

# CheckBox

## Property Summary

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

## Method Summary

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

## 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.addItem(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.addItem(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.addItem(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.addItem(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.; This can be also html if enclosed between html tags

**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.addItem(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.addItem("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.addItem("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.addItem("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**[CheckBox](#) **setAccelerator** ()

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

**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 a menu item
var entry = menu.addItem("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");
```

**setAlign**[CheckBox](#) **setAlign** (align)

Sets the alignment of the checkbox.

**Parameters**{[Boolean](#)} align



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

// enable alignment of the new entry
entry.setAlign(true);
```

**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.addItem("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**[CheckBox](#) **setIcon** ()

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

**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 a menu item
var entry = menu.addItem("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**[CheckBox](#) **setMethod** (method)

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

**Parameters**[Function](#) method

**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 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**[CheckBox](#) **setMethod** (method, arguments)

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

**Parameters**

[{Function}](#) method  
[{Object\[\]}](#) arguments

**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 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****CheckBox** **setMnemonic** ()

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

**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 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

[CheckBox](#) **setVisible** (visible)

Set the item visible.

### Parameters

{[Boolean](#)} visible

### 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 a menu item
var entry_one = menu.addItem("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);
```