Class

NSPopUpButton

A control for selecting an item from a list.

macOS

```
@MainActor
class NSPopUpButton
```

Overview

An `NSPopUpButton` object uses an `NSPopUpButtonCell` object to implement its user interface.

Note that while a menu is tracking user input, programmatic changes to the menu, such as adding, removing, or changing items on the menu, is not reflected.

Important

Setting a pop up button's `image` property has no effect. The image displayed in a pop up button is taken from the selected menu item (in the case of a pop up menu) or from the first menu item (in the case of a pull-down menu).

Topics

Initializing an NSPopUpButton

`init(frame: NSRect, pullsDown: Bool)`

Returns an `NSPopUpButton` object initialized to the specified dimensions.

Configuring the Cell

`class NSPopUpButtonCell`

The `NSPopUpButtonCell` class defines the visual appearance of pop-up buttons that display pop-up or pull-down menus. Pop-up menus present the user with a set of choices, much the way radio buttons do, but using much less space. Pull-down menus also provide a set of choices but present the information in a slightly different way, usually to provide a set of commands from which the user can choose.

Setting the type of menu

`var pullsDown: Bool`

A Boolean value indicating whether the button displays a pull-down or pop-up menu.

`var autoenablesItems: Bool`

A Boolean value indicating whether the button enables and disables its items every time a user event occurs.

Inserting and deleting items

`func addItem(withTitle: String)`

Adds an item with the specified title to the end of the menu.

`func addItems(withTitles: [String])`

Adds multiple items to the end of the menu.

`func insertItem(withTitle: String, at: Int)`

Inserts an item at the specified position in the menu.

`func removeAllItems()`

Removes all items in the receiver’s item menu.

`func removeItem(withTitle: String)`

Removes the item with the specified title from the menu.

`func removeItem(at: Int)`

Removes the item at the specified index.

Getting the user’s selection

`var selectedItem: NSMenuItem?`

The menu item that was last selected by the user.

`var titleOfSelectedItem: String?`

The title of the item that was last selected by the user.

`var indexOfSelectedItem: Int`

The index of the item that was last selected by the user.

Setting the current selection

`func select(NSMenuItem?)`

Selects the specified menu item.

`func selectItem(at: Int)`

Selects the item in the menu at the specified index.

`func selectItem(withTag: Int) -> Bool`

Selects the menu item with the specified tag.

`func selectItem(withTitle: String)`

Selects the item with the specified title.

Getting menu items

`var menu: NSMenu?`

The menu associated with the pop-up button.

`var numberOfItems: Int`

The number of items in the menu.

`var itemArray: [NSMenuItem]`

The array of menu item objects associated with the button.

`func item(at: Int) -> NSMenuItem?`

Returns the menu item at the specified index.

`func itemTitle(at: Int) -> String`

Returns the title of the item at the specified index.

```
var itemTitles: [String]
```

An array of strings corresponding to the titles of the items in the menu.

```
func item(withTitle: String) -> NSMenuItem?
```

Returns the menu item with the specified title.

```
var lastItem: NSMenuItem?
```

The last item in the menu.

Getting the indices of menu items

```
func index(of: NSMenuItem) -> Int
```

Returns the index of the specified menu item.

```
func indexOfItem(withTag: Int) -> Int
```

Returns the index of the menu item with the specified tag.

```
func indexOfItem(withTitle: String) -> Int
```

Returns the index of the item with the specified title.

```
func indexOfItem(withRepresentedObject: Any?) -> Int
```

Returns the index of the menu item that holds the specified represented object.

```
func indexOfItem(withTarget: Any?, andAction: Selector?) -> Int
```

Returns the index of the menu item with the specified target and action.

Setting the cell edge to pop out in restricted situations

```
var preferredEdge: NSRectEdge
```

The edge of the button on which to display the menu when screen space is constrained.

Setting the title

```
func setTitle(String)
```

Sets the string displayed in the receiver when the user isn't pressing the mouse button.

Setting the state

`func synchronizeTitleAndSelectedItem()`

Ensures that the item being displayed by the receiver agrees with the selected item.

Notifications

`class let willPopUpNotification: NSNotification.Name`

Posted when an `NSPopUpButton` object receives a mouse-down event—that is, when the user is about to select an item from the menu.

Instance Methods

`func selectedTag() -> Int`

Initializers

`convenience init(image: NSImage, pullDownMenu: NSMenu)`

Creates a standard pull-down button with a title, optional image, and menu.

`convenience init(popUpMenu: NSMenu, target: AnyObject?, action: Selector?)`

Creates a standard pop-up button with a menu, target, and action.

`convenience init(title: String, image: NSImage?, pullDownMenu: NSMenu)`

Creates a standard pull-down button with a title, optional image, and menu.

Instance Properties

`var altersStateOfSelectedItem: Bool`

When the value of this property is YES, the selected menu item's `state` is set to `NSControlStateValueOn`. When the value of this property is NO, the menu item's `state` is not changed. When this property changes, the `state` of the currently selected item is updated appropriately. This property is ignored for pull-down buttons.

`var usesItemFromMenu: Bool`

When `usesItemFromMenu` is YES, a pull-down button uses the title of the first menu item and hides the first menu item. A pop-up button uses the title of the currently selected menu. The default value is YES.

Relationships

Inherits From

`NSButton`

Conforms To

```
CVarArg
CustomDebugStringConvertible
CustomStringConvertible
Equatable
Hashable
NSAccessibilityButton
NSAccessibilityElementProtocol
NSAccessibilityProtocol
NSAnimatablePropertyContainer
NSAppearanceCustomization
NSCoding
NSDraggingDestination
NSObjectProtocol
NSStandardKeyBindingResponding
NSTouchBarProvider
NSUserActivityRestoring
NSUserInterfaceCompression
NSUserInterfaceItemIdentification
NSUserInterfaceValidations
Sendable
SendableMetatype
```

See Also

Controls

Responding to control-based events using target-action

Handle user input by connecting buttons, sliders, and other controls to your app’s code using the target-action design pattern.

`class NSButton`

A control that defines an area on the screen that a user clicks to trigger an action.

`class NSColorWell`

A control that displays a color value and lets the user change that color value.

Combo Box

Display a list of values in a pop-up menu that lets the user select a value or type in a custom value.

`class NSComboButton`

A button with a pull-down menu and a default action.

Date Picker

Display a calendar date and provide controls for editing the date value.

`class NSImageView`

A display of image data in a frame.

`class NSLevelIndicator`

A visual representation of a level or quantity, using discrete values.

Path Control

A display of a file system path or virtual path information.

`class NSProgressIndicator`

An interface that provides visual feedback to the user about the status of an ongoing task.

`class NSRuleEditor`

An interface for configuring a rule-based list of options.

`class NSPredicateEditor`

A defined set of rules that allows the editing of predicate objects.

Search Field

Provide a text field that is optimized for text-based search interfaces.

`class NSSegmentedControl`

Display one or more buttons in a single horizontal group.

Slider

Display a range of values from which the user selects a single value.