MiscScreenColor

Inherits From: MiscColor

Conforms To: NXTransport

Declared In: misckit/MiscScreenColor.h

Class Description

A MiscScreenColor is a color associated with a specific pixel within a specific view. Whenever you ask it what it's color is, it returns the current color value of the pixel it is monitoring. It is preferable to initialize a new instance of MiscScreenColor by using the ±**initColorFromPixel:inView:** method. After that, use it as a MiscColor would be used. It is possible to force a refresh/update of the stored color via the ±**updateColor** method, but this should never actually be necessary, since MiscScreenColor automatically updates itself as needed. The point being monitored may be accessed via the ±**location**, ±**view**, ±**setLocation:**, and ±**setView:** methods.

Several methods necessary to perform the monitoring task have been overridden, but are not documented here. Only new or particularly important methods are listed in this specification; for further information, please read the MiscColor documentation. Although the methods which alter a MiscColor are still functional, they do not actually alter the monitored pixel's contents. This class can only be used as a voyeur. *(If you would like a future implementation of MiscScreenColor to actually alter the pixel at a particular point in the view, please contact the author and this will be considered. If this were to happen, a new class with the class name ªMiscScreenPixelº would probably be created, rather than modifying this class. Note that such functionality would be truly slow and would not be a highly recommended method of painting the screen.)*

Instance Variables

NXPoint **theLocation**;
id **theView**;

theLocation	The point, in *view*'s coordinates, which is being monitored.
theView	The view which is being monitored. If *nil*, then the view currently focused whenever the color value is requested will be used.

Method Types

Halftoning	± initColorFromPixel: ± initColorFromPixel:inView:

Determining monitored point	± location: ± setLocation: ± setView: ± view
Monitoring point	± actualColor ± updateColor

Instance Methods

actualColor
- (NXColor)**actualColor**

Returns the NXColor of the monitored point, updating as necessary.

See also: -**updateColor**

initColorFromPixel:
- **initColorFromPixel:**(NXPoint *)*location*

The same as calling ±**initColorFromPixel:inView:** with *view* set to *nil*.

See also: -**initColorFromPixel:inView:**

initColorFromPixel:inView:
- **initColorFromPixel:**(NXPoint *)*location* **inView:***view*

Sets up a new instance of MiscScreenColor to monitor the point *location* in view *view*. If *view* is nil, whenever the MiscScreenColor is updated, it will take on the value of the pixel at *location* in the currently focused view. If *location* is a NULL pointer, then the point (0.0, 0.0) will be monitored. Returns *self*.

See also: **-initColorFromPixel:**

location
- (NXPoint)**location**

Returns the coordinates of the point being monitored.

See also: **-setLocation:**

setLocation
- **setLocation:**(NXPoint *)*location*

Set the receiver to monitor the point at *location* in *theView*. Returns *self*. If *location* is a NULL pointer, then the point (0.0, 0.0) will be monitored by the receiver, and *nil* is returned.

See also: **-location**

setView
- **setView:***aView*

Set the receiver to monitor *aView*. Returns the view which was previously being monitored.

See also: **-view**

view
- view

Returns the view being monitored. If not particular view is monitored (ie, whichever is the currently lock focused view at the time is being monitored instead) then *nil* is returned.

See also: **-setView:**

updateColor
- (NXColor)**updateColor**

Re-reads the color from *theView*, keeping the MiscScreenColor up to date.

See also: **-actualColor**