Abaqus 6.11

GUI Toolkit Reference Manual

Part I: Abaqus GUI Toolkit Reference Manual

1. All Classes

1.1 AFXApp

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

1.1.1 AFXApp(...)

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.

1.1.2 create()

Creates windows for the application.
Reimplemented from FXApp.

Arguments

None.

1.1.3 getAFXMainWindow()

Returns a pointer to the AFXMainWindow.

Arguments

None.

1.1.4 getBasePrerelease()

Returns True if the base product is a prerelease.

Arguments

None.

1.1.5 getBaseProductName()

Returns the base product name.

Arguments

None.

1.1.6 getBaseVersionNumbers(...)

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.

1.1.7 getKernelInitializationCommand()

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

Arguments

None.

1.1.8 getPrerelease()

Returns True if this is a prerelease.

Arguments

None.

1.1.9 getProductName()

Returns the product name.

Arguments

None.

1.1.10 getVersionNumbers()

Returns the major, minor, and update numbers.

Arguments

None.

1.1.11 init(...)

Initializes the application and connects to the kernel.

Argument	Type	Default	Description
argc	Int		
argv	String		

1.1.12 isLocked()

Returns True if the GUI is locked or False if otherwise.
Reimplemented from FXApp.

Arguments

None.

1.1.13 isProductCAE()

Returns True if the base product is Abaqus/CAE.

Arguments

None.

1.1.14 isProductViewer()

Returns True if the base product is Abaqus/Viewer.

Arguments

None.

1.1.15 isStudentEdition()

Returns True if the base product is a student edition.

Arguments

None.

1.1.16 lock()

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

Arguments

None.

1.1.17 run()

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

Arguments

None.

1.1.18 runUntil(...)

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

Argument	Type	Default	Description
condition	Int		

1.1.19 unlock()

Unlocks the GUI.

Arguments

None.

Class flags

Message ID's.

ID_QUERY_GUILOCK	Used to query whether the GUI is locked.
ID_SHOW_HOURGLASS	Used to change the cursor.

1.2 AFXBoolKeyword

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

1.2.1 AFXBoolKeyword(...)

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.

1.2.2 getTypeName()

Returns the name of the keyword type.
Implements AFXKeyword.

Arguments

None.

1.2.3 getValue()

Returns the keyword's current value.

Arguments

None.

1.2.4 getValueAsString()

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

Arguments

None.

1.2.5 isValueChanged()

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

Arguments

None.

1.2.6 setDefaultValue(...)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	Bool		Default value.

1.2.7 setDefaultValueByString(...)

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.

1.2.8 setDefaultValueByString(...)

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.

1.2.9 setValue(...)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	Bool		New value.

1.2.10 setValueByString(...)

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.

1.2.11 setValueByString(...)

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.

1.2.12 setValueToDefault(...)

Sets the keyword value to its default.

Argument	Type	Default	Description
ignoreUnspecified	Bool	False	Not used.

1.2.13 setValueToPrevious()

Sets the keyword value to its previous value.
Implements AFXKeyword.

Arguments

None.

1.2.14 syncPreviousValue()

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

Arguments

None.

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.

1.3 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").

1.3.1 AFXColorButton(...)

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).

1.3.2 create()

Creates the color button widget.
Reimplemented from FXComposite.

Arguments

None.

1.3.3 disable()

Disables the color button.
Reimplemented from FXWindow.

Arguments

None.

1.3.4 enable()

Enables the color button.
Reimplemented from FXWindow.

Arguments

None.

1.3.5 getHelpText()

Returns the status line help text.

Arguments

None.

1.3.6 getLabelFont()

Returns the label font.

Arguments

None.

1.3.7 getLabelText()

Returns the label string.

Arguments

None.

1.3.8 getRGBA()

Returns the color of the button.

Arguments

None.

1.3.9 getTipText()

Returns the tool tip message.

Arguments

None.

1.3.10 setHelpText(...)

Sets the status line help text.

Argument	Type	Default	Description
text	String		

1.3.11 setLabelFont(...)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		

1.3.12 setLabelText(...)

Sets the label string.

Argument	Type	Default	Description
txt	String		

1.3.13 setRGBA(...)

Sets the color of the button.

Argument	Type	Default	Description
clr	FXColor		

1.3.14 setTipText(...)

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.

1.4 AFXColorComboBox

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

1.4.1 AFXColorComboBox(...)

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).

Argument	Type	Default	Description
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.

1.5 AFXColumnItems

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

1.5.1 AFXColumnItems(...)

Constructor.

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

1.5.2 AFXColumnItems(...)

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).

1.5.3 getReferenceColumn()

Returns the index of the table reference column.

Arguments

None.

1.5.4 getSelector()

Returns the message ID.

Arguments

None.

1.5.5 getTarget()

Returns the message target.

Arguments

None.

1.5.6 setReferenceColumn(...)

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

Argument	Type	Default	Description
index	Int		Table column index.

1.5.7 setSelector(...)

Sets the message ID.

Argument	Type	Default	Description
sel	Int		New message ID.

1.5.8 setTarget(...)

Sets the message target.

Argument	Type	Default	Description
tgt	FXObject		New message target.

Class flags

Message ID's.

ID_TABLE	Table ID.

1.6 AFXComboBox

This class contains a label that precedes a combo box, which allows the user to select entries from a drop-down list.

1.6.1 AFXComboBox(...)

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).

1.6.2 appendItem(...)

Adds an item to the end of the list.

Argument	Type	Default	Description
text	String		Text.
sel	Int	0	Selector.

1.6.3 clearItems()

Removes all items from the list.

Arguments

None.

1.6.4 create()

Creates the combo box.
Reimplemented from FXComposite.

Arguments

None.

1.6.5 disable()

Disables the combo box.
Reimplemented from FXWindow.

Arguments

None.

1.6.6 enable()

Enables the combo box.
Reimplemented from FXWindow.

Arguments

None.

1.6.7 getCheck()

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

Arguments

None.

1.6.8 getCurrentItem()

Returns the index of the current item.

Arguments

None.

1.6.9 getHelpText()

Returns the status line help text.

Arguments

None.

1.6.10 getItemData(...)

Returns the data for the specified item.

Argument	Type	Default	Description
index	Int		Index.

1.6.11 getItemIndexForData(...)

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

Argument	Type	Default	Description
data			

1.6.12 getItemIndexForFloat(...)

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

Argument	Type	Default	Description
val	Float		

1.6.13 getItemProvider()

Returns the provider of the combo box's items.

Arguments

None.

1.6.14 getItemText(...)

Returns the text for the specified item.

Argument	Type	Default	Description
index	Int		Index.

1.6.15 getLabelFont()

Returns the label font.

Arguments

None.

1.6.16 getLabelText()

Returns the label string.

Arguments

None.

1.6.17 getNumColumns()

Returns the number of columns.

Arguments

None.

1.6.18 getNumItems()

Returns the number of items in the list.

Arguments

None.

1.6.19 getNumVisible()

Returns the number of visible items.

Arguments

None.

1.6.20 getText()

Returns the text displayed in input field.

Arguments

None.

1.6.21 getTipText()

Returns the tool tip message.

Arguments

None.

1.6.22 insertItem(...)

Inserts a new item at the specified index position.

Argument	Type	Default	Description
index	Int		Index.
text	String		Text.
sel	Int	0	Selector.

1.6.23 isEditable()

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

Arguments

None.

1.6.24 isItemCurrent(...)

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

Argument	Type	Default	Description
index	Int		Index.

1.6.25 isReadOnlyState()

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

Arguments

None.

1.6.26 removeItem(...)

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

Argument	Type	Default	Description
index	Int		Index.

1.6.27 replaceItem(...)

Replaces the item at the specified index position.

Argument	Type	Default	Description
index	Int		Index.
text	String		Text.
sel	Int	0	Selector.

1.6.28 setCheck(...)

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

Argument	Type	Default	Description
state	Bool		Button state.

1.6.29 setCheckButtonSelector(...)

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

Argument	Type	Default	Description
sel	Int		Selector.

1.6.30 setCheckButtonTarget(...)

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.

1.6.31 setCurrentItem(...)

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

Argument	Type	Default	Description
index	Int		Index.

1.6.32 setEditable(...)

Sets the editable state for the input field.

Argument	Type	Default	Description
edit	Bool	True	Editable state.

1.6.33 setFocusAndSelection()

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

Arguments

None.

1.6.34 setFocusToCheckButton()

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

Arguments

None.

1.6.35 setFocusToComboBox()

Moves the focus to the input field of the widget.

Arguments

None.

1.6.36 setHelpText(...)

Sets the status line help text.

Argument	Type	Default	Description
text	String		Help text.

1.6.37 setItemData(...)

Sets the data for the specified item.

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

1.6.38 setItemProvider(...)

Sets the provider of this object items.

Argument	Type	Default	Description
cp	FXObject		Item provider.

1.6.39 setItemText(...)

Sets the text for the specified item.

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

1.6.40 setLabelFont(...)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		Label font.

1.6.41 setLabelText(...)

Sets the label string.

Argument	Type	Default	Description
txt	String		Label text.

1.6.42 setMaxVisible(...)

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.

1.6.43 setNumColumns(...)

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.

1.6.44 setNumVisible(...)

Sets the number of visible items.

Argument	Type	Default	Description
nvis	Int		Number of visible items.

1.6.45 setReadOnlyState(...)

Sets the read-only state of the combo box.

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

1.6.46 setText(...)

Sets the text displayed in the input field.

Argument	Type	Default	Description
txt	String		Input field text.

1.6.47 setTipText(...)

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.

1.7 AFXCommand

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

1.7.1 AFXCommand(...)

Constructor.

Argument	Type	Default	Description
mode	AFXMode		Host mode.
method	String		Method.
objectName	String	”	Object name.

Argument	Type	Default	Description
registerQuery	Bool	False	True if a query should be registered when the command is used for the GUI.

1.7.2 activate()

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

Arguments

None.

1.7.3 deactivate()

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

Arguments

None.

1.7.4 getCommandString()

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

Arguments

None.

1.7.5 getExpandedObjectName()

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

Arguments

None.

1.7.6 getKeyword(...)

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

Argument	Type	Default	Description
name	String		Keyword name.

1.7.7 getKeyword(...)

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).

1.7.8 getMethod()

Returns the command's method.

Arguments

None.

1.7.9 getNumKeywords()

Returns the number of keywords.

Arguments

None.

1.7.10 getObjectName()

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

Arguments

None.

1.7.11 isActive()

Returns True if the command is active.

Arguments

None.

1.7.12 isQueryNeeded()

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

Arguments

None.

1.7.13 isRequired()

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

Arguments

None.

1.7.14 setKeywordValuesToDefaults(...)

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.

1.7.15 setKeywordValuesToPrevious()

Sets the values of all keywords to their previous values.

Arguments

None.

1.7.16 setMethod(...)

Sets the command's method.

Argument	Type	Default	Description
method	String		Method.

1.7.17 setObjectName(...)

Sets the object name.

Argument	Type	Default	Description
objectName	String		Object name.

1.7.18 setRequired(...)

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		

1.7.19 syncKeywordPreviousValues()

Synchronizes the current values and previous values of all keywords.

Arguments

None.

1.7.20 verify()

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

Arguments

None.

1.8 AFXComTableKeyword

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

1.8.1 AFXComTableKeyword(...)

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).

Argument	Type	Default	Description
opts	Int	0	Options.

1.8.2 equal(...)

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.

1.8.3 getColumnStyle(...)

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

Argument	Type	Default	Description
index	Int		Column index.

1.8.4 getColumnType(...)

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

Argument	Type	Default	Description
index	Int		Column index.

1.8.5 getDefaultStyle()

Returns the default style for the table elements.

Arguments

None.

1.8.6 getDefaultType()

Returns the default type for the table elements.

Arguments

None.

1.8.7 getDefaultValues()

Returns the default values for this table.

Arguments

None.

1.8.8 getFormattedValue(...)

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.

1.8.9 getMaxNumColumns()

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

Arguments

None.

1.8.10 getMinNumColumns()

Returns the minimum number of columns.

Arguments

None.

1.8.11 getNumColumns(...)

Returns the number of columns in the row.

Argument	Type	Default	Description
row	Int		Row index.

1.8.12 getNumRows()

Returns the number of rows in the table.

Arguments

None.

1.8.13 getRow(...)

Returns a string with the contents of a table row.

Argument	Type	Default	Description
row	Int		Row index.

1.8.14 getTypeName()

Returns the name of the table keyword type.
Implements AFXKeyword.
Reimplemented in AFXTableKeyword.

Arguments

None.

1.8.15 getValue(...)

Returns the value of a table element.

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

1.8.16 getValueAsDouble()

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

Arguments

None.

1.8.17 getValueAsInt()

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

Arguments

None.

1.8.18 getValueAsString()

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

Arguments

None.

1.8.19 getValueForBlank(...)

Returns the element value substituted for blank for the column.

Argument	Type	Default	Description
column	Int		Column index.

1.8.20 getValues()

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

Arguments

None.

1.8.21 getValuesForBlanks()

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

Arguments

None.

1.8.22 insertColumns(...)

Inserts columns starting at the given index.

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

1.8.23 insertRows(...)

Inserts rows starting at the given index.

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

1.8.24 isValueChanged()

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

Arguments

None.

1.8.25 removeColumns(...)

Removes columns starting at the given index.

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

1.8.26 removeRows(...)

Removes rows starting at the given index.

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

1.8.27 setColumnStyle(...)

Sets the style of the column elements.

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

1.8.28 setColumnType(...)

Sets the type of the column elements.

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

1.8.29 setDefaultStyle(...)

Sets the default style for the table elements.

Argument	Type	Default	Description
style	Int		New default style.

1.8.30 setDefaultType(...)

Sets the default type for table elements.

Argument	Type	Default	Description
type	Int		New default type.

1.8.31 setDefaultValues(...)

Sets the default values for this table.

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

1.8.32 setMaxNumColumns(...)

Sets the maximum number of columns.

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

1.8.33 setMinNumColumns(...)

Sets the minimum number of columns.

Argument	Type	Default	Description
length	Int		New minimum length.

1.8.34 setNumColumnsRange(...)

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.

1.8.35 setRow(...)

Sets the contents of a table row.

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

1.8.36 setValue(...)

Sets the value of a table element.

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

1.8.37 setValueForBlank(...)

Sets the element value substituted for blank for the column.

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

1.8.38 setValues(...)

Sets all values for the table elements.

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

1.8.39 setValuesForBlanks(...)

Sets the values substituted for blanks for all table columns.

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

1.8.40 setValueToDefault(...)

Sets the keyword value to its default.

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

1.8.41 setValueToPrevious()

Sets the keyword value to its previous value.
Implements AFXKeyword.

Arguments

None.

1.8.42 syncPreviousValue()

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

Arguments

None.

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.

1.9 AFXCreateSketchStep

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

1.9.1 AFXCreateSketchStep(...)

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.

1.9.2 onCancel()

Called when the step is cancelled.
Reimplemented from AFXStep.

Arguments

None.

1.9.3 onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.
Reimplemented from AFXStep.

Arguments

None.

1.9.4 onResume()

Called when the step is resumed.
Reimplemented from AFXStep.

Arguments

None.

1.9.5 onSuspend()

Called when the step is suspended.
Reimplemented from AFXStep.

Arguments

None.

1.10 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.

1.10.1 AFXDataDialog(...)

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.

1.10.2 AFXDataDialog(...)

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.

1.10.3 addTransition(...)

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.

Argument	Type	Default	Description
value	Int		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

1.10.4 addTransition(...)

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.

1.10.5 addTransition(...)

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.

1.10.6 addTransition(...)

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.

Argument	Type	Default	Description
sel	Int		Message selector.
ptr	String	None	Message data.

1.10.7 addTransition(...)

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.

1.10.8 addTransition(...)

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.

1.10.9 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.

Arguments

None.

1.10.10 getMode()

Returns the dialog box's host mode.

Arguments

None.

1.10.11 onKeywordError(...)

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.

1.10.12 onTableError(...)

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.

1.10.13 onTupleError(...)

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.

1.10.14 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.

Arguments

None.

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.

1.11 AFXDialog

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

1.11.1 AFXDialog(...)

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.

1.11.2 AFXDialog(...)

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.

1.11.3 AFXDialog(...)

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.

1.11.4 appendActionButton(...)

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.

1.11.5 appendActionButton(...)

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

Argument	Type	Default	Description
buttonID	ButtonID		Button ID.

1.11.6 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.

Arguments

None.

1.11.7 create()

Creates the dialog box.

Reimplemented from FXTopWindow.

Arguments

None.

1.11.8 createButton(...)

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.

1.11.9 getActionButton(...)

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.

1.11.10 getInitialFocusWidget()

Returns the widget that will receive the initial focus.

Arguments

None.

1.11.11 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.

Arguments

None.

1.11.12 hide()

Unpost the dialog box.

Reimplemented from FXTopWindow.

Reimplemented in AFXManagerMenuDB, and AFXMessageDialog.

Arguments

None.

1.11.13 onKeywordError(...)

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.

1.11.14 onTableError(...)

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.

1.11.15 onTupleError(...)

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.

1.11.16 selectContents(...)

Selects the contents of the given widget.

Argument	Type	Default	Description
widget	FXWindow		Widget to select.

1.11.17 selectTableElement(...)

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.

1.11.18 selectTupleElement(...)

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.

1.11.19 setInitialFocusWidget(...)

Sets the widget that will receive the initial focus.

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

1.11.20 setOnPopdownTarget(...)

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.

1.11.21 setUnpostHints(...)

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

Argument	Type	Default	Description
hints	Int		

1.11.22 show()

Posts the dialog box.

Reimplemented from FXTopWindow.

Reimplemented in AFXFileDialog, and AFXMessageDialog.

Arguments

None.

1.11.23 showModal(...)

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.

1.12 AFXDialogStep

This class provides dialog steps in GUI procedures.

1.12.1 AFXDialogStep(...)

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		

1.12.2 AFXDialogStep(...)

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.

1.12.3 onCancel()

Called when the step is cancelled.
Reimplemented from AFXStep.

Arguments

None.

1.12.4 onExecute()

Called to execute steps returned by getFirstStep and getNextStep.
Reimplemented from AFXStep.

Arguments

None.

1.12.5 onResume()

Called when the step is resumed.
Reimplemented from AFXStep.

Arguments

None.

1.12.6 onSuspend()

Called when the step is suspended.
Reimplemented from AFXStep.

Arguments

None.

1.13 AFXEditSketchStep

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

1.13.1 AFXEditSketchStep(...)

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.

1.13.2 onCancel()

Called when the step is cancelled.
Reimplemented from AFXStep.

Arguments

None.

1.13.3 onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.
Reimplemented from AFXStep.

Arguments

None.

1.13.4 onResume()

Called when the step is resumed.
Reimplemented from AFXStep.

Arguments

None.

1.13.5 onSuspend()

Called when the step is suspended.
Reimplemented from AFXStep.

Arguments

None.

1.14 AFXFileDialog

This class contains a file selection dialog box.

1.14.1 AFXFileDialog(...)

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.

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.

1.14.2 AFXFileDialog(...)

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.

1.14.3 AFXFileDialog(...)

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.

1.14.4 AFXFileDialog(...)

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.

1.14.5 getCurrentPattern()

Returns the current pattern number.

Arguments

None.

1.14.6 getDirectory()

Returns the current directory.

Arguments

None.

1.14.7 getFileBoxStyle()

Return file list style.

Arguments

None.

1.14.8 getFilename()

Returns the file name.

Arguments

None.

1.14.9 getFilenames()

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

Arguments

None.

1.14.10 getItemSpace()

Returns the inter-item spacing (in pixels).

Arguments

None.

1.14.11 getPattern()

Returns the file pattern.

Arguments

None.

1.14.12 getPatternList()

Returns the list of patterns.

Arguments

None.

1.14.13 getPatternText(...)

Returns the pattern text for a given pattern number.

Argument	Type	Default	Description
patno	Int		

1.14.14 getPressedButtonId()

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

Arguments

None.

1.14.15 getReadOnly()

Returns the read-only state.

Arguments

None.

1.14.16 getReadOnlyPatterns()

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

Arguments

None.

1.14.17 getSelectMode()

Returns the file selection mode.

Arguments

None.

1.14.18 setCurrentPattern(...)

Sets the current active pattern.

Argument	Type	Default	Description
n	Int		

1.14.19 setDirectory(...)

Sets the current directory.

Argument	Type	Default	Description
path	String		

1.14.20 setFileBoxStyle(...)

Sets the file list style.

Argument	Type	Default	Description
style	Int		

1.14.21 setFilename(...)

Sets the file name.

Argument	Type	Default	Description
path	String		

1.14.22 setItemSpace(...)

Sets the inter-item spacing (in pixels).

Argument	Type	Default	Description
s	Int		

1.14.23 setPattern(...)

Sets the file pattern.

Argument	Type	Default	Description
ptrn	String		

1.14.24 setPatternList(...)

Sets the list of file patterns.

Argument	Type	Default	Description
patterns	String		

1.14.25 setPatternListMaxVisible(...)

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

Argument	Type	Default	Description
maxVisible	Int		

1.14.26 setPatternText(...)

Sets the pattern text for a pattern number.

Argument	Type	Default	Description
patno	Int		
text	String		

1.14.27 setReadOnly(...)

Sets the initial state of read-only button.

Argument	Type	Default	Description
state	Bool		

1.14.28 setReadOnlyPatterns(...)

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		

1.14.29 setSelectMode(...)

Sets the file selection mode.

Argument	Type	Default	Description
mode	Int		

1.14.30 show()

Posts the dialog box.

Reimplemented from AFXDialog.

Arguments

None.

1.14.31 showModal(...)

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).

1.14.32 shownReadOnly()

Returns True if the read-only button is shown.

Arguments

None.

1.14.33 showReadOnly(...)

Shows the read-only button.

Argument	Type	Default	Description
show	Bool		

1.15 AFXFileSelectorDialog

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

1.15.1 AFXFileSelectorDialog(...)

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.

1.15.2 AFXFileSelectorDialog(...)

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.

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

1.15.3 AFXFileSelectorDialog(...)

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.

1.15.4 AFXFileSelectorDialog(...)

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.

Argument	Type	Default	Description
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.

1.16 AFXFloatKeyword

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

1.16.1 AFXFloatKeyword(...)

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.

1.16.2 getPrecision()

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

Arguments

None.

1.16.3 getTypeName()

Returns the name of the keyword type.
Implements AFXKeyword.

Arguments

None.

1.16.4 getValue()

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

Arguments

None.

1.16.5 getValueAsString()

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

Arguments

None.

1.16.6 isValueChanged()

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

Arguments

None.

1.16.7 setDefaultValue(...)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	String		Default value.

1.16.8 setDefaultValue(...)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	Float		Default value.

1.16.9 setPrecision(...)

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

Argument	Type	Default	Description
precision	Int		

1.16.10 setValue(...)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	String		New value.

1.16.11 setValue(...)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	Float		New value.

1.16.12 setValueToDefault(...)

Sets the keyword value to its default.

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

1.16.13 setValueToPrevious()

Sets the keyword value to its previous value.

Implements AFXKeyword.

Arguments

None.

1.16.14 syncPreviousValue()

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

Arguments

None.

1.17 AFXFloatSpinner

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

1.17.1 AFXFloatSpinner(...)

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.

1.17.2 canFocus()

Returns True if the spinner can recieve focus.
Reimplemented from FXWindow.

Arguments

None.

1.17.3 create()

Creates the spinner.
Reimplemented from FXComposite.

Arguments

None.

1.17.4 disable()

Disables the spinner.
Reimplemented from FXWindow.

Arguments

None.

1.17.5 enable()

Enables the spinner.
Reimplemented from FXWindow.

Arguments

None.

1.17.6 getCheck()

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

Arguments

None.

1.17.7 getHelpText()

Returns the status line help text.

Arguments

None.

1.17.8 getIncrement()

Returns the spinner increment.

Arguments

None.

1.17.9 getLabelFont()

Returns the label font.

Arguments

None.

1.17.10 getLabelText()

Returns the label string.

Arguments

None.

1.17.11 getRange()

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

Arguments

None.

1.17.12 getTipText()

Returns the tool tip message.

Arguments

None.

1.17.13 getValue()

Returns the spinner value.

Arguments

None.

1.17.14 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.

Arguments

None.

1.17.15 isEditable()

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

Arguments

None.

1.17.16 isReadOnlyState()

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

Arguments

None.

1.17.17 setCheck(...)

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

Argument	Type	Default	Description
state	Bool		Button state.

1.17.18 setCheckButtonSelector(...)

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

Argument	Type	Default	Description
sel	Int		Selector.

1.17.19 setCheckButtonTarget(...)

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

Argument	Type	Default	Description
tgt	FXObject		Target.

1.17.20 setEditable(...)

Sets the editable state for the input field of spinner.

Argument	Type	Default	Description
edit	Bool	True	Editable state.

1.17.21 setHelpText(...)

Sets the status line help text.

Argument	Type	Default	Description
text	String		Help text.

1.17.22 setIncrement(...)

Sets the spinner increment.

Argument	Type	Default	Description
incr	Float		Increment.

1.17.23 setLabelFont(...)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		Label font.

1.17.24 setLabelText(...)

Sets the label string.

Argument	Type	Default	Description
txt	String		Label text.

1.17.25 setRange(...)

Sets the spinner range.

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

1.17.26 setReadOnlyState(...)

Sets the read-only state of the spinner.

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

1.17.27 setTipText(...)

Sets the tool tip message.

Argument	Type	Default	Description
text	String		Tooltip text.

1.17.28 setValue(...)

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.

1.18 AFXFloatTarget

This class is designed for floating-point targets.

1.18.1 AFXFloatTarget(...)

Constructor.

Argument	Type	Default	Description
initialValue	Float	0.0	Initial value.

1.18.2 getTypeName()

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

Arguments

None.

1.18.3 getValue()

Returns the target's current value.

Arguments

None.

1.18.4 setValue(...)

Sets the target's current value.

Argument	Type	Default	Description
newValue	Float		New value.

1.19 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.

1.19.1 AFXFlyoutButton(...)

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).

1.19.2 canFocus()

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

Arguments

None.

1.19.3 create()

Creates the flyout button.
Reimplemented from FXLabel.

Arguments

None.

1.19.4 detach()

Detaches server-side resources for the flyout button.
Reimplemented from FXLabel.

Arguments

None.

1.19.5 disable()

Disables the flyout button.
Reimplemented from FXLabel.

Arguments

None.

1.19.6 enable()

Enables the flyout button.
Reimplemented from FXLabel.

Arguments

None.

1.19.7 getButtonStyle()

Returns the flyout button style.

Arguments

None.

1.19.8 getCurrentItem()

Returns the current item.

Arguments

None.

1.19.9 getPane()

Returns the popup menu.

Arguments

None.

1.19.10 getState()

Returns the flyout button state.

Arguments

None.

1.19.11 setButtonStyle(...)

Sets the flyout button style.

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

1.19.12 setCurrentItem(...)

Sets the current item.

Argument	Type	Default	Description
item	AFXFlyoutItem		Item.

1.19.13 setCurrentItem(...)

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.

1.19.14 setPane(...)

Sets the popup menu.

Argument	Type	Default	Description
pup	FXPopup		Popup menu.

1.19.15 setState(...)

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.

1.20 AFXFlyoutItem

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

1.20.1 AFXFlyoutItem(...)

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).

1.20.2 canFocus()

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

Arguments

None.

1.20.3 hide()

Hides the flyout item.
Reimplemented from FXWindow.

Arguments

None.

1.20.4 setIcon(...)

Sets the icon for the flyout item.
Reimplemented from FXLabel.

Argument	Type	Default	Description
ic	FXIcon		

1.21 AFXForm

This class is the abstract base class for forms.

1.21.1 AFXForm(...)

Constructor.

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

1.21.2 activate()

Performs the necessary tasks when the form is activated.
Reimplemented from AFXGuiMode.

Arguments

None.

1.21.3 cancel(...)

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.

Argument	Type	Default	Description
msg	Int	0	Completion message ID.

1.21.4 commit()

Performs the necessary tasks when the form is committed.
Implements AFXGuiMode.

Arguments

None.

1.21.5 continueMode()

Used to get the next dialog box in the mode.
Implements AFXGuiMode.

Arguments

None.

1.21.6 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.

Arguments

None.

1.21.7 getFirstDialog()

Returns the first dialog box to be posted.

Arguments

None.

1.21.8 getNextDialog(...)

Returns the next dialog box to be posted.

Argument	Type	Default	Description
previousDialog	AFXDialog		Previous dialog box.

1.21.9 issueCommands(...)

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.

1.21.10 setModal(...)

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.

1.22 AFXGuiCommand

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

1.22.1 AFXGuiCommand(...)

Constructor.

Argument	Type	Default	Description
mode	AFXGuiMode		Host mode.
method	String		Method.
objectName	String	”	Object name.

Argument	Type	Default	Description
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.

1.22.2 getMode()

Retrieves the command's mode.

Arguments

None.

1.23 AFXGuiMode

This class is the abstract base class for modes.

1.23.1 AFXGuiMode(...)

Constructor.

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

1.23.2 activate()

Performs the necessary tasks when the mode is activated.
Reimplemented in AFXForm, and AFXProcedure.

Arguments

None.

1.23.3 cancel(...)

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.

1.23.4 commit()

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

Implemented in AFXForm, and AFXProcedure.

Arguments

None.

1.23.5 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.

Arguments

None.

1.23.6 deactivate()

Performs the necessary tasks when the mode is deactivated.

Reimplemented in AFXProcedureForm, and AFXProcedure.

Arguments

None.

1.23.7 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.

Arguments

None.

1.23.8 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.

Arguments

None.

1.23.9 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.

Arguments

None.

1.23.10 getCommandString()

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

Arguments

None.

1.23.11 getCurrentDialog()

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

Arguments

None.

1.23.12 getModeName()

Returns the mode's name.

Arguments

None.

1.23.13 getOwner()

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

Arguments

None.

1.23.14 getPressedButtonId()

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

Arguments

None.

1.23.15 handleException(...)

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.

1.23.16 handleKeywordException(...)

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.

1.23.17 isKeyword(...)

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

Argument	Type	Default	Description
object	FXObject		

1.23.18 issueCommands(...)

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.

1.23.19 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.

Arguments

None.

1.23.20 onKeywordsUpdated(...)

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.

1.23.21 registerDefaultsObject(...)

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.

1.23.22 registerQueries()

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

Arguments

None.

1.23.23 removeAllProxies()

Unregisters queries and deletes all the associated proxies.

Arguments

None.

1.23.24 sendCommandString(...)

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.

1.23.25 setModeName(...)

Sets the mode's name.

Argument	Type	Default	Description
name	String		

1.23.26 unregisterAllQueries()

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

Arguments

None.

1.23.27 verify()

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

Arguments

None.

1.23.28 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.

Arguments

None.

1.23.29 verifyKeywordValues()

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

Arguments

None.

Class flags

Message ID's.

ID_DESTROY_DIALOGS	Used to destroy dialogs.
ID_HANDLE_BAILOUT	Used to handle bailout.

1.24 AFXGuiObjectManager

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

1.24.1 AFXGuiObjectManager(...)

Constructor.

Argument	Type	Default	Description

1.24.2 AFXGuiObjectManager(...)

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

Argument	Type	Default	Description
source	AFXGuiObjectManager		Object to be copied.

1.24.3 getDialog(...)

Returns the dialog box with the specified widget key.

Argument	Type	Default	Description
widgetAlias	String		Dialog box alias.

1.24.4 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.

Arguments

None.

1.24.5 getToolbarGroup(...)

Returns the toolbar group specified by the given name argument.

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

1.24.6 hide(...)

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

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

1.24.7 registerAndShowDialog(...)

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.

1.24.8 registerAndShowModalDialog(...)

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.

1.24.9 registerShortcutFunction(...)

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.

Argument	Type	Default	Description
typesToDisplay	Int	0	Flags specifying the types of objects displayed in the module; see AFXModuleGui.

1.24.10 sendCommandString(...)

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.

1.24.11 show(...)

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

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

1.24.12 unregisterDialog(...)

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.

1.25 AFXIntKeyword

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

1.25.1 AFXIntKeyword(...)

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.

1.25.2 getTypeName()

Returns the name of the keyword type.

Implements AFXKeyword.

Reimplemented in AFXSymConstKeyword.

Arguments

None.

1.25.3 getValue()

Returns the keyword's current value.

Arguments

None.

1.25.4 getValueAsString()

Returns the text string that represents the keyword's current value.
Implements AFXKeyword.
Reimplemented in AFXSymConstKeyword.

Arguments

None.

1.25.5 isValueChanged()

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

Arguments

None.

1.25.6 setDefaultValue(...)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	Int		Default value.

1.25.7 setValue(...)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	Int		New value.

1.25.8 setValueToDefault(...)

Sets the keyword value to its default.

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

1.25.9 setValueToPrevious()

Sets the keyword value to its previous value.
Implements AFXKeyword.

Arguments

None.

1.25.10 syncPreviousValue()

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

Arguments

None.

1.26 AFXIntTarget

This class is designed for integer targets.

1.26.1 AFXIntTarget(...)

Constructor.

Argument	Type	Default	Description
initialValue	Int	0	Initial value.

1.26.2 getTypeName()

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

Arguments

None.

1.26.3 getValue()

Returns the target's current value.

Arguments

None.

1.26.4 setValue(...)

Sets the target's current value.

Argument	Type	Default	Description
newValue	Int		New value.

1.27 AFXItemProvider

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

1.27.1 AFXItemProvider(...)

Constructor.

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

1.27.2 append(...)

Appends a string to the item string.

Argument	Type	Default	Description
str	String		

1.27.3 append(...)

Appends a character to the item string.

Argument	Type	Default	Description
ch	String		

1.27.4 empty()

Returns True if the item string is empty.

Arguments

None.

1.27.5 getItems()

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

Arguments

None.

1.27.6 getVersion()

Returns the version of provider's items.

Arguments

None.

1.27.7 reset(...)

Clears the contents of the item string and reallocates space.

Argument	Type	Default	Description
sz	Int	0	

1.27.8 setItems(...)

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

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

1.28 AFXKeyword

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

1.28.1 AFXKeyword(...)

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.

1.28.2 activate()

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

Arguments

None.

1.28.3 deactivate()

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

Arguments

None.

1.28.4 getCommandSnippet()

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

Arguments

None.

1.28.5 getName()

Returns the keyword name.

Arguments

None.

1.28.6 getSetupCommands()

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

Arguments

None.

1.28.7 getTypeName()

Returns the keyword type name.

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

Arguments

None.

1.28.8 getValueAsString()

Returns the text string that represents the current keyword value.

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

Arguments

None.

1.28.9 isActive()

Returns True if the keyword is active.

Arguments

None.

1.28.10 isRequired()

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

Arguments

None.

1.28.11 isValueChanged()

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

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

Arguments

None.

1.28.12 setRequired(...)

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

Argument	Type	Default	Description
val	Bool		

1.28.13 setSetupCommands(...)

Sets variable initialization commands needed by the keyword.

Argument	Type	Default	Description
cmds	String		

1.28.14 setValueToDefault(...)

Sets the keyword value to its default.

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

1.28.15 setValueToPrevious()

Sets the keyword value to its previous value.

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

Arguments

None.

1.28.16 syncPreviousValue()

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

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

Arguments

None.

Class flags

Message ID's.

ID_ACTIVATE	Used to activate the keyword.
ID_DEACTIVATE	Used to deactivate the keyword.

1.29 AFXList

This class is a list widget that allows displaying items in a scrollable window.

1.29.1 AFXList(...)

Constructor.

Argument	Type	Default	Description
p	FXComposite		Parent widget.
nvis	Int		Number of visible items.
tgt	FXObject	None	Message target.

Argument	Type	Default	Description
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.

1.29.2 appendItem(...)

Adds a new item to the end of the list.

Argument	Type	Default	Description
text	String		
icon	FXIcon	None	
sel	Int	0	

1.29.3 disable()

Disables the list.
Reimplemented from FXWindow.

Arguments

None.

1.29.4 enable()

Enables the list.
Reimplemented from FXWindow.

Arguments

None.

1.29.5 getAutoCommit()

Returns the auto-commit flag.

Arguments

None.

1.29.6 getDefaultHeight()

Returns the default height of the list.
Reimplemented from FXList.

Arguments

None.

1.29.7 getItemIndexForData(...)

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

Argument	Type	Default	Description
data			

1.29.8 getItemProvider()

Returns the provider of the list's items.

Arguments

None.

1.29.9 getSingleSelection()

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

Arguments

None.

1.29.10 insertItem(...)

Inserts a new item at the given index.

Argument	Type	Default	Description
index	Int		
text	String		
icon	FXIcon	None	
sel	Int	0	

1.29.11 replaceItem(...)

Replaces the item at the given index.

Argument	Type	Default	Description
index	Int		
text	String		
icon	FXIcon	None	
sel	Int	0	

1.29.12 setAutoCommit(...)

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

Argument	Type	Default	Description
commit	Bool		

1.29.13 setItemProvider(...)

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.

1.30 AFXListBox

This class contains a label that precedes a list box, which allows the user to select entries from a drop-down list.

1.30.1 AFXListBox(...)

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).

1.30.2 appendItem(...)

Adds an item to the end of the list.

Argument	Type	Default	Description
text	String		
icon	FXIcon	None	
sel	Int	0	

1.30.3 clearItems()

Removes all items from the list.

Arguments

None.

1.30.4 create()

Creates the list box.

Reimplemented from FXComposite.

Arguments

None.

1.30.5 disable()

Disables the list box.
Reimplemented from FXWindow.

Arguments

None.

1.30.6 enable()

Enables the list box.
Reimplemented from FXWindow.

Arguments

None.

1.30.7 getCurrentItem()

Returns the index of the current item.

Arguments

None.

1.30.8 getHelpText()

Returns the status line help text.

Arguments

None.

1.30.9 getItemData(...)

Returns the data for the specified item.

Argument	Type	Default	Description
index	Int		

1.30.10 getItemIndexForData(...)

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

Argument	Type	Default	Description
data			

1.30.11 getLabelFont()

Returns the label font.

Arguments

None.

1.30.12 getLabelText()

Returns the label string.

Arguments

None.

1.30.13 getNumItems()

Returns the number of items in the list.

Arguments

None.

1.30.14 getNumVisible()

Returns the number of visible items.

Arguments

None.

1.30.15 getTipText()

Returns the tool tip message.

Arguments

None.

1.30.16 insertItem(...)

Inserts a new item at the specified index position.

Argument	Type	Default	Description
index	Int		
text	String		
icon	FXIcon	None	
sel	Int	0	

1.30.17 isItemCurrent(...)

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

Argument	Type	Default	Description
index	Int		

1.30.18 isReadOnlyState()

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

Arguments

None.

1.30.19 removeItem(...)

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

Argument	Type	Default	Description
index	Int		

1.30.20 replaceItem(...)

Replaces the item at the specified index position.

Argument	Type	Default	Description
index	Int		
text	String		
icon	FXIcon	None	

Argument	Type	Default	Description
sel	Int	0	

1.30.21 setCurrentItem(...)

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

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.30.22 setHelpText(...)

Sets the status line help text.

Argument	Type	Default	Description
text	String		

1.30.23 setItemData(...)

Sets the data for the specified item.

Argument	Type	Default	Description
index	Int		
ptr	String		

1.30.24 setLabelFont(...)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		

1.30.25 setLabelText(...)

Sets the label string.

Argument	Type	Default	Description
txt	String		

1.30.26 setNumVisible(...)

Sets the number of visible items.

Argument	Type	Default	Description
nvis	Int		

1.30.27 setReadOnlyState(...)

Sets the read-only state of the list box.

Argument	Type	Default	Description
readonly	Bool	True	

1.30.28 setTipText(...)

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.

1.31 AFXMainWindow

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

1.31.1 AFXMainWindow(...)

Constructor.

Argument	Type	Default	Description
app	FXApp		FXApp object.
title	String		Main window title.

Argument	Type	Default	Description
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.

1.31.2 appendApplicableModuleForTreeTab(...)

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.
displayedName	String		Module name to be appended to the list of the tab's applicable modules.

1.31.3 appendTreeTab(...)

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.

1.31.4 appendVisibleModuleForTreeTab(...)

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.

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

1.31.5 create()

Virtual base class method for creating windowing system resources.
Reimplemented from FXTopWindow.

Arguments

None.

1.31.6 getContextBar()

Returns a pointer to the context bar container.

Arguments

None.

1.31.7 getCurrentTreeTab()

Returns the current tab item.

Arguments

None.

1.31.8 getDefaultHeight()

Returns the default main window height.
Reimplemented from FXTopWindow.

Arguments

None.

1.31.9 getDefaultWidth()

Returns the default main window width.
Reimplemented from FXTopWindow.

Arguments

None.

1.31.10 getDisplayedNameAtIndex(...)

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

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

1.31.11 getDrawingAreaHeight()

Returns the height of the drawing area in pixels.

Arguments

None.

1.31.12 getDrawingAreaWidth()

Returns the width of the drawing area in pixels.

Arguments

None.

1.31.13 getHelpToolset()

Returns a pointer to the help toolset.

Arguments

None.

1.31.14 getMenubar()

Returns a pointer to the menubar.

Arguments

None.

1.31.15 getModule(...)

Returns the module specified by the given name argument.

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

1.31.16 getModuleName(...)

Returns the module name for the given displayed name.

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

1.31.17 getNumModules()

Returns the number of modules.

Arguments

None.

1.31.18 getPluginToolset()

Returns the Plugin toolset.

Arguments

None.

1.31.19 getSelectorFromFunction(...)

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.

1.31.20 getTargetFromFunction(...)

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.

1.31.21 getToolbox()

Returns a pointer to the module toolbox container.

Arguments

None.

1.31.22 getToolMenuPane()

Returns a pointer to the tools menu pane.

Arguments

None.

1.31.23 getToolMenuTitle()

Returns a pointer to the Tools menu title.

Arguments

None.

1.31.24 getToolset(...)

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.

1.31.25 getToolsetKernelInitializationCommands()

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

Arguments

None.

1.31.26 getWorkDirectory()

Returns the current working directory.

Arguments

None.

1.31.27 hideCli()

Hides the command line interface.

Arguments

None.

1.31.28 hideMessageArea()

Hides the message area interface.

Arguments

None.

1.31.29 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.

Arguments

None.

1.31.30 registerHelpToolset(...)

Registers the Help toolset.

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

1.31.31 registerModule(...)

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

Argument	Type	Default	Description
displayedName	String		English Name of the module.
moduleName	String		Python module name.

1.31.32 registerModule(...)

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		English Name of the module.
moduleName	String		Python module name.
kernelInitializationCommand	String		Python command sent to kernel on module load

1.31.33 registerToolset(...)

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.

1.31.34 setApplicabilityForTreeTab(...)

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.

1.31.35 setCurrentTreeTab(...)

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.

1.31.36 setVisibilityForTreeTab(...)

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.

1.31.37 setWorkDirectory(...)

Sets the current working directory.

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

1.31.38 showCli()

Shows the command line interface.

Arguments

None.

1.31.39 showMessageArea()

Shows the message area interface.

Arguments

None.

1.31.40 writeToMessageArea(...)

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

1.32 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.

1.32.1 AFXMenuCascade(...)

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.

1.32.2 getOwner()

Returns the owner of the menu cascade.
Reimplemented from FXWindow.

Arguments

None.

1.33 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.

1.33.1 AFXMenuCommand(...)

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.

1.33.2 getOwner()

Returns the owner of the menu command.
Reimplemented from FXWindow.

Arguments

None.

1.34 AFXMenuPane

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

1.34.1 AFXMenuPane(...)

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Manager of the pane.

1.34.2 getOwner()

Returns the owner of the menu pane.
Reimplemented from FXWindow.

Arguments

None.

1.35 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.

1.35.1 AFXMenuTitle(...)

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.

1.35.2 AFXMenuTitle(...)

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.

1.35.3 getOwner()

Returns the owner of the menu title.
Reimplemented from FXWindow.

Arguments

None.

1.35.4 hide()

Hides the widget.
Reimplemented from FXWindow.

Arguments

None.

1.35.5 show()

Shows the widget.
Reimplemented from FXWindow.

Arguments

None.

1.36 AFXMode

This class is the base class for modes.

1.36.1 AFXMode(...)

Constructor.

Argument	Type	Default	Description

1.36.2 getCommand(...)

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).

1.36.3 getNumCommands()

Returns the number of commands associated with the mode.

Arguments

None.

1.36.4 isActive()

Returns True if the mode is active.

Arguments

None.

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.

1.37 AFXModuleGui

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

1.37.1 AFXModuleGui(...)

Constructor.

Argument	Type	Default	Description
moduleName	String		Module name passed in from derived modules.
displayTypes	Int		Types of primary objects that this module may display.

1.37.2 activate()

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

Arguments

None.

1.37.3 deactivate()

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

Arguments

None.

1.37.4 getModuleName()

Returns the name of the module given on construction.

Arguments

None.

1.37.5 getToolsetKernelInitializationCommands()

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

Arguments

None.

1.37.6 getTypesToDisplay()

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

Arguments

None.

1.37.7 hide(...)

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.

1.37.8 registerProcedureToolset(...)

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

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

1.37.9 registerToolset(...)

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.

1.37.10 show(...)

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.

1.37.11 unregisterToolset(...)

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.

1.38 AFXNote

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

1.38.1 AFXNote(...)

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.

1.38.2 create()

Creates the note.
Reimplemented from FXComposite.

Arguments

None.

1.38.3 detach()

Detaches the server-resources of the note.
Reimplemented from FXComposite.

Arguments

None.

1.38.4 disable()

Disables the note.
Reimplemented from FXWindow.

Arguments

None.

1.38.5 enable()

Enables the note.
Reimplemented from FXWindow.

Arguments

None.

1.38.6 getText()

Returns the note's message string.

Arguments

None.

1.38.7 setText(...)

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.

1.39 AFXObjectKeyword

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

1.39.1 AFXObjectKeyword(...)

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.

1.39.2 getTypeName()

Returns the name of the keyword type.
Implements AFXKeyword.

Arguments

None.

1.39.3 getValue()

Returns the keyword's current value.

Arguments

None.

1.39.4 getValueAsString()

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

Arguments

None.

1.39.5 isValueChanged()

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

Arguments

None.

1.39.6 setDefaultValue(...)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	String		Default value.

1.39.7 setValue(...)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	String		New value.

1.39.8 setValueToDefault(...)

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.

1.39.9 setValueToPrevious()

Sets the keyword value to its previous value.
Implements AFXKeyword.

Arguments

None.

1.39.10 syncPreviousValue()

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

Arguments

None.

1.40 AFXOptionTreeItem

This class is a tree widget with check buttons.

1.40.1 AFXOptionTreeItem(...)

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.

Argument	Type	Default	Description
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.

1.40.2 addItemAfter(...)

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.

1.40.3 addItemBefore(...)

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.

1.40.4 addItemFirst(...)

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.

1.40.5 addItemLast(...)

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.

1.40.6 childAtIndex(...)

Returns the child tree at the given index.

Argument	Type	Default	Description
index	Int		Index.

1.40.7 collapse()

Collapses (hides) the children.

Arguments

None.

1.40.8 computeDefaultArrowSize()

Computes default size of the arrow button.

Arguments

None.

1.40.9 containsChild(...)

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

Argument	Type	Default	Description
tree	AFXOptionTreeItem		Item.

1.40.10 create()

Creates the tree item.

Reimplemented from FXComposite.

Arguments

None.

1.40.11 disable()

Disables the tree item.
Reimplemented from FXWindow.

Arguments

None.

1.40.12 enable()

Enables the tree item.
Reimplemented from FXWindow.

Arguments

None.

1.40.13 expand()

Expands (shows) the children.

Arguments

None.

1.40.14 getArrowSize()

Returns the size of the arrow button.

Arguments

None.

1.40.15 getCheck()

Returns the check state of the tree item.

Arguments

None.

1.40.16 getDefaultWidth()

Returns the default width of the tree item.
Reimplemented from FXPacker.

Arguments

None.

1.40.17 getFirst()

Returns the first child tree.
Reimplemented from FXWindow.

Arguments

None.

1.40.18 getLast()

Returns the last child tree.
Reimplemented from FXWindow.

Arguments

None.

1.40.19 getNext()

Returns the next sibling tree.
Reimplemented from FXWindow.

Arguments

None.

1.40.20 getParent()

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

Arguments

None.

1.40.21 getPrev()

Returns the previous sibling tree.
Reimplemented from FXWindow.

Arguments

None.

1.40.22 getText()

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

Arguments

None.

1.40.23 hasVisibleChildren()

Checks whether the tree item has any visible children.

Arguments

None.

1.40.24 hide()

Hides the tree item.
Reimplemented from FXWindow.

Arguments

None.

1.40.25 indexOfChild(...)

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

Argument	Type	Default	Description
tree	AFXOptionTreeItem		Item.

1.40.26 isChildOf(...)

Checks whether this object is contained in the given tree.

Argument	Type	Default	Description
tree	AFXOptionTreeItem		Item.

1.40.27 isExpanded()

Checks whether the tree item shows its children.

Arguments

None.

1.40.28 numChildren()

Returns the number of child trees.

Reimplemented from FXWindow.

Arguments

None.

1.40.29 setArrowSize(...)

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

Argument	Type	Default	Description
size	Int		Size.

1.40.30 setCheck(...)

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.

1.40.31 setCheck(...)

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

Argument	Type	Default	Description
check	Int	True	Check state.

1.40.32 setText(...)

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

Argument	Type	Default	Description
txt	String		Label text.

1.40.33 show()

Shows the tree item.
Reimplemented from FXWindow.

Arguments

None.

1.40.34 updateCheck(...)

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.

1.41 AFXOptionTreeList

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

1.41.1 AFXOptionTreeList(...)

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.

1.41.2 addItemFirst(...)

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.

1.41.3 addItemLast(...)

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.

1.41.4 clearItems()

Removes all items from the list.

Arguments

None.

1.41.5 computeItemHeight(...)

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

Argument	Type	Default	Description
p	AFXOptionTreeItem	None	Item.

1.41.6 createItem(...)

Creates a new tree item object.

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

1.41.7 getContentHeight()

Returns the content height.

Reimplemented from FXScrollWindow.

Arguments

None.

1.41.8 getContents()

Returns the content window.

Arguments

None.

1.41.9 getContentWidth()

Returns the content width.

Reimplemented from FXScrollWindow.

Arguments

None.

1.41.10 getDefaultHeight()

Returns the default height.
Reimplemented from FXScrollArea.

Arguments

None.

1.41.11 getDefaultWidth()

Returns the default width.
Reimplemented from FXScrollArea.

Arguments

None.

1.41.12 getFirstItem()

Returns the first root item.

Arguments

None.

1.41.13 getHSpacing()

Returns the horizontal inter-child spacing.

Arguments

None.

1.41.14 getLastItem()

Returns the last root item.

Arguments

None.

1.41.15 getNumItems()

Returns the number of top-level items.

Arguments

None.

1.41.16 getNumVisible()

Returns the number of visible items.

Arguments

None.

1.41.17 getPadBottom()

Returns the bottom padding.

Arguments

None.

1.41.18 getPadLeft()

Returns the left padding.

Arguments

None.

1.41.19 getPadRight()

Returns the right padding.

Arguments

None.

1.41.20 getPadTop()

Returns the top padding.

Arguments

None.

1.41.21 getVSpacing()

Returns the vertical inter-child spacing.

Arguments

None.

1.41.22 layout()

Recalculates layout.
Reimplemented from FXScrollWindow.

Arguments

None.

1.41.23 moveContents(...)

Moves contents to the specified position.

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

1.41.24 removeItem(...)

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.

1.41.25 setHSpacing(...)

Sets the horizontal inter-child spacing.

Argument	Type	Default	Description
hs	Int		Horizontal spacing.

1.41.26 setNumVisible(...)

Sets the number of visible items.

Argument	Type	Default	Description
nvis	Int		Number of visible items.

1.41.27 setPadBottom(...)

Sets the bottom padding.

Argument	Type	Default	Description
pb	Int		Bottom padding.

1.41.28 setPadLeft(...)

Sets the left padding.

Argument	Type	Default	Description
pl	Int		Left padding.

1.41.29 setPadRight(...)

Sets the right padding.

Argument	Type	Default	Description
pr	Int		Right padding.

1.41.30 setPadTop(...)

Sets the top padding.

Argument	Type	Default	Description
pt	Int		Top padding.

1.41.31 setVSpacing(...)

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.

1.42 AFXOrderedPickStep

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

1.42.1 AFXOrderedPickStep(...)

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.

1.42.2 onCancel()

Called when the step is cancelled.
Reimplemented from AFXPickStep.

Arguments

None.

1.42.3 onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.
Reimplemented from AFXPickStep.

Arguments

None.

1.42.4 reset()

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

Arguments

None.

1.43 AFXPickStep

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

1.43.1 AFXPickStep(...)

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.

1.43.2 addElementSetSelection(...)

Adds an element set to the step's selections.

Argument	Type	Default	Description
name	String		Name of set.

1.43.3 addGeometrySetSelection(...)

Adds a geometry set to the step's selections.

Argument	Type	Default	Description
name	String		Name of set.

1.43.4 addNodeSetSelection(...)

Adds a node set to the step's selections.

Argument	Type	Default	Description
name	String		Name of set.

1.43.5 addPointKeyIn(...)

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

Argument	Type	Default	Description
keyword	AFXTupleKeyword		Keyword

1.43.6 addSurfaceSelection(...)

Adds a surface to the step's selections.

Argument	Type	Default	Description
name	String		Name of surface.

1.43.7 allowRepeatedSelections(...)

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.

1.43.8 onCancel()

Called when the step is cancelled.

Reimplemented from AFXStep.

Reimplemented in AFXOrderedPickStep.

Arguments

None.

1.43.9 onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.
Reimplemented from AFXStep.
Reimplemented in AFXOrderedPickStep.

Arguments

None.

1.43.10 onResume()

Called when the step is resumed.
Reimplemented from AFXStep.

Arguments

None.

1.43.11 onSuspend()

Called when the step is suspended.
Reimplemented from AFXStep.

Arguments

None.

1.43.12 reset()

Allows a step to reset any of its data (if needed) when looping.
Reimplemented from AFXStep.
Reimplemented in AFXOrderedPickStep.

Arguments

None.

1.43.13 setEdgeRefinements(...)

Sets the refinements to be used when picking edges.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

1.43.14 setElementEdgeRefinements(...)

Sets the refinements to be used when picking element edges.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

1.43.15 setElementFaceRefinements(...)

Sets the refinements to be used when picking element faces.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

1.43.16 setElementRefinements(...)

Sets the refinements to be used when picking elements.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

1.43.17 setFaceRefinements(...)

Sets the refinements to be used when picking faces.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

1.43.18 setInstanceRefinements(...)

Sets the refinements to be used when picking instances.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

1.43.19 setNodeRefinements(...)

Sets the refinements to be used when picking nodes.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

1.43.20 setSketchRefinements(...)

Sets the refinements to be used when picking sketches.

Argument	Type	Default	Description
refinement	Int		Refinement flag.

1.43.21 setXyRefinements(...)

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.

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.

1.44 AFXProcedure

This class provides the basis for writing procedures.

1.44.1 AFXProcedure(...)

Constructor.

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

1.44.2 activate()

Activates the mode.

Reimplemented from AFXGuiMode.

Arguments

None.

1.44.3 cancel(...)

Tries to cancel the procedure depending on checkCancel results.

Argument	Type	Default	Description
tgt	FXObject	None	Completion message target.

Argument	Type	Default	Description
msg	Int	0	Completion message ID.

1.44.4 checkBackup()

Returns 1 if ok to backup else returns 0.

Arguments

None.

1.44.5 checkCancel()

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

Arguments

None.

1.44.6 commit()

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

Arguments

None.

1.44.7 continueMode()

Used to get the next step in the mode.
Implements AFXGuiMode.

Arguments

None.

1.44.8 deactivate()

deactivates the mode.
Reimplemented from AFXGuiMode.

Arguments

None.

1.44.9 getCurrentStep()

Returns the current step.

Arguments

None.

1.44.10 getFirstStep()

Returns the first step to be executed in the procedure.

Arguments

None.

1.44.11 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.

Arguments

None.

1.44.12 getNextStep(...)

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.

1.44.13 getNumSteps()

Returns the number of steps in the step stack.

Arguments

None.

1.44.14 handleException(...)

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

Argument	Type	Default	Description
exc	nex_Exception		Exception.

1.44.15 onBackup()

Called when a procedure backs up a step.

Arguments

None.

1.44.16 onCancel()

Called when a procedure cancels.

Arguments

None.

1.44.17 onCmdBackup(...)

Message handler for handling backup button activation.

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

1.44.18 onCmdHandle2BtnBailout(...)

Message handler for handling the user 2 button bailout choice.

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

1.44.19 onCmdHandleBailout(...)

Message handler for handling the user 3 button bailout choice.

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

1.44.20 onResume()

Called when a procedure resumes.

Arguments

None.

1.44.21 onSuspend()

Called when a procedure suspends.

Arguments

None.

1.44.22 onValueChanged()

Called when a procedure's step changes in value.

Arguments

None.

1.44.23 setCurrentDb(...)

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

Argument	Type	Default	Description
db	AFXDialog		Dialog box.

1.44.24 setSelectionOptions(...)

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.

1.44.25 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.

Arguments

None.

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.

1.45 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.

1.45.1 AFXProcedureToolsetGui(...)

Deserialization.

Argument	Type	Default	Description

1.45.2 AFXProcedureToolsetGui(...)

Constructor.

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

1.45.3 swapInToolsetItems()

Swaps in the toolset's GUI items.

Arguments

None.

1.45.4 swapOutToolsetItems()

Swaps out the toolset's GUI items.

Arguments

None.

Class flags

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

1.46 AFXProgressBar

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

1.46.1 AFXProgressBar(...)

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).

1.46.2 create()

Creates the progress bar.
Reimplemented from FXProgressBar.

Arguments

None.

1.46.3 getBarStyle()

Returns the progress bar style.
Reimplemented from FXProgressBar.

Arguments

None.

1.46.4 getDefaultHeight()

Returns the default height.
Reimplemented from FXProgressBar.

Arguments

None.

1.46.5 getDefaultWidth()

Returns the default width.
Reimplemented from FXProgressBar.

Arguments

None.

1.46.6 getNumCursorBoxes()

Returns the number of cursor boxes displayed.

Arguments

None.

1.46.7 getProgress()

Returns the current progress amount.
Reimplemented from FXProgressBar.

Arguments

None.

1.46.8 getTotal()

Returns the total progress amount.
Reimplemented from FXProgressBar.

Arguments

None.

1.46.9 hide()

Hides the progress bar.
Reimplemented from FXWindow.

Arguments

None.

1.46.10 hideNumber()

Hides the progress bar iteration or percentage text.
Reimplemented from FXProgressBar.

Arguments

None.

1.46.11 setBarStyle(...)

Sets the progress bar style.

Argument	Type	Default	Description
style	Int		Style flag.

1.46.12 setNumCursorBoxes(...)

Sets the number of cursor boxes to display.

Argument	Type	Default	Description
nb	Int		Number of boxes.

1.46.13 setProgress(...)

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		

1.46.14 setTotal(...)

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		

1.46.15 show()

Shows the progress bar.

Reimplemented from FXWindow.

Arguments

None.

1.46.16 showNumber(...)

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.

1.47 AFXSequenceString

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

1.47.1 AFXSequenceString(...)

Constructor.

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

1.47.2 AFXSequenceString(...)

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

Argument	Type	Default	Description

1.47.3 forceNumElements(...)

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.

1.47.4 getContentString()

Returns a string containing values of the sequence elements.
Reimplemented in AFX2DArrayConstString.

Arguments

None.

1.47.5 getElementSeparator()

Returns the element separator character.

Arguments

None.

1.47.6 getLength(...)

Returns the length in characters of a sequence element.

Argument	Type	Default	Description
index	Int		Element index.

1.47.7 getNumElements()

Returns the number of elements in this sequence.

Arguments

None.

1.47.8 getPosition(...)

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

Argument	Type	Default	Description
index	Int		Element index.

1.47.9 getValue(...)

Returns the value of a sequence element.

Argument	Type	Default	Description
index	Int		Element index.

1.47.10 insert(...)

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.

1.47.11 isValidSequence()

Returns True if this object contains a valid sequence.

Arguments

None.

1.47.12 remove(...)

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.

1.47.13 setContentString(...)

Resets all values for the sequence elements.

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

1.47.14 setElementSeparator(...)

Sets the element separator character.

Argument	Type	Default	Description
sep	String		Separator character.

1.47.15 setLength(...)

Sets the length of the sequence element.

Argument	Type	Default	Description
index	Int		Element index.
length	Int		New length (in characters).

1.47.16 setPosition(...)

Sets the position of the sequence element.

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

1.47.17 setValue(...)

Sets the value of a sequence element.

Argument	Type	Default	Description
index	Int		Element index.
value	String		New value.

Argument	Type	Default	Description
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.

1.47.18 trimWhiteSpace(...)

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

Argument	Type	Default	Description
index	Int		Element index.

1.48 AFXSlider

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

1.48.1 AFXSlider(...)

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).

Argument	Type	Default	Description
pb	Int	0	Bottom padding (margin).

1.48.2 canFocus()

Returns True because a slider can receive focus.
Reimplemented from FXWindow.

Arguments

None.

1.48.3 disable()

Disables the slider.
Reimplemented from FXWindow.

Arguments

None.

1.48.4 enable()

Enables the slider.
Reimplemented from FXWindow.

Arguments

None.

1.48.5 getDecimalPlaces()

Returns the number of decimal points displayed.

Arguments

None.

1.48.6 getDefaultHeight()

Returns the default height.
Reimplemented from FXPacker.

Arguments

None.

1.48.7 getDefaultWidth()

Returns the default width.
Reimplemented from FXPacker.

Arguments

None.

1.48.8 getIncrement()

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

Arguments

None.

1.48.9 getMaxLabelText()

Returns the maximum label's text.

Arguments

None.

1.48.10 getMinLabelText()

Returns the minimum label's text.

Arguments

None.

1.48.11 getRange()

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

Arguments

None.

1.48.12 getSliderStyle()

Returns the slider's style.

Arguments

None.

1.48.13 getTipText()

Returns the slider's tip text.

Arguments

None.

1.48.14 getTitleLabelJustify()

Returns the title label's justification mode.

Arguments

None.

1.48.15 getTitleLabelText()

Returns the title label's text.

Arguments

None.

1.48.16 getValue()

Returns the slider's value.

Arguments

None.

1.48.17 recalc()

Recalculates the slider. Redefined to handle slider movement.
Reimplemented from FXWindow.

Arguments

None.

1.48.18 setDecimalPlaces(...)

Sets the number of decimal points displayed.

Argument	Type	Default	Description
dp	Int		Number of decimal places.

1.48.19 setIncrement(...)

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

Argument	Type	Default	Description
inc	Int		Increment.

1.48.20 setMaxLabelText(...)

Sets the maximum label's text.

Argument	Type	Default	Description
text	String		Max label text.

1.48.21 setMinLabelText(...)

Sets the minimum label's text.

Argument	Type	Default	Description
text	String		Min label text.

1.48.22 setRange(...)

Sets the slider's maximum and minimum values.

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

1.48.23 setSliderStyle(...)

Sets the slider's style.

Argument	Type	Default	Description
style	Int		Style flag.

1.48.24 setTipText(...)

Sets the slider's tip text.

Argument	Type	Default	Description
text	String		Tip text.

1.48.25 setTitleLabelJustify(...)

Sets the title label's justification mode.

Argument	Type	Default	Description
mode	Int		Justification mode.

1.48.26 setTitleLabelText(...)

Sets the title label's text.

Argument	Type	Default	Description
text	String		Title text.

1.48.27 setValue(...)

Sets the slider's value.

Argument	Type	Default	Description
value	Int		Value.

1.48.28 show()

Shows the slider.

Reimplemented from FXWindow.

Arguments

None.

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.

1.49 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.

1.49.1 AFXSpinner(...)

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

Argument	Type	Default	Description
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).

1.49.2 create()

Creates the spinner.
Reimplemented from FXComposite.

Arguments

None.

1.49.3 disable()

Disables the spinner.
Reimplemented from FXWindow.

Arguments

None.

1.49.4 enable()

Enables the spinner.
Reimplemented from FXWindow.

Arguments

None.

1.49.5 getCheck()

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

Arguments

None.

1.49.6 getHelpText()

Returns the status line help text.

Arguments

None.

1.49.7 getIncrement()

Returns the spinner increment.

Arguments

None.

1.49.8 getLabelFont()

Returns the label font.

Arguments

None.

1.49.9 getLabelText()

Returns the label string.

Arguments

None.

1.49.10 getRange()

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

Arguments

None.

1.49.11 getTipText()

Returns the tool tip message.

Arguments

None.

1.49.12 getValue()

Returns the spinner value.

Arguments

None.

1.49.13 isEditable()

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

Arguments

None.

1.49.14 isReadOnlyState()

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

Arguments

None.

1.49.15 setCheck(...)

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

Argument	Type	Default	Description
state	Bool		State.

1.49.16 setCheckButtonSelector(...)

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

Argument	Type	Default	Description
sel	Int		Selector.

1.49.17 setCheckButtonTarget(...)

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

Argument	Type	Default	Description
tgt	FXObject		Target.

1.49.18 setEditable(...)

Sets the editable state for the input field.

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

1.49.19 setHelpText(...)

Sets the status line help text.

Argument	Type	Default	Description
text	String		Help text.

1.49.20 setIncrement(...)

Sets the spinner increment.

Argument	Type	Default	Description
incr	Int		Increment.

1.49.21 setLabelFont(...)

Sets the label font.

Argument	Type	Default	Description
fnt	FXFont		Label font.

1.49.22 setLabelText(...)

Sets the label string.

Argument	Type	Default	Description
txt	String		Label text.

1.49.23 setRange(...)

Sets the spinner range.

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

1.49.24 setReadOnlyState(...)

Sets the read-only state of the spinner.

Argument	Type	Default	Description
readonly	Bool	True	State.

1.49.25 setTipText(...)

Sets the tool tip message.

Argument	Type	Default	Description
text	String		Tooltip text.

1.49.26 setValue(...)

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.

1.50 AFXStep

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

1.50.1 AFXStep(...)

Constructor.

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

1.50.2 onCancel()

Called when the step is cancelled.

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

Arguments

None.

1.50.3 onDone()

Called when the step completes.

Arguments

None.

1.50.4 onExecute()

Called to execute the steps returned by getFirstStep and getNextStep.

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

Arguments

None.

1.50.5 onResume()

Called when the step is resumed.

Reimplemented in AFXCreateSketchStep, AFXDialogStep, AFXEditSketchStep, and AFXPickStep.

Arguments

None.

1.50.6 onSuspend()

Called when the step is suspended.

Reimplemented in AFXCreateSketchStep, AFXDialogStep, AFXEditSketchStep, and AFXPickStep.

Arguments

None.

1.50.7 onValueChanged()

Called when the step's value changes.

Arguments

None.

1.50.8 reset()

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

Reimplemented in AFXOrderedPickStep, and AFXPickStep.

Arguments

None.

1.51 AFXStringKeyword

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

1.51.1 AFXStringKeyword(...)

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.

1.51.2 getTypeName()

Returns the name of the keyword type.
Implements AFXKeyword.

Arguments

None.

1.51.3 getValue()

Returns the keyword’s current value.

Arguments

None.

1.51.4 getValueAsString()

Returns the text string that represents the keyword’s current value.
Implements AFXKeyword.

Arguments

None.

1.51.5 isValueChanged()

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

Arguments

None.

1.51.6 setDefaultValue(...)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	String		Default value.

1.51.7 setValue(...)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	String		New value.

1.51.8 setValueToDefault(...)

Sets the keyword value to its default.

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

1.51.9 setValueToPrevious()

Sets the keyword value to its previous value.
Implements AFXKeyword.

Arguments

None.

1.51.10 syncPreviousValue()

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

Implements AFXKeyword.

Arguments

None.

1.52 AFXStringTarget

This class is designed for string targets.

1.52.1 AFXStringTarget(...)

Constructor.

Argument	Type	Default	Description
initialValue	String	”	Initial value.

1.52.2 getTypeName()

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

Arguments

None.

1.52.3 getValue()

Returns the target’s current value.

Arguments

None.

1.52.4 setValue(...)

Sets the target’s current value.

Argument	Type	Default	Description
newValue	String		New value.

1.53 AFXSymConstKeyword

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

1.53.1 AFXSymConstKeyword(...)

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.

1.53.2 getTypeName()

Returns the name of the keyword type.
Reimplemented from AFXIntKeyword.

Arguments

None.

1.53.3 getValueAsString()

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

Arguments

None.

1.53.4 setDefaultValue(...)

Sets the keyword's default value.

Argument	Type	Default	Description
defaultValue	Int		Default value.

1.53.5 setDefaultValueByString(...)

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.

1.53.6 setDefaultValueByString(...)

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.

1.53.7 setValue(...)

Sets the keyword's current value.

Argument	Type	Default	Description
newValue	Int		New value.

1.53.8 setValueByString(...)

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.

1.53.9 setValueByString(...)

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.

1.53.10 setValueToDefault(...)

Sets the keyword value to its default.

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

1.54 AFXTable

This class implements an editable table.

1.54.1 AFXTable(...)

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).

Argument	Type	Default	Description
pb	Int	DEFAULT_MARGIN	Bottom padding (margin).

1.54.2 addList(...)

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.

1.54.3 addList(...)

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.

1.54.4 appendClientPopupItem(...)

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.

1.54.5 appendClientPopupSeparator()

Appends a client separator to the MB3 popup menu.

Arguments

None.

1.54.6 appendListItem(...)

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.

1.54.7 beginEdit(...)

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.

1.54.8 cancelEdit()

Cancels the edit mode.

Arguments

None.

1.54.9 clearClientPopupItems()

Removes all client items from the MB3 popup menu.

Arguments

None.

1.54.10 clearContents(...)

Clears the text in the items in the specified range.

Argument	Type	Default	Description
startRow	Int		Row in which to start clearing.

Argument	Type	Default	Description
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.

1.54.11 clearListItems(...)

Removes all items from the specified table list.

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

1.54.12 create()

Creates server-side resources.

Reimplemented from AFXBaseTable.

Arguments

None.

1.54.13 deleteColumns(...)

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.

1.54.14 deleteRows(...)

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.

1.54.15 deselectItem(...)

Deselects the specified item.

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

1.54.16 deselectRow(...)

Deselects all items in the row.

Argument	Type	Default	Description
row	Int		Row index.

1.54.17 destroy()

Destroys server-side resources.
Reimplemented from FXComposite.

Arguments

None.

1.54.18 detach()

Detaches server-side resources.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.19 disable()

Disables the table and the table items in the table.
Reimplemented from FXWindow.

Arguments

None.

1.54.20 disableItem(...)

Disables the specified item.

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

1.54.21 enable()

Enables the table and the table items in the table.
Reimplemented from FXWindow.

Arguments

None.

1.54.22 enableItem(...)

Enables the specified item.

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

1.54.23 getColumnAtX(...)

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.

1.54.24 getColumnSortOrder(...)

Returns the sort order of the given column.

Argument	Type	Default	Description
column	Int		Column index.

1.54.25 getColumnWidth(...)

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

Argument	Type	Default	Description
column	Int		Column index.

1.54.26 getCurrentColumn()

Returns the column index of the current item.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.27 getCurrentRow()

Returns the row index of the current item.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.28 getCurrentSortColumn()

Returns the current sort column or -1 if none.

Arguments

None.

1.54.29 getDefaultColumnWidth()

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

Arguments

None.

1.54.30 getDefaultHeight()

Returns the default height of the table.

Reimplemented from AFXBaseTable.

Arguments

None.

1.54.31 getDefaultRowHeight()

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

Arguments

None.

1.54.32 getDefaultWidth()

Returns the default width of the table.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.33 getFont()

Gets the font for all text items in the table.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.34 getGridColor()

Gets the color of the grid lines in the table.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.35 getItemBackColor(...)

Gets the background color of an item.

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

1.54.36 getItemBoolValue(...)

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.

1.54.37 getItemColor(...)

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.

1.54.38 getItemFloatValue(...)

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.

1.54.39 getItemIcon(...)

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.

1.54.40 getItemIntValue(...)

Returns the value of an item of type INT.

Argument	Type	Default	Description
row	Int		Row index of item.

Argument	Type	Default	Description
column	Int		Column index of item.

1.54.41 getItemListId(...)

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.

1.54.42 getItemListIndex(...)

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.

1.54.43 getItemProvider()

Returns the item provider of this object.

Arguments

None.

1.54.44 getItemSelector(...)

Returns the message ID for an item.

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

1.54.45 getItemTarget(...)

Returns the target for an item.

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

1.54.46 getItemText(...)

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.

1.54.47 getItemTextColor(...)

Returns the text color of an item.

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

1.54.48 getItemType(...)

Returns the type of an item.

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

1.54.49 getItemValue(...)

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.

1.54.50 getLeadingColumns()

Returns the number of leading columns in the table.

Reimplemented from AFXBaseTable.

Arguments

None.

1.54.51 getLeadingFont()

Returns the font of the leading rows and columns.

Arguments

None.

1.54.52 getLeadingRows()

Returns the number of leading rows in the table.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.53 getListItemIcon(...)

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.

1.54.54 getListItemText(...)

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.

1.54.55 getNumColumns()

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

Arguments

None.

1.54.56 getNumEmptyRowsAtBottom()

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

Arguments

None.

1.54.57 getNumListItems(...)

Returns the number of items in the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list.

1.54.58 getNumRows()

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

Arguments

None.

1.54.59 getPopupOptions()

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

Arguments

None.

1.54.60 getPreferredColumnWidth(...)

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.

1.54.61 getPreferredRowHeight(...)

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

Argument	Type	Default	Description
row	Int		Row index.

1.54.62 getRowAtY(...)

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.

1.54.63 getRowHeight(...)

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

Argument	Type	Default	Description
row	Int		Row index.

1.54.64 getSelBackColor()

Gets the selection background color of the table.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.65 getSelTextColor()

Gets the selection text color of the table.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.66 getStretchableColumn()

Returns the index of the stretchable column.

Arguments

None.

1.54.67 getTableStyle()

Returns the options related only to the table.

Arguments

None.

1.54.68 getVisibleColumns()

Gets the number of visible columns in the table.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.69 getVisibleRows()

Gets the number of visible rows in the table.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.70 insertColumns(...)

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.

1.54.71 insertRows(...)

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.

1.54.72 isAnyItemInColumnSelected(...)

Returns True if any item in the column is selected.

Argument	Type	Default	Description
column	Int		Column index.

1.54.73 isAnyItemInRowSelected(...)

Returns True if any item in the row is selected.

Argument	Type	Default	Description
row	Int		Row index.

1.54.74 isColumnSelected(...)

Returns True if all items in the column are selected.

Argument	Type	Default	Description
column	Int		Column index.

1.54.75 isColumnSortable(...)

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

Argument	Type	Default	Description
column	Int		Column index.

1.54.76 isItemBool(...)

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.

1.54.77 isItemColor(...)

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.

1.54.78 isItemEditable(...)

Returns True if the item is editable.

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

1.54.79 isItemEmpty(...)

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.

1.54.80 isItemIcon(...)

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.

1.54.81 isItemList(...)

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.

1.54.82 isItemSelected(...)

Returns True if the specified item is selected.

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

1.54.83 isItemText(...)

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.

1.54.84 isItemVisible(...)

Returns True if the specified item is visible.

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

1.54.85 isRowSelected(...)

Returns True if all items in the row are selected.

Argument	Type	Default	Description
row	Int		Row index.

1.54.86 killFocus()

Removes the focus from this window.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.87 killSelection(...)

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).

1.54.88 layout()

Lays out the table contents.

Reimplemented from AFXBaseTable.

Arguments

None.

1.54.89 makePositionVisible(...)

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.

1.54.90 makeRowVisible(...)

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

Argument	Type	Default	Description
row	Int		Row index.

1.54.91 moveContents(...)

Scrolls the contents.

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

1.54.92 recalc()

Propagates size changes.
Reimplemented from AFXBaseTable.

Arguments

None.

1.54.93 removeListItem(...)

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.

1.54.94 selectItem(...)

Selects the specified item.

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

1.54.95 selectRow(...)

Selects all items in the row.

Argument	Type	Default	Description
row	Int		Row index.

1.54.96 setColumnBoolIcons(...)

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.

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.

1.54.97 setColumnBoolValue(...)

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.

1.54.98 setColumnColor(...)

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.

1.54.99 setColumnColorItemDefault(...)

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.

1.54.100 setColumnColorOptions(...)

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).

1.54.101 setColumnEditable(...)

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.

1.54.102 setColumnFloatValue(...)

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.

1.54.103 setColumnIcon(...)

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.

1.54.104 setColumnIntValue(...)

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.

1.54.105 setColumnJustify(...)

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).

1.54.106 setColumnListId(...)

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.

1.54.107 setColumnListIndex(...)

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.

1.54.108 setColumnSortable(...)

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.

1.54.109 setColumnSortOrder(...)

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).

1.54.110 setColumnText(...)

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.

1.54.111 setColumnType(...)

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).

1.54.112 setColumnWidth(...)

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.

1.54.113 setColumnWidthInChars(...)

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.

1.54.114 setCurrentItem(...)

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.

1.54.115 setCurrentSortColumn(...)

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.

1.54.116 setDefaultBoolIcons(...)

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.

1.54.117 setDefaultBoolValue(...)

Sets the default bool value for the table.

Argument	Type	Default	Description
value	Bool		Specify True or False.

1.54.118 setDefaultColor(...)

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.

1.54.119 setDefaultColumnWidth(...)

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

Argument	Type	Default	Description
width	Int		Width in pixels.

1.54.120 setDefaultFloatValue(...)

Sets the default floating-point value for the table.

Argument	Type	Default	Description
value	Float		Floating-point value.

1.54.121 setDefaultIntValue(...)

Sets the default integer value for the table.

Argument	Type	Default	Description
value	Int		Integer value.

1.54.122 setDefaultJustify(...)

Sets the default justification for the table.

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

1.54.123 setDefaultRowHeight(...)

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

Argument	Type	Default	Description
height	Int		Height in pixels.

1.54.124 setDefaultText(...)

Sets the default text for the table.

Argument	Type	Default	Description
text	String		Text.

1.54.125 setDefaultType(...)

Sets the default type for the table.

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

1.54.126 setEmptyItemDefault(...)

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.

1.54.127 setEmptyItemPolicy(...)

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).

1.54.128 setFont(...)

Sets the font for all text items in the table.

Argument	Type	Default	Description
font	FXFont		Font.

1.54.129 setGridColor(...)

Sets the color for the grid lines in the table.

Argument	Type	Default	Description
color	FXColor		Color.

1.54.130 setItemBackColor(...)

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.

1.54.131 setItemBackColor(...)

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.

1.54.132 setItemBoolIcons(...)

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.

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

1.54.133 setItemBoolValue(...)

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.

1.54.134 setItemColor(...)

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.

1.54.135 setItemEditable(...)

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.

1.54.136 setItemFloatValue(...)

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.

Argument	Type	Default	Description
value	Float		Floating-point value.

1.54.137 setItemIcon(...)

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.

1.54.138 setItemIntValue(...)

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.

1.54.139 setItemJustify(...)

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).

1.54.140 setItemListId(...)

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.

1.54.141 setItemListIndex(...)

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.

1.54.142 setItemProvider(...)

Sets the item provider of this object.

Argument	Type	Default	Description
ip	FXObject		Item provideer.

1.54.143 setItemSpan(...)

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.

1.54.144 setItemTarget(...)

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.

1.54.145 setItemText(...)

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.

1.54.146 setItemTextColor(...)

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.

1.54.147 setItemTextColor(...)

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.

1.54.148 setItemType(...)

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).

1.54.149 setItemValue(...)

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.

1.54.150 setLeadingColumnLabels(...)

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).

1.54.151 setLeadingColumns(...)

Sets the number of leading columns.

Argument	Type	Default	Description
numColumns	Int		Number of columns.

1.54.152 setLeadingFont(...)

Sets the font of the leading rows and columns.

Argument	Type	Default	Description
font	FXFont		Font.

1.54.153 setLeadingRowLabels(...)

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).

1.54.154 setLeadingRows(...)

Sets the number of leading rows.

Argument	Type	Default	Description
numRows	Int		Number of rows.

1.54.155 setListMaxVisible(...)

Sets the maximum number of visible items for all table lists.

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

1.54.156 setPopupOptions(...)

Sets the menu items to be displayed in the popup menu.

Argument	Type	Default	Description
opts	Int		Options.

1.54.157 setRowHeight(...)

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

Argument	Type	Default	Description
row	Int		Row index.
height	Int		Height in pixels.

1.54.158 setSelBackColor(...)

Sets the selection background color of the table.

Argument	Type	Default	Description
color	FXColor		Color index.

1.54.159 setSelTextColor(...)

Sets the selection text color of the table.

Argument	Type	Default	Description
color	FXColor		Color index.

1.54.160 setStretchableColumn(...)

Sets the stretchable column. (This method only works for the last column.)

Argument	Type	Default	Description
column	Int		Column index.

1.54.161 setTableSize(...)

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.

1.54.162 setTableStyle(...)

Sets the table options.

Argument	Type	Default	Description
style	Int		Style flag (see Flags for AFX table options).

1.54.163 setVisibleColumns(...)

Sets the number of visible columns in the table.

Argument	Type	Default	Description
visibleColumns	Int		Number of visible columns.

1.54.164 setVisibleRows(...)

Sets the number of visible rows in the table.

Argument	Type	Default	Description
visibleRows	Int		Number of visible rows.

1.54.165 shadeReadOnlyItems(...)

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.

1.54.166 showHorizontalGrid(...)

Controls the display of horizontal grid lines in the table.

Argument	Type	Default	Description
on	Bool	True	True if grid lines should be displayed.

1.54.167 showVerticalGrid(...)

Controls the display of vertical grid lines in the table.

Argument	Type	Default	Description
on	Bool	True	True if grid lines 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.

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.
IGNORE_BOTTOM_EMPTY_ROWS	Exclude empty rows at the bottom of the table (default).
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.

1.55 AFXTableKeyword

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

1.55.1 AFXTableKeyword(...)

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.

1.55.2 getTypeName()

Returns the name of the table keyword type.
Reimplemented from AFXComTableKeyword.

Arguments

None.

1.56 AFXTarget

This class is the base class for all target objects.

1.56.1 AFXTarget(...)

Constructor.

Argument	Type	Default	Description

1.56.2 connect(...)

Associates the data with a string variable.

Argument	Type	Default	Description
value	String		Variable to be associated with.

1.56.3 connect(...)

Associates the data with a floating-point variable.

Argument	Type	Default	Description
value	Float		Variable to be associated with.

1.56.4 connect(...)

Associates the data with an integer variable.

Argument	Type	Default	Description
value	Int		Variable to be associated with.

1.56.5 getSelector()

Returns the message ID of this target object.

Arguments

None.

1.56.6 getTarget()

Returns the target of this target object.

Arguments

None.

1.56.7 getType()

Returns the target type; this method is deprecated in Abaqus 6.6, and its use should be replaced by getTypeName().

Arguments

None.

1.56.8 getTypeName()

Returns the name of the target type.
Implemented in AFXFloatTarget, AFXIntTarget, and AFXStringTarget.

Arguments

None.

1.56.9 setSelector(...)

Sets the message ID of this target object.

Argument	Type	Default	Description
msgId	Int		Message ID.

1.56.10 setTarget(...)

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.

1.57 AFXTextField

This class contains a label that precedes a text field that allows the user to enter in text.

1.57.1 AFXTextField(...)

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).

1.57.2 create()

Creates the text field.

Reimplemented from FXComposite.

Arguments

None.

1.57.3 disable()

Disables the text field.
Reimplemented from FXWindow.

Arguments

None.

1.57.4 enable()

Enables the text field.
Reimplemented from FXWindow.

Arguments

None.

1.57.5 getCheck()

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

Arguments

None.

1.57.6 getCursorPos()

Returns the cursor position.

Arguments

None.

1.57.7 getDefaultWidth()

Returns the default width of the text field.
Reimplemented from FXPacker.

Arguments

None.

1.57.8 getExponentType()

Returns the exponent type of the text field for real and complex types.

Arguments

None.

1.57.9 getImaginaryText()

Returns the imaginary text for the complex text field.

Arguments

None.

1.57.10 getJustify()

Returns the text justification mode.

Arguments

None.

1.57.11 getLabelFont()

Returns the label's font.

Arguments

None.

1.57.12 getLabelText()

Returns the label text.

Arguments

None.

1.57.13 getNumColumns()

Returns the number of columns.

Arguments

None.

1.57.14 getPrecision()

Returns the precision of the text field for real and complex types.

Arguments

None.

1.57.15 getText()

Returns the text.

Arguments

None.

1.57.16 getValueType()

Returns the value type (AFXTEXTFIELD_FLOAT, etc.) of the text field.

Arguments

None.

1.57.17 isEditable()

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

Arguments

None.

1.57.18 isReadOnlyState()

Returns True if the text field appears in the read-only state.

Arguments

None.

1.57.19 isVerticalLayout()

Returns True if the layout orientation is vertical.

Arguments

None.

1.57.20 setCheck(...)

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

Argument	Type	Default	Description
state	Bool		Check state.

1.57.21 setCheckButtonSelector(...)

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

Argument	Type	Default	Description
sel	Int		Selector.

1.57.22 setCheckButtonTarget(...)

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

Argument	Type	Default	Description
checkVal	Bool	False	Check state.

1.57.23 setCursorPos(...)

Sets the cursor position.

Argument	Type	Default	Description
pos	Int		Position.

1.57.24 setEditable(...)

Sets the editable state for the text field.

Argument	Type	Default	Description
edit	Bool	True	If True, text can be edited.

1.57.25 setExponentType(...)

Sets the exponent type of the text field for real and complex types.

Argument	Type	Default	Description
e	FXExponent		Exponent type.

1.57.26 setFocus()

Moves the focus to the text field.
Reimplemented from FXWindow.

Arguments

None.

1.57.27 setFocusAndSelection()

Sets the focus to the input field and selects its contents.

Arguments

None.

1.57.28 setFocusToCheckButton()

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

Arguments

None.

1.57.29 setFocusToImaginaryTextField()

Moves the focus to the input field for the imaginary part.

Arguments

None.

1.57.30 setFocusToTextField()

Moves the focus to the input field of the widget.

Arguments

None.

1.57.31 setImaginaryFocusAndSelection()

Sets the focus to the input field for the imaginary part and selects its contents.

Arguments

None.

1.57.32 setImaginaryText(...)

Sets the imaginary text for the complex text field.

Argument	Type	Default	Description
text	String		Imaginary text field text.

1.57.33 setJustify(...)

Sets the text justification mode.

Argument	Type	Default	Description
mode	Int		Justification flag.

1.57.34 setLabelFont(...)

Sets the label's text font.

Argument	Type	Default	Description
fnt	FXFont		Label font.

1.57.35 setLabelText(...)

Sets the label text.

Argument	Type	Default	Description
txt	String		Label text.

1.57.36 setNumColumns(...)

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.

1.57.37 setPrecision(...)

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.

1.57.38 setReadOnlyState(...)

Sets the read-only state of the text field.

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

1.57.39 setSelection(...)

Select the specified number of characters starting at given position.

Argument	Type	Default	Description
pos	Int		Position.
len	Int		Length.

1.57.40 setText(...)

Sets the text in the input field.

Argument	Type	Default	Description
text	String		Text field text.

1.57.41 setValueType(...)

Sets the value type (AFXTEXTFIELD_FLOAT, etc.) of the text field.

Argument	Type	Default	Description
type	Int		Value type.

1.57.42 setVerticalLayout(...)

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.

1.58 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.

1.58.1 AFXToolbarGroup(...)

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.

1.58.2 getDefaultHeight()

Returns the default height.
Reimplemented from FXToolbar.

Arguments

None.

1.58.3 getDefaultWidth()

Returns the default width.
Reimplemented from FXToolbar.

Arguments

None.

1.58.4 getName()

Returns the English identifier for the group.

Arguments

None.

1.58.5 getOwner()

Returns the creator of the group.
Reimplemented from FXWindow.

Arguments

None.

1.58.6 getTitle()

Returns the name appearing in the title bar when the group is floating.

Arguments

None.

1.58.7 hide()

Hide this window.

Reimplemented from FXWindow.

Reimplemented in AFXToolbarGroupRender, and AFXToolbarGroupVisibility.

Arguments

None.

1.58.8 isActive()

Return True if the window is active.

Reimplemented from FXWindow.

Arguments

None.

1.58.9 layout()

Calculates layout.

Reimplemented from FXToolbar.

Arguments

None.

1.58.10 setName(...)

Sets the English identifier for the group.

Argument	Type	Default	Description
name	String		

1.58.11 setTitle(...)

Sets the name appearing in the title bar when the group is floating.

Argument	Type	Default	Description
title	String		

1.58.12 show()

Show this window.

Reimplemented from FXWindow.

Reimplemented in AFXToolbarGroupRender, and AFXToolbarGroupVisibility.

Arguments

None.

1.59 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.

1.59.1 AFXToolboxGroup(...)

Constructor.

Argument	Type	Default	Description
owner	AFXGuiObjectManager		Creator of the toolbox group.
parent	FXComposite	None	Parent widget.

1.59.2 getOwner()

Returns the owner of the toolbox group.

Reimplemented from FXWindow.

Arguments

None.

1.60 AFXToolButton

This class contains a button for use in the tool bar or the toolbox.

1.60.1 AFXToolButton(...)

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.

1.61 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.

1.61.1 AFXToolsetGui(...)

Constructor.

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

1.61.2 activate()

Activates the toolset (if there is no mode factory, then this method need not be redefined).

Arguments

None.

1.61.3 deactivate()

Deactivates the toolset (if there is no mode factory, then this method need not be redefined).

Arguments

None.

1.61.4 getToolsetName()

Returns the name of the toolset given on construction.

Arguments

None.

1.61.5 hide(...)

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.

1.61.6 show(...)

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.

1.62 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.

1.62.1 AFXTransition(...)

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.

1.62.2 AFXTransition(...)

Constructor.

Argument	Type	Default	Description
floatKeyword	AFXFloatKeyword		Keyword.
op	Operator		Operator type.

Argument	Type	Default	Description
refValue	Float		Reference value.
tgt	FXObject		Message target.
sel	Int		Message selector.
ptr	String	None	Message data.

1.62.3 AFXTransition(...)

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.

1.62.4 AFXTransition(...)

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.

1.62.5 AFXTransition(...)

Constructor.

Argument	Type	Default	Description
floatTarget	AFXFloatTarget		Target.
op	Operator		Operator type.
refValue	Float		Reference value.
tgt	FXObject		Message target.

Argument	Type	Default	Description
sel	Int		Message selector.
ptr	String	None	Message data.

1.62.6 AFXTransition(...)

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.

1.62.7 process(...)

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.

1.63 AFXTreeTable

This class combines a tree widget with a table widget to allow associating a row of data with an item in a tree.

1.63.1 AFXTreeTable(...)

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.

1.63.2 addItemAfter(...)

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	

1.63.3 addItemAfter(...)

Appends the new tree item after the other tree item.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

1.63.4 addItemBefore(...)

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	

1.63.5 addItemBefore(...)

Prepends the new item prior to the other tree item.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

1.63.6 addItemFirst(...)

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	

1.63.7 addItemFirst(...)

Prepends the new tree item as first child of parent.

Argument	Type	Default	Description
p	FXTreeItem		
item	FXTreeItem		

Argument	Type	Default	Description
notify	Bool	False	

1.63.8 addItemLast(...)

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	

1.63.9 addItemLast(...)

Appends the new tree item as the last child of the parent.

Argument	Type	Default	Description
p	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

1.63.10 addList(...)

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.

1.63.11 addList(...)

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	Usual flag.

1.63.12 appendListItem(...)

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.

1.63.13 beginEdit(...)

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.

1.63.14 cancelEdit()

Cancels the edit mode.

Arguments

None.

1.63.15 clearContents(...)

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.

1.63.16 clearItems(...)

Removes all tree items and table rows.

Argument	Type	Default	Description
notify	Bool	False	

1.63.17 clearListItems(...)

Removes all items from the specified table list.

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

1.63.18 closeItem(...)

Closes the specified item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.63.19 collapseTree(...)

Collapses the specified item to hide its children.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.63.20 deleteColumns(...)

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.

1.63.21 deselectItem(...)

Deselects the specified item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.
notify	Bool	False	

1.63.22 deselectRow(...)

Deselects all items in the row.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
notify	Bool	False	

1.63.23 expandTree(...)

Expands the specified item to show its children.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.63.24 getColumnWidth(...)

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

Argument	Type	Default	Description
column	Int		Column index.

1.63.25 getCurrentColumn()

Returns the column index of the current item.

Arguments

None.

1.63.26 getCurrentItem()

Returns the current item, if any.

Arguments

None.

1.63.27 getDefaultColumnWidth()

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

Arguments

None.

1.63.28 getDefaultHeight()

Return default height.
Reimplemented from FXScrollArea.

Arguments

None.

1.63.29 getDefaultWidth()

Return default width.
Reimplemented from FXScrollArea.

Arguments

None.

1.63.30 getFirstItem()

Returns the first root tree item.

Arguments

None.

1.63.31 getItemBoolValue(...)

Returns the value of a table item of type BOOL.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.

Argument	Type	Default	Description
column	Int		Column index of table item.

1.63.32 getItemCheck(...)

Returns the item checked state.

Argument	Type	Default	Description
item	FXTreeItem		

1.63.33 getItemClosedIcon(...)

Returns the tree item's closed icon.

Argument	Type	Default	Description
item	FXTreeItem		

1.63.34 getItemColor(...)

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.

1.63.35 getItemFloatValue(...)

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.

1.63.36 getItemFormat(...)

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.

1.63.37 getItemIcon(...)

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.

1.63.38 getItemIntValue(...)

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.

1.63.39 getItemListId(...)

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.

1.63.40 getItemListIndex(...)

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.

1.63.41 getItemNumDigits(...)

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.

1.63.42 getItemOpenIcon(...)

Returns the tree item's open icon.

Argument	Type	Default	Description
item	FXTreeItem		

1.63.43 getItemPrecision(...)

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.

1.63.44 getItemText(...)

Returns the text of an item of type TEXT.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.

1.63.45 getItemType(...)

Returns the type of a table item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

1.63.46 getItemValue(...)

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.

1.63.47 getLastItem()

Returns the last root tree item.

Arguments

None.

1.63.48 getListItemIcon(...)

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.

1.63.49 getListItemIndex(...)

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.

1.63.50 getListItemIndex(...)

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.

1.63.51 getListItemText(...)

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.

1.63.52 getNumColumns()

Returns the number of columns.

Arguments

None.

1.63.53 getNumItems()

Returns the number of items.

Arguments

None.

1.63.54 getNumListItems(...)

Returns the number of items in the specified table list.

Argument	Type	Default	Description
listId	Int		ID of the list.

1.63.55 getTableStyle()

Returns the options related only to the table.

Arguments

None.

1.63.56 getTreeColumn()

Returns the column index of the tree.

Arguments

None.

1.63.57 getVisibleColumns()

Returns the number of visible columns.

Arguments

None.

1.63.58 getVisibleItems()

Returns the number of visible items.

Arguments

None.

1.63.59 insertColumns(...)

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.

1.63.60 isAnyItemInColumnSelected(...)

Returns True if any item in the column is selected.

Argument	Type	Default	Description
column	Int		Column index.

1.63.61 isAnyItemInRowSelected(...)

Returns True if any item in the row is selected.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.

1.63.62 isColumnSelected(...)

Returns True if all items in the column are selected.

Argument	Type	Default	Description
column	Int		Column index.

1.63.63 isItemBool(...)

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.

1.63.64 isItemColor(...)

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.

1.63.65 isItemEditable(...)

Returns True if the table item is editable.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of table item.

1.63.66 isItemEmpty(...)

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.

1.63.67 isItemExpanded(...)

Returns True if the tree item is expanded, False otherwise.

Argument	Type	Default	Description
item	FXTreeItem		

1.63.68 isItemFloat(...)

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.

1.63.69 isItemIcon(...)

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.

1.63.70 isItemInt(...)

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.

1.63.71 isItemLeaf(...)

Returns True if the tree item is a leaf-item (has no children), False otherwise.

Argument	Type	Default	Description
item	FXTreeItem		

1.63.72 isItemList(...)

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.

1.63.73 isItemOpened(...)

Returns True if the tree item is opened, False otherwise.

Argument	Type	Default	Description
item	FXTreeItem		

1.63.74 isItemSelected(...)

Returns True if the specified item is selected.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.

1.63.75 isItemText(...)

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.

1.63.76 isItemVisible(...)

Returns True if the specified item is visible.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.

1.63.77 isRowSelected(...)

Returns True if all items in the row are selected.

Argument	Type	Default	Description
item	FXTreeItem		

1.63.78 killSelection(...)

Deselects all items.

Argument	Type	Default	Description
notify	Bool	False	

1.63.79 makePositionVisible(...)

Scrolls to make the specified row, column fully visible.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.

1.63.80 makeRowVisible(...)

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

Argument	Type	Default	Description
item	FXTreeItem		Tree item.

1.63.81 openItem(...)

Opens the specified item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.63.82 removeItem(...)

Removes the specified tree item and corresponding table row.

Argument	Type	Default	Description
item	FXTreeItem		

Argument	Type	Default	Description
notify	Bool	False	

1.63.83 removeItems(...)

Removes the specified tree items and their corresponding table rows, inclusively.

Argument	Type	Default	Description
from	FXTreeItem		
to	FXTreeItem		
notify	Bool	False	

1.63.84 removeListItem(...)

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.

1.63.85 selectItem(...)

Selects the specified item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.
notify	Bool	False	

1.63.86 selectRow(...)

Selects all items in the row.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
notify	Bool	False	

1.63.87 setColumnBoolIcons(...)

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.

1.63.88 setColumnBoolValue(...)

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.

1.63.89 setColumnColor(...)

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.

1.63.90 setColumnColorItemDefault(...)

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.

1.63.91 setColumnColorOptions(...)

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).

1.63.92 setColumnEditable(...)

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.

1.63.93 setColumnFloatValue(...)

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.

1.63.94 setColumnFormat(...)

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).

1.63.95 setColumnIcon(...)

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.

1.63.96 setColumnIntValue(...)

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.

1.63.97 setColumnJustify(...)

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).

1.63.98 setColumnListId(...)

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.

1.63.99 setColumnListIndex(...)

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.

1.63.100 setColumnNumDigits(...)

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.

1.63.101 setColumnPrecision(...)

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.

1.63.102 setColumnText(...)

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.

1.63.103 setColumnType(...)

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.

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

1.63.104 setColumnWidth(...)

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.

1.63.105 setColumnWidthInChars(...)

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.

1.63.106 setCurrentItem(...)

Sets the current item.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.
column	Int		Column index of item.
notify	Bool	False	

1.63.107 setDefaultBoolIcons(...)

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.

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

1.63.108 setDefaultBoolValue(...)

Sets the default bool value for the table.

Argument	Type	Default	Description
value	Bool		Specify True or False.

1.63.109 setDefaultColor(...)

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.

1.63.110 setDefaultColumnWidth(...)

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

Argument	Type	Default	Description
width	Int		Width in pixels.

1.63.111 setDefaultFloatValue(...)

Sets the default floating-point value for the table.

Argument	Type	Default	Description
value	Float		Floating-point value.

1.63.112 setDefaultFormat(...)

Sets the default real format for the table.

Argument	Type	Default	Description
format	Int		Format flag.

1.63.113 setDefaultIntValue(...)

Sets the default integer value for the table.

Argument	Type	Default	Description
value	Int		Integer value.

1.63.114 setDefaultJustify(...)

Sets the default justification for the table.

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

1.63.115 setDefaultNumDigits(...)

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.

1.63.116 setDefaultPrecision(...)

Sets the precision for the table.

Argument	Type	Default	Description
precision	Int		Precision.

1.63.117 setDefaultText(...)

Sets the default text for the table.

Argument	Type	Default	Description
text	String		Text.

1.63.118 setDefaultType(...)

Sets the default type for the table.

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

1.63.119 setItemBoolIcons(...)

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.

1.63.120 setItemBoolValue(...)

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.

1.63.121 setItemCheck(...)

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	

1.63.122 setItemClosedIcon(...)

Changes the tree item's closed icon.

Argument	Type	Default	Description
item	FXTreeItem		
icon	FXIcon		

1.63.123 setItemColor(...)

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.

1.63.124 setItemEditable(...)

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.

1.63.125 setItemFloatValue(...)

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.

1.63.126 setItemFormat(...)

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.

Argument	Type	Default	Description
format	Int		Format of table item (see RealFormat).

1.63.127 setItemIcon(...)

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.

1.63.128 setItemIntValue(...)

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.

1.63.129 setItemJustify(...)

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).

1.63.130 setItemListId(...)

Sets the list ID of a table item of type LIST.

Argument	Type	Default	Description
item	FXTreeItem		Tree item.

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

1.63.131 setItemListIndex(...)

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.

1.63.132 setItemNumDigits(...)

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.

1.63.133 setItemOpenIcon(...)

Sets the tree item's open icon.

Argument	Type	Default	Description
item	FXTreeItem		
icon	FXIcon		

1.63.134 setItemPrecision(...)

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.

Argument	Type	Default	Description
precision	Int		Number of digits to the right of the decimal point.

1.63.135 setItemText(...)

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.

1.63.136 setItemType(...)

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).

1.63.137 setItemValue(...)

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.

1.63.138 setLeadingRowLabels(...)

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")

1.63.139 setListMaxVisible(...)

Sets the maximum number of visible items for all table lists.

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

1.63.140 setTableStyle(...)

Sets the table options.

Argument	Type	Default	Description
style	Int		Style flag (see Flags for AFX table options).

1.63.141 setVisibleColumns(...)

Sets the number of visible columns.

Argument	Type	Default	Description
visibleColumns	Int		Number of visible columns.

1.63.142 setVisibleItems(...)

Sets the number of visible items.

Argument	Type	Default	Description
visibleItems	Int		Number of visible items.

1.63.143 shadeReadOnlyItems(...)

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.

1.64 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).

1.64.1 AFXTupleKeyword(...)

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.

1.64.2 equal(...)

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.

1.64.3 getDefaultStyle()

Returns the default style for elements.

Arguments

None.

1.64.4 getDefaultType()

Returns the default type for elements.

Arguments

None.

1.64.5 getDefaultValues()

Returns the default values for this tuple.

Arguments

None.

1.64.6 getElementStyle(...)

Returns the style of one element. Will never return AFXTUPLE_STYLE_DEFAULT!

Argument	Type	Default	Description
index	Int		Element index.

1.64.7 getElementType(...)

Returns the type of one element. Will never return AFXTUPLE_TYPE_DEFAULT!

Argument	Type	Default	Description
index	Int		Element index.

1.64.8 getFormattedValue(...)

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.

1.64.9 getLength()

Returns the length of the tuple.

Arguments

None.

1.64.10 getMaxLength()

Returns the maximum length of this tuple, or -1 for unbounded length.

Arguments

None.

1.64.11 getMinLength()

Returns the minimum length of this tuple.

Arguments

None.

1.64.12 getTypeName()

Returns the name of the tuple keyword type.
Implements AFXKeyword.

Arguments

None.

1.64.13 getValue(...)

Returns the value of a tuple element.

Argument	Type	Default	Description
index	Int		Element index.

1.64.14 getValueAsDouble()

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

Arguments

None.

1.64.15 getValueAsInt()

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

Arguments

None.

1.64.16 getValueAsString()

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

Arguments

None.

1.64.17 getValueForBlank(...)

Returns the value substituted for blank tuple element.

Argument	Type	Default	Description
index	Int		Element index.

1.64.18 getValues()

Returns a string containing values (separated by commas) of the tuple elements.

Arguments

None.

1.64.19 getValuesForBlanks()

Returns a string containing values substituted for blanks for the tuple elements.

Arguments

None.

1.64.20 insertElements(...)

Inserts elements starting at the given index.

Argument	Type	Default	Description
index	Int		Starting index.
numCols	Int		Number of elements to insert.

1.64.21 isValueChanged()

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

Implements AFXKeyword.

Arguments

None.

1.64.22 removeElements(...)

Removes elements starting at the given index.

Argument	Type	Default	Description
index	Int		Starting index.
numCols	Int		Number of elements to remove.

1.64.23 setDefaultStyle(...)

Sets the default style for elements.

Argument	Type	Default	Description
style	Int		New default element style.

1.64.24 setDefaultType(...)

Sets the default type for elements.

Argument	Type	Default	Description
type	Int		New default type.

1.64.25 setDefaultValues(...)

Sets the default values for this tuple.

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

1.64.26 setElementStyle(...)

Sets the style of one element.

Argument	Type	Default	Description
index	Int		Element index.

Argument	Type	Default	Description
style	Int		New element style.

1.64.27 setElementType(...)

Sets the type of one element.

Argument	Type	Default	Description
index	Int		Element index.
type	Int		New element type.

1.64.28 setLengthRange(...)

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.

1.64.29 setMaxLength(...)

Sets the maximum length of this tuple.

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

1.64.30 setMinLength(...)

Sets the minimum length of this tuple.

Argument	Type	Default	Description
length	Int		New minimum length.

1.64.31 setValue(...)

Sets the value of the tuple element; returns False upon failure.

Argument	Type	Default	Description
index	Int		Element index.
value	String		New value.

1.64.32 setValueForBlank(...)

Sets the value substituted for a blank tuple element.

Argument	Type	Default	Description
index	Int		Element index.
value	String		New value.

1.64.33 setValues(...)

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.

1.64.34 setValuesForBlanks(...)

Sets all values substituted for blanks for the tuple elements.

Argument	Type	Default	Description
values	String		Tuple string with values.

1.64.35 setValueToDefault(...)

Sets the keyword value to its default.

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

1.64.36 setValueToPrevious()

Sets the keyword value to its previous value.
Implements AFXKeyword.

Arguments

None.

1.64.37 syncPreviousValue()

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

Arguments

None.

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.

1.65 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.

1.65.1 AFXVerticalAligner(...)

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.

1.65.2 create()

Creates the aligner.

Reimplemented from FXComposite.

Arguments

None.

1.65.3 getDefaultHeight()

Returns the default height.

Reimplemented from FXVerticalFrame.

Arguments

None.

1.65.4 getDefaultWidth()

Returns the default width.

Reimplemented from FXVerticalFrame.

Arguments

None.

1.66 FXApp

Application Object

1.66.1 FXApp(...)

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	

1.66.2 addChore(...)

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		

1.66.3 addInput(...)

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		

1.66.4 addSocket(...)

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		

1.66.5 addTimeout(...)

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		

1.66.6 beep()

Beep.

Arguments

None.

1.66.7 beginWaitCursor()

Begin of wait-cursor block; wait-cursor blocks may be nested.

Arguments

None.

1.66.8 create()

Create application's windows.
Reimplemented in AFXApp.

Arguments

None.

1.66.9 endWaitCursor()

End of wait-cursor block.

Arguments

None.

1.66.10 forceRefresh()

Force GUI refresh.

Arguments

None.

1.66.11 getAppName()

Get application name.

Arguments

None.

1.66.12 getBorderColor()

Obtain default colors.

Arguments

None.

1.66.13 getMainWindow()

Get main window, if any.

Arguments

None.

1.66.14 getMonoVisual()

Get monochrome visual.

Arguments

None.

1.66.15 getNormalFont()

Return default font.

Arguments

None.

1.66.16 getRoot()

Get root Window.

Arguments

None.

1.66.17 getTypingSpeed()

Obtain application-wide settings.

Arguments

None.

1.66.18 init(...)

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	

1.66.19 peekEvent()

Peek to determine if there's an event.

Arguments

None.

1.66.20 refresh()

Schedule a refresh.

Arguments

None.

1.66.21 removeChore(...)

Remove idle processing message.

Argument	Type	Default	Description
c	FXChore		

1.66.22 removeInput(...)

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		

1.66.23 removeSocket(...)

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		

Argument	Type	Default	Description
mode	Int		

1.66.24 removeTimeout(...)

Remove timeout, returns NULL.

Argument	Type	Default	Description
t	FXTimer		

1.66.25 repaint()

Paint all windows marked for repainting. On return all the applications windows have been painted.

Arguments

None.

1.66.26 run()

Run the main application event loop until stop() is called, and return the exit code passed as argument to stop().

Reimplemented in AFXApp.

Arguments

None.

1.66.27 runOneEvent()

Perform one event dispatch.

Arguments

None.

1.66.28 runUntil(...)

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

Reimplemented in AFXApp.

Argument	Type	Default	Description
condition	Int		

1.66.29 runWhileEvents(...)

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	

1.66.30 setBorderColor(...)

Change default colors.

Argument	Type	Default	Description
color	FXColor		

1.66.31 setNormalFont(...)

Change default font.

Argument	Type	Default	Description
font	FXFont		

1.66.32 setTypingSpeed(...)

Change application-wide settings.

Argument	Type	Default	Description
speed	Int		

1.66.33 stop(...)

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	

1.66.34 useWidgetBackColor(...)

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.

1.67 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.

1.67.1 FXArrowButton(...)

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	

1.67.2 canFocus()

Returns True because a button can receive focus.
Reimplemented from FXWindow.

Arguments

None.

1.67.3 disable()

Disable the button.
Reimplemented from FXWindow.

Arguments

None.

1.67.4 enable()

Enable the button.
Reimplemented from FXWindow.

Arguments

None.

1.67.5 getArrowColor()

Get the fill color for the arrow.

Arguments

None.

1.67.6 getArrowSize()

Get the default arrow size.

Arguments

None.

1.67.7 getArrowStyle()

Get the arrow style flags.

Arguments

None.

1.67.8 getDefaultHeight()

Get default height.
Reimplemented from FXFrame.

Arguments

None.

1.67.9 getDefaultWidth()

Get default width.
Reimplemented from FXFrame.

Arguments

None.

1.67.10 getJustify()

Get the current justification mode.

Arguments

None.

1.67.11 getState()

Get the button state (where True means the button is down).

Arguments

None.

1.67.12 getTipText()

Get tool tip message for this arrow button.

Arguments

None.

1.67.13 setArrowColor(...)

Set the fill color for the arrow.

Argument	Type	Default	Description
clr	FXColor		

1.67.14 setArrowSize(...)

Set the default arrow size.

Argument	Type	Default	Description
size	Int		

1.67.15 setArrowStyle(...)

Set the arrow style flags.

Argument	Type	Default	Description
style	Int		

1.67.16 setJustify(...)

Set the current justification mode.

Argument	Type	Default	Description
mode	Int		

1.67.17 setState(...)

Set the button state (where True means the button is down).

Argument	Type	Default	Description
s	Bool		

1.67.18 setTipText(...)

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.

1.68 FXBMPIcon

Microsoft Bitmap icon.

1.68.1 FXBMPIcon(...)

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	

1.69 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.

1.69.1 FXButton(...)

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	

1.69.2 canFocus()

Returns True because a button can receive focus.

Reimplemented from FXWindow.

Reimplemented in AFXFlyoutItem.

Arguments

None.

1.69.3 getButtonStyle()

Get the button style flags.

Arguments

None.

1.69.4 getState()

Get the button state.

Arguments

None.

1.69.5 setButtonStyle(...)

Set the button style flags.

Argument	Type	Default	Description
style	Int		

1.69.6 setDefault(...)

Set as default button.
Reimplemented from FXWindow.

Argument	Type	Default	Description
enable	Bool	True	

1.69.7 setState(...)

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.

1.70 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.

1.70.1 FXCheckButton(...)

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	

1.70.2 canFocus()

Returns True because a check button can receive focus.
Reimplemented from FXWindow.

Arguments

None.

1.70.3 getCheck()

Get check button state (True, False or MAYBE).
Reimplemented in AFXCheckButton.

Arguments

None.

1.70.4 getDefaultHeight()

Get default height.
Reimplemented from FXLabel.

Arguments

None.

1.70.5 getDefaultWidth()

Get default width.
Reimplemented from FXLabel.

Arguments

None.

1.70.6 setCheck(...)

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.

1.71 FXComposite

Base composite.

1.71.1 FXComposite(...)

Constructor.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

1.71.2 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.

Arguments

None.

1.71.3 destroy()

Destroy server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXComboBox, FXDirBox, FXDirList, FXDriveBox, FXFileList, FXListBox, FXRootWindow, FXTreeList, FXTreeListBox, AFXColorFlyout, and AFXTable.

Arguments

None.

1.71.4 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.

Arguments

None.

1.71.5 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.

Arguments

None.

1.71.6 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.

Arguments

None.

1.71.7 maxChildHeight()

Return the height of the tallest child window.

Arguments

None.

1.71.8 maxChildWidth()

Return the width of the widest child window.

Arguments

None.

1.72 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.

1.72.1 FXDataTarget(...)

Associate with nothing.

Argument	Type	Default	Description
tgt	FXObject	None	
sel	Int	0	

1.72.2 FXDataTarget(...)

Associate with character variable.

Argument	Type	Default	Description
value	String		
tgt	FXObject	None	
sel	Int	0	

1.72.3 FXDataTarget(...)

Associate with unsigned character variable.

Argument	Type	Default	Description
value	Int		
tgt	FXObject	None	
sel	Int	0	

1.72.4 FXDataTarget(...)

Associate with signed short variable.

Argument	Type	Default	Description
value	Int		
tgt	FXObject	None	
sel	Int	0	

1.72.5 FXDataTarget(...)

Associate with unsigned short variable.

Argument	Type	Default	Description
value	Int		
tgt	FXObject	None	
sel	Int	0	

1.72.6 FXDataTarget(...)

Associate with int variable.

Argument	Type	Default	Description
value	Int		
tgt	FXObject	None	
sel	Int	0	

1.72.7 FXDataTarget(...)

Associate with unsigned int variable.

Argument	Type	Default	Description
value	Int		
tgt	FXObject	None	
sel	Int	0	

1.72.8 FXDataTarget(...)

Associate with float variable.

Argument	Type	Default	Description
value	Float		
tgt	FXObject	None	
sel	Int	0	

1.72.9 FXDataTarget(...)

Associate with double variable.

Argument	Type	Default	Description
value	Float		
tgt	FXObject	None	
sel	Int	0	

1.72.10 FXDataTarget(...)

Associate with string variable.

Argument	Type	Default	Description
value	String		
tgt	FXObject	None	
sel	Int	0	

1.72.11 connect(...)

Associate with string variable.

Argument	Type	Default	Description
value	String		

1.72.12 connect(...)

Associate with double variable.

Argument	Type	Default	Description
value	Float		

1.72.13 connect(...)

Associate with float variable.

Argument	Type	Default	Description
value	Float		

1.72.14 connect(...)

Associate with unsigned int variable.

Argument	Type	Default	Description
value	Int		

1.72.15 connect(...)

Associate with int variable.

Argument	Type	Default	Description
value	Int		

1.72.16 connect(...)

Associate with unsigned short variable.

Argument	Type	Default	Description
value	Int		

1.72.17 connect(...)

Associate with signed short variable.

Argument	Type	Default	Description
value	Int		

1.72.18 connect(...)

Associate with unsigned character variable.

Argument	Type	Default	Description
value	Int		

1.72.19 connect(...)

Associate with character variable.

Argument	Type	Default	Description
value	String		

1.72.20 connect()

Associate with nothing.

Arguments

None.

1.72.21 getData()

Return pointer to data its connected to.

Arguments

None.

1.72.22 getSelector()

Get the message identifier for this data target.

Arguments

None.

1.72.23 getTarget()

Get the message target object for this data target, if any.

Arguments

None.

1.72.24 getType()

Return type of data its connected to.

Arguments

None.

1.72.25 setSelector(...)

Set the message identifier for this data target.

Argument	Type	Default	Description
sel	Int		

1.72.26 setTarget(...)

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.

1.73 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.

1.73.1 FXDialogBox(...)

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	

1.73.2 FXDialogBox(...)

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	

Argument	Type	Default	Description
w	Int	0	
h	Int	0	
pl	Int	10	
pr	Int	10	
pt	Int	10	
pb	Int	10	
hs	Int	4	
vs	Int	4	

1.73.3 execute(...)

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.

1.74 FXDrawable

Drawable is an abstract base class for any surface that can be drawn upon, such as a FXWindow, or FXImage.

1.74.1 getHeight()

Height of drawable.

Arguments

None.

1.74.2 getVisual()

Get the visual.

Arguments

None.

1.74.3 getWidth()

Width of drawable.

Arguments

None.

1.74.4 resize(...)

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		

1.75 FXFileItem

File item

1.75.1 FXFileItem(...)

File time.

Argument	Type	Default	Description

1.75.2 FXFileItem(...)

Constructor.

Argument	Type	Default	Description
text	String		
bi	FXIcon	None	
mi	FXIcon	None	
ptr	String	None	

1.75.3 getDate()

Return the date for this item.

Arguments

None.

1.75.4 getSize()

Return the file size for this item.

Arguments

None.

1.75.5 isBlockdev()

Return True if this is a block device item.

Arguments

None.

1.75.6 isChardev()

Return True if this is a character device item.

Arguments

None.

1.75.7 isDirectory()

Return True if this is a directory item.
Reimplemented in AFXFileItem.

Arguments

None.

1.75.8 isExecutable()

Return True if this is an executable item.

Arguments

None.

1.75.9 isFifo()

Return True if this is an FIFO item.

Arguments

None.

1.75.10 isFile()

Return True if this is a file item.

Arguments

None.

1.75.11 isSocket()

Return True if this is a socket.

Arguments

None.

1.75.12 isSymlink()

Return True if this is a symbolic link item.

Arguments

None.

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.

1.76 FXFont

Font class

1.76.1 FXFont(...)

Construct font from font description.

Argument	Type	Default	Description
a	FXApp		
fontdesc	FXFontDesc		

1.76.2 FXFont(...)

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	

1.76.3 FXFont(...)

Construct a font with given X11 font string.

Argument	Type	Default	Description
a	FXApp		
nm	String		

1.76.4 create()

Create the font.

Reimplemented from FXId.

Arguments

None.

1.76.5 destroy()

Destroy the font.
Reimplemented from FXId.

Arguments

None.

1.76.6 detach()

Detach the font.
Reimplemented from FXId.

Arguments

None.

1.76.7 getEncoding()

Get character set encoding.

Arguments

None.

1.76.8 getFontAscent()

Ascent from baseline.

Arguments

None.

1.76.9 getFontDesc(...)

Get font description.

Argument	Type	Default	Description
fontdesc	FXFontDesc		

1.76.10 getFontDescent()

Descent from baseline.

Arguments

None.

1.76.11 getFontHeight()

Height of highest character in font.

Arguments

None.

1.76.12 getFontLeading()

Get font leading [that is lead-ing as in Pb!].

Arguments

None.

1.76.13 getFontSpacing()

Get font line spacing.

Arguments

None.

1.76.14 getFontWidth()

Width of widest character in font.

Arguments

None.

1.76.15 getHints()

Get hints.

Arguments

None.

1.76.16 getMaxChar()

Get last character glyph in font.

Arguments

None.

1.76.17 getMinChar()

Get first character glyph in font.

Arguments

None.

1.76.18 getName()

Get face name.

Arguments

None.

1.76.19 getSetWidth()

Get setwidth.

Arguments

None.

1.76.20 getSize()

Get size in deci-points.

Arguments

None.

1.76.21 getSlant()

Get slant.

Arguments

None.

1.76.22 getTextHeight(...)

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.

1.76.23 getTextWidth(...)

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.

1.76.24 getWeight()

Get font weight.

Arguments

None.

1.76.25 hasChar(...)

See if font has glyph for ch.

Argument	Type	Default	Description
ch	Int		

1.76.26 isFontMono()

Find out if the font is monotype or proportional.

Arguments

None.

1.76.27 leftBearing(...)

Left bearing.

Argument	Type	Default	Description
ch	String		

1.76.28 rightBearing(...)

Right bearing.

Argument	Type	Default	Description
ch	String		

1.76.29 setFontDesc(...)

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.

1.77 FXFrame

Base Frame

1.77.1 FXFrame(...)

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	

1.77.2 getBaseColor()

Get base gui color.

Arguments

None.

1.77.3 getBorderColor()

Get border color.

Arguments

None.

1.77.4 getBorderWidth()

Get border width.

Arguments

None.

1.77.5 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.

Arguments

None.

1.77.6 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.

Arguments

None.

1.77.7 getFrameStyle()

Get current frame style.

Arguments

None.

1.77.8 getHiliteColor()

Get highlight color.

Arguments

None.

1.77.9 getPadBottom()

Get bottom interior padding.

Arguments

None.

1.77.10 getPadLeft()

Get left interior padding.

Arguments

None.

1.77.11 getPadRight()

Get right interior padding.

Arguments

None.

1.77.12 getPadTop()

Get top interior padding.

Arguments

None.

1.77.13 getShadowColor()

Get shadow color.

Arguments

None.

1.77.14 setBaseColor(...)

Change base gui color.

Argument	Type	Default	Description
clr	FXColor		

1.77.15 setBorderColor(...)

Change border color.

Argument	Type	Default	Description
clr	FXColor		

1.77.16 setFrameStyle(...)

Change frame style.

Argument	Type	Default	Description
style	Int		

1.77.17 setHiliteColor(...)

Change highlight color.

Argument	Type	Default	Description
clr	FXColor		

1.77.18 setPadBottom(...)

Change bottom padding.

Argument	Type	Default	Description
pb	Int		

1.77.19 setPadLeft(...)

Change left padding.

Argument	Type	Default	Description
pl	Int		

1.77.20 setPadRight(...)

Change right padding.

Argument	Type	Default	Description
pr	Int		

1.77.21 setPadTop(...)

Change top padding.

Argument	Type	Default	Description
pt	Int		

1.77.22 setShadowColor(...)

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.

1.78 FXGIFIcon

GIF Icon class.

1.78.1 FXGIFIcon(...)

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	

1.79 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.

1.79.1 FXGroupBox(...)

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	

Argument	Type	Default	Description
pr	Int	DEFAULT_SPACING	
pt	Int	DEFAULT_SPACING	
pb	Int	DEFAULT_SPACING	
hs	Int	DEFAULT_SPACING	
vs	Int	DEFAULT_SPACING	

1.79.2 create()

Create server-side resources.
Reimplemented from FXComposite.

Arguments

None.

1.79.3 detach()

Detach server-side resources.
Reimplemented from FXComposite.

Arguments

None.

1.79.4 disable()

Disable the window.
Reimplemented from FXWindow.

Arguments

None.

1.79.5 enable()

Enable the window.
Reimplemented from FXWindow.

Arguments

None.

1.79.6 getDefaultHeight()

Return default height.
Reimplemented from FXPacker.

Arguments

None.

1.79.7 getDefaultWidth()

Return default width.
Reimplemented from FXPacker.

Arguments

None.

1.79.8 getText()

Return current groupbox title text.

Arguments

None.

1.79.9 setText(...)

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.

1.80 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.

1.80.1 FXHorizontalFrame(...)

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	

1.80.2 getDefaultHeight()

Return default height.

Reimplemented from FXPacker.
Reimplemented in FXStatusbar.

Arguments

None.

1.80.3 getDefaultWidth()

Return default width.

Reimplemented from FXPacker.
Reimplemented in FXStatusbar.

Arguments

None.

1.81 FXHorizontalSeparator

Horizontal separator

1.81.1 FXHorizontalSeparator(...)

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	

1.81.2 getDefaultHeight()

Return default height.
Reimplemented from FXFrame.

Arguments

None.

1.81.3 getDefaultWidth()

Return default width.
Reimplemented from FXFrame.

Arguments

None.

Global flags

Separator Options

SEPARATOR_NONE	Nothing visible.
SEPARATOR_GROOVE	Etched-in looking groove.
SEPARATOR_RIDGE	Embossed looking ridge.
SEPARATOR_LINE	Simple line.

1.82 FXIcon

Icon class.

1.82.1 FXIcon(...)

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	

1.82.2 create()

Create the icon resource.
Reimplemented from FXImage.

Arguments

None.

1.82.3 destroy()

Destroy the icon resource.

Reimplemented from FXImage.

Arguments

None.

1.82.4 detach()

Detach the icon resource.
Reimplemented from FXImage.

Arguments

None.

1.82.5 render()

Render the image from client-side pixel buffer.
Reimplemented from FXImage.

Arguments

None.

1.82.6 resize(...)

Resize pixmap to the specified width and height.
Reimplemented from FXImage.

Argument	Type	Default	Description
w	Int		
h	Int		

1.83 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.

1.84 FXIconList

Icon List Widget

1.84.1 FXIconList(...)

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	

1.84.2 appendHeader(...)

Append header with given text and optional icon.

Argument	Type	Default	Description
text	String		
icon	FXIcon	None	
size	Int	1	

1.84.3 appendItem(...)

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	

1.84.4 appendItem(...)

Append a [possibly subclassed] item to the end of the list.

Argument	Type	Default	Description
item	FXIconItem		
notify	Bool	False	

1.84.5 canFocus()

Icon list can receive focus.
Reimplemented from FXWindow.

Arguments

None.

1.84.6 clearItems(...)

Remove all items from list.

Argument	Type	Default	Description
notify	Bool	False	

1.84.7 create()

Create server-side resources.
Reimplemented from FXComposite.
Reimplemented in FXFileList.

Arguments

None.

1.84.8 deselectItem(...)

Deselect item at index.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.84.9 detach()

Detach server-side resources.

Reimplemented from FXComposite.

Reimplemented in FXFileList.

Arguments

None.

1.84.10 disableItem(...)

Disable item at index.

Argument	Type	Default	Description
index	Int		

1.84.11 enableItem(...)

Enable item at index.

Argument	Type	Default	Description
index	Int		

1.84.12 extendSelection(...)

Extend selection from anchor index to index.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.84.13 findItem(...)

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	

Argument	Type	Default	Description
flags	Int	SEARCH_FORWARD| SEARCH_WRAP	

1.84.14 getAnchorItem()

Return anchor item index, or -1 if none.

Arguments

None.

1.84.15 getContentHeight()

Return content height.
Reimplemented from FXScrollArea.

Arguments

None.

1.84.16 getContentWidth()

Compute and return content width.
Reimplemented from FXScrollArea.

Arguments

None.

1.84.17 getCurrentItem()

Return current item index, or -1 if none.

Arguments

None.

1.84.18 getCursorItem()

Return index of item under cursor, or -1 if none.

Arguments

None.

1.84.19 getFont()

Return text font.

Arguments

None.

1.84.20 getHeader()

Return header control.

Arguments

None.

1.84.21 getHeaderIcon(...)

Return icon of header at index.

Argument	Type	Default	Description
index	Int		

1.84.22 getHeaderSize(...)

Return width of header at index.

Argument	Type	Default	Description
index	Int		

1.84.23 getHeaderText(...)

Return text of header at index.

Argument	Type	Default	Description
index	Int		

1.84.24 getHelpText()

Get the status line help text for this widget.

Arguments

None.

1.84.25 getItemAt(...)

Return index of item at x,y, or -1 if none.

Argument	Type	Default	Description
x	Int		
y	Int		

1.84.26 getItemBigIcon(...)

Return big icon of item at index.

Argument	Type	Default	Description
index	Int		

1.84.27 getItemData(...)

Return item user-data pointer.

Argument	Type	Default	Description
index	Int		

1.84.28 getItemHeight()

Return item height.

Arguments

None.

1.84.29 getItemMiniIcon(...)

Return mini icon of item at index.

Argument	Type	Default	Description
index	Int		

1.84.30 getItemSpace()

Return maximum item space.

Arguments

None.

1.84.31 getItemText(...)

Return item text.

Argument	Type	Default	Description
index	Int		

1.84.32 getItemWidth()

Return item width.

Arguments

None.

1.84.33 getListStyle()

Get the current icon list style.

Arguments

None.

1.84.34 getNumCols()

Return number of columns.

Arguments

None.

1.84.35 getNumHeaders()

Return number of headers.

Arguments

None.

1.84.36 getNumItems()

Return number of items.

Arguments

None.

1.84.37 getNumRows()

Return number of rows.

Arguments

None.

1.84.38 getSelBackColor()

Return selected text background.

Arguments

None.

1.84.39 getSelTextColor()

Return selected text color.

Arguments

None.

1.84.40 getSortFunc()

Return sort function.

Arguments

None.

1.84.41 getTextColor()

Return normal text color.

Arguments

None.

1.84.42 getViewportHeight()

Return viewport size.

Reimplemented from FXScrollArea.

Arguments

None.

1.84.43 hitItem(...)

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	

1.84.44 insertItem(...)

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	

1.84.45 insertItem(...)

Insert a new [possibly subclassed] item at the give index.

Argument	Type	Default	Description
index	Int		
item	FXIconItem		
notify	Bool	False	

1.84.46 isItemCurrent(...)

Return True if item at index is current.

Argument	Type	Default	Description
index	Int		

1.84.47 isItemEnabled(...)

Return True if item at index is enabled.

Argument	Type	Default	Description
index	Int		

1.84.48 isItemSelected(...)

Return True if item at index is selected.

Argument	Type	Default	Description
index	Int		

1.84.49 isItemVisible(...)

Return True if item at index is visible.

Argument	Type	Default	Description
index	Int		

1.84.50 killFocus()

Remove the focus from this window.
Reimplemented from FXWindow.

Arguments

None.

1.84.51 killSelection(...)

Deselect all items.

Argument	Type	Default	Description
notify	Bool	False	

1.84.52 makeItemVisible(...)

Scroll to make item at index visible.

Argument	Type	Default	Description
index	Int		

1.84.53 moveContents(...)

Move contents to the specified position.

Reimplemented from FXScrollArea.

Argument	Type	Default	Description
x	Int		
y	Int		

1.84.54 position(...)

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		

1.84.55 prependItem(...)

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	

1.84.56 prependItem(...)

Append a [possibly subclassed] item to the end of the list.

Argument	Type	Default	Description
item	FXIconItem		
notify	Bool	False	

1.84.57 recalc()

Recalculate layout.
Reimplemented from FXWindow.

Arguments

None.

1.84.58 removeHeader(...)

Remove header at index.

Argument	Type	Default	Description
index	Int		

1.84.59 removeItem(...)

Remove item from list.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.84.60 replaceItem(...)

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	

1.84.61 replaceItem(...)

Replace the item with a [possibly subclassed] item.

Argument	Type	Default	Description
index	Int		
item	FXIconItem		
notify	Bool	False	

1.84.62 resize(...)

Resize this window to the specified width and height.
Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

1.84.63 retrieveItem(...)

Return the item at the given index.

Argument	Type	Default	Description
index	Int		

1.84.64 selectInRectangle(...)

Select items in rectangle.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		
notify	Bool	False	

1.84.65 selectItem(...)

Select item at index.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.84.66 setAnchorItem(...)

Change anchor item index.

Argument	Type	Default	Description
index	Int		

1.84.67 setCurrentItem(...)

Change current item index.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.84.68 setFocus()

Move the focus to this window.
Reimplemented from FXWindow.

Arguments

None.

1.84.69 setFont(...)

Change text font.

Argument	Type	Default	Description
fnt	FXFont		

1.84.70 setHeaderIcon(...)

Change icon of header at index.

Argument	Type	Default	Description
index	Int		
icon	FXIcon		

1.84.71 setHeaderSize(...)

Change size of header at index.

Argument	Type	Default	Description
index	Int		
size	Int		

1.84.72 setHeaderText(...)

Change text of header at index.

Argument	Type	Default	Description
index	Int		
text	String		

1.84.73 setHelpText(...)

Set the status line help text for this widget.

Argument	Type	Default	Description
text	String		

1.84.74 setItemBigIcon(...)

Change item big icon.

Argument	Type	Default	Description
index	Int		
icon	FXIcon		

1.84.75 setItemData(...)

Change item user-data pointer.

Argument	Type	Default	Description
index	Int		
ptr	String		

1.84.76 setItemMiniIcon(...)

Change item mini icon.

Argument	Type	Default	Description
index	Int		
icon	FXIcon		

1.84.77 setItemSpace(...)

Change maximum item space for each item.

Argument	Type	Default	Description
s	Int		

1.84.78 setItemText(...)

Change item text.

Argument	Type	Default	Description
index	Int		
text	String		

1.84.79 setListStyle(...)

Set the current icon list style.

Argument	Type	Default	Description
style	Int		

1.84.80 setSelBackColor(...)

Change selected text background.

Argument	Type	Default	Description
clr	FXColor		

1.84.81 setSelTextColor(...)

Change selected text color.

Argument	Type	Default	Description
clr	FXColor		

1.84.82 setSortFunc(...)

Change sort function.

Argument	Type	Default	Description
func	FXIconListSortFunc		

1.84.83 setTextColor(...)

Change normal text color.

Argument	Type	Default	Description
clr	FXColor		

1.84.84 sortItems()

Sort items.

Arguments

None.

1.84.85 toggleItem(...)

Toggle item at index.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.84.86 updateItem(...)

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.

1.85 FXId

Encapsulates server side resource.

1.85.1 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.

Arguments

None.

1.85.2 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.

Arguments

None.

1.85.3 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.

Arguments

None.

1.85.4 getApp()

Get application.

Arguments

None.

1.85.5 getUserData()

Get user data pointer.

Arguments

None.

1.85.6 id()

Get XID handle.

Reimplemented in FXFont.

Arguments

None.

1.85.7 setUserData(...)

Set user data pointer.

Argument	Type	Default	Description
ptr	String		

1.86 FXImage

Image class

1.86.1 FXImage(...)

Create an image.

Argument	Type	Default	Description
a	FXApp		
pix		None	
opts	Int	0	
w	Int	1	
h	Int	1	

1.86.2 blend(...)

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	

1.86.3 create()

Create image resource.
Reimplemented from FXId.
Reimplemented in FXIcon.

Arguments

None.

1.86.4 destroy()

Destroy image resource.
Reimplemented from FXId.
Reimplemented in FXIcon.

Arguments

None.

1.86.5 detach()

Detach image resource.
Reimplemented from FXId.
Reimplemented in FXIcon.

Arguments

None.

1.86.6 getOptions()

To get to the option flags.

Arguments

None.

1.86.7 getPixel(...)

Get pixel at x,y.

Argument	Type	Default	Description
x	Int		
y	Int		

1.86.8 render()

Render the image from client-side pixel buffer.
Reimplemented in FXIcon.

Arguments

None.

1.86.9 resize(...)

Resize pixmap to the specified width and height.
Reimplemented from FXDrawable.
Reimplemented in FXIcon.

Argument	Type	Default	Description
w	Int		
h	Int		

1.86.10 scale(...)

Rescale pixels image to the specified width and height.

Argument	Type	Default	Description
w	Int		
h	Int		

1.86.11 setPixel(...)

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.

1.87 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.

1.87.1 FXLabel(...)

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	

1.87.2 create()

Create server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXMenuButton, FXOptionMenu, FXToggleButton, and AFXFlyoutButton.

Arguments

None.

1.87.3 detach()

Detach server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXMenuButton, FXOptionMenu, FXToggleButton, and AFXFlyoutButton.

Arguments

None.

1.87.4 disable()

Disable the window.

Reimplemented from FXWindow.
Reimplemented in AFXFlyoutButton.

Arguments

None.

1.87.5 enable()

Enable the window.
Reimplemented from FXWindow.
Reimplemented in AFXFlyoutButton.

Arguments

None.

1.87.6 getDefaultHeight()

Return default height.
Reimplemented from FXFrame.
Reimplemented in FXCheckButton, FXMDIDeleteButton, FXMDIRestoreButton, FXMDIMaximizeButton, FXMDIMinimizeButton, FXMDIWindowButton, FXMenuButton, FXOption, FXOptionMenu, FXRadioButton, and FXToggleButton.

Arguments

None.

1.87.7 getDefaultWidth()

Return default width.
Reimplemented from FXFrame.
Reimplemented in FXCheckButton, FXMDIDeleteButton, FXMDIRestoreButton, FXMDIMaximizeButton, FXMDIMinimizeButton, FXMDIWindowButton, FXMenuButton, FXOption, FXOptionMenu, FXRadioButton, and FXToggleButton.

Arguments

None.

1.87.8 getFont()

Get the text font.

Arguments

None.

1.87.9 getHelpText()

Get the status line help text for this label.

Arguments

None.

1.87.10 getIcon()

Get the icon for this label.

Arguments

None.

1.87.11 getIconPosition()

Get the current icon position.

Arguments

None.

1.87.12 getJustify()

Get the current text-justification mode.

Arguments

None.

1.87.13 getText()

Get the text for this label.

Arguments

None.

1.87.14 getTextColor()

Get the current text color.

Arguments

None.

1.87.15 getTipText()

Get the tool tip message for this label.

Arguments

None.

1.87.16 setFont(...)

Set the text font.

Argument	Type	Default	Description
fnt	FXFont		

1.87.17 setHelpText(...)

Set the status line help text for this label.
Reimplemented in AFXFlyoutItem.

Argument	Type	Default	Description
text	String		

1.87.18 setIcon(...)

Set the icon for this label.
Reimplemented in AFXFlyoutItem.

Argument	Type	Default	Description
ic	FXIcon		

1.87.19 setIconPosition(...)

Set the current icon position.

Argument	Type	Default	Description
mode	Int		

1.87.20 setJustify(...)

Set the current text-justification mode.

Argument	Type	Default	Description
mode	Int		

1.87.21 setText(...)

Set the text for this label.
Reimplemented in AFXFlyoutItem.

Argument	Type	Default	Description
text	String		

1.87.22 setTextColor(...)

Set the current text color.

Argument	Type	Default	Description
clr	FXColor		

1.87.23 setTipText(...)

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.

1.88 FXList

List Widget

1.88.1 FXList(...)

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	

1.88.2 appendItem(...)

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	

1.88.3 appendItem(...)

Append a [possibly subclassed] item to the list.

Argument	Type	Default	Description
item	FXListItem		
notify	Bool	False	

1.88.4 canFocus()

List widget can receive focus.
Reimplemented from FXWindow.

Arguments

None.

1.88.5 clearItems(...)

Remove all items from list.

Argument	Type	Default	Description
notify	Bool	False	

1.88.6 create()

Create server-side resources.
Reimplemented from FXComposite.

Arguments

None.

1.88.7 deselectItem(...)

Deselect item.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.88.8 detach()

Detach server-side resources.

Reimplemented from FXComposite.

Arguments

None.

1.88.9 findItem(...)

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	

1.88.10 getContentHeight()

Return content height.

Reimplemented from FXScrollArea.

Arguments

None.

1.88.11 getContentWidth()

Compute and return content width.

Reimplemented from FXScrollArea.

Arguments

None.

1.88.12 getCurrentItem()

Return current item, if any.

Arguments

None.

1.88.13 getDefaultHeight()

Return default height.
Reimplemented from FXScrollArea.
Reimplemented in AFXList.

Arguments

None.

1.88.14 getDefaultWidth()

Return default width.
Reimplemented from FXScrollArea.

Arguments

None.

1.88.15 getItemData(...)

Return item user-data pointer.

Argument	Type	Default	Description
index	Int		

1.88.16 getItemIcon(...)

Return item icon, if any.

Argument	Type	Default	Description
index	Int		

1.88.17 getItemText(...)

Return item text.

Argument	Type	Default	Description
index	Int		

1.88.18 getListStyle()

Return list style.

Arguments

None.

1.88.19 getNumItems()

Return the number of items in the list.

Arguments

None.

1.88.20 getNumVisible()

Return number of visible items.

Arguments

None.

1.88.21 insertItem(...)

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	

1.88.22 insertItem(...)

Insert a new [possibly subclassed] item at the give index.

Argument	Type	Default	Description
index	Int		
item	FXListItem		
notify	Bool	False	

1.88.23 isItemSelected(...)

Return True if item is selected.

Argument	Type	Default	Description
index	Int		

1.88.24 isItemVisible(...)

Return True if item is visible.

Argument	Type	Default	Description
index	Int		

1.88.25 killSelection(...)

Deselect all items.

Argument	Type	Default	Description
notify	Bool	False	

1.88.26 makeItemVisible(...)

Scroll to bring item into view.

Argument	Type	Default	Description
index	Int		

1.88.27 recalc()

Recalculate layout.
Reimplemented from FXWindow.

Arguments

None.

1.88.28 removeItem(...)

Remove item from list.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.88.29 replaceItem(...)

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	

1.88.30 replaceItem(...)

Replace the item with a [possibly subclassed] item.

Argument	Type	Default	Description
index	Int		
item	FXListItem		
notify	Bool	False	

1.88.31 retrieveItem(...)

Return the item at the given index.

Argument	Type	Default	Description
index	Int		

1.88.32 selectItem(...)

Select item.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.88.33 setCurrentItem(...)

Change current item.

Argument	Type	Default	Description
index	Int		
notify	Bool	False	

1.88.34 setItemData(...)

Change item user-data pointer.

Argument	Type	Default	Description
index	Int		
ptr	String		

1.88.35 setItemIcon(...)

Change item icon.

Argument	Type	Default	Description
index	Int		
icon	FXIcon		

1.88.36 setItemText(...)

Change item text.

Argument	Type	Default	Description
index	Int		
text	String		

1.88.37 setListStyle(...)

Change list style.

Argument	Type	Default	Description
style	Int		

1.88.38 setNumVisible(...)

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.

1.89 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.

1.90 FXMainWindow

Main application window.

1.90.1 FXMainWindow(...)

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	

Argument	Type	Default	Description
h	Int	0	
pl	Int	0	
pr	Int	0	
pt	Int	0	
pb	Int	0	
hs	Int	0	
vs	Int	0	

1.91 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.

1.91.1 FXMatrix(...)

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	

Argument	Type	Default	Description
pb	Int	DEFAULT_SPACING	
hs	Int	DEFAULT_SPACING	
vs	Int	DEFAULT_SPACING	

1.91.2 getDefaultHeight()

Return default height.
Reimplemented from FXPacker.

Arguments

None.

1.91.3 getDefaultWidth()

Return default width.
Reimplemented from FXPacker.

Arguments

None.

1.91.4 getNumColumns()

Return the number of columns.

Arguments

None.

1.91.5 getNumRows()

Return the number of rows.

Arguments

None.

1.91.6 setNumColumns(...)

Change the number of columns.

Argument	Type	Default	Description
nc	Int		

1.91.7 setNumRows(...)

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.

1.92 FXMDIChild

The MDI child window contains the application work area in a Multiple Document Interface application.

1.92.1 FXMDIChild(...)

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	

1.92.2 canFocus()

MDI Child can receive focus.
Reimplemented from FXWindow.

Arguments

None.

1.92.3 contentWindow()

Return content window.

Arguments

None.

1.92.4 create()

Create window.
Reimplemented from FXComposite.

Arguments

None.

1.92.5 detach()

Detach window.
Reimplemented from FXComposite.

Arguments

None.

1.92.6 getDefaultHeight()

Return default height.
Reimplemented from FXComposite.

Arguments

None.

1.92.7 getDefaultWidth()

Compute default size.
Reimplemented from FXComposite.

Arguments

None.

1.92.8 getFont()

Get title font.

Arguments

None.

1.92.9 getHiliteColor()

Get colors.

Arguments

None.

1.92.10 getIconX()

Return iconified position.

Arguments

None.

1.92.11 getMDINext()

Get next MDI Child.

Arguments

None.

1.92.12 getMDIPrev()

Get previous MDI Child.

Arguments

None.

1.92.13 getNormalX()

Return normal (restored) position.

Arguments

None.

1.92.14 getTitle()

Get current title.

Arguments

None.

1.92.15 getWindowIcon()

Get window icon.

Arguments

None.

1.92.16 getWindowMenu()

Get window menu.

Arguments

None.

1.92.17 isMaximized()

Return True if maximized.

Arguments

None.

1.92.18 isMinimized()

Return True if minimized.

Arguments

None.

1.92.19 maximize(...)

Maximize MDI Child.

Argument	Type	Default	Description
notify	Bool	False	

1.92.20 minimize(...)

Minimize/iconify MDI Child.

Argument	Type	Default	Description
notify	Bool	False	

1.92.21 move(...)

Move this window to the specified position in the parent's coordinates.
Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		

1.92.22 position(...)

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		

1.92.23 resize(...)

Resize this window to the specified width and height.
Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

1.92.24 restore(...)

Restore MDI Child to normal.

Argument	Type	Default	Description
notify	Bool	False	

1.92.25 setFont(...)

Set title font.

Argument	Type	Default	Description
fnt	FXFont		

1.92.26 setHiliteColor(...)

Change colors.

Argument	Type	Default	Description
clr	FXColor		

1.92.27 setIconX(...)

Change iconified position.

Argument	Type	Default	Description
x	Int		

1.92.28 setNormalX(...)

Change normal (restored) position.

Argument	Type	Default	Description
x	Int		

1.92.29 setTitle(...)

Change MDI Child's title.

Argument	Type	Default	Description
name	String		

1.92.30 setWindowIcon(...)

Set window icon.

Argument	Type	Default	Description
icon	FXIcon		

1.92.31 setWindowMenu(...)

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.

1.93 FXMenubar

Menu bar.

1.93.1 FXMenubar(...)

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		

Argument	Type	Default	Description
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	

1.93.2 FXMenubar(...)

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	

1.94 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.

1.94.1 FXMenuButton(...)

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	

1.94.2 canFocus()

Returns True because a menu button can receive focus.

Reimplemented from FXWindow.

Arguments

None.

1.94.3 create()

Create server-side resources.
Reimplemented from FXLabel.

Arguments

None.

1.94.4 detach()

Detach server-side resources.
Reimplemented from FXLabel.

Arguments

None.

1.94.5 getAttachment()

Get attachment.

Arguments

None.

1.94.6 getButtonStyle()

Get menu button style.

Arguments

None.

1.94.7 getDefaultHeight()

Return default height.
Reimplemented from FXLabel.
Reimplemented in FXMDIWindowButton.

Arguments

None.

1.94.8 getDefaultWidth()

Return default width.

Reimplemented from FXLabel.

Reimplemented in FXMDIWindowButton.

Arguments

None.

1.94.9 getMenu()

Return current popup menu.

Arguments

None.

1.94.10 getPopupStyle()

Get popup style.

Arguments

None.

1.94.11 getXOffset()

Return current X offset.

Arguments

None.

1.94.12 getYOffset()

Return current Y offset.

Arguments

None.

1.94.13 setAttachment(...)

Change attachment.

Argument	Type	Default	Description
att	Int		

1.94.14 setButtonStyle(...)

Change menu button style.

Argument	Type	Default	Description
style	Int		

1.94.15 setMenu(...)

Change the popup menu.

Argument	Type	Default	Description
pup	FXPopup		

1.94.16 setPopupStyle(...)

Change popup style.

Argument	Type	Default	Description
style	Int		

1.94.17 setXOffset(...)

Set X offset where menu pops up relative to button.

Argument	Type	Default	Description
offx	Int		

1.94.18 setYOffset(...)

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.

1.95 FXMenuCaption

The menu caption is a widget which can be used as a caption above a number of menu commands in a menu.

1.95.1 FXMenuCaption(...)

Construct a menu caption.

Argument	Type	Default	Description
p	FXComposite		
text	String		

Argument	Type	Default	Description
ic	FXIcon	None	
opts	Int	0	

1.95.2 create()

Create server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXMenuCascade, and FXMenuTitle.

Arguments

None.

1.95.3 detach()

Detach server-side resources.

Reimplemented from FXWindow.

Reimplemented in FXMenuCascade, and FXMenuTitle.

Arguments

None.

1.95.4 disable()

Disable the menu.

Reimplemented from FXWindow.

Arguments

None.

1.95.5 enable()

Enable the menu.

Reimplemented from FXWindow.

Arguments

None.

1.95.6 getDefaultHeight()

Return default height.

Reimplemented from FXWindow.

Reimplemented in FXMenuCommand, and FXMenuTitle.

Arguments

None.

1.95.7 getDefaultWidth()

Return default width.

Reimplemented from FXWindow.

Reimplemented in FXMenuCommand, and FXMenuTitle.

Arguments

None.

1.95.8 getFont()

Return the text font.

Arguments

None.

1.95.9 getIcon()

Get the icon for this menu.

Arguments

None.

1.95.10 getText()

Get the text for this menu.

Arguments

None.

1.95.11 getTextColor()

Get the current text color.

Arguments

None.

1.95.12 setFont(...)

Set the text font.

Argument	Type	Default	Description
fnt	FXFont		

1.95.13 setIcon(...)

Set the icon for this menu.

Argument	Type	Default	Description
ic	FXIcon		

1.95.14 setText(...)

Set the text for this menu.

Argument	Type	Default	Description
text	String		

1.95.15 setTextColor(...)

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.

1.96 FXMenuCascade

The cascade menu widget is used to bring up a sub menu from a pull down menu.

1.96.1 FXMenuCascade(...)

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	

1.96.2 canFocus()

Yes it can receive the focus.
Reimplemented from FXWindow.

Arguments

None.

1.96.3 create()

Create server-side resources.
Reimplemented from FXMenuCaption.

Arguments

None.

1.96.4 destroy()

Destroy server-side resources.
Reimplemented from FXWindow.
Reimplemented in AFXManagerMenuCascade.

Arguments

None.

1.96.5 detach()

Detach server-side resources.
Reimplemented from FXMenuCaption.

Arguments

None.

1.96.6 getMenu()

Return popup menu.

Arguments

None.

1.96.7 setMenu(...)

Set popup menu to pop up.

Argument	Type	Default	Description
pup	FXPopup		

1.97 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.

1.97.1 FXMenuCommand(...)

Construct a menu command.

Argument	Type	Default	Description
p	FXComposite		
text	String		
ic	FXIcon	None	
tgt	FXObject	None	
sel	Int	0	
opts	Int	0	

1.97.2 canFocus()

Yes it can receive the focus.
Reimplemented from FXWindow.

Arguments

None.

1.97.3 check()

Place checkmark next to text.

Arguments

None.

1.97.4 checkRadio()

Place radio bullit next to text.

Arguments

None.

1.97.5 getAccelText()

Return accelarator text.

Arguments

None.

1.97.6 getDefaultHeight()

Return default height.
Reimplemented from FXMenuCaption.

Arguments

None.

1.97.7 getDefaultWidth()

Return default width.

Reimplemented from FXMenuCaption.

Arguments

None.

1.97.8 isChecked()

Return True if checked.

Arguments

None.

1.97.9 isRadioChecked()

Return True if radio-checked.

Arguments

None.

1.97.10 setAccelText(...)

Set accelerator text.

Argument	Type	Default	Description
text	String		

1.97.11 uncheck()

Uncheck the item.

Arguments

None.

1.97.12 uncheckRadio()

Uncheck radio bullit.

Arguments

None.

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.

1.98 FXMenuSeparator

The menu separator is a simple decorative groove used to delineate items in a popup menu.

1.98.1 FXMenuSeparator(...)

Construct a menu separator.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	

1.98.2 getDefaultHeight()

Return default height.
Reimplemented from FXWindow.

Arguments

None.

1.98.3 getDefaultWidth()

Return default width.
Reimplemented from FXWindow.

Arguments

None.

1.99 FXMenuTitle

A menu title is a child of a menu bar which is responsible for popping up a pulldown menu.

1.99.1 FXMenuTitle(...)

Constructor.

Argument	Type	Default	Description
p	FXComposite		
text	String		
ic	FXIcon	None	
pup	FXPopup	None	
opts	Int	0	

1.99.2 canFocus()

Yes it can receive the focus.
Reimplemented from FXWindow.

Arguments

None.

1.99.3 create()

Create server-side resources.
Reimplemented from FXMenuCaption.

Arguments

None.

1.99.4 detach()

Detach server-side resources.
Reimplemented from FXMenuCaption.

Arguments

None.

1.99.5 getDefaultHeight()

Return default height.
Reimplemented from FXMenuCaption.

Arguments

None.

1.99.6 getDefaultWidth()

Return default width.
Reimplemented from FXMenuCaption.

Arguments

None.

1.99.7 getMenu()

Return popup menu.

Arguments

None.

1.99.8 setMenu(...)

Set popup menu to pop up.

Argument	Type	Default	Description
menu	FXPopup		

1.100 FXObject

Base of all FOX object.

1.100.1 getClassName()

Get class name of some object.

Arguments

None.

1.100.2 isMemberOf(...)

Check if object is member of metaclass.

Argument	Type	Default	Description
metaclass	FXMetaClass		

1.100.3 onDefault()

Called for unhandled messages.

Reimplemented in FXDelegator, FXGLViewer, FXMDIChild, and FXMDIClient.

Arguments

None.

1.100.4 handle(...)

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.

1.101 FXObjectList

List of pointers to objects.

1.101.1 FXObjectList(...)

Default constructor.

Argument	Type	Default	Description

1.101.2 FXObjectList(...)

Copy constructor.

Argument	Type	Default	Description
orig	FXObjectList		

1.101.3 append(...)

Append element.

Argument	Type	Default	Description
p	FXObject		

1.101.4 clear()

Remove all elements.

Arguments

None.

1.101.5 findb(...)

Find object in list, searching backward; return position or -1.

Argument	Type	Default	Description
p	FXObject		
pos	Int	2147483647	

1.101.6 findf(...)

Find object in list, searching forward; return position or -1.

Argument	Type	Default	Description
p	FXObject		
pos	Int	0	

1.101.7 insert(...)

Insert element at certain position.

Argument	Type	Default	Description
pos	Int		
p	FXObject		

1.101.8 no(...)

Set number of elements.

Argument	Type	Default	Description
n	Int		

1.101.9 no()

Return number of elements.

Arguments

None.

1.101.10 remove(...)

Remove element p.

Argument	Type	Default	Description
p	FXObject		

1.101.11 remove(...)

Remove element at pos.

Argument	Type	Default	Description
pos	Int		

1.101.12 size(...)

Set max number of elements.

Argument	Type	Default	Description
m	Int		

1.101.13 size()

Return size of list.

Arguments

None.

1.102 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.

1.102.1 FXPacker(...)

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	

1.102.2 getBaseColor()

Get base gui color.

Arguments

None.

1.102.3 getBorderColor()

Get border color.

Arguments

None.

1.102.4 getBorderWidth()

Get border width.

Arguments

None.

1.102.5 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.

Arguments

None.

1.102.6 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.

Arguments

None.

1.102.7 getFrameStyle()

Get current frame style.

Arguments

None.

1.102.8 getHiliteColor()

Get highlight color.

Arguments

None.

1.102.9 getHSpacing()

Return current horizontal inter-child spacing.

Arguments

None.

1.102.10 getPackingHints()

Return packing hints.

Arguments

None.

1.102.11 getPadBottom()

Get bottom interior padding.

Arguments

None.

1.102.12 getPadLeft()

Get left interior padding.

Arguments

None.

1.102.13 getPadRight()

Get right interior padding.

Arguments

None.

1.102.14 getPadTop()

Get top interior padding.

Arguments

None.

1.102.15 getShadowColor()

Get shadow color.

Arguments

None.

1.102.16 getVSpacing()

Return current vertical inter-child spacing.

Arguments

None.

1.102.17 setBaseColor(...)

Change base gui color.

Argument	Type	Default	Description
clr	FXColor		

1.102.18 setBorderColor(...)

Change border color.

Argument	Type	Default	Description
clr	FXColor		

1.102.19 setFrameStyle(...)

Change frame style.

Argument	Type	Default	Description
style	Int		

1.102.20 setHiliteColor(...)

Change highlight color.

Argument	Type	Default	Description
clr	FXColor		

1.102.21 setHSpacing(...)

Change horizontal inter-child spacing.

Argument	Type	Default	Description
hs	Int		

1.102.22 setPackingHints(...)

Change packing hints.

Argument	Type	Default	Description
ph	Int		

1.102.23 setPadBottom(...)

Change bottom padding.

Argument	Type	Default	Description
pb	Int		

1.102.24 setPadLeft(...)

Change left padding.

Argument	Type	Default	Description
pl	Int		

1.102.25 setPadRight(...)

Change right padding.

Argument	Type	Default	Description
pr	Int		

1.102.26 setPadTop(...)

Change top padding.

Argument	Type	Default	Description
pt	Int		

1.102.27 setShadowColor(...)

Change shadow color.

Argument	Type	Default	Description
clr	FXColor		

1.102.28 setVSpacing(...)

Change vertical inter-child spacing.

Argument	Type	Default	Description
vs	Int		

1.103 FXPNGIcon

Portable Network Graphics (PNG) Icon class.

1.103.1 FXPNGIcon(...)

Construct an icon from memory stream formatted in PNG format.

Argument	Type	Default	Description
a	FXApp		
pix		None	

Argument	Type	Default	Description
clr	FXColor	FXRGB(192, 192, 192)	
opts	Int	0	
w	Int	1	
h	Int	1	

1.104 FXPopup

Popup window

1.104.1 FXPopup(...)

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	

1.104.2 getBaseColor()

Return base color.

Arguments

None.

1.104.3 getBorderColor()

Return border color.

Arguments

None.

1.104.4 getBorderWidth()

Return border width.

Arguments

None.

1.104.5 getDefaultHeight()

Return the default height of this window.
Reimplemented from FXComposite.

Arguments

None.

1.104.6 getDefaultWidth()

Return the default width of this window.
Reimplemented from FXComposite.

Arguments

None.

1.104.7 getFrameStyle()

Return frame style.

Arguments

None.

1.104.8 getGrabOwner()

Return current grab owner.

Arguments

None.

1.104.9 getHiliteColor()

Return highlight color.

Arguments

None.

1.104.10 getOrientation()

Return popup orientation.

Arguments

None.

1.104.11 getShadowColor()

Return shadow color.

Arguments

None.

1.104.12 getShrinkWrap()

Return shrinkwrap mode.

Arguments

None.

1.104.13 popdown()

Pop down the menu.

Arguments

None.

1.104.14 popup(...)

Popup the menu and grab to the given owner.

Argument	Type	Default	Description
grabto	FXWindow		
x	Int		
y	Int		
w	Int	0	

Argument	Type	Default	Description
h	Int	0	

1.104.15 setBaseColor(...)

Change base color.

Argument	Type	Default	Description
clr	FXColor		

1.104.16 setBorderColor(...)

Change border color.

Argument	Type	Default	Description
clr	FXColor		

1.104.17 setFrameStyle(...)

Change frame style.

Argument	Type	Default	Description
style	Int		

1.104.18 setHiliteColor(...)

Change highlight color.

Argument	Type	Default	Description
clr	FXColor		

1.104.19 setOrientation(...)

Change popup orientation.

Argument	Type	Default	Description
orient	Int		

1.104.20 setShadowColor(...)

Change shadow color.

Argument	Type	Default	Description
clr	FXColor		

1.104.21 setShrinkWrap(...)

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.

1.105 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.

1.105.1 FXRadioButton(...)

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	

Argument	Type	Default	Description
pl	Int	DEFAULT_PAD	
pr	Int	DEFAULT_PAD	
pt	Int	DEFAULT_PAD	
pb	Int	DEFAULT_PAD	

1.105.2 canFocus()

Returns True because a radio button can receive focus.
Reimplemented from FXWindow.

Arguments

None.

1.105.3 getCheck()

Get radio button state (True, False or MAYBE).

Arguments

None.

1.105.4 getDefaultHeight()

Get default height.
Reimplemented from FXLabel.

Arguments

None.

1.105.5 getDefaultWidth()

Get default width.
Reimplemented from FXLabel.

Arguments

None.

1.105.6 setCheck(...)

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.

1.106 FXRootWindow

Root window.

1.106.1 FXRootWindow(...)

Construct root window.

Argument	Type	Default	Description
a	FXApp		
vis	FXVisual		

1.106.2 create()

Root window need not be created.
Reimplemented from FXComposite.

Arguments

None.

1.106.3 destroy()

Root window can not be destroyed.
Reimplemented from FXComposite.

Arguments

None.

1.106.4 detach()

Root window may not be detached.

Reimplemented from FXComposite.

Arguments

None.

1.106.5 getDefaultHeight()

Return height of the root window.
Reimplemented from FXComposite.

Arguments

None.

1.106.6 getDefaultWidth()

Return width of the root window.
Reimplemented from FXComposite.

Arguments

None.

1.106.7 move(...)

Root window can not be moved.
Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		

1.106.8 position(...)

Root window can not be positioned.
Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		
w	Int		
h	Int		

1.106.9 recalc()

No op.
Reimplemented from FXWindow.

Arguments

None.

1.106.10 resize(...)

Root window can not be resized.
Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

1.107 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.

1.107.1 getContentWidth()

Return content size.
Reimplemented in FXIconList, FXImageView, FXList, FXMDIClient, FXScrollWindow, FXTable, FXText, FXTreeList, AFXBaseTable, and AFXOptionTreeList.

Arguments

None.

1.107.2 getDefaultHeight()

Return default height.

Reimplemented from FXComposite.

Reimplemented in FXList, FXTable, FXText, FXTreeList, AFXBaseTable, AFXList, AFXOptionTreeList, AFXTable, and AFXTreeTable.

Arguments

None.

1.107.3 getDefaultWidth()

Return default width.

Reimplemented from FXComposite.

Reimplemented in FXList, FXTable, FXText, FXTreeList, AFXBaseTable, AFXOptionTreeList, AFXTable, and AFXTreeTable.

Arguments

None.

1.107.4 getPosition(...)

Get the current position.

Argument	Type	Default	Description
x	Int		
y	Int		

1.107.5 getScrollStyle()

Return scroll style.

Arguments

None.

1.107.6 getViewportHeight()

Return viewport size.

Reimplemented in FXIconList.

Arguments

None.

1.107.7 getXPosition()

Return the current x-position.

Arguments

None.

1.107.8 getYPosition()

Return the current y-position.

Arguments

None.

1.107.9 horizontalScrollbar()

Return a pointer to the horizontal scrollbar.

Arguments

None.

1.107.10 isHorizontalScrollable()

Return True if horizontally scrollable.

Arguments

None.

1.107.11 isVerticalScrollable()

Return True if vertically scrollable.

Arguments

None.

1.107.12 moveContents(...)

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		

1.107.13 setPosition(...)

Set the current position.

Argument	Type	Default	Description
x	Int		
y	Int		

1.107.14 setScrollStyle(...)

Change scroll style.

Argument	Type	Default	Description
style	Int		

1.107.15 verticalScrollbar()

Return a pointer to the vertical scrollbar.

Arguments

None.

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.

1.108 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.

1.108.1 FXScrollbar(...)

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	

1.108.2 getBorderColor()

Change border color.

Arguments

None.

1.108.3 getDefaultHeight()

Return default height.

Reimplemented from FXWindow.

Arguments

None.

1.108.4 getDefaultWidth()

Return default width.
Reimplemented from FXWindow.

Arguments

None.

1.108.5 getHiliteColor()

Return highlight color.

Arguments

None.

1.108.6 getLine()

Return line increment.

Arguments

None.

1.108.7 getPage()

Return page size.

Arguments

None.

1.108.8 getPosition()

return scroll position

Arguments

None.

1.108.9 getRange()

Return content size range.

Arguments

None.

1.108.10 getScrollbarStyle()

Change the scrollbar style.

Arguments

None.

1.108.11 getShadowColor()

Return shadow color.

Arguments

None.

1.108.12 setBorderColor(...)

Return border color.

Argument	Type	Default	Description
clr	FXColor		

1.108.13 setHiliteColor(...)

Change highlight color.

Argument	Type	Default	Description
clr	FXColor		

1.108.14 setLine(...)

Set scoll increment for line.

Argument	Type	Default	Description
l	Int		

1.108.15 setPage(...)

Set viewport page size.

Argument	Type	Default	Description
p	Int		

1.108.16 setPosition(...)

Change current scroll position.

Argument	Type	Default	Description
p	Int		
notifyTgt	Bool	False	

1.108.17 setRange(...)

Set content size range.

Argument	Type	Default	Description
r	Int		

1.108.18 setScrollbarStyle(...)

Get the current scrollbar style.

Argument	Type	Default	Description
style	Int		

1.108.19 setShadowColor(...)

Change shadow color.

Argument	Type	Default	Description
clr	FXColor		

Global flags

Scrollbar styles

SCROLLBAR_HORIZONTAL	Horizontally oriented.
SCROLLBAR_VERTICAL	Vertically oriented.

1.109 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.

1.109.1 FXScrollWindow(...)

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	

1.109.2 contentWindow()

Return a pointer to the contents window.

Arguments

None.

1.109.3 create()

Create server-side resources.
Reimplemented from FXComposite.

Arguments

None.

1.109.4 getContentHeight()

Return the height of the contents.
Reimplemented from FXScrollArea.
Reimplemented in AFXOptionTreeList.

Arguments

None.

1.109.5 getContentWidth()

Return the width of the contents.
Reimplemented from FXScrollArea.
Reimplemented in AFXOptionTreeList.

Arguments

None.

1.109.6 moveContents(...)

Move contents to the specified position.
Reimplemented from FXScrollArea.
Reimplemented in AFXOptionTreeList.

Argument	Type	Default	Description
x	Int		
y	Int		

1.110 FXShell

A child of the Root window.

1.110.1 create()

Create server-side resources.
Reimplemented from FXComposite.
Reimplemented in FXPrintDialog, FXToolbarShell, FXTooltip, FXTopWindow, AFXManagerMenuPane, AFXMainWindow, and AFXDialog.

Arguments

None.

1.110.2 recalc()

Mark this window's layout as dirty.

Reimplemented from FXWindow.

Arguments

None.

1.111 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.

1.111.1 FXSplitter(...)

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	

1.111.2 FXSplitter(...)

Construct new splitter widget, which will notify target about size changes.

Argument	Type	Default	Description
p	FXComposite		
tgt	FXObject		
sel	Int		

Argument	Type	Default	Description
opts	Int	SPLITTER_NORMAL	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

1.111.3 getDefaultHeight()

Get default height.

Reimplemented from FXComposite.

Arguments

None.

1.111.4 getDefaultWidth()

Get default width.

Reimplemented from FXComposite.

Arguments

None.

1.111.5 getSplitterStyle()

Return current splitter style.

Arguments

None.

1.111.6 setSplitterStyle(...)

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.

1.112 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.

1.112.1 FXSwitcher(...)

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	

1.112.2 getCurrent()

Return the index of the child window currently on top.

Arguments

None.

1.112.3 getDefaultHeight()

Return default height.
Reimplemented from FXPacker.

Arguments

None.

1.112.4 getDefaultWidth()

Return default width.
Reimplemented from FXPacker.

Arguments

None.

1.112.5 setCurrent(...)

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.

1.113 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.

1.113.1 FXTabBar(...)

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	

1.113.2 create()

Create all of the server-side resources for this window // CAE.
Reimplemented from FXComposite.

Arguments

None.

1.113.3 getCurrent()

Return the currently active tab item.

Arguments

None.

1.113.4 getDefaultHeight()

Return default height.

Reimplemented from FXPacker.
Reimplemented in FXTabBook.

Arguments

None.

1.113.5 getDefaultWidth()

Return default width.

Reimplemented from FXPacker.
Reimplemented in FXTabBook.

Arguments

None.

1.113.6 getTabStyle()

Return tab bar style.

Arguments

None.

1.113.7 setCurrent(...)

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	

1.113.8 setTabStyle(...)

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.

1.114 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.

1.114.1 FXTabBook(...)

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	

1.114.2 getDefaultHeight()

Return default height.
Reimplemented from FXTabBar.

Arguments

None.

1.114.3 getDefaultWidth()

Return default width.
Reimplemented from FXTabBar.

Arguments

None.

1.115 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.

1.115.1 FXTabItem(...)

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	

1.115.2 canFocus()

Returns True because a tab item can receive focus.
Reimplemented from FXWindow.

Arguments

None.

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.

1.116 FXText

Multiline text widget

1.116.1 FXText(...)

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	

1.116.2 appendText(...)

Append n characters of text at the end of the buffer.

Argument	Type	Default	Description
text	String		
n	Int		
notify	Bool	False	

1.116.3 canFocus()

Returns True because a text widget can receive focus.
Reimplemented from FXWindow.

Arguments

None.

1.116.4 create()

Create server-side resources.
Reimplemented from FXComposite.

Arguments

None.

1.116.5 detach()

Detach server-side resources.
Reimplemented from FXComposite.

Arguments

None.

1.116.6 disable()

Disable the text widget.
Reimplemented from FXWindow.

Arguments

None.

1.116.7 enable()

Enable the text widget.
Reimplemented from FXWindow.

Arguments

None.

1.116.8 extractText(...)

Extract n characters of text from position pos.

Argument	Type	Default	Description
text	String		
pos	Int		
n	Int		

1.116.9 getBarColor()

Return bar color.

Arguments

None.

1.116.10 getBarColumns()

Return number of columns used for line numbers.

Arguments

None.

1.116.11 getChar(...)

Get character at position in text buffer.

Argument	Type	Default	Description
pos	Int		

1.116.12 getContentHeight()

Get default height.
Reimplemented from FXScrollArea.

Arguments

None.

1.116.13 getContentWidth()

Get default width.
Reimplemented from FXScrollArea.

Arguments

None.

1.116.14 getCursorCol()

Return cursor row, i.e. indent position.

Arguments

None.

1.116.15 getCursorColor()

Return cursor color.

Arguments

None.

1.116.16 getCursorPos()

Return the cursor position.

Arguments

None.

1.116.17 getCursorRow()

Return cursor row.

Arguments

None.

1.116.18 getDefaultHeight()

Return default height.
Reimplemented from FXScrollArea.

Arguments

None.

1.116.19 getDefaultWidth()

Return default width.
Reimplemented from FXScrollArea.

Arguments

None.

1.116.20 getFont()

Return text font.

Arguments

None.

1.116.21 getLength()

Return length of buffer.

Arguments

None.

1.116.22 getMarginBottom()

Return bottom margin.

Arguments

None.

1.116.23 getMarginLeft()

Return left margin.

Arguments

None.

1.116.24 getMarginRight()

Return right margin.

Arguments

None.

1.116.25 getMarginTop()

Return top margin.

Arguments

None.

1.116.26 getNumberColor()

Return line number color.

Arguments

None.

1.116.27 getPosAt(...)

Return text position at given visible x,y coordinate.

Argument	Type	Default	Description
x	Int		
y	Int		

1.116.28 getSelBackColor()

Return selected background color.

Arguments

None.

1.116.29 getSelEndPos()

Return selendpos.

Arguments

None.

1.116.30 getSelStartPos()

Return selstartpos.

Arguments

None.

1.116.31 getSelTextColor()

Return selected text color.

Arguments

None.

1.116.32 getText()

Return text in the widget.

Arguments

None.

1.116.33 getText(...)

Retrieve text into buffer.

Argument	Type	Default	Description
text	String		
n	Int		

1.116.34 getTextColor()

Return text color.

Arguments

None.

1.116.35 getTopLine()

Return position of top line.

Arguments

None.

1.116.36 getVisCols()

Return number of visible columns.

Arguments

None.

1.116.37 getVisRows()

Return number of visible rows.

Arguments

None.

1.116.38 insertText(...)

Insert n characters of text at position pos into the buffer.

Argument	Type	Default	Description
pos	Int		
text	String		
n	Int		
notify	Bool	False	

1.116.39 isEditable()

Return True if text is editable.

Arguments

None.

1.116.40 isModified()

Return True if text was modified.

Arguments

None.

1.116.41 isPosSelected(...)

Return True if position pos is selected.

Argument	Type	Default	Description
pos	Int		

1.116.42 killSelection(...)

Unselect the text.

Argument	Type	Default	Description
notify	Bool	False	

1.116.43 lineEnd(...)

Return position of end of line containing position pos.

Argument	Type	Default	Description
pos	Int		

1.116.44 lineStart(...)

Return position of begin of line containing position pos.

Argument	Type	Default	Description
pos	Int		

1.116.45 makePositionVisible(...)

Scroll text to make the given position visible.

Argument	Type	Default	Description
pos	Int		

1.116.46 moveContents(...)

Scroll the contents.

Reimplemented from FXScrollArea.

Argument	Type	Default	Description
x	Int		
y	Int		

1.116.47 nextLine(...)

Return start of next line.

Argument	Type	Default	Description
pos	Int		
nl	Int	1	

1.116.48 nextRow(...)

Return start of next row.

Argument	Type	Default	Description
pos	Int		
nr	Int	1	

1.116.49 position(...)

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		

1.116.50 prevLine(...)

Return start of previous line.

Argument	Type	Default	Description
pos	Int		
nl	Int	1	

1.116.51 prevRow(...)

Return start of previous row.

Argument	Type	Default	Description
pos	Int		
nr	Int	1	

1.116.52 recalc()

Need to recalculate size.
Reimplemented from FXWindow.

Arguments

None.

1.116.53 removeText(...)

Remove n characters of text at position pos from the buffer.

Argument	Type	Default	Description
pos	Int		
n	Int		
notify	Bool	False	

1.116.54 replaceText(...)

Replace m characters at pos by n characters.

Argument	Type	Default	Description
pos	Int		
m	Int		
text	String		
n	Int		
notify	Bool	False	

1.116.55 resize(...)

Resize this window to the specified width and height.
Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

1.116.56 rowEnd(...)

Return row end.

Argument	Type	Default	Description
pos	Int		

1.116.57 rowStart(...)

Return row start.

Argument	Type	Default	Description
pos	Int		

1.116.58 setBarColor(...)

Change bar color.

Argument	Type	Default	Description
clr	FXColor		

1.116.59 setBarColumns(...)

Change number of columns used for line numbers.

Argument	Type	Default	Description
cols	Int		

1.116.60 setBottomLine(...)

Make line containing pos the bottom line.

Argument	Type	Default	Description
pos	Int		

1.116.61 setCursorCol(...)

Set cursor column.

Argument	Type	Default	Description
col	Int		
notify	Bool	False	

1.116.62 setCursorColor(...)

Change cursor color.

Argument	Type	Default	Description
clr	FXColor		

1.116.63 setCursorPos(...)

Set the cursor position.

Argument	Type	Default	Description
pos	Int		
notify	Bool	False	

1.116.64 setCursorRow(...)

Set cursor row.

Argument	Type	Default	Description
row	Int		
notify	Bool	False	

1.116.65 setEditable(...)

Set editable flag.

Argument	Type	Default	Description
edit	Bool	True	

1.116.66 setFont(...)

Change text font.

Argument	Type	Default	Description
fnt	FXFont		

1.116.67 setMarginBottom(...)

Change bottom margin.

Argument	Type	Default	Description
pb	Int		

1.116.68 setMarginLeft(...)

Change left margin.

Argument	Type	Default	Description
pl	Int		

1.116.69 setMarginRight(...)

Change right margin.

Argument	Type	Default	Description
pr	Int		

1.116.70 setMarginTop(...)

Change top margin.

Argument	Type	Default	Description
pt	Int		

1.116.71 setModified(...)

Set modified flag.

Argument	Type	Default	Description
mod	Bool	True	

1.116.72 setNumberColor(...)

Change line number color.

Argument	Type	Default	Description
clr	FXColor		

1.116.73 setSelBackColor(...)

Change selected background color.

Argument	Type	Default	Description
clr	FXColor		

1.116.74 setSelection(...)

Select len characters starting at given position pos.

Argument	Type	Default	Description
pos	Int		
len	Int		
notify	Bool	False	

1.116.75 setSelTextColor(...)

Change selected text color.

Argument	Type	Default	Description
clr	FXColor		

1.116.76 setText(...)

Change the text.

Argument	Type	Default	Description
text	String		
notify	Bool	False	

1.116.77 setText(...)

Change the text in the buffer to new text.

Argument	Type	Default	Description
text	String		
n	Int		
notify	Bool	False	

1.116.78 setTextColor(...)

Change text color.

Argument	Type	Default	Description
clr	FXColor		

1.116.79 setTopLine(...)

Make line containing pos the top line.

Argument	Type	Default	Description
pos	Int		

1.116.80 setVisCols(...)

Change number of visible columns.

Argument	Type	Default	Description
cols	Int		

1.116.81 setVisRows(...)

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.

1.117 FXTopWindow

Abstract base class for all top-level windows

1.117.1 create()

Create server-side resources.

Reimplemented from FXShell.

Reimplemented in FXPrintDialog, FXToolbarShell, AFXMainWindow, and AFXDialog.

Arguments

None.

1.117.2 deiconify()

Deiconify window.

Arguments

None.

1.117.3 detach()

Detach the server-side resources for this window.

Reimplemented from FXComposite.

Arguments

None.

1.117.4 getDecorations()

Return current title and border decorations.

Arguments

None.

1.117.5 getDefaultHeight()

Return the default height of this window.
Reimplemented from FXComposite.
Reimplemented in FXToolbarShell, and AFXMainWindow.

Arguments

None.

1.117.6 getDefaultWidth()

Return the default width of this window.
Reimplemented from FXComposite.
Reimplemented in FXToolbarShell, and AFXMainWindow.

Arguments

None.

1.117.7 getHSpacing()

Return horizontal spacing between children.

Arguments

None.

1.117.8 getIcon()

Return window icon.

Arguments

None.

1.117.9 getMiniIcon()

Return window mini (title) icon.

Arguments

None.

1.117.10 getPackingHints()

Return packing hints for children.

Arguments

None.

1.117.11 getPadBottom()

Get bottom interior padding.

Arguments

None.

1.117.12 getPadLeft()

Get left interior padding.

Arguments

None.

1.117.13 getPadRight()

Get right interior padding.

Arguments

None.

1.117.14 getPadTop()

Get top interior padding.

Arguments

None.

1.117.15 getTitle()

Return window title.

Arguments

None.

1.117.16 getVSpacing()

Return vertical spacing between children.

Arguments

None.

1.117.17 hide()

Hide this window.

Reimplemented from FXWindow.

Reimplemented in AFXManagerMenuDB, AFXDialog, and AFXMessageDialog.

Arguments

None.

1.117.18 iconify()

Iconify window.

Arguments

None.

1.117.19 isIconified()

Return True if window has been iconified.

Arguments

None.

1.117.20 killFocus()

Remove the focus from this window.
Reimplemented from FXShell.

Arguments

None.

1.117.21 move(...)

Move this window to the specified position in the parent's coordinates.
Reimplemented from FXWindow.

Argument	Type	Default	Description
x	Int		
y	Int		

1.117.22 place(...)

Position the window based on placement.

Argument	Type	Default	Description
placement	Int		

1.117.23 position(...)

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		

1.117.24 resize(...)

Resize this window to the specified width and height.

Reimplemented from FXWindow.

Argument	Type	Default	Description
w	Int		
h	Int		

1.117.25 setDecorations(...)

Change title and border decorations.

Argument	Type	Default	Description
decorations	Int		

1.117.26 setFocus()

Move the focus to this window.
Reimplemented from FXShell.

Arguments

None.

1.117.27 setHSpacing(...)

Change horizontal spacing between children.

Argument	Type	Default	Description
hs	Int		

1.117.28 setIcon(...)

Change window icon.

Argument	Type	Default	Description
ic	FXIcon		

1.117.29 setMiniIcon(...)

Change window mini (title) icon.

Argument	Type	Default	Description
ic	FXIcon		

1.117.30 setPackingHints(...)

Change packing hints for children.

Argument	Type	Default	Description
ph	Int		

1.117.31 setPadBottom(...)

Change bottom padding.

Argument	Type	Default	Description
pb	Int		

1.117.32 setPadLeft(...)

Change left padding.

Argument	Type	Default	Description
pl	Int		

1.117.33 setPadRight(...)

Change right padding.

Argument	Type	Default	Description
pr	Int		

1.117.34 setPadTop(...)

Change top padding.

Argument	Type	Default	Description
pt	Int		

1.117.35 setTitle(...)

Change window title.

Argument	Type	Default	Description
name	String		

1.117.36 setVSpacing(...)

Change vertical spacing between children.

Argument	Type	Default	Description
vs	Int		

1.117.37 show(...)

Show this window with given placement.

Argument	Type	Default	Description
placement	Int		

1.117.38 show()

Show this window.

Reimplemented from FXWindow.

Reimplemented in AFXDialog, AFXFileDialog, and AFXMessageDialog.

Arguments

None.

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.

1.118 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.

1.119 FXTreeList

Tree list Widget

1.119.1 FXTreeList(...)

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	

1.119.2 addItemAfter(...)

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	

1.119.3 addItemAfter(...)

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	

1.119.4 addItemAfter(...)

Append new [possibly subclassed] item after to other item.

Argument	Type	Default	Description
other	FXTreeItem		

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.119.5 addItemBefore(...)

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	

1.119.6 addItemBefore(...)

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	

1.119.7 addItemBefore(...)

Prepend new [possibly subclassed] item prior to other item.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

1.119.8 addItemFirst(...)

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	

1.119.9 addItemFirst(...)

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	

1.119.10 addItemFirst(...)

Prepend new [possibly subclassed] item as first child of p.

Argument	Type	Default	Description
p	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

1.119.11 addItemLast(...)

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	

1.119.12 addItemLast(...)

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	

1.119.13 addItemLast(...)

Append new [possibly subclassed] item as last child of p.

Argument	Type	Default	Description
p	FXTreeItem		
item	FXTreeItem		
notify	Bool	False	

1.119.14 canFocus()

Tree list can receive focus.
Reimplemented from FXWindow.

Arguments

None.

1.119.15 clearItems(...)

Remove all items from list.

Argument	Type	Default	Description
notify	Bool	False	

1.119.16 closeItem(...)

Close item.

Argument	Type	Default	Description
item	FXTreeItem		

Argument	Type	Default	Description
notify	Bool	False	

1.119.17 collapseTree(...)

Collapse tree.

Argument	Type	Default	Description
tree	FXTreeItem		
notify	Bool	False	

1.119.18 create()

Create server-side resources.
Reimplemented from FXComposite.
Reimplemented in FXDirList.

Arguments

None.

1.119.19 deselectItem(...)

Deselect item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.119.20 destroy()

Destroy server-side resources.
Reimplemented from FXComposite.
Reimplemented in FXDirList.

Arguments

None.

1.119.21 detach()

Detach server-side resources.
Reimplemented from FXComposite.

Reimplemented in FXDirList.

Arguments

None.

1.119.22 disableItem(...)

Disable item.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.23 enableItem(...)

Enable item.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.24 expandTree(...)

Expand tree.

Argument	Type	Default	Description
tree	FXTreeItem		
notify	Bool	False	

1.119.25 extendSelection(...)

Extend selection from anchor item to item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.119.26 findItem(...)

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	

Argument	Type	Default	Description
flags	Int	SEARCH_FORWARD| SEARCH_WRAP	

1.119.27 getAnchorItem()

Return anchor item, if any.

Arguments

None.

1.119.28 getContentHeight()

Return content height.
Reimplemented from FXScrollArea.

Arguments

None.

1.119.29 getContentWidth()

Compute and return content width.
Reimplemented from FXScrollArea.

Arguments

None.

1.119.30 getCurrentItem()

Return current item, if any.

Arguments

None.

1.119.31 getCursorItem()

Return item under cursor, if any.

Arguments

None.

1.119.32 getDefaultHeight()

Return default height.
Reimplemented from FXScrollArea.

Arguments

None.

1.119.33 getDefaultWidth()

Return default width.
Reimplemented from FXScrollArea.

Arguments

None.

1.119.34 getFirstItem()

REturn first root item.

Arguments

None.

1.119.35 getFont()

Return text font.

Arguments

None.

1.119.36 getHelpText()

Get the status line help text for this list.

Arguments

None.

1.119.37 getIndent()

Return parent-child indent amount.

Arguments

None.

1.119.38 getItemAt(...)

Get item at x,y, if any.

Argument	Type	Default	Description
x	Int		
y	Int		

1.119.39 getItemClosedIcon(...)

Return item's closed icon.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.40 getItemData(...)

Return item user-data pointer.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.41 getItemHeight(...)

Return item height.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.42 getItemOpenIcon(...)

Return item's open icon.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.43 getItemText(...)

Return item's text.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.44 getItemWidth(...)

Return item width.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.45 getLastItem()

Return last root item.

Arguments

None.

1.119.46 getLineColor()

Return line color.

Arguments

None.

1.119.47 getListStyle()

Return list style.

Arguments

None.

1.119.48 getNumItems()

Return number of items.

Arguments

None.

1.119.49 getNumVisible()

Return number of visible items.

Arguments

None.

1.119.50 getSelBackColor()

Return selected text background.

Arguments

None.

1.119.51 getSelTextColor()

Return selected text color.

Arguments

None.

1.119.52 getTextColor()

Return normal text color.

Arguments

None.

1.119.53 hitItem(...)

Return item hit code: 0 outside, 1 icon, 2 text, 3 box.

Argument	Type	Default	Description
item	FXTreeItem		
x	Int		
y	Int		

1.119.54 isItemCurrent(...)

Return True if item is current.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.55 isItemEnabled(...)

Return True if item is enabled.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.56 isItemExpanded(...)

Return True if item expanded.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.57 isItemLeaf(...)

Return True if item is a leaf-item, i.e. has no children.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.58 isItemOpened(...)

Return True if item opened.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.59 isItemSelected(...)

Return True if item is selected.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.60 isItemVisible(...)

Return True if item is visible.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.61 killFocus()

Remove the focus from this window.
Reimplemented from FXWindow.

Arguments

None.

1.119.62 killSelection(...)

Deselect all items.

Argument	Type	Default	Description
notify	Bool	False	

1.119.63 makeItemVisible(...)

Scroll to make item visible.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.64 moveItemAfter(...)

Move item after other.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		

1.119.65 moveItemBefore(...)

Move item before other.

Argument	Type	Default	Description
other	FXTreeItem		
item	FXTreeItem		

1.119.66 openItem(...)

Open item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.119.67 recalc()

Recalculate layout.

Reimplemented from FXWindow.

Arguments

None.

1.119.68 removeItem(...)

Remove item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.119.69 removeItems(...)

Remove items in range [fm, to] inclusively.

Argument	Type	Default	Description
fm	FXTreeItem		
to	FXTreeItem		
notify	Bool	False	

1.119.70 reparentItem(...)

Reparent item under parent p.

Argument	Type	Default	Description
item	FXTreeItem		
p	FXTreeItem		

1.119.71 selectItem(...)

Select item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.119.72 setAnchorItem(...)

Change anchor item.

Argument	Type	Default	Description
item	FXTreeItem		

1.119.73 setCurrentItem(...)

Change current item.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.119.74 setFocus()

Move the focus to this window.

Reimplemented from FXWindow.

Arguments

None.

1.119.75 setFont(...)

Change text font.

Argument	Type	Default	Description
fnt	FXFont		

1.119.76 setHelpText(...)

Set the status line help text for this list.

Argument	Type	Default	Description
text	String		

1.119.77 setIndent(...)

Change parent-child indent amount.

Argument	Type	Default	Description
in	Int		

1.119.78 setItemClosedIcon(...)

Chance item's closed icon.

Argument	Type	Default	Description
item	FXTreeItem		
icon	FXIcon		

1.119.79 setItemData(...)

Change item user-data pointer.

Argument	Type	Default	Description
item	FXTreeItem		
ptr	String		

1.119.80 setItemOpenIcon(...)

Change item's open icon.

Argument	Type	Default	Description
item	FXTreeItem		
icon	FXIcon		

1.119.81 setItemText(...)

Change item's text.

Argument	Type	Default	Description
item	FXTreeItem		
text	String		

1.119.82 setLineColor(...)

Change line color.

Argument	Type	Default	Description
clr	FXColor		

1.119.83 setListStyle(...)

Change list style.

Argument	Type	Default	Description
style	Int		

1.119.84 setNumVisible(...)

Change number of visible items.

Argument	Type	Default	Description
nvis	Int		

1.119.85 setSelBackColor(...)

Change selected text background.

Argument	Type	Default	Description
clr	FXColor		

1.119.86 setSelTextColor(...)

Change selected text color.

Argument	Type	Default	Description
clr	FXColor		

1.119.87 setTextColor(...)

Change normal text color.

Argument	Type	Default	Description
clr	FXColor		

1.119.88 toggleItem(...)

Toggle item selection.

Argument	Type	Default	Description
item	FXTreeItem		
notify	Bool	False	

1.119.89 updateItem(...)

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.

1.120 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.

1.120.1 FXVerticalFrame(...)

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	

1.120.2 getDefaultHeight()

Return default height.

Reimplemented from FXPacker.

Reimplemented in AFXVerticalAligner.

Arguments

None.

1.120.3 getDefaultWidth()

Return default width.

Reimplemented from FXPacker.

Reimplemented in AFXVerticalAligner.

Arguments

None.

1.121 FXVerticalSeparator

Vertical separator

1.121.1 FXVerticalSeparator(...)

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	

1.121.2 getDefaultHeight()

Return default height.
Reimplemented from FXFrame.

Arguments

None.

1.121.3 getDefaultWidth()

Return default width.
Reimplemented from FXFrame.

Arguments

None.

Global flags

Separator Options

SEPARATOR_NONE	Nothing visible.
SEPARATOR_GROOVE	Etched-in looking groove.
SEPARATOR_RIDGE	Embossed looking ridge.

SEPARATOR_LINE	Simple line.

1.122 FXWindow

Base class for all windows

1.122.1 FXWindow(...)

Constructor.

Argument	Type	Default	Description
p	FXComposite		
opts	Int	0	
x	Int	0	
y	Int	0	
w	Int	0	
h	Int	0	

1.122.2 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.

Arguments

None.

1.122.3 childAtIndex(...)

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		

1.122.4 containsChild(...)

Return True if specified window is a child of this window.

Argument	Type	Default	Description
child	FXWindow		

1.122.5 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.

Arguments

None.

1.122.6 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.

Arguments

None.

1.122.7 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.

Arguments

None.

1.122.8 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.

Arguments

None.

1.122.9 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.

Arguments

None.

1.122.10 forceRefresh()

Force a GUI update of this window and its children.

Arguments

None.

1.122.11 getBackColor()

Get background color.

Reimplemented in FXComboBox, and FXListBox.

Arguments

None.

1.122.12 getCursorPosition()

Returns a sequence of (status, x, y, mouseButtonState) representing the relative location of the cursor in the widget.

Arguments

None.

1.122.13 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.

Arguments

None.

1.122.14 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.

Arguments

None.

1.122.15 getFirst()

Return a pointer to this window's first child window , if any.

Reimplemented in AFXOptionTreeItem.

Arguments

None.

1.122.16 getHeightForWidth(...)

Return height for given width.

Reimplemented in FXDockSite.

Argument	Type	Default	Description
givenwidth	Int		

1.122.17 getKey()

Return window key.

Arguments

None.

1.122.18 getLast()

Return a pointer to this window's last child window, if any.
Reimplemented in AFXOptionTreeItem.

Arguments

None.

1.122.19 getLayoutHints()

Get layout hints for this window.

Arguments

None.

1.122.20 getNext()

Return a pointer to the next (sibling) window, if any.
Reimplemented in AFXOptionTreeItem.

Arguments

None.

1.122.21 getOwner()

Return a pointer to the owner window.
Reimplemented in AFXMenuCascade, AFXMenuCommand, AFXMenuPane, AFXMenuTitle, AFXToolbarGroup, and AFXToolboxGroup.

Arguments

None.

1.122.22 getParent()

Return a pointer to the parent window.
Reimplemented in AFXOptionTreeItem.

Arguments

None.

1.122.23 getPrev()

Return a pointer to the previous (sibling) window , if any.
Reimplemented in AFXOptionTreeItem.

Arguments

None.

1.122.24 getRoot()

Return a pointer to the root window.

Arguments

None.

1.122.25 getSelector()

Get the message identifier for this window.

Arguments

None.

1.122.26 getShell()

Return a pointer to the shell window.

Arguments

None.

1.122.27 getTarget()

Get the message target object for this window, if any.

Arguments

None.

1.122.28 getWidthForHeight(...)

Return width for given height.
Reimplemented in FXDockSite.

Argument	Type	Default	Description
givenheight	Int		

1.122.29 getX()

Get this window's x-coordinate, in the parent's coordinate system.

Arguments

None.

1.122.30 getY()

Get this window's y-coordinate, in the parent's coordinate system.

Arguments

None.

1.122.31 grab(...)

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	

1.122.32 grabbed()

Return True if the window has been grabbed.

Arguments

None.

1.122.33 hasFocus()

Return True if this window has the focus.

Arguments

None.

1.122.34 hide()

Hide this window.

Reimplemented in FXTopWindow, AFXManagerMenuDB, AFXMenuTitle, AFXToolbarGroup, AFXToolbarGroupRender, AFXToolbarGroupVisibility, AFXDialog, AFXFlyoutItem, AFXMessageDialog, AFXOptionTreeItem, and AFXProgressBar.

Arguments

None.

1.122.35 indexOfChild(...)

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		

1.122.36 isActive()

Return True if the window is active.

Reimplemented in AFXToolbarGroup.

Arguments

None.

1.122.37 isChildOf(...)

Return True if specified window is this window's parent.

Argument	Type	Default	Description
window	FXWindow		

1.122.38 isDefault()

Return True if this is the default window.

Arguments

None.

1.122.39 isEnabled()

Return True if this window is able to receive mouse and keyboard events.

Arguments

None.

1.122.40 isInitial()

Return True if this is the initial default window.

Arguments

None.

1.122.41 linkAfter(...)

Relink this window after sibling in the window list.

Argument	Type	Default	Description
sibling	FXWindow		

1.122.42 linkBefore(...)

Relink this window before sibling in the window list.

Argument	Type	Default	Description
sibling	FXWindow		

1.122.43 move(...)

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		

1.122.44 numChildren()

Return the number of child windows for this window.
Reimplemented in AFXOptionTreeItem.

Arguments

None.

1.122.45 position(...)

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		

1.122.46 recalc()

Mark this window's layout as dirty.
Reimplemented in FXIconList, FXList, FXMDIClient, FXRootWindow, FXShell, FXTable, FXText, FXTreeList, AFXBaseTable, AFXSlider, and AFXTable.

Arguments

None.

1.122.47 repaint()

If marked but not yet painted, paint the entire window.

Arguments

None.

1.122.48 repaint(...)

If marked but not yet painted, paint the given area.

Argument	Type	Default	Description
x	Int		

Argument	Type	Default	Description
y	Int		
w	Int		
h	Int		

1.122.49 resize(...)

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		

1.122.50 setBackColor(...)

Set window background color.

Reimplemented in FXComboBox, and FXListBox.

Argument	Type	Default	Description
clr	FXColor		

1.122.51 setCursorPosition(...)

Warp the cursor to the new position.

Argument	Type	Default	Description
x	Int		
y	Int		

1.122.52 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.

Arguments

None.

1.122.53 setHeight(...)

Set the window height.

Argument	Type	Default	Description
h	Int		

1.122.54 setInitial(...)

Make this window the initial default window.

Argument	Type	Default	Description
enable	Bool	True	

1.122.55 setKey(...)

Change window key.

Argument	Type	Default	Description
k	Int		

1.122.56 setLayoutHints(...)

Set layout hints for this window.

Argument	Type	Default	Description
lout	Int		

1.122.57 setSelector(...)

Set the message identifier for this window.

Argument	Type	Default	Description
sel	Int		

1.122.58 setTarget(...)

Set the message target object for this window.

Argument	Type	Default	Description
t	FXObject		

1.122.59 setWidth(...)

Set the window width.

Argument	Type	Default	Description
w	Int		

1.122.60 setX(...)

Set this window's x-coordinate, in the parent's coordinate system.

Argument	Type	Default	Description
x	Int		

1.122.61 setY(...)

Set this window's y-coordinate, in the parent's coordinate system.

Argument	Type	Default	Description
y	Int		

1.122.62 show()

Show this window.

Reimplemented in FXTooltip, FXTopWindow, AFXMenuTitle, AFXToolbarGroup, AFXToolbarGroupRender, AFXToolbarGroupVisibility, AFXDialog, AFXFileDialog, AFXMessageDialog, AFXOptionTreeItem, AFXProgressBar, and AFXSlider.

Arguments

None.

1.122.63 shown()

Return True if the window is shown.

Arguments

None.

1.122.64 translateCoordinatesTo(...)

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		

1.122.65 ungrab()

Release the mouse grab.

Arguments

None.

1.122.66 update()

Mark the entire window client area dirty.

Arguments

None.

1.122.67 update(...)

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

1.123 FXXPMIcon

X Pixmap icon.

1.123.1 FXXPMIcon(...)

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	

1.124 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 Manual for more details.

1.124.1 FXMAPFUNC(...)

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).

1.124.2 FXMAPFUNCS(...)

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.

Argument	Type	Default	Description
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).

1.124.3 MKUINT(...)

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).

1.124.4 SELID(...)

Returns the message ID from a message selector.

Argument	Type	Default	Description
selector	Int		A message selector.

1.124.5 SELTYPE(...)

Returns the message type from a message selector.

Argument	Type	Default	Description
selector	Int		A message selector.

1.125 Auxiliary functions

This section lists various auxiliary GUI toolkit functions.

1.125.1 addExitCallback(...)

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.

1.125.2 afxCreateIcon(...)

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.

1.125.3 afxCreateBMPIcon(...)

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.

1.125.4 afxCreateGIFIcon(...)

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.

1.125.5 afxCreatePNGIcon(...)

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.

1.125.6 afxCreateXPMIcon(...)

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.

1.125.7 afxDisableWidgetTree(...)

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.

1.125.8 afxEnableWidgetTree(...)

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.

1.125.9 displayURL(...)

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.

1.125.10 getAFXApp()

Returns the application object.

Arguments

None.

1.125.11 getAFXAliasMap().setPrefix(...)

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.

1.125.12 getAFXAliasMap().setName(...)

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.

1.125.13 getAFXFont(...)

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.

1.125.14 afxGetColorHexSpecFromID(...)

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 Manual for details.

1.125.15 afxGetColorHexSpecFromName(...)

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 Manual for details.

1.125.16 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.

Arguments

None.

1.125.17 getCurrentModuleGui()

Returns the current moduleGui object. You can call that object's getModuleName method to check its name.

Arguments

None.

1.125.18 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.

Arguments

None.

1.125.19 getDisplayedObjectType()

Returns the type of the object displayed in the current viewport. Possible return values are: PART, ASSEMBLY, ODB, XY_PLOT, SKETCH, or None.

Arguments

None.

1.125.20 getSeparator(...)

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).

1.125.21 getWidgetFromText(...)

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).

1.125.22 openWithURL(...)

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.

1.125.23 registerCurrentContext(...)

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.

1.125.24 registerPluginToolset()

Registers the plug-in toolset with the application.

Arguments

None.

1.125.25 objectPath.registerQuery(...)

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,” Section 6.8 of the Abaqus

GUI Toolkit User's Manual. 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.

1.125.26 objectPath.registerInclusive(...)

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.

1.125.27 removeExitCallback(...)

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.

1.125.28 sendCommand(...)

Sends a command string to the kernel.

Argument	Type	Default	Description
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.

1.125.29 setCurrentModel(...)

Sets the current model to the specified model name.

Argument	Type	Default	Description
modelName	String		The name of the model to be made current.

1.125.30 setSwitchModuleHook(...)

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.

1.125.31 shutdown()

Exits Abaqus/CAE. This is equivalent to selecting File->Exit.

Arguments

None.

1.125.32 switchModule(...)

Switches into a GUI module.

Argument	Type	Default	Description
moduleName	String		Module to switch into.

1.125.33 unregisterCurrentContext(...)

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.

1.125.34 objectPath.unregisterQuery(...)

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.

1.126 Message dialogs

The following methods provide access to the standard message dialogs.

1.126.1 showAFXDismissableWarningDialog(...)

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.

Argument	Type	Default	Description
message	String		Text to be displayed in the dialog box.
tgt	FXObject	None	Message target.
sel	Int	0	Message ID.

1.126.2 showAFXErrorDialog(...)

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.

1.126.3 showAFXItemsWarningDialog(...)

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.

1.126.4 showAFXWarningDialog(...)

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.

1.126.5 showAFXInformationDialog(...)

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.

2. Plug-in registration commands

Plug-in commands register kernel and GUI plug-ins in either the **Plug-ins** menu or in a toolbox.

2.1 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()
```

2.1.1 registerKernelMenuButton(...)

This method registers a kernel plug-in in the **Plug-ins** menu.

Path

```
toolset.registerKernelMenuButton
```

Arguments

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 manual. 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.

2.1.2 registerKernelToolButton(...)

Registers a kernel plug-in in a toolbox.

Path

`toolset.registerKernelToolButton`

Arguments

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 manual. 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.

2.2 GUI plug-in registration commands

GUI plug-in commands register GUI plug-ins either in the **Plug-ins** menu or in a toolbox.

2.2.1 registerGuiMenuButton(...)

Registers a GUI plug-in in the **Plug-ins** menu.

Path

`toolset.registerGuiMenuButton`

Arguments

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 manual. 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.

2.2.2 registerGuiToolButton(...)

Registers a GUI plug-in in a toolbox.

Path

`toolset.registerGuiToolButton`

Arguments

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 manual. 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 6.10 and 6.11

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 6.10 GUI Toolkit Reference manual for the description of methods that have been removed in Abaqus 6.11.

AFXComboBox

AFXComboBox.ID_DECREMENT: Down arrow button ID.

AFXComboBox.ID_INCREMENT: Up arrow button ID.

AFXCOMBOBOX_SPINNER: Include spinner buttons.

AFXCommand

expandObjectName(objectName)

AFXGuiCommand

expandObjectName(objectName)

AFXGuiObjectManager

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

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

AFXMainWindow

appendApplicableModuleForTreeTab(name, moduleName)

appendApplicableModuleForTreeTab(name, displayedName)

appendVisibleModuleForTreeTab(name, moduleName)

appendVisibleModuleForTreeTab(name, displayedName)

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

AFXMenuCascade

getOwner()

AFXMenuCommand

getOwner()

AFXMenuPane

getOwner()

AFXMenuTitle

getOwner()

AFXModuleGui

AFXModuleGui.MISSING ENUMERATOR: MISSING ENUMERATOR DESCRIPTION

AFXProcedureToolsetGui

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

AFXToolbarGroup

getOwner()

AFXToolboxGroup

getOwner()

FXApp

useWidgetBackColor(state)

FXFileItem

FXFileItem.BLOCKDEV: Character special item.

FXFileItem.CHARDEV: Symbolic linked item.

FXFileItem.EXECUTABLE: Directory item.

FXFileItem.FIFO: Block special item.

FXFileItem.SOCK: FIFO item.

FXFileItem.SYMLINK: Executable item.

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)

FXText

FXText.MOUSE_CHARS: Selecting characters.
FXText.MOUSE_DRAG: Dragging text.
FXText.MOUSE_LINES: Selecting lines.
FXText.MOUSE_NONE: No mouse operation.
FXText.MOUSE_PASTE: Pasting.
FXText.MOUSE_SCROLL: Scrolling.
FXText.MOUSE_TRYDRAG: Tentative drag.
FXText.MOUSE_WORDS: Selecting words.
FXText.STYLE_ACTIVE: Active.
FXText.STYLE_CONTROL: Control character.
FXText.STYLE_HILITE: Highlighted.
FXText.STYLE_MASK: Mask color table.
FXText.STYLE_SELECTED: Selected.
FXText.STYLE_STRIKEOUT: Strike out text.
FXText.STYLE_TEXT: Draw some content.
FXText.STYLE_UNDERLINE: Underline text.

FXWindow

FXWindow.CAE_GET_VERSION: CAE, used to get an object's version number.
FXWindow.CAE_SET_VERSION: CAE, used to set an object's version number.
FXWindow.FLAG_ACTIVE: Window is active.
FXWindow.FLAG_CARET: Caret is on.
FXWindow.FLAG_CHANGED: Window data changed.
FXWindow.FLAG_DEFAULT: Default widget.
FXWindow.FLAG_DIRTY: Needs layout.
FXWindow.FLAG_DODRAG: Doing drag mode.
FXWindow.FLAG_DROPTARGET: Drop target.
FXWindow.FLAG_ENABLED: Able to receive input.
FXWindow.FLAG_FOCUSED: Has focus.
FXWindow.FLAG_HELP: Show help.
FXWindow.FLAG_INITIAL: Initial widget.

FXWindow.FLAG_KEY: Keyboard key pressed.
FXWindow.FLAG_LASSO: Lasso mode.
FXWindow.FLAG_PRESSED: Button has been pressed.
FXWindow.FLAG_RECALC: Needs recalculation.
FXWindow.FLAG_SCROLLING: Right mouse scrolling.
FXWindow.FLAG_SCROLLINSIDE: Scroll only when inside.
FXWindow.FLAG_SHELL: Shell window.
FXWindow.FLAG_SHOWN: Is shown.
FXWindow.FLAG_TIP: Show tip.
FXWindow.FLAG_TRYDRAG: Tentative drag mode.
FXWindow.FLAG_UPDATE: Is subject to GUI update.
FXWindow.ID_CLOSE_ALL_DOCUMENTS: CAE: Removed from later FOX version.
FXWindow.ID_CLOSE_DOCUMENT: CAE: Removed from later FOX version.
FXWindow.ID_GETHELPSTRING: CAE: Copied from FOX 1.4.34.
FXWindow.ID_GETICONVALUE: CAE: Copied from FOX 1.4.34.
FXWindow.ID_GETTIPSTRING: CAE: Copied from FOX 1.4.34.
FXWindow.ID_QUERY_HELP: CAE: Removed from later FOX version.
FXWindow.ID_QUERY_TIP: CAE: Removed from later FOX version.
FXWindow.ID_SETHELPSTRING: CAE: Copied from FOX 1.4.34.
FXWindow.ID_SETICONVALUE: CAE: Copied from FOX 1.4.34.
FXWindow.ID_SETTIPSTRING: CAE: Copied from FOX 1.4.34.
FXWindow.ID_TIPTIMER: CAE: Copied from FOX 1.4.34.
FXWindow.ID_TOGGLEENABLED: CAE: Copied from FOX 1.4.34.

Global functions

highlight(entity)
unhighlight(entity)

About SIMULIA

SIMULIA is the Dassault Systèmes brand that delivers a scalable portfolio of Realistic Simulation solutions including the Abaqus product suite for Unified Finite Element Analysis; multiphysics solutions for insight into challenging engineering problems; and lifecycle management solutions for managing simulation data, processes, and intellectual property. By building on established technology, respected quality, and superior customer service, SIMULIA makes realistic simulation an integral business practice that improves product performance, reduces physical prototypes, and drives innovation. Headquartered in Providence, RI, USA, with R&D centers in Providence and in Vélizy, France, SIMULIA provides sales, services, and support through a global network of regional offices and distributors. For more information, visit www.simulia.com.

About Dassault Systèmes

As a world leader in 3D and Product Lifecycle Management (PLM) solutions, Dassault Systèmes brings value to more than 100,000 customers in 80 countries. A pioneer in the 3D software market since 1981, Dassault Systèmes develops and markets PLM application software and services that support industrial processes and provide a 3D vision of the entire lifecycle of products from conception to maintenance to recycling. The Dassault Systèmes portfolio consists of CATIA for designing the virtual product, SolidWorks for 3D mechanical design, DELMIA for virtual production, SIMULIA for virtual testing, ENOVIA for global collaborative lifecycle management, and 3DVIA for online 3D lifelike experiences. Dassault Systèmes' shares are listed on Euronext Paris (#13065, DSY.PA), and Dassault Systèmes' ADRs may be traded on the US Over-The-Counter (OTC) market (DASTY). For more information, visit www.3ds.com.

DASSAULT SYSTEMES