MiscTargetActionView

Inherits From: View : Responder : Object

Declared In: misckit/MiscTargetActionView.h

Class Description

MiscTargetActionView is an abstract superclass that implements a target action paradeigm for Views. It is not quite the same as the Control Class since   there is no   "value".   MiscTargetActionView accepts keyboard and mouse events and converts them into target/action messages (potentially using a responder chain).

MiscTargetActionView takes it's lessons from several other classes. The target/action methods are similar to those of the Control Class;   the *actionMask* support is similar and complimentary to the *eventMask* of Windows; and the event handling methods override (but may forward to) those of   the Responder Class.

Only a handful of the methods in this class will ever be called directly by a user: all of the action mask methods; the target and action methods; initialization methods; possibly the diagnostic methods; and sometimes the **setFirstResponder** and **setFirstMouse** methods.

Note that events in the *actionMask* will only be recieved if the corresponding events are enabled in the window's event mask. The following code fragment shows how to use this class in an application in which one wants to have the user object's method attention: sent upon any key down, mouse down or mouse movement including the very first one. This is typical of code one might use to make a screensaver get out of the way:

```
myWindow = [[Window allocFromZone: [self zone]] init];
newView      = [[MiscTargetActionView   allocFromZone: [self zone]] init];
[newView setActionMask: NX_LMOUSEDOWNMASK | NX_RMOUSEDOWNMASK | \
                                        NX_MOUSEMOVEDMASK | NX_KEYDOWNMASK];
[newView setFirstMouse: YES];
[newView setTarget: self];
[newView setAction: @selector(attention:)];
[[myWindow setContentView: newView] free];

oldMask =   [myWindow addToEventMask:NX_KEYDOWNMASK | NX_MOUSEMOVEDMASK];
[newView setFirstResponder];
[myWindow makeKeyWindow];
[NXApp activateSelf: YES];
```

The **setFirstResponder** method will only be in effect until some other view is given firstResponder status by the Window, so in some applications it might be necessary to send it as part of the setup, just like the **makeKeyWindow** is sent to the Window.

Instance Variables

BOOL **firstMouse**;
int **actionMask**;

```
NXEvent   * actionEvent;
id target;
SEL action;
BOOL   debug;
```

firstMouse	Yes if we accept first mouse click.
actionMask	Mask of events that will fire the target/action.
actionEvent	Event causing the current target/action.
target	Id of the target object .
action	Selector of the message to be sent to the target.
debug	YES if a diagnostic trace is to be printed on the console or the gdb terminal.

Method Types

Initialization	- initFrame:
Determining the first responder	- acceptsFirstResponder - setFirstResponder - acceptsFirstMouse - setFirstMouse:
Setting the action mask	- setActionMask: - addToActionMask: - removeFromActionMask: - actionMask
Target and action	- setAction: - action - setTarget: - target

	- actionEvent
Processing event messages	- mouseDown:
	- rightMouseDown:
	- mouseDragged:
	- rightMouseDragged:
	- mouseUp:
	- rightMouseUp:
	- mouseMoved:
	- mouseEntered:
	- mouseExited:
	- keyDown:
	- keyUp:
	- flagsChanged:
Diagnostics	- setDebug:
	- debug:
	- toggleDebug:
Archiving	- awake
	- read:
	- write:

Instance Methods

acceptsFirstMouse

- (BOOL)**acceptsFirstMouse**

This returns YES if an initial mouse-down event in the ViewÐan event that causes the View's Window to become the key windowÐis sent to the View (through a **mouseDown:** message). If only those mouse-downs that occur when the MiscTargetActionView's Window is already key are sent, this returns NO (the default). The default behavior may be changed with the **setFirstMouse:** method.

See also: - **setFirstMouse:**

acceptsFirstResponder

- (BOOL)**acceptsFirstResponder**

If you set NX_KEYDOWNMASK, NX_KEYUPMASK, or NX_FLAGSCHANGEDMASK, the TargetActionView will automatically assume you want to acceptFirstResponder and returns YES. Otherwise it returns NO.

YES indicates that the reciever agrees to become the first responder. Before making any object the first responder, the Application Kit gives it an opportunity to refuse by sending it an **acceptsFirstResponder** message.

See also: - **setFirstResponder:**

action

- (SEL)**action**

Returns the action message sent by the MiscTargetActionView.

See also: - **setAction:**, - **target**

actionEvent

- (const NXEvent *)**actionEvent**

Returns a pointer to the event which fired the current action message. This method is to be used by the target if it needs to know exactly what event fired it. The target may not free this record. If the event data is needed for later, the target must copy it because the pointer will not be valid outside of the time of execution of the action message.

See also: - **setAction:**, - **target**

actionMask
- (int)**actionMask**

Returns the current event mask for the MiscTargetActionView. See **setActionMask:** for a list of the possible contents of the mask.

See also: - **setActionMask:**, - **addToActionMask:**, - **removeFromActionMask:**

addToActionMask:
- (int)**addToActiontMask:**(int)*newActions*

Adds *newActions* to the MiscTargetActionView's current action mask and returns the original action mask. The return value should be used to restore the MiscTargetActionView's original event mask at the end of a period of special processing. See **setActionMask:** for a list of action mask constants.

See also: - **setActionMask:**, - **actionMask**, - **removeFromActionMask:**

awake:

- **awake**

Executed when an object completes unarchiving. Returns **self**.

See also: - **read:**, - **write:**

debug

- (BOOL)**debug**

Return the current state of the debug toggle.

See also: - **setDebug:**, - **toggleDebug:**

flagsChanged:

- **flagsChanged:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **flagsChanged:** event message is passed to the receiver's next responder.

initFrame:

- **initFrame:**(const NXRect *)*frameRect*

Create a MiscTargetActionView with the specified frame size. The default *actionMask* responds to no events; the default *action* is NULL, and the default *target* is nil.

keyDown:

- **keyDown:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **keyDown:** event message is passed to the receiver's next responder.

keyUp:

- **keyUp:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **keyUp:** event message is passed to the receiver's next responder.

mouseDown:

- **mouseDown:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **mouseDown:** event message is passed to the receiver's next responder.

mouseDragged:

- **mouseDragged:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **mouseDragged:** event message is passed to the receiver's next responder.

mouseEntered:

- **mouseEntered:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **mouseEntered:** event message is passed to the receiver's next responder.

mouseExited:
- **mouseExited:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **mouseExited:** event message is passed to the receiver's next responder.

mouseMoved:
- **mouseMoved:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **mouseMoved:** event message is passed to the receiver's next responder.

mouseUp:
- **mouseUp:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **mouseUp:** event message is passed to the receiver's next responder.

read:

- **read:**(NXTypedStream *)*stream*

Reads the object from the typed stream *stream*. Returns **self**.

See also: - **write:**, - **awake**

removeFromActionMask:
- (int)**removeFromActionMask:**(int)*oldActions*

Removes the event types specified by *oldActions* from the TargetActionViews's action mask, and returns the old mask.

See also: - **actionMask**, - **setActionMask:**, - **addToActionMask:**

rightMouseDown:
- **rightMouseDown:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **rightMouseDown:** event message is passed to the receiver's next responder.

rightMouseDragged:
- **rightMouseDragged:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **rightMouseDragged:** event message is passed to the receiver's next responder.

rightMouseUp:
- **rightMouseUp:**(NXEvent *)*theEvent*

If the associated action mask bit is set (see **setActionMask:**) the method attempts to send the reciever's *action* message to the reciever's *target*. If the bit is not set, a responder cannot be found or the action message returns nil, the **rightMouseUp:** event message is passed to the receiver's next responder.

setAction:
- **setAction:**(SEL)*aSelector*

Makes *aSelector* the TargetActionView's action method. If *aSelector* is NULL, then no action messages will be sent from the TargetAction. Returns **self**.

See also: - **action**, - **setTarget:**

setActionMask:
- (int)**setActionMask:**(int)*newMask*

Assigns a new action mask to the MiscTargetActionView; the original event mask is returned. The mask tells the MiscTargetActionView which types of events should cause it to fire the action message . It's formed by joining the masks for individual events using the bitwise OR operator. The constants for individual event masks are listed below. The default action mask is NX_NULLEVENTMASK.

NX_LMOUSEDOWNMASK
NX_LMOUSEUPMASK
NX_RMOUSEDOWNMASK
NX_RMOUSEUPMASK
NX_MOUSEMOVEDMASK
NX_LMOUSEDRAGGEDMASK

NX_RMOUSEDRAGGEDMASK
NX_MOUSEENTEREDMASK
NX_MOUSEEXITEDMASK
NX_KEYDOWNMASK
NX_KEYUPMASK
NX_FLAGSCHANGEDMASK
NX_NULLEVENTMASK

A very important note: Setting a flag here does not enable the event for the window. If the event is one that is not
part of the normal window mask, you will have to also set and clear it there. See Window for more information on setting the overall event masks.

If you set NX_KEYDOWNMASK, NX_KEYUPMASK, or NX_FLAGSCHANGEDMASK, the TargetActionView will automatically assume you want to acceptFirstResponder.

See also: - actionMask, **- addToActionMask:**, **- removeFromActionMask:**

setDebug:

- setDebug: (BOOL)*flg*

Turn diagnostic printouts on or off. YES means on. If on, a localized message is printed on the Workspace console or on the GDB console each time one of the event messages or queries to accept first responder or first mouse is recieved. Note that the printouts are a trace of methods sent to this object and do not necessarily mean that any action was taken by this object: **mouseUp:** might be reported and yet do nothing because NX_MOUSEUPMASK is not set in the action mask; **acceptsFirstMouse** might be reported, but first mouse may not actually be accepted if the *firstMouse* instance variable says NO. Returns **self**.

See also: - debug, **- toggleDebug:**

setFirstMouse:
- **setFirstMouse:** (BOOL)*flg*

If flag is YES, then in the future an initial mouse-down event in the ViewĐan event that causes the View's Window to become the key windowĐis sent to the View (through a **mouseDown:** message). If flag is NO, then only those mouse-downs that occur when the MiscTargetActionView's Window is already key are sent. NO is the default.

See also: - acceptsFirstMouse

setFirstResponder
- **setFirstResponder**

Makes the reciever the first receiver of keyboard events and action messages sent to its' Window. It is basically a local cover for sending a makeFirstResponder message to the View's Window.

If successful in making the reciever the first responder, this method returns **self**. If not (if the old first responder refuses to resign), it returns **nil**.

See also: - acceptsFirstResponder:

setTarget:
- **setTarget:***anObject*

Sets the target for the action message of the TargetActionView. Returns **self**.

If *anObject* is **nil**, then when an action message is sent, NXApp looks for an object that can respond to the message by following the responder chain.

See also: - **target**, - **setAction:**

target

- **target**

Returns the target for the action message of the MiscTargetActionView. If **nil**, then any action messages sent by the MiscTargetActionView will be sent up the responder chain.

See also: - **setTarget:**, - **action**

toggleDebug:

- **setDebug:** *sender*

Toggle the diagnostic printouts on and off. YES means on. If on, a localizable message is printed on the Workspace console or on the GDB console each time one of the event messages or queries to accept first responder or first mouse is recieved. Returns **self**.

This method is meant to be used with menu items for debugging. Note that if you do use it in a MenuItem, you will need an update method set so that the menu item will change title: otherwise you won't know what state you are in.

See also: - **debug**, - **setDebug:**

write:

- **write:**(NXTypedStream *)*stream*

Writes the object to the typed stream *stream*. Returns **self**.

See also: **- read:**, **- awake**