Menu

Property Summary

String #text
Retrieve/set the text.

Method Summary

CheckBox #addCheckBox()
Add a checkbox.

CheckBox #addCheckBox(index)
Add a checkbox at the selected index (starting at 0) or at the end.

CheckBox #addCheckBox(name)
Add a checkbox with given name.

CheckBox #addCheckBox(name, feedback_item)
Add a checkbox at the selected index (starting at 0) or at the end.

CheckBox #addCheckBox(name, feedback_item, icon)
Add a checkbox at the selected index (starting at 0) or at the end.

CheckBox #addCheckBox(name, feedback_item, icon, mnemonic)
Add a checkbox at the selected index (starting at 0) or at the end.

CheckBox #addCheckBox(name, feedback_item, icon, mnemonic, enabled)
Add a checkbox at the selected index (starting at 0) or at the end.

CheckBox #addCheckBox(name, feedback_item, icon, mnemonic, enabled, align)
Add a checkbox at the selected index (starting at 0) or at the end.

Menu #addMenu()
Add a submenu at the end.

Menu #addMenu(index)
Add a submenu at the selected index (starting at 0).

Menu #addMenu(name)
Add a submenu with given name.

MenuItem #addMenuItem()
Add a menu item.

MenuItem #addMenuItem(index)
Add a menu item at the selected index (starting at 0) or at the end.

MenuItem #addMenuItem(name)
Add a menu item with given name.

MenuItem #addMenuItem(name, feedback_item)
Add a menu item at the selected index (starting at 0) or at the end.

MenuItem #addMenuItem(name, feedback_item, icon)
Add a menu item at the selected index (starting at 0) or at the end.

MenuItem #addMenuItem(name, feedback_item, icon, mnemonic)
Add a menu item at the selected index (starting at 0) or at the end.

MenuItem #addMenuItem(name, feedback_item, icon, mnemonic, enabled)
Add a menu item at the selected index (starting at 0) or at the end.

MenuItem #addMenuItem(name, feedback_item, icon, mnemonic, enabled, align)
Add a menu item at the selected index (starting at 0) or at the end.

RadioButton #addRadioButton()
Add a radio button.

RadioButton #addRadioButton(index)
Add a radiobutton at the selected index (starting at 0) or at the end.

RadioButton #addRadioButton(name)
Add a radio button with given name.

RadioButton #addRadioButton(name, feedback_item)
Add a radiobutton at the selected index (starting at 0) or at the end.

RadioButton #addRadioButton(name, feedback_item, icon)
Add a radiobutton at the selected index (starting at 0) or at the end.

RadioButton #addRadioButton(name, feedback_item, icon, mnemonic)
Add a radiobutton at the selected index (starting at 0) or at the end.

RadioButton #addRadioButton(name, feedback_item, icon, mnemonic, enabled)
Add a radiobutton at the selected index (starting at 0) or at the end.

RadioButton #addRadioButton(name, feedback_item, icon, mnemonic, enabled, align)
Add a radiobutton at the selected index (starting at 0) or at the end.

void #addRadioGroup()
Add a radiogroup for radiobuttons.

void #addSeparator()
Add the separator at the selected index (starting at 0) or at the end (empty).

void #addSeparator(index)
Add the separator at the selected index (starting at 0) or at the end (empty).

void	#doClick() Script the selection (emulate a mouse click) of the menu.
CheckBox	#getCheckBox(index) Get the checkbox at the selected index (starting at 0).
Object	#getClientProperty(key) Gets the specified client property for the element based on a key.
MenuItem	#getItem(index) Get the item at the selected index (starting at 0).
Number	#getItemCount() Get the number of items in the menu.
Number	#getItemIndexByText(text) Retrieve the index of the item by text.
Menu	#getMenu(index) Get the submenu at the selected index (starting at 0).
RadioButton	#getRadioButton(index) Get the radiobutton at the selected index (starting at 0).
void	#putClientProperty(key, value) Sets the value for the specified element client property key.
void	#removeAllItems() Remove all items from the menu.
void	#removeItem(index) Remove the item(s) at the selected index/indices.
void	#setEnabled() Set the the selected menu enabled or disabled.
void	#setIcon() Set the icon of the menu.
void	#setMnemonic() Set the mnemonic of the selected menu.

Property Details

text

Retrieve/set the text.

Returns

String

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
// set the menu's text
menu.text = "New Menu";
// disable the menu
menu.setEnabled(false);
// set a mnemonic
menu.setMnemonic("u");
// add an icon to the menu
menu.setIcon("media:///yourimage.gif");
```

Method Details

addCheckBox

CheckBox **addCheckBox**()

Add a checkbox.

Returns

CheckBox – checkbox

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();

// when you don't define an index the checkbox will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addCheckBox("checkbox", feedback_checkbox);
// add a checkbox with an icon
menu.addCheckBox("checkbox with icon", feedback_checkbox, "media:///yourimage.gif");
// add a checkbox with a mnemonic
menu.addCheckBox("checkbox with mnemonic", feedback_checkbox, "media:///yourimage.gif", "c");
// add a disabled checkbox
menu.addCheckBox("checkbox disabled", feedback_checkbox, "media:///yourimage.gif", "d", false);
// add a checkbox with text aligned to the right
menu.addCheckBox("align right", feedback_checkbox, null, null, true, MenuItem.ALIGN_RIGHT);

// add a checkbox at a given index (checkbox properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var chk = menu.addCheckBox(2);
chk.text = "checkbox at index";
chk.setMethod(feedback_checkbox);
```

addCheckBox

CheckBox **addCheckBox**(index)

Add a checkbox at the selected index (starting at 0) or at the end.

Parameters

{Number} index – the index at which to add the checkbox

Returns

CheckBox – checkbox

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();

// when you don't define an index the checkbox will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addCheckBox("checkbox", feedback_checkbox);
// add a checkbox with an icon
menu.addCheckBox("checkbox with icon", feedback_checkbox, "media:///yourimage.gif");
// add a checkbox with a mnemonic
menu.addCheckBox("checkbox with mnemonic", feedback_checkbox, "media:///yourimage.gif", "c");
// add a disabled checkbox
menu.addCheckBox("checkbox disabled", feedback_checkbox, "media:///yourimage.gif", "d", false);
// add a checkbox with text aligned to the right
menu.addCheckBox("align right", feedback_checkbox, null, null, true, MenuItem.ALIGN_RIGHT);

// add a checkbox at a given index (checkbox properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var chk = menu.addCheckBox(2);
chk.text = "checkbox at index";
chk.setMethod(feedback_checkbox);
```

addCheckBox

CheckBox **addCheckBox**(name)

Add a checkbox with given name.

Parameters

{String} name – the checkbox text

Returns

CheckBox – checkbox

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();

// when you don't define an index the checkbox will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addCheckBox("checkbox", feedback_checkbox);
// add a checkbox with an icon
menu.addCheckBox("checkbox with icon", feedback_checkbox, "media:///yourimage.gif");
// add a checkbox with a mnemonic
menu.addCheckBox("checkbox with mnemonic", feedback_checkbox, "media:///yourimage.gif", "c");
// add a disabled checkbox
menu.addCheckBox("checkbox disabled", feedback_checkbox, "media:///yourimage.gif", "d", false);
// add a checkbox with text aligned to the right
menu.addCheckBox("align right", feedback_checkbox, null, null, true, MenuItem.ALIGN_RIGHT);

// add a checkbox at a given index (checkbox properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var chk = menu.addCheckBox(2);
chk.text = "checkbox at index";
chk.setMethod(feedback_checkbox);
```

addCheckBox

CheckBox **addCheckBox**(name, feedback_item)

Add a checkbox at the selected index (starting at 0) or at the end.

Parameters

{String} name – the checkbox text

{Object} feedback_item – this can be either a method or an array representing a submenu

Returns

CheckBox – checkbox

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();

// when you don't define an index the checkbox will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addCheckBox("checkbox", feedback_checkbox);
// add a checkbox with an icon
menu.addCheckBox("checkbox with icon", feedback_checkbox, "media:///yourimage.gif");
// add a checkbox with a mnemonic
menu.addCheckBox("checkbox with mnemonic", feedback_checkbox, "media:///yourimage.gif", "c");
// add a disabled checkbox
menu.addCheckBox("checkbox disabled", feedback_checkbox, "media:///yourimage.gif", "d", false);
// add a checkbox with text aligned to the right
menu.addCheckBox("align right", feedback_checkbox, null, null, true, MenuItem.ALIGN_RIGHT);

// add a checkbox at a given index (checkbox properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var chk = menu.addCheckBox(2);
chk.text = "checkbox at index";
chk.setMethod(feedback_checkbox);
```

addCheckBox

CheckBox **addCheckBox**(name, feedback_item, icon)

Add a checkbox at the selected index (starting at 0) or at the end.

Parameters

{String} name – the checkbox text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the checkbox icon (can be an image URL or the image content byte array)

Returns

CheckBox – checkbox

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();

// when you don't define an index the checkbox will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addCheckBox("checkbox", feedback_checkbox);
// add a checkbox with an icon
menu.addCheckBox("checkbox with icon", feedback_checkbox, "media:///yourimage.gif");
// add a checkbox with a mnemonic
menu.addCheckBox("checkbox with mnemonic", feedback_checkbox, "media:///yourimage.gif", "c");
// add a disabled checkbox
menu.addCheckBox("checkbox disabled", feedback_checkbox, "media:///yourimage.gif", "d", false);
// add a checkbox with text aligned to the right
menu.addCheckBox("align right", feedback_checkbox, null, null, true, MenuItem.ALIGN_RIGHT);

// add a checkbox at a given index (checkbox properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var chk = menu.addCheckBox(2);
chk.text = "checkbox at index";
chk.setMethod(feedback_checkbox);
```

addCheckBox

CheckBox **addCheckBox**(name, feedback_item, icon, mnemonic)

Add a checkbox at the selected index (starting at 0) or at the end.

Parameters

{String} name – the checkbox text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the checkbox icon (can be an image URL or the image content byte array)
{String} mnemonic – the checkbox mnemonic

Returns

CheckBox – checkbox

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();

// when you don't define an index the checkbox will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addCheckBox("checkbox", feedback_checkbox);
// add a checkbox with an icon
menu.addCheckBox("checkbox with icon", feedback_checkbox, "media:///yourimage.gif");
// add a checkbox with a mnemonic
menu.addCheckBox("checkbox with mnemonic", feedback_checkbox, "media:///yourimage.gif", "c");
// add a disabled checkbox
menu.addCheckBox("checkbox disabled", feedback_checkbox, "media:///yourimage.gif", "d", false);
// add a checkbox with text aligned to the right
menu.addCheckBox("align right", feedback_checkbox, null, null, true, MenuItem.ALIGN_RIGHT);

// add a checkbox at a given index (checkbox properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var chk = menu.addCheckBox(2);
chk.text = "checkbox at index";
chk.setMethod(feedback_checkbox);
```

addCheckBox

CheckBox **addCheckBox**(name, feedback_item, icon, mnemonic, enabled)

Add a checkbox at the selected index (starting at 0) or at the end.

Parameters

{String} name – the checkbox text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the checkbox icon (can be an image URL or the image content byte array)
{String} mnemonic – the checkbox mnemonic
{Boolean} enabled – the enabled state of the checkbox

Returns

CheckBox – checkbox

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();

// when you don't define an index the checkbox will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addCheckBox("checkbox", feedback_checkbox);
// add a checkbox with an icon
menu.addCheckBox("checkbox with icon", feedback_checkbox, "media:///yourimage.gif");
// add a checkbox with a mnemonic
menu.addCheckBox("checkbox with mnemonic", feedback_checkbox, "media:///yourimage.gif", "c");
// add a disabled checkbox
menu.addCheckBox("checkbox disabled", feedback_checkbox, "media:///yourimage.gif", "d", false);
// add a checkbox with text aligned to the right
menu.addCheckBox("align right", feedback_checkbox, null, null, true, MenuItem.ALIGN_RIGHT);

// add a checkbox at a given index (checkbox properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var chk = menu.addCheckBox(2);
chk.text = "checkbox at index";
chk.setMethod(feedback_checkbox);
```

addCheckBox

CheckBox **addCheckBox**(name, feedback_item, icon, mnemonic, enabled, align)

Add a checkbox at the selected index (starting at 0) or at the end.

Parameters

{String} name – the checkbox text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the checkbox icon (can be an image URL or the image content byte array)
{String} mnemonic – the checkbox mnemonic
{Boolean} enabled – the enabled state of the checkbox
{Number} align – the alignment type

Returns

CheckBox – checkbox

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();

// when you don't define an index the checkbox will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addCheckBox("checkbox", feedback_checkbox);
// add a checkbox with an icon
menu.addCheckBox("checkbox with icon", feedback_checkbox, "media:///yourimage.gif");
// add a checkbox with a mnemonic
menu.addCheckBox("checkbox with mnemonic", feedback_checkbox, "media:///yourimage.gif", "c");
// add a disabled checkbox
menu.addCheckBox("checkbox disabled", feedback_checkbox, "media:///yourimage.gif", "d", false);
// add a checkbox with text aligned to the right
menu.addCheckBox("align right", feedback_checkbox, null, null, true, MenuItem.ALIGN_RIGHT);

// add a checkbox at a given index (checkbox properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var chk = menu.addCheckBox(2);
chk.text = "checkbox at index";
chk.setMethod(feedback_checkbox);
```

addMenu

Menu **addMenu**()

Add a submenu at the end.

Returns

Menu – the submenu

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 first submenu
var submenu1 = menu.addMenu("submenu 1");
submenu1.addMenuItem("sub item 1 - 1", feedback_item);
// add a submenu as child of the first submenu
var submenu1_2 = submenu1.addMenu("submenu 1 - 2");
submenu1_2.addMenuItem("sub item 1 - 2 - 1", feedback_item);
// add another submenu as a child of the first submenu
var submenu1_3 = submenu1.addMenu("submenu 1 - 3");
submenu1_3.addMenuItem("sub item 1 - 3 - 1", feedback_item);
// add a submenu to the second submenu of the first submenu
var submenu1_3_2 = submenu1_2.addMenu("submenu 1 - 2 - 2");
submenu1_3_2.addMenuItem("sub item 1 - 2 - 2 - 1", feedback_item);
// add a submenu directly to the menu, at the first position
var submenu0 = menu.addMenu(0);
submenu0.text = "submenu 0";
submenu0.addMenuItem("sub item 0 - 1", feedback_item);
```

addMenu

Menu **addMenu**(index)

Add a submenu at the selected index (starting at 0).

Parameters

{Number} index – the index at which to add the submenu

Returns
Menu – the submenu
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 first submenu
var submenu1 = menu.addMenu("submenu 1");
submenu1.addMenuItem("sub item 1 - 1", feedback_item);
// add a submenu as child of the first submenu
var submenu1_2 = submenu1.addMenu("submenu 1 - 2");
submenu1_2.addMenuItem("sub item 1 - 2 - 1", feedback_item);
// add another submenu as a child of the first submenu
var submenu1_3 = submenu1.addMenu("submenu 1 - 3");
submenu1_3.addMenuItem("sub item 1 - 3 - 1", feedback_item);
// add a submenu to the second submenu of the first submenu
var submenu1_3_2 = submenu1_2.addMenu("submenu 1 - 2 - 2");
submenu1_3_2.addMenuItem("sub item 1 - 2 - 2 - 1", feedback_item);
// add a submenu directly to the menu, at the first position
var submenu0 = menu.addMenu(0);
submenu0.text = "submenu 0";
submenu0.addMenuItem("sub item 0 - 1", feedback_item);
```

addMenu
Menu **addMenu**(name)
Add a submenu with given name.
Parameters
{String} name – the text of the submenu
Returns
Menu – the submenu
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 first submenu
var submenu1 = menu.addMenu("submenu 1");
submenu1.addMenuItem("sub item 1 - 1", feedback_item);
// add a submenu as child of the first submenu
var submenu1_2 = submenu1.addMenu("submenu 1 - 2");
submenu1_2.addMenuItem("sub item 1 - 2 - 1", feedback_item);
// add another submenu as a child of the first submenu
var submenu1_3 = submenu1.addMenu("submenu 1 - 3");
submenu1_3.addMenuItem("sub item 1 - 3 - 1", feedback_item);
// add a submenu to the second submenu of the first submenu
var submenu1_3_2 = submenu1_2.addMenu("submenu 1 - 2 - 2");
submenu1_3_2.addMenuItem("sub item 1 - 2 - 2 - 1", feedback_item);
// add a submenu directly to the menu, at the first position
var submenu0 = menu.addMenu(0);
submenu0.text = "submenu 0";
submenu0.addMenuItem("sub item 0 - 1", feedback_item);
```

addMenuItem
MenuItem **addMenuItem**()
Add a menu item.
Returns
MenuItem – menu item

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();

// when you don't define an index the item will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addMenuItem("item", feedback_item);
// add an item with an icon
menu.addMenuItem("item with icon", feedback_item, "media:///yourimage.gif");
// add an item with a mnemonic
menu.addMenuItem("item with mnemonic", feedback_item, "media:///yourimage.gif", "i");
// add a disabled item
menu.addMenuItem("disabled item", feedback_item, "media:///yourimage.gif", "d", false);
// add an item with text aligned to the right
menu.addMenuItem("align right", feedback_item, null, null, true, SM_ALIGNMENT.RIGHT);

// add an item at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var item = menu.addMenuItem(2);
item.text = "item at index";
item.setMethod(feedback_item);
```

addMenuItem

MenuItem **addMenuItem**(index)

Add a menu item at the selected index (starting at 0) or at the end.

Parameters

{Number} index – the index at which to add the menu item

Returns

MenuItem – menu item

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();

// when you don't define an index the item will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addMenuItem("item", feedback_item);
// add an item with an icon
menu.addMenuItem("item with icon", feedback_item, "media:///yourimage.gif");
// add an item with a mnemonic
menu.addMenuItem("item with mnemonic", feedback_item, "media:///yourimage.gif", "i");
// add a disabled item
menu.addMenuItem("disabled item", feedback_item, "media:///yourimage.gif", "d", false);
// add an item with text aligned to the right
menu.addMenuItem("align right", feedback_item, null, null, true, SM_ALIGNMENT.RIGHT);

// add an item at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var item = menu.addMenuItem(2);
item.text = "item at index";
item.setMethod(feedback_item);
```

addMenuItem

MenuItem **addMenuItem**(name)

Add a menu item with given name.

Parameters

{String} name – the menu item text

Returns

MenuItem – menu item

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();

// when you don't define an index the item will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addMenuItem("item", feedback_item);
// add an item with an icon
menu.addMenuItem("item with icon", feedback_item, "media:///yourimage.gif");
// add an item with a mnemonic
menu.addMenuItem("item with mnemonic", feedback_item, "media:///yourimage.gif", "i");
// add a disabled item
menu.addMenuItem("disabled item", feedback_item, "media:///yourimage.gif", "d", false);
// add an item with text aligned to the right
menu.addMenuItem("align right", feedback_item, null, null, true, SM_ALIGNMENT.RIGHT);

// add an item at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var item = menu.addMenuItem(2);
item.text = "item at index";
item.setMethod(feedback_item);
```

addMenuItem

MenuItem **addMenuItem**(name, feedback_item)

Add a menu item at the selected index (starting at 0) or at the end.

Parameters

{String} name – the menu item text

{Object} feedback_item – this can be either a method or an array representing a submenu

Returns

MenuItem – menu item

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();

// when you don't define an index the item will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addMenuItem("item", feedback_item);
// add an item with an icon
menu.addMenuItem("item with icon", feedback_item, "media:///yourimage.gif");
// add an item with a mnemonic
menu.addMenuItem("item with mnemonic", feedback_item, "media:///yourimage.gif", "i");
// add a disabled item
menu.addMenuItem("disabled item", feedback_item, "media:///yourimage.gif", "d", false);
// add an item with text aligned to the right
menu.addMenuItem("align right", feedback_item, null, null, true, SM_ALIGNMENT.RIGHT);

// add an item at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var item = menu.addMenuItem(2);
item.text = "item at index";
item.setMethod(feedback_item);
```

addMenuItem

MenuItem **addMenuItem**(name, feedback_item, icon)

Add a menu item at the selected index (starting at 0) or at the end.

Parameters

{String} name – the menu item text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the menu item icon (can be an image URL or the image content byte array)

Returns

MenuItem – menu item

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();

// when you don't define an index the item will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addMenuItem("item", feedback_item);
// add an item with an icon
menu.addMenuItem("item with icon", feedback_item, "media:///yourimage.gif");
// add an item with a mnemonic
menu.addMenuItem("item with mnemonic", feedback_item, "media:///yourimage.gif", "i");
// add a disabled item
menu.addMenuItem("disabled item", feedback_item, "media:///yourimage.gif", "d", false);
// add an item with text aligned to the right
menu.addMenuItem("align right", feedback_item, null, null, true, SM_ALIGNMENT.RIGHT);

// add an item at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var item = menu.addMenuItem(2);
item.text = "item at index";
item.setMethod(feedback_item);
```

addMenuItem

MenuItem **addMenuItem**(name, feedback_item, icon, mnemonic)

Add a menu item at the selected index (starting at 0) or at the end.

Parameters

{String} name – the menu item text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the menu item icon (can be an image URL or the image content byte array)
{String} mnemonic – the menu item mnemonic

Returns

MenuItem – menu item

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();

// when you don't define an index the item will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addMenuItem("item", feedback_item);
// add an item with an icon
menu.addMenuItem("item with icon", feedback_item, "media:///yourimage.gif");
// add an item with a mnemonic
menu.addMenuItem("item with mnemonic", feedback_item, "media:///yourimage.gif", "i");
// add a disabled item
menu.addMenuItem("disabled item", feedback_item, "media:///yourimage.gif", "d", false);
// add an item with text aligned to the right
menu.addMenuItem("align right", feedback_item, null, null, true, SM_ALIGNMENT.RIGHT);

// add an item at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var item = menu.addMenuItem(2);
item.text = "item at index";
item.setMethod(feedback_item);
```

addMenuItem

MenuItem **addMenuItem**(name, feedback_item, icon, mnemonic, enabled)

Add a menu item at the selected index (starting at 0) or at the end.

Parameters

{String} name – the menu item text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the menu item icon (can be an image URL or the image content byte array)
{String} mnemonic – the menu item mnemonic
{Boolean} enabled – the enabled state of the menu item

Returns

MenuItem – menu item

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();

// when you don't define an index the item will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addMenuItem("item", feedback_item);
// add an item with an icon
menu.addMenuItem("item with icon", feedback_item, "media:///yourimage.gif");
// add an item with a mnemonic
menu.addMenuItem("item with mnemonic", feedback_item, "media:///yourimage.gif", "i");
// add a disabled item
menu.addMenuItem("disabled item", feedback_item, "media:///yourimage.gif", "d", false);
// add an item with text aligned to the right
menu.addMenuItem("align right", feedback_item, null, null, true, SM_ALIGNMENT.RIGHT);

// add an item at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var item = menu.addMenuItem(2);
item.text = "item at index";
item.setMethod(feedback_item);
```

addMenuItem

MenuItem **addMenuItem**(name, feedback_item, icon, mnemonic, enabled, align)

Add a menu item at the selected index (starting at 0) or at the end.

Parameters

{String} name – the menu item text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the menu item icon (can be an image URL or the image content byte array)
{String} mnemonic – the menu item mnemonic
{Boolean} enabled – the enabled state of the menu item
{Number} align – the alignment type

Returns

MenuItem – menu item

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();

// when you don't define an index the item will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addMenuItem("item", feedback_item);
// add an item with an icon
menu.addMenuItem("item with icon", feedback_item, "media:///yourimage.gif");
// add an item with a mnemonic
menu.addMenuItem("item with mnemonic", feedback_item, "media:///yourimage.gif", "i");
// add a disabled item
menu.addMenuItem("disabled item", feedback_item, "media:///yourimage.gif", "d", false);
// add an item with text aligned to the right
menu.addMenuItem("align right", feedback_item, null, null, true, SM_ALIGNMENT.RIGHT);

// add an item at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var item = menu.addMenuItem(2);
item.text = "item at index";
item.setMethod(feedback_item);
```

addRadioButton

RadioButton **addRadioButton**()

Add a radio button.

Returns

RadioButton – a radio button menu item

Sample

addRadioButton

RadioButton **addRadioButton**(index)

Add a radiobutton at the selected index (starting at 0) or at the end.

Parameters

{Number} index – the index at which to add the radio button

Returns

RadioButton – a radio button menu item

Sample

addRadioButton

RadioButton **addRadioButton**(name)

Add a radio button with given name.

Parameters

{String} name – the radio button text

Returns

RadioButton – a radio button menu item

Sample

addRadioButton

RadioButton **addRadioButton**(name, feedback_item)

Add a radiobutton at the selected index (starting at 0) or at the end.

Parameters

{String} name – the radio button text

{Object} feedback_item – this can be either a method or an array representing a submenu

Returns

RadioButton – a radio button menu item

Sample

addRadioButton

RadioButton **addRadioButton**(name, feedback_item, icon)

Add a radiobutton at the selected index (starting at 0) or at the end.

Parameters

{String} name – the radio button text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the radio button icon (can be an image URL or the image content byte array)

Returns

RadioButton – a radio button menu item

Sample

addRadioButton

RadioButton **addRadioButton**(name, feedback_item, icon, mnemonic)

Add a radiobutton at the selected index (starting at 0) or at the end.

Parameters

{String} name – the radio button text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the radio button icon (can be an image URL or the image content byte array)
{String} mnemonic – the radio button mnemonic

Returns

RadioButton – a radio button menu item

Sample

addRadioButton

RadioButton **addRadioButton**(name, feedback_item, icon, mnemonic, enabled)

Add a radiobutton at the selected index (starting at 0) or at the end.

Parameters

{String} name – the radio button text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the radio button icon (can be an image URL or the image content byte array)
{String} mnemonic – the radio button mnemonic
{Boolean} enabled – the enabled state of radio button

Returns

RadioButton – a radio button menu item

Sample

addRadioButton

RadioButton **addRadioButton**(name, feedback_item, icon, mnemonic, enabled, align)

Add a radiobutton at the selected index (starting at 0) or at the end.

Parameters

{String} name – the radio button text
{Object} feedback_item – this can be either a method or an array representing a submenu
{Object} icon – the radio button icon (can be an image URL or the image content byte array)
{String} mnemonic – the radio button mnemonic
{Boolean} enabled – the enabled state of radio button
{Number} align – the alignment type

Returns

RadioButton – a radio button menu item

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();

// when you don't define an index the radiobutton will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addRadioButton("radio", feedback_radiobutton);
// add a radiobutton with an icon
menu.addRadioButton("radio with icon", feedback_radiobutton, "media:///yourimage.gif");

// add a new radiobutton group
// a group will 'bind' all added radiobuttons after the group together
// as a result checking one item will uncheck the other
// if no group is added, a group is created automatically when the first radiobutton is added to the menu
// so in this case we will have two groups, one with the radiobuttons added until now and one with the ones
added from now on
menu.addRadioGroup();

// add a radiobutton with a mnemonic
menu.addRadioButton("radio with mnemonic", feedback_radiobutton, "media:///yourimage.gif", "i");
// add a disabled radiobutton
menu.addRadioButton("disabled radio", feedback_radiobutton, "media:///yourimage.gif", "d", false);
// add a radiobutton with text aligned to the right
menu.addRadioButton("align right", feedback_radiobutton, null, null, true, SM_ALIGNMENT.RIGHT);
// add a radiobutton at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var rd = menu.addRadioButton(2);
rd.text = "radio at index";
rd.setMethod(feedback_item);
```

addRadioGroup

void **addRadioGroup**()

Add a radiogroup for radiobuttons. A radiogroup groups together all radiobuttons that are added after the group is added. From all radiobuttons that belong to the same radiogroup only one can be checked at a time.

If no radiogroup is added, one is created automatically when the first radiobutton is added.

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();

// when you don't define an index the radiobutton will be added at the last position
// this is what you usually do to build a new menu
// minimum settings are the text and method
// the method can be a global or form method
// be sure to enter the method WITHOUT '()' at the end
menu.addRadioButton("radio", feedback_radiobutton);
// add a radiobutton with an icon
menu.addRadioButton("radio with icon", feedback_radiobutton, "media:///yourimage.gif");

// add a new radiobutton group
// a group will 'bind' all added radiobuttons after the group together
// as a result checking one item will uncheck the other
// if no group is added, a group is created automatically when the first radiobutton is added to the menu
// so in this case we will have two groups, one with the radiobuttons added until now and one with the ones
added from now on
menu.addRadioGroup();

// add a radiobutton with a mnemonic
menu.addRadioButton("radio with mnemonic", feedback_radiobutton, "media:///yourimage.gif", "i");
// add a disabled radiobutton
menu.addRadioButton("disabled radio", feedback_radiobutton, "media:///yourimage.gif", "d", false);
// add a radiobutton with text aligned to the right
menu.addRadioButton("align right", feedback_radiobutton, null, null, true, SM_ALIGNMENT.RIGHT);
// add a radiobutton at a given index (item properties must be configured after creation)
// indexes start at 0 (zero) so index 2 is in fact position 3
var rd = menu.addRadioButton(2);
rd.text = "radio at index";
rd.setMethod(feedback_item);
```

addSeparator

void **addSeparator**()

Add the separator at the selected index (starting at 0) or at the end (empty).

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 an item and a checkbox
menu.addMenuItem("item", feedback_item);
menu.addCheckBox("checkbox", feedback_checkbox);
// add a separator
menu.addSeparator();
// add a radiobutton. it will be separated from the rest of the control by the separator
menu.addRadioButton("radio", feedback_radiobutton);
// add another separator between the item and the checkbox
menu.addSeparator(1);
```

addSeparator

void **addSeparator**(index)

Add the separator at the selected index (starting at 0) or at the end (empty).

Parameters

{Number} index – the index at which to add the separator

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 an item and a checkbox
menu.addMenuItem("item", feedback_item);
menu.addCheckBox("checkbox", feedback_checkbox);
// add a separator
menu.addSeparator();
// add a radiobutton. it will be separated from the rest of the control by the separator
menu.addRadioButton("radio", feedback_radiobutton);
// add another separator between the item and the checkbox
menu.addSeparator(1);
```

doClick

void **doClick**()
Script the selection (emulate a mouse click) of the menu.
Returns
void
Sample

```
// retrieve the File menu
var menubar = plugins.window.getMenuBar();
var menu = menubar.getMenu(0);
// simulate a click on the File menu
menu.doClick();
```

getCheckBox

CheckBox **getCheckBox**(index)
Get the checkbox at the selected index (starting at 0).
Parameters
{Number} index
Returns
CheckBox

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 two radiobuttons
menu.addRadioButton("radio one", feedback_radiobutton);
menu.addRadioButton("radio two", feedback_radiobutton);
// add a menu item, with a separator before it
menu.addSeparator();
menu.addMenuItem("item", feedback_item);
// add a checkbox, with a separator before it
menu.addSeparator();
menu.addCheckBox("check", feedback_checkbox);
// add a submenu with an item under it
var submenu = menu.addMenu("submenu");
submenu.addMenuItem("subitem", feedback_item);

// depending on some state, update the entries in the menu
var some_state = true;
if (some_state) {
        // select the first radiobutton
        menu.getRadioButton(0).selected = true;
} else {
        // select the first radiobutton
        menu.getRadioButton(1).selected = true;
}
// enable/disable the menu item
// remember to include the separators also when counting the index
menu.getItem(3).enabled = !some_state;
// select/unselect the checkbox
// remember to include the separators also when counting the index
menu.getCheckBox(5).selected = some_state;
// change the text of the submenu and its item
application.output(menu.getItemCount());
if (some_state) {
        menu.getMenu(6).text = "some state";
        menu.getMenu(6).getItem(0).text = "some text";
}
else {
        menu.getMenu(6).text = "not some state";
        menu.getMenu(6).getItem(0).text = "other text";
}
```

getClientProperty

Object **getClientProperty**(key)

Gets the specified client property for the element based on a key.

Parameters

{Object} key

Returns

Object

Sample

```
// NOTE: Depending on the operating system, a user interface property name may be available.
// 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 an item to the menu
menu.addMenuItem("item", feedback_item);

// set the tooltip of the menu via client properties
// keep the original tooltip in a form or global variable
originalTooltip = menu.getClientProperty("ToolTipText");
menu.putClientProperty("ToolTipText", "changed tooltip");

// later restore the original tooltip from the variable
//var menubar = plugins.window.getMenuBar();
//var menu = menubar.getMenu(menubar.getMenuCount()-1);
//menu.putClientProperty("ToolTipText", originalTooltip);
```

getItem

MenuItem **getItem**(index)

Get the item at the selected index (starting at 0).

Parameters

{Number} index

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 two radiobuttons
menu.addRadioButton("radio one", feedback_radiobutton);
menu.addRadioButton("radio two", feedback_radiobutton);
// add a menu item, with a separator before it
menu.addSeparator();
menu.addMenuItem("item", feedback_item);
// add a checkbox, with a separator before it
menu.addSeparator();
menu.addCheckBox("check", feedback_checkbox);
// add a submenu with an item under it
var submenu = menu.addMenu("submenu");
submenu.addMenuItem("subitem", feedback_item);

// depending on some state, update the entries in the menu
var some_state = true;
if (some_state) {
        // select the first radiobutton
        menu.getRadioButton(0).selected = true;
} else {
        // select the first radiobutton
        menu.getRadioButton(1).selected = true;
}
// enable/disable the menu item
// remember to include the separators also when counting the index
menu.getItem(3).enabled = !some_state;
// select/unselect the checkbox
// remember to include the separators also when counting the index
menu.getCheckBox(5).selected = some_state;
// change the text of the submenu and its item
application.output(menu.getItemCount());
if (some_state) {
        menu.getMenu(6).text = "some state";
        menu.getMenu(6).getItem(0).text = "some text";
}
else {
        menu.getMenu(6).text = "not some state";
        menu.getMenu(6).getItem(0).text = "other text";
}
```

getItemCount

Number **getItemCount**()

Get the number of items in the menu.

Returns

Number

Sample

```
// REMARK: indexes start at 0, disabled items, non visible items and seperators are counted also
// REMARK: this is especially important when getting items by the index
// 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 two radiobuttons
menu.addRadioButton("radio one", feedback_radiobutton);
menu.addRadioButton("radio two", feedback_radiobutton);
// add a checkbox
menu.addCheckBox("check", feedback_checkbox);
// add a menu item
menu.addMenuItem("item", feedback_item);
// add another menu item
menu.addMenuItem("item 2", feedback_item);

// remove the last item
menu.removeItem(menu.getItemCount() - 1);
```

getItemIndexByText

Number **getItemIndexByText**(text)

Retrieve the index of the item by text.

Parameters

{String} text

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 two radiobuttons
menu.addRadioButton("radio one", feedback_radiobutton);
menu.addRadioButton("radio two", feedback_radiobutton);
// add a checkbox
menu.addCheckBox("check", feedback_checkbox);
// add a menu item
menu.addMenuItem("item", feedback_item);
// add another menu item
menu.addMenuItem("item 2", feedback_item);

// find the index of the checkbox
var idx = menu.getItemIndexByText("check");
// remove the checkbox by its index
menu.removeItem(idx);
// remove both radiobuttons by their indices
menu.removeItem([0, 1]);
// remove all remaining entries
menu.removeAllItems();
// add back an item
menu.addMenuItem("new item", feedback_item);
```

getMenu

Menu **getMenu**(index)

Get the submenu at the selected index (starting at 0).

Parameters

{Number} index

Returns
Menu
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 two radiobuttons
menu.addRadioButton("radio one", feedback_radiobutton);
menu.addRadioButton("radio two", feedback_radiobutton);
// add a menu item, with a separator before it
menu.addSeparator();
menu.addMenuItem("item", feedback_item);
// add a checkbox, with a separator before it
menu.addSeparator();
menu.addCheckBox("check", feedback_checkbox);
// add a submenu with an item under it
var submenu = menu.addMenu("submenu");
submenu.addMenuItem("subitem", feedback_item);

// depending on some state, update the entries in the menu
var some_state = true;
if (some_state) {
        // select the first radiobutton
        menu.getRadioButton(0).selected = true;
} else {
        // select the first radiobutton
        menu.getRadioButton(1).selected = true;
}
// enable/disable the menu item
// remember to include the separators also when counting the index
menu.getItem(3).enabled = !some_state;
// select/unselect the checkbox
// remember to include the separators also when counting the index
menu.getCheckBox(5).selected = some_state;
// change the text of the submenu and its item
application.output(menu.getItemCount());
if (some_state) {
        menu.getMenu(6).text = "some state";
        menu.getMenu(6).getItem(0).text = "some text";
}
else {
        menu.getMenu(6).text = "not some state";
        menu.getMenu(6).getItem(0).text = "other text";
}
```

getRadioButton

RadioButton **getRadioButton**(index)

Get the radiobutton at the selected index (starting at 0).

Parameters
{Number} index
Returns
RadioButton

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 two radiobuttons
menu.addRadioButton("radio one", feedback_radiobutton);
menu.addRadioButton("radio two", feedback_radiobutton);
// add a menu item, with a separator before it
menu.addSeparator();
menu.addMenuItem("item", feedback_item);
// add a checkbox, with a separator before it
menu.addSeparator();
menu.addCheckBox("check", feedback_checkbox);
// add a submenu with an item under it
var submenu = menu.addMenu("submenu");
submenu.addMenuItem("subitem", feedback_item);

// depending on some state, update the entries in the menu
var some_state = true;
if (some_state) {
        // select the first radiobutton
        menu.getRadioButton(0).selected = true;
} else {
        // select the first radiobutton
        menu.getRadioButton(1).selected = true;
}
// enable/disable the menu item
// remember to include the separators also when counting the index
menu.getItem(3).enabled = !some_state;
// select/unselect the checkbox
// remember to include the separators also when counting the index
menu.getCheckBox(5).selected = some_state;
// change the text of the submenu and its item
application.output(menu.getItemCount());
if (some_state) {
        menu.getMenu(6).text = "some state";
        menu.getMenu(6).getItem(0).text = "some text";
}
else {
        menu.getMenu(6).text = "not some state";
        menu.getMenu(6).getItem(0).text = "other text";
}
```

putClientProperty

void **putClientProperty**(key, value)

Sets the value for the specified element client property key.

Parameters

{Object} key

{Object} value

Returns

void

Sample

```
// NOTE: Depending on the operating system, a user interface property name may be available.
// 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 an item to the menu
menu.addMenuItem("item", feedback_item);

// set the tooltip of the menu via client properties
// keep the original tooltip in a form or global variable
originalTooltip = menu.getClientProperty("ToolTipText");
menu.putClientProperty("ToolTipText", "changed tooltip");

// later restore the original tooltip from the variable
//var menubar = plugins.window.getMenuBar();
//var menu = menubar.getMenu(menubar.getMenuCount()-1);
//menu.putClientProperty("ToolTipText", originalTooltip);
```

removeAllItems

void **removeAllItems**()

Remove all items from the menu.

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 two radiobuttons
menu.addRadioButton("radio one", feedback_radiobutton);
menu.addRadioButton("radio two", feedback_radiobutton);
// add a checkbox
menu.addCheckBox("check", feedback_checkbox);
// add a menu item
menu.addMenuItem("item", feedback_item);
// add another menu item
menu.addMenuItem("item 2", feedback_item);

// find the index of the checkbox
var idx = menu.getItemIndexByText("check");
// remove the checkbox by its index
menu.removeItem(idx);
// remove both radiobuttons by their indices
menu.removeItem([0, 1]);
// remove all remaining entries
menu.removeAllItems();
// add back an item
menu.addMenuItem("new item", feedback_item);
```

removeItem

void **removeItem**(index)

Remove the item(s) at the selected index/indices.

Parameters

{Object[]} index – array of one or moe indexes corresponding to items to remove

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 two radiobuttons
menu.addRadioButton("radio one", feedback_radiobutton);
menu.addRadioButton("radio two", feedback_radiobutton);
// add a checkbox
menu.addCheckBox("check", feedback_checkbox);
// add a menu item
menu.addMenuItem("item", feedback_item);
// add another menu item
menu.addMenuItem("item 2", feedback_item);

// find the index of the checkbox
var idx = menu.getItemIndexByText("check");
// remove the checkbox by its index
menu.removeItem(idx);
// remove both radiobuttons by their indices
menu.removeItem([0, 1]);
// remove all remaining entries
menu.removeAllItems();
// add back an item
menu.addMenuItem("new item", feedback_item);
```

setEnabled

void **setEnabled**()

Set the the selected menu enabled or disabled.

Returns

void

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
// set the menu's text
menu.text = "New Menu";
// disable the menu
menu.setEnabled(false);
// set a mnemonic
menu.setMnemonic("u");
// add an icon to the menu
menu.setIcon("media:///yourimage.gif");
```

setIcon

void **setIcon**()

Set the icon of the menu.

Returns

void

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
// set the menu's text
menu.text = "New Menu";
// disable the menu
menu.setEnabled(false);
// set a mnemonic
menu.setMnemonic("u");
// add an icon to the menu
menu.setIcon("media:///yourimage.gif");
```

setMnemonic

void **setMnemonic**()

Set the mnemonic of the selected menu.

Returns

void

Sample

```
// add a new menu to the menubar
var menubar = plugins.window.getMenuBar();
var menu = menubar.addMenu();
// set the menu's text
menu.text = "New Menu";
// disable the menu
menu.setEnabled(false);
// set a mnemonic
menu.setMnemonic("u");
// add an icon to the menu
menu.setIcon("media:///yourimage.gif");
```