The default value is "N/A".

description

A String specifying the description of the plug-in. The description is displayed in the About Plug-ins dialog box. The default value is "N/A".

helpUrl

A String specifying the universal resource locator (URL) that points to the help for this plug-in. This URL can be loaded in a web browser from the View button in the About Plug-ins dialog box. The default value is "N/A".

Return value

`None`

Exceptions

None.

GUI plug-in registration commands

GUI plug-in commands register GUI plug-ins either in the **Plug-ins** menu or in a toolbox.

registerGuiMenuButton(...)

Registers a GUI plug-in in the **Plug-ins** menu.

Path

`toolset.registerGuiMenuButton`

Required arguments

object

The GUI object to which a (messageId, SEL_COMMAND) message will be sent. The object must have been inherited from FXObject.

buttonText

A String specifying the text to be displayed in the **Plug-ins** menu. Use a pipe (|) between words to specify submenus. The default value is the empty string.

Optional arguments

messageId

An Int specifying the ID to be used when sending a command to the GUI object. The default value is *AFXMode.ID_ACTIVATE*.

icon

A `FXIcon` object specifying an icon to be displayed to the left of the text in the menu. For more information, see `afxCreateIcon` in the "Auxiliary functions" section of this guide. The default value is `None`.

kernelInitString

A String specifying the string sent to the kernel the first time this plug-in is invoked. The string is intended to initialize the kernel (typically by importing modules) in preparation for commands that will be sent by this plug-in's GUI. The default value is the empty string.

applicableModules

The SymbolicConstant ALL or a sequence of one or more Strings specifying the list of modules to which this plug-in applies. If a plug-in is not applicable to a module, it will be hidden when the user switches into that module. Possible values of the Strings in the sequence are "Part", "Property", "Assembly", "Step", "Interaction", "Load", "Mesh", "Job", "Visualization", and "Sketch". The default value is ALL.

version

A String specifying the version of the plug-in. The version is displayed in the About Plug-ins dialog box. The default value is "N/A".

author

A String specifying the author of the plug-in. The author is displayed in the About Plug-ins dialog box. The default value is "N/A".

description

A String specifying the description of the plug-in. The description is displayed in the About Plug-ins dialog box. The default value is "N/A".

helpUrl

A String specifying the universal resource locator (URL) that points to the help for this plug-in. This URL can be loaded in a web browser from the View button in the About Plug-ins dialog box. The default value is "N/A".

Return value

`None`

Exceptions

None.

registerGuiToolButton(...)

Registers a GUI plug-in in a toolbox.

Path

`toolset.registerGuiToolButton`

Required arguments

toolboxName

A String specifying the name of the toolbox in which the button will be shown. The name appears in the toolbox title bar.

object

The GUI object to which a (messageId, SEL_COMMAND) message will be sent. The object must have been inherited from FXObject.

Optional arguments

messageId

An Int specifying the ID to be used when sending a command to the GUI object. The default value is *AFXMode.ID_ACTIVATE*.

buttonText

A String specifying the text to be displayed in the **Plug-ins** menu. The default value is the empty string.

icon

A `FXIcon` object specifying an icon to be displayed to the left of the text in the menu. For more information, see `afxCreateIcon` in the "Auxiliary functions" section of this guide. The default value is `None`.

kernelInitString

A String specifying the string sent to the kernel the first time this plug-in is invoked. The string is intended to initialize the kernel (typically by importing modules) in preparation for commands that will be sent by this plug-in's GUI. The default value is the empty string.

applicableModules

The SymbolicConstant ALL or a sequence of one or more Strings specifying the list of modules to which this plug-in applies. If a plug-in is not applicable to a module, it will be hidden when the user switches into that module. Possible values of the Strings in the sequence are "Part", "Property", "Assembly", "Step", "Interaction", "Load", "Mesh", "Job", "Visualization", and "Sketch". The default value is ALL.

version

A String specifying the version of the plug-in. The version is displayed in the About Plug-ins dialog box. The default value is "N/A".

author

A String specifying the author of the plug-in. The author is displayed in the About Plug-ins dialog box. The default value is "N/A".

description

A String specifying the description of the plug-in. The description is displayed in the About Plug-ins dialog box. The default value is "N/A".

helpUrl

A String specifying the universal resource locator (URL) that points to the help for this plug-in. This URL can be loaded in a web browser from the View button in the About Plug-ins dialog box. The default value is "N/A".

Return value

`None`

Exceptions

None.

Summary of changes between Abaqus 6.14 and 2016

To upgrade your scripts to the latest version, select Plug-ins->Abaqus->Upgrade Scripts in Abaqus/CAE. Listed below are the changes to the toolkit in this release. They are color-coded as follows:

(**Blue bold text**)	New method or argument
(*Red italic text*)	Removed method or argument
(*Green italic text*)	Changed argument; for example, the argument type has changed or the possible values of the SymbolicConstant have changed.

Refer to the Abaqus 6.14 GUI Toolkit Reference manual for the description of methods that have been removed in Abaqus 2016.

AFXMainWindow

AFXMainWindow.ID_TOGGLE_FULL_SCREEN: Toggle full screen.

Summary of changes between Abaqus 6.13 and 6.14

To upgrade your scripts to the latest version, select **Plug-ins>Abaqus>Upgrade Scripts** in Abaqus/CAE. Listed below are the changes to the toolkit in this release. They are color-coded as follows:

(**Blue bold text**)	New method or argument
(*Red italic text*)	Removed method or argument
(*Green italic text*)	Changed argument; for example, the argument type has changed or the possible values of the SymbolicConstant have changed.

Refer to the Abaqus 6.13 GUI Toolkit Reference Guide for the description of methods that have been removed in Abaqus 6.14.

FXFrame

FXFrame(p, opts=FRAME_NORMAL, x=0, y=0, w=0, h=0, pl=2, pr=2, pt=2, pb=2)

FXFrame(p, opts=FRAME_NORMAL, x=0, y=0, w=0, h=0, pl=DEFAULT_PAD, pr=DEFAULT_PAD, pt=DEFAULT_PAD, pb=DEFAULT_PAD)

FXPacker

FXPacker(p, opts=0, x=0, y=0, w=0, h=0, pl=4, pr=4, pt=4, pb=4, hs=4, vs=4)

FXPacker(p, opts=0, x=0, y=0, w=0, h=0, pl=DEFAULT_SPACING, pr=DEFAULT_SPACING, pt=DEFAULT_SPACING, pb=DEFAULT_SPACING, hs=DEFAULT_SPACING, vs=DEFAULT_SPACING)

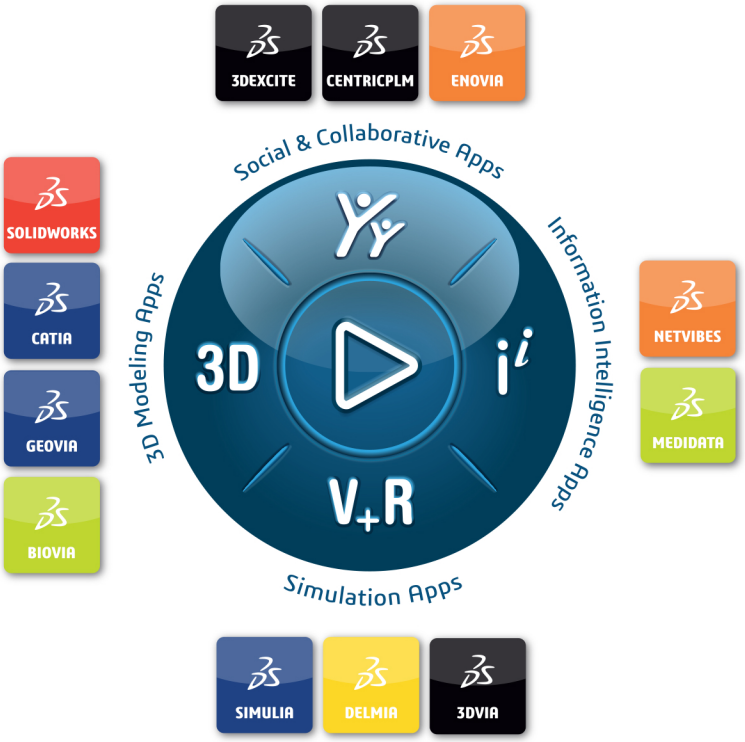
FXTreeList

getItemCheck(item)

setItemCheck(item, check, notify=False)

Global functions

afxGetIcon(fileName, size)



Our **3D**EXPERIENCE® platform powers our brand applications, serving 12 industries, and provides a rich portfolio of industry solution experiences.

Dassault Systèmes is a catalyst for human progress. We provide business and people with collaborative virtual environments to imagine sustainable innovations. By creating virtual twin experiences of the real world with our **3D**EXPERIENCE platform and applications, our customers can redefine the creation, production and life-cycle-management processes of their offer and thus have a meaningful impact to make the world more sustainable. The beauty of the Experience Economy is that it is a human-centered economy for the benefit of all – consumers, patients and citizens.

Dassault Systèmes brings value to more than 300,000 customers of all sizes, in all industries, in more than 150 countries. For more information, visit **www.3ds.com**.

Europe/Middle East/Africa
Dassault Systèmes
10, rue Marcel Dassault
CS 40501
78946 Vélizy-Villacoublay Cedex
France

Asia-Pacific
Dassault Systèmes
17F, Foxconn Building,
No. 1366, Lujiazui Ring Road
Pilot Free Trade Zone, Shanghai 200120
China

Americas
Dassault Systèmes
175 Wyman Street
Waltham, Massachusetts
02451-1223
USA

DASSAULT SYSTEMES

©2026 Dassault Systèmes. All rights reserved. **3DEXPERIENCE**, the 3DS logo, the Compass icon, IFWE, 3DEXCITE, 3DVIA, BIOVIA, CATIA, CENTRIC PLM, DELMIA, ENOVIA, GEOVIA, MEDIDATA, NETVIBES, OUTSCALE, SIMULIA and SOLIDWORKS are commercial trademarks or registered trademarks of Dassault Systèmes, a European company (Societas Europaea) incorporated under French law, and registered with the Versailles trade and companies registry under number 322 306 440, or its subsidiaries in the United States and/or other countries.