NSColorPickingDefault

Adopted By: NSColorPicker

Declared In: AppKit/NSColorPicking.h

Protocol Description

The NSColorPickingDefault protocol, together with the NSColorPickingCustom protocol, provides an interface for adding color pickersÐcustom user interfaces for color selectionÐto an application's NSColorPanel. The NSColorPickingDefault protocol provides basic behavior for a color picker. The NSColorPickingCustom protocol provides implementation-specific behavior.

The NSColorPicker class implements the NSColorPickingDefault protocol. The simplest way to implement your own color picker is to create a subclass of NSColorPicker, implementing the NSColorPickingCustom protocol in that subclass. However, it's possible to create a subclass of another class, such as NSView, and use it as a base upon which to add the methods of both NSColorPickingDefault and NSColorPickingCustom.

Color Picker Bundles

A class that implements the NSColorPickingDefault and NSColorPickingCustom protocols needs to be compiled and linked in an application's object file. However, your application need not explicitly create an instance of this class. Instead, your application's file package should include a directory named **ColorPickers**; within this directory you should place a directory *MyPickerClass***.bundle** for each custom color picker your application implements. This bundle should contain all resources required for your color picker: nib files, TIFF files, and so on.

NSColorPanel will allocate and initialize an instance of each class for which a bundle is found in the **ColorPickers** directory. The class name is assumed to be the bundle directory name minus the **.bundle** extension.

Color Picker Buttons

NSColorPanel lets the user select a color picker from a matrix of NSButtonCells. This protocol includes methods for providing and

manipulating the image that gets displayed on the button.

See also: NSColorPickingCustom, NSColorPicker (class), NSColorPanel (class)

Initializing a Color Picker

- (id)**initWithPickerMask:**(int)*mask*
  colorPanel:(NSColorPanel *)*colorPanel*

Initializes the receiver for the specified mask and color panel. This method is sent by the NSColorPanel to all implementors of the color picking protocols when the application's color panel is first initialized. If the color picker responds to any of the modes represented in *mask*, it should perform its initialization (if desired) and return **self**; otherwise it should do nothing and return **nil**. However, a custom color picker can instead delay initialization until it receives a **provideNewView:** message.

Adding Button Images

- (void)**insertNewButtonImage:**(NSImage *)*newImage*
  in:(NSButtonCell *)*newButtonCell*

Sets *newImage* as *newButtonCell*'s image. *newButtonCell* is the NSButtonCell object that lets the user choose the picker from the color panel. This method should perform application-specific manipulation of the image before it's inserted and displayed by the button cell.

- (NSImage *)**provideNewButtonImage**

Returns the image for the mode button that the user uses to select this picker in the color panel. (This is the same image that the color panel uses as an argument when sending the **insertNewButtonImage:in:** message.)

Setting the Mode

- (void)**setMode:**(int)*mode*

Sets the color picker's mode. This method is invoked by NSColorPanel's **setMode:** method to ensure that the color picker reflects the current mode. Most color pickers have only one mode, and thus don't need to do any work in this method. Others, like the standard sliders picker, have multiple modes.

Using Color Lists

- (void)**attachColorList:**(NSColorList *)*aColorList* Attaches the given color list to the receiver, if it isn't already displaying the list. This method is invoked automatically by the NSColorPanel when its **attachColorList:** method is invoked. Since NSColorPanel's list mode manages NSColorLists, this method need only be implemented by a custom color picker that manages NSColorLists itself.

- (void)**detachColorList:**(NSColorList *)*aColorList* Removes the given color list from the receiver, unless the receiver isn't displaying the list. This method is invoked automatically by the NSColorPanel when its **detachColorList:** method is invoked. Since NSColorPanel's list mode manages NSColorLists, this method need only be implemented by a custom color picker that manages NSColorLists itself.

Showing Opacity Controls

- (void)**alphaControlAddedOrRemoved:**(id)*sender* Sent by the color panel when the opacity controls have been hidden or displayed. If the color picker has its own opacity controls, it should hide or display them, depending on whether the sender's **showsAlpha** method returns NO or YES.

Responding to a Resized View

- (void)**viewSizeChanged:**(id)*sender* Sent when the color picker's superview has been resized in a way that might affect the color picker. *sender* is the NSColorPanel that contains the color picker.