
NSColorPicker

Inherits From:	NSObject
Conforms To:	NSColorPickingDefault NSObject (NSObject)
Declared In:	AppKit/NSColorPicker.h

NSColorPicker is an abstract superclass that implements the NSColorPickingDefault protocol. The NSColorPickingDefault and NSColorPickingCustom protocols define a way to add color pickers (custom user interfaces for color selection) to the NSColorPanel. The simplest way to implement a color picker is to create a subclass of NSColorPicker, instead of implementing the NSColorPickingDefault protocol in another kind of object. (To add functionality, implement the NSColorPickingCustom methods in your subclass.)

The NSColorPickingDefault protocol specification describes the details of implementing a color picker and adding it to your application's NSColorPanel; you should look there first for an overview of how NSColorPicker works. This specification is provided to document the specific behavior of NSColorPicker's methods.

Adopted Protocols

NSColorPickingDefault	<ul style="list-style-type: none">– alphaControlAddedOrRemoved:– attachColorList:– detachColorList:– initWithPickerMask:colorPanel:– insertNewButtonImage:in:– provideNewButtonImage– setMode:– viewSizeChanged:
-----------------------	---

Method Types

Initializing an NSColorPicker	– initWithPickerMask:colorPanel:
Getting the color panel	– colorPanel

Adding button images	– insertNewButtonImage:in: – provideNewButtonImage
Setting the mode	– setMode:
Using color lists	– attachColorList: – detachColorList:
Responding to a resized view	– viewSizeChanged:

Instance Methods

attachColorList:

– (void)**attachColorList:**(NSColorList *)*colorList*

Does nothing. Override to attach a color list to a color picker.

See also: – detachColorList

colorPanel

– (NSColorPanel *)**colorPanel**

Returns the NSColorPanel that owns this NSColorPicker.

detachColorList:

– (void)**detachColorList:**(NSColorList *)*colorList*

Does nothing. Override to detach a color list from a color picker.

See also: – attachColorList

initWithPickerMask:colorPanel:

– (id)**initWithPickerMask:**(int)*mask*
colorPanel:(NSColorPanel *)*owningColorPanel*

Sets the color picker's color panel to *owningColorPanel*, caching the *owningColorPanel* value so it can later be returned by the **colorPanel** method. Returns self. Override this method to respond to the values in *mask* or do other custom initialization. If you override this method in a subclass, you should forward the message to **super** as part of the implementation.

See also: – colorPanel

insertNewButtonImage:in:

– (void)**insertNewButtonImage:**(NSImage *)*newButtonImage*
in:(NSButtonCell *)*buttonCell*

Sets *newButtonImage* as *buttonCell*'s image by invoking NSButtonCell's **setImage:** method. Called by the color panel to insert a new image into the specified cell. Override this method to customize *newButtonImage* before insertion in *buttonCell*.

See also: – provideNewButtonImage

provideNewButtonImage

– (NSImage *)**provideNewButtonImage**

Returns the button image for the color picker. The color panel will place this image in the mode button that the user uses to select this picker. (This is the same image that the color panel uses as an argument when sending the **insertNewButtonImage:in:** message.) The default implementation looks in the color picker's bundle for a TIFF file named after the color picker's class, with the extension **“.tiff”**.

See also: – insertNewButtonImage:in:

setMode:

– (void)**setMode:**(int)*mode*

Does nothing. Override to set the color picker's mode. Here are the standard color picking modes and mode constants (defined in **AppKit/NSColorPanel.h**):

Mode	Color Mode Constant
Grayscale-Alpha	NSGrayModeColorPanel
Red-Green-Blue	NSRGBModeColorPanel
Cyan-Yellow-Magenta-Black	NSCMYKModeColorPanel
Hue-Saturation-Brightness	NSHSBModeColorPanel
Custom palette	NSCustomPaletteModeColorPanel
Custom color list	NSColorListModeColorPanel
Color wheel	NSWheelModeColorPanel

In grayscale-alpha, red-green-blue, cyan-magenta-yellow-black, and hue-saturation-brightness modes, the user adjusts colors by manipulating sliders. In the custom palette mode, the user can load an NSImage file (TIFF or EPS) into the NSColorPanel, then select colors from the image. In custom color list mode, the user can create and load lists of named colors. The two custom modes provide NSPopUpLists for loading and saving files. Finally, color wheel mode provides a simplified control for selecting colors.

viewSizeChanged:

– (void)**viewSizeChanged:**(id)*sender*

Does nothing. Override to respond to a size change.