MenuItem

Constants Summary

Number	#ALIGN_CENTER Constant for centering menu items/checkbox/radiobuttons at creation.
Number	#ALIGN_LEFT Constant for left aligning menu items/checkbox/radiobuttons at creation.
Number	#ALIGN_RIGHT Constant for right aligning menu items/checkbox/radiobuttons at creation.

Property Summary

Boolean	#enabled Enable/disable the menu item/checkbox/radiobutton.
Object[]	#methodArguments Set arguments that are sent to the callback method.
String	#name The name of the menu item/checkbox/radiobutton.
Boolean	#selected Select/unselect the checkbox/radiobutton.
String	#text Get/set the text of the menu item/checkbox/radiobutton.

Method Summary

void	#doClick() Script the selection (emulate a mouse click) of the item.
Object	#getClientProperty(key) Gets the specified client property for the menu item/checkbox/radiobutton based on a key.
void	#putClientProperty(key, value) Sets the value for the specified client property key of the menu item/checkbox/radiobutton.
MenuItem	#setAccelerator() Set the accelerator key of the menu item/checkbox/radiobutton.
void	#setBackgroundColor() Set the background color of the menu item/checkbox/radiobutton.
void	#setForegroundColor() Set the foreground color of the menu item/checkbox/radiobutton.
MenuItem	#setIcon() Set the icon of the menu item/checkbox/radiobutton.
MenuItem	#setMethod(method) Set the method for the menu item/checkbox/radiobutton.
MenuItem	#setMethod(method, arguments) Set the method for the menu item/checkbox/radiobutton.
MenuItem	#setMnemonic() Set the mnemonic key of the menu item/checkbox/radiobutton.
MenuItem	#setVisible(visible) Set the item visible.

Constants Details

ALIGN_CENTER
Constant for centering menu items/checkbox/radiobuttons at creation.

Returns
Number

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a left aligned menu item
menu.addMenuItem("significantly long text", feedback, null, null, true, MenuItem.ALIGN_LEFT);
// add a centered checkbox
menu.addCheckBox("a bit shorter", feedback, null, null, true, MenuItem.ALIGN_CENTER);
// add a right aligned radiobutton
menu.addRadioButton("short", feedback, null, null, true, MenuItem.ALIGN_RIGHT);
```

ALIGN_LEFT
Constant for left aligning menu items/checkbox/radiobuttons at creation.

Returns
Number

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a left aligned menu item
menu.addMenuItem("significantly long text", feedback, null, null, true, MenuItem.ALIGN_LEFT);
// add a centered checkbox
menu.addCheckBox("a bit shorter", feedback, null, null, true, MenuItem.ALIGN_CENTER);
// add a right aligned radiobutton
menu.addRadioButton("short", feedback, null, null, true, MenuItem.ALIGN_RIGHT);
```

ALIGN_RIGHT
Constant for right aligning menu items/checkbox/radiobuttons at creation.

Returns
Number

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a left aligned menu item
menu.addMenuItem("significantly long text", feedback, null, null, true, MenuItem.ALIGN_LEFT);
// add a centered checkbox
menu.addCheckBox("a bit shorter", feedback, null, null, true, MenuItem.ALIGN_CENTER);
// add a right aligned radiobutton
menu.addRadioButton("short", feedback, null, null, true, MenuItem.ALIGN_RIGHT);
```

Property Details

enabled
Enable/disable the menu item/checkbox/radiobutton.
Returns
Boolean
Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item at the first position in the menu
var entry = menu.addMenuItem(0);
// alternatively add a checkbox at the first position
//var entry = menu.addCheckBox(0);
// or alternatively add a radiobutton at the first position
//var entry = menu.addRadioButton(0);

// disable the newly added entry
entry.enabled = false;
// give a name to the entry (the name is not visible anywhere)
entry.name = "my_name";
// make the entry selected (affects checkboxes and radiobuttons)
entry.selected = true;
// set the text of the entry
entry.text = "menu entry";
// set the callback method
entry.setMethod(feedback);
// set the arguments to be sent to the callback method
// (an array of elements which will be passed as arguments 5, 6 and so on to the callback method)
// the first 5 arguments are fixed:
//        [0] item index
//        [1] parent item index
//        [2] isSelected boolean
//        [3] parent menu text
//        [4] menu text
entry.methodArguments = [17, "data"];
```

methodArguments

Set arguments that are sent to the callback method.

Returns

Object[]

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item at the first position in the menu
var entry = menu.addMenuItem(0);
// alternatively add a checkbox at the first position
//var entry = menu.addCheckBox(0);
// or alternatively add a radiobutton at the first position
//var entry = menu.addRadioButton(0);

// disable the newly added entry
entry.enabled = false;
// give a name to the entry (the name is not visible anywhere)
entry.name = "my_name";
// make the entry selected (affects checkboxes and radiobuttons)
entry.selected = true;
// set the text of the entry
entry.text = "menu entry";
// set the callback method
entry.setMethod(feedback);
// set the arguments to be sent to the callback method
// (an array of elements which will be passed as arguments 5, 6 and so on to the callback method)
// the first 5 arguments are fixed:
//          [0] item index
//          [1] parent item index
//          [2] isSelected boolean
//          [3] parent menu text
//          [4] menu text
entry.methodArguments = [17, "data"];
```

name

The name of the menu item/checkbox/radiobutton. The name is used only internally, it is not visible in the user interface.

Returns

String

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item at the first position in the menu
var entry = menu.addMenuItem(0);
// alternatively add a checkbox at the first position
//var entry = menu.addCheckBox(0);
// or alternatively add a radiobutton at the first position
//var entry = menu.addRadioButton(0);

// disable the newly added entry
entry.enabled = false;
// give a name to the entry (the name is not visible anywhere)
entry.name = "my_name";
// make the entry selected (affects checkboxes and radiobuttons)
entry.selected = true;
// set the text of the entry
entry.text = "menu entry";
// set the callback method
entry.setMethod(feedback);
// set the arguments to be sent to the callback method
// (an array of elements which will be passed as arguments 5, 6 and so on to the callback method)
// the first 5 arguments are fixed:
//        [0] item index
//        [1] parent item index
//        [2] isSelected boolean
//        [3] parent menu text
//        [4] menu text
entry.methodArguments = [17, "data"];
```

selected

Select/unselect the checkbox/radiobutton.

Returns

Boolean

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item at the first position in the menu
var entry = menu.addMenuItem(0);
// alternatively add a checkbox at the first position
//var entry = menu.addCheckBox(0);
// or alternatively add a radiobutton at the first position
//var entry = menu.addRadioButton(0);

// disable the newly added entry
entry.enabled = false;
// give a name to the entry (the name is not visible anywhere)
entry.name = "my_name";
// make the entry selected (affects checkboxes and radiobuttons)
entry.selected = true;
// set the text of the entry
entry.text = "menu entry";
// set the callback method
entry.setMethod(feedback);
// set the arguments to be sent to the callback method
// (an array of elements which will be passed as arguments 5, 6 and so on to the callback method)
// the first 5 arguments are fixed:
//        [0] item index
//        [1] parent item index
//        [2] isSelected boolean
//        [3] parent menu text
//        [4] menu text
entry.methodArguments = [17, "data"];
```

text

Get/set the text of the menu item/checkbox/radiobutton.

Returns

String

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item at the first position in the menu
var entry = menu.addMenuItem(0);
// alternatively add a checkbox at the first position
//var entry = menu.addCheckBox(0);
// or alternatively add a radiobutton at the first position
//var entry = menu.addRadioButton(0);

// disable the newly added entry
entry.enabled = false;
// give a name to the entry (the name is not visible anywhere)
entry.name = "my_name";
// make the entry selected (affects checkboxes and radiobuttons)
entry.selected = true;
// set the text of the entry
entry.text = "menu entry";
// set the callback method
entry.setMethod(feedback);
// set the arguments to be sent to the callback method
// (an array of elements which will be passed as arguments 5, 6 and so on to the callback method)
// the first 5 arguments are fixed:
//        [0] item index
//        [1] parent item index
//        [2] isSelected boolean
//        [3] parent menu text
//        [4] menu text
entry.methodArguments = [17, "data"];
```

Method Details

doClick

void **doClick**()

Script the selection (emulate a mouse click) of the item.

Returns

void

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item
var entry = menu.addMenuItem("menu entry", feedback);
// alternatively add a checkbox
//var entry = menu.addCheckBox("menu entry", feedback);
// or alternatively add a radiobutton
//var entry = menu.addRadioButton("menu entry", feedback);

// simulate a click on the entry
entry.doClick();
```

getClientProperty

Object **getClientProperty**(key)

Gets the specified client property for the menu item/checkbox/radiobutton based on a key.

Parameters

{Object} key

Returns

Object

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item
var entry = menu.addMenuItem("menu entry", feedback);
// alternatively add a checkbox
//var entry = menu.addCheckBox("menu entry", feedback);
// or alternatively add a radiobutton
//var entry = menu.addRadioButton("menu entry", feedback);

// NOTE: Depending on the operating system, a user interface property name may be available.
// set the tooltip of the menu item/checkbox/radiobutton via client properties
// keep the original tooltip in a form or global variable
originalTooltip = entry.getClientProperty("ToolTipText");
entry.putClientProperty("ToolTipText", "changed tooltip");

// later restore the original tooltip from the variable
//var menubar = plugins.window.getMenuBar();
//var menuIndex = menubar.getMenuIndexByText("New Menu");
//var menu = menubar.getMenu(menuIndex);
//var entry = menu.getItem(0);
//entry.putClientProperty("ToolTipText", originalTooltip);
```

putClientProperty

void **putClientProperty**(key, value)

Sets the value for the specified client property key of the menu item/checkbox/radiobutton.

Parameters

{Object} key

{Object} value

Returns

void

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item
var entry = menu.addMenuItem("menu entry", feedback);
// alternatively add a checkbox
//var entry = menu.addCheckBox("menu entry", feedback);
// or alternatively add a radiobutton
//var entry = menu.addRadioButton("menu entry", feedback);

// NOTE: Depending on the operating system, a user interface property name may be available.
// set the tooltip of the menu item/checkbox/radiobutton via client properties
// keep the original tooltip in a form or global variable
originalTooltip = entry.getClientProperty("ToolTipText");
entry.putClientProperty("ToolTipText", "changed tooltip");

// later restore the original tooltip from the variable
//var menubar = plugins.window.getMenuBar();
//var menuIndex = menubar.getMenuIndexByText("New Menu");
//var menu = menubar.getMenu(menuIndex);
//var entry = menu.getItem(0);
//entry.putClientProperty("ToolTipText", originalTooltip);
```

setAccelerator

MenuItem **setAccelerator**()

Set the accelerator key of the menu item/checkbox/radiobutton.

Returns

MenuItem

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item
var entry = menu.addMenuItem("menu entry", feedback);
// alternatively add a checkbox
//var entry = menu.addCheckBox("menu entry", feedback);
// or alternatively add a radiobutton
//var entry = menu.addRadioButton("menu entry", feedback);

// define an accelerator for the menu entry
entry.setAccelerator("ctrl alt Y");
// also define a mnemonic
entry.setMnemonic("y");
// set a custom background color
entry.setBackgroundColor("#111111");
// set a custom foreground color
entry.setForegroundColor("#EE5555");
// set an icon
entry.setIcon("media:///yourimage.gif");
```

setBackgroundColor

void **setBackgroundColor**()

Set the background color of the menu item/checkbox/radiobutton.

Returns

void

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item
var entry = menu.addMenuItem("menu entry", feedback);
// alternatively add a checkbox
//var entry = menu.addCheckBox("menu entry", feedback);
// or alternatively add a radiobutton
//var entry = menu.addRadioButton("menu entry", feedback);

// define an accelerator for the menu entry
entry.setAccelerator("ctrl alt Y");
// also define a mnemonic
entry.setMnemonic("y");
// set a custom background color
entry.setBackgroundColor("#111111");
// set a custom foreground color
entry.setForegroundColor("#EE5555");
// set an icon
entry.setIcon("media:///yourimage.gif");
```

setForegroundColor

void **setForegroundColor**()

Set the foreground color of the menu item/checkbox/radiobutton.

Returns

void

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item
var entry = menu.addMenuItem("menu entry", feedback);
// alternatively add a checkbox
//var entry = menu.addCheckBox("menu entry", feedback);
// or alternatively add a radiobutton
//var entry = menu.addRadioButton("menu entry", feedback);

// define an accelerator for the menu entry
entry.setAccelerator("ctrl alt Y");
// also define a mnemonic
entry.setMnemonic("y");
// set a custom background color
entry.setBackgroundColor("#111111");
// set a custom foreground color
entry.setForegroundColor("#EE5555");
// set an icon
entry.setIcon("media:///yourimage.gif");
```

setIcon

MenuItem **setIcon**()

Set the icon of the menu item/checkbox/radiobutton.

Returns

MenuItem

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item
var entry = menu.addMenuItem("menu entry", feedback);
// alternatively add a checkbox
//var entry = menu.addCheckBox("menu entry", feedback);
// or alternatively add a radiobutton
//var entry = menu.addRadioButton("menu entry", feedback);

// define an accelerator for the menu entry
entry.setAccelerator("ctrl alt Y");
// also define a mnemonic
entry.setMnemonic("y");
// set a custom background color
entry.setBackgroundColor("#111111");
// set a custom foreground color
entry.setForegroundColor("#EE5555");
// set an icon
entry.setIcon("media:///yourimage.gif");
```

setMethod

MenuItem **setMethod**(method)

Set the method for the menu item/checkbox/radiobutton.

Parameters

{Function} method

Returns

MenuItem

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item at the first position in the menu
var entry = menu.addMenuItem(0);
// alternatively add a checkbox at the first position
//var entry = menu.addCheckBox(0);
// or alternatively add a radiobutton at the first position
//var entry = menu.addRadioButton(0);

// disable the newly added entry
entry.enabled = false;
// give a name to the entry (the name is not visible anywhere)
entry.name = "my_name";
// make the entry selected (affects checkboxes and radiobuttons)
entry.selected = true;
// set the text of the entry
entry.text = "menu entry";
// set the callback method
entry.setMethod(feedback);
// set the arguments to be sent to the callback method
// (an array of elements which will be passed as arguments 5, 6 and so on to the callback method)
// the first 5 arguments are fixed:
//        [0] item index
//        [1] parent item index
//        [2] isSelected boolean
//        [3] parent menu text
//        [4] menu text
entry.methodArguments = [17, "data"];
```

setMethod

MenuItem **setMethod**(method, arguments)

Set the method for the menu item/checkbox/radiobutton.

Parameters

{Function} method
{Object[]} arguments

Returns

MenuItem

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item at the first position in the menu
var entry = menu.addMenuItem(0);
// alternatively add a checkbox at the first position
//var entry = menu.addCheckBox(0);
// or alternatively add a radiobutton at the first position
//var entry = menu.addRadioButton(0);

// disable the newly added entry
entry.enabled = false;
// give a name to the entry (the name is not visible anywhere)
entry.name = "my_name";
// make the entry selected (affects checkboxes and radiobuttons)
entry.selected = true;
// set the text of the entry
entry.text = "menu entry";
// set the callback method
entry.setMethod(feedback);
// set the arguments to be sent to the callback method
// (an array of elements which will be passed as arguments 5, 6 and so on to the callback method)
// the first 5 arguments are fixed:
//        [0] item index
//        [1] parent item index
//        [2] isSelected boolean
//        [3] parent menu text
//        [4] menu text
entry.methodArguments = [17, "data"];
```

setMnemonic

MenuItem **setMnemonic**()

Set the mnemonic key of the menu item/checkbox/radiobutton.

Returns

MenuItem

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item
var entry = menu.addMenuItem("menu entry", feedback);
// alternatively add a checkbox
//var entry = menu.addCheckBox("menu entry", feedback);
// or alternatively add a radiobutton
//var entry = menu.addRadioButton("menu entry", feedback);

// define an accelerator for the menu entry
entry.setAccelerator("ctrl alt Y");
// also define a mnemonic
entry.setMnemonic("y");
// set a custom background color
entry.setBackgroundColor("#111111");
// set a custom foreground color
entry.setForegroundColor("#EE5555");
// set an icon
entry.setIcon("media:///yourimage.gif");
```

setVisible

MenuItem **setVisible**(visible)

Set the item visible.

Parameters

{Boolean} visible

Returns

MenuItem

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
menu.text = "New Menu";
// alternatively create a popup menu
//var menu = plugins.window.createPopupMenu();

// add a menu item
var entry_one = menu.addMenuItem("an entry", feedback);
// add a checkbox
var entry_two = menu.addCheckBox("another entry", feedback);
// add a radiobutton
var entry_three = menu.addRadioButton("yet another entry", feedback);

// hide the menu item
entry_one.setVisible(false);
// make sure the checkbox is visible
entry_two.setVisible(true);
// hide the radiobutton
entry_three.setVisible(false);
```