MiscDragView

Inherits From: NSControl
Declared In: MiscAppKit/MiscDragView.h

Class Description

MiscDragView defines an NSView subclass that can be either the source or sink of a dragging session. The behavior of the drag view almost completely depends upon the class of drag cell you make use of. You can set the drag cell that we use in many ways. You can either create a MiscDragView instance and use **setCell:** (from NSControl) to set a concrete instance of a subclass of MiscDragCell, or you can create a subclass of MiscDragView that overrides **+cellClass** to return a subclass of MiscDragCell. For an example of the latter, see MiscFileDragView.

Instance Variables

id _delegate;

_delegate	No description.

Method Types

Class initialization	+ initialize + cellClass + setCellClass:

Our images	- image - setImage: - acceptingImage - setAcceptingImage: - dragImage - setDragImage:
Drag options	- setAllowsSourceDragging: - allowsSourceDragging - setAllowsDestinationDragging: - allowsDestinationDragging - acceptsForeignDrag - acceptsLocalDrag - acceptsSelfDrag - retainsData - shadowsIncoming - setShadowsIncoming: - shadowColor - dragImageSlidesBack
Title options	- title - setTitle: - displaysTitle - setDisplaysTitle: - isTitleEditable - setTitleEditable: - isTitleMultiline - setTitleMultiline:
Our delegate	- delegate - setDelegate:
Source dragging	- initializeDragTypes
Drag helpers	- acceptsFirstMouse: - shouldDelayWindowOrderingForEvent:

Display options	- setBorderType: - borderType
Archiving	- initWithCoder: - encodeWithCoder:

Class Methods

cellClass

+ (Class)**cellClass**

Returns our cell class. By default it is MiscDragCell, though you should override these methods (**cellClass**/setCellClass:) in any subclasses because MiscDragCell is an abstract class.

initialize

+ (void)**initialize**

Sets our class verison for archiving and sets our cell class to MiscDragCell by default.

setCellClass:

+ (void)**setCellClass:**(Class)*dragCellClass*

Sets our cell class. By default it is MiscDragCell.

Instance Methods

acceptingImage

- (NSImage*)**acceptingImage**

Returns the image that's dislayed when we are accepting a drag, or **nil** if we don't have an accepting image. If we do have an accepting image then the usual "dimmed" icoming image won't be shown.

acceptsFirstMouse:

- (BOOL)**acceptsFirstMouse:**(NSEvent *)*theEvent*

Returns YES otherwise we wouldn't even get a **mouseDown:** message.

acceptsForeignDrag

- (BOOL)**acceptsForeignDrag**

Returns YES if we accept drags that don't originate from within our own application. The default is YES.

acceptsLocalDrag

- (BOOL)**acceptsLocalDrag**

Returns YES if we accept drags that originate from within our own application. The default is YES.

acceptsSelfDrag

- (BOOL)**acceptsSelfDrag**

Returns YES if we accept drags that originated from our own drag cell. The default is YES. You see this behavior in the Workspace shelf. If you drag a file from one of the cells you are still able to put it back in the same cell.

allowsDestinationDragging

- (BOOL)**allowsDestinationDragging**

Returns YES if we are allowed to accept a drag. The default is YES.

allowsSourceDragging

- (BOOL)**allowsSourceDragging**

Returns YES if we are allowed to begin a dragging session. The default is YES.

borderType

- (NSBorderType)**borderType**

Returns our current border type, which will be one of NSNoBorder, NSLineBorder, NSBezelBorder or NSGrooveBorder. By default we use NSGrooveBorder.

delegate

- (id)**delegate**

Returns our delegate or **nil** if we don't have one. Right now there are no delegate messages but there will be soon (similar to the ones from the old drag view).

displaysTitle

- (BOOL)**displaysTitle**

Returns YES if we display a title in our view. The default is YES.

dragImage

- (NSImage *)**dragImage**

Returns the image we use as the drag image. If it is **nil** then by default we use the image returned by sending ourself an **image** message.

dragImageSlidesBack

- (BOOL)**dragImageSlidesBack**

Returns YES if the dragging image should slide back to it's destination when it isn't deposited in another view. The default is to slide back only if we retain our data (**retainsData** returns YES).

encodeWithCoder:

- (void)**encodeWithCoder:**(NSCoder *)*aCoder*

Encodes an instance of MiscDragView.

image

- (NSImage*)**image**

Returns our current image or **nil** if we don't have one.

initWithCoder:

- (id)**initWithCoder:**(NSCoder *)*aDecoder*

Unarchives an instance of MiscDragView. Returns **self**.

initializeDragTypes

- (void)**initializeDragTypes**

This method registers the dragging types our cell is interested in receiving, by calling our drag cell's **acceptingPasteboardTypes** method and registering them. This is called from both our **init** method and **initWithCoder:** so both new and unarchived drag views will be registered.

isTitleEditable

- (BOOL)**isTitleEditable**

Returns whether single or double clicking on the title will allow you to edit it. (Title editing NOT IMPLEMENTED YET)

isTitleMultiline

- (BOOL)**isTitleMultiline**

NOT IMPLEMENTED YET

retainsData

- (BOOL)**retainsData**

Returns YES if when we drag that we leave a copy of ourselves behind. If you were to emulate the Workspace shelf then this would return NO. The default is NO.

setAcceptingImage:

- (void)**setAcceptingImage:**(NSImage *)*anImage*

Sets the image that's dislayed when we are accepting a drag. If we do have an accepting image then the usual "dimmed" icoming image won't be shown. If newImage is **nil** then we revert back to using the "dimmed" incoming image.

setAllowsDestinationDragging:

- (void)**setAllowsDestinationDragging:**(BOOL)*aBool*

Sets whether we are allowed to accept a drag. The default is YES.

setAllowsSourceDragging:

- (void)**setAllowsSourceDragging:**(BOOL)*aBool*

Sets whether we are allowed to begin a dragging session. The default is YES.

setBorderType:

- (void)**setBorderType:**(NSBorderType)*aType*

Sets our current border type. *aType* can be one of NSNoBorder, NSLineBorder, NSBezelBorder or NSGrooveBorder. By default we use NSBezelBorder.

setDelegate:

- (void)**setDelegate:**(id)*newDelegate*

Sets our delegate to *newDelegate*.

setDisplaysTitle:

- (void)**setDisplaysTitle:**(BOOL)*displays*

Sets whether we display a title in our view. The default is YES.

setDragImage:

- (void)**setDragImage:**(NSImage *)*anImage*

Sets our drag image. If we don't have a drag image then we use the image we are currently displaying.

setImage:

- (void)**setImage:**(NSImage*)*image*

Sets our *image*. If newImage is **nil** it will clear any *image* we used to have.

setShadowsIncoming:

- (void)**setShadowsIncoming:**(BOOL)*aBool*

Sets whether we show a dimmed image when there is an incoming drag. The default is YES.

setTitle:

- (void)**setTitle:**(NSString*)*newTitle*

Sets our title. In subclasses it is your responsibility to set the title when you accept a drag.

setTitleEditable:

- (void)**setTitleEditable:**(BOOL)*isEditable*

Sets whether single or double clicking on the title will allow you to edit it. (Title editing NOT IMPLEMENTED YET)

setTitleMultiline:

- (void)**setTitleMultiline:**(BOOL)*isMultiline*

NOT IMPLEMENTED YET

shadowColor

- (NSColor*)**shadowColor**

Returns the color used to created a dimmed appearance for a destination image. The default is to return a partially transparent gray. If you want a different appearance for shadowing, you can override this method.

shadowsIncoming

- (BOOL)**shadowsIncoming**

Returns YES if we show a dimmed image when there is an incoming drag. This assumes that we are currently accepting drags. The default is YES.

shouldDelayWindowOrderingForEvent:

- (BOOL)**shouldDelayWindowOrderingForEvent:**(NSEvent *)*theEvent*

Returns YES but I'm not sure why any more?

title

- (NSString*)**title**

Returns our title. This may or may not be displayed depending on the return value of **displaysTitle**.