NSDraggingDestination

(informal protocol)

Category Of: NSObject

Declared In: AppKit/NSDragging.h

Description

The NSDraggingDestination informal protocol declares methods that the destination (or recipient) of a dragged image must implement. The destination automatically receives NSDraggingDestination messages as an image enters, moves around inside, and then exits or is released within the destination's boundaries.

In the text here and in the other dragging protocol descriptions, the term *dragging session* is the entire process during which an image is selected, dragged, released, and absorbed or rejected by the destination. A *dragging operation* is the action that the destination takes in absorbing the image when it's released. The *dragging source* is the object that "owns" the image that's being dragged. It's specified as an argument to the **dragImage:at:offset:event:pasteboard:source:slideBack:** message, sent to a window or view object, that instigated the dragging session.

The Dragged Image

The image that's dragged in an image-dragging session is simply an image that represents data that resides on the pasteboard. Although a dragging destination can access the image (through the **draggedImage** method described in the NSDraggingInfo protocol), its primary concern is with the pasteboard data that the image represents—the dragging operation that a destination ultimately performs is on the pasteboard data, not on the image itself.

Valid Destinations

Dragging is a visual phenomenon. To be an image-dragging destination, an object must represent a portion of screen real estate; thus, only window and view objects can be destinations. Furthermore, you must register the pasteboard types that the object will accept by sending the object a **registerForDraggedTypes:** message, defined in both NSWindow and NSView. During a dragging session, a candidate destination only receives NSDraggingDestination messages if the destination is registered for a pasteboard type that matches the type of the pasteboard data being dragged. See the NSPasteboard class specification for more information about pasteboard types.

Although NSDraggingDestination is declared as an informal protocol, the NSWindow and NSView subclasses that you create to adopt the protocol need only implement those methods that are pertinent. (The NSWindow and NSView classes provide private implementations for all of the methods.) Either a window

object or its delegate may implement these methods; however, the delegate's implementation takes precedence if there are implementations in both places.

The Sender of Destination Messages

Each of the NSDraggingDestination methods sports a single argument: *sender*, the object that invoked the method. Within its implementations of the NSDraggingDestination methods, the destination can send NSDraggingInfo messages to *sender* to get more information on the current dragging session.

The Order of Destination Messages

The six NSDraggingDestination methods are invoked in a distinct order:

- As the image is dragged into the destination's boundaries, the destination is sent a **draggingEntered:** message
- While the image remains within the destination, a series of **draggingUpdated:** messages are sent.
- If the image is dragged out of the destination, **draggingExited:** is sent and the sequence of NSDraggingDestination messages stops. If it re-enters, the sequence begins again (with a new **draggingEntered:** message).
- When the image is released, it either slides back to its source (and breaks the sequence) or a **prepareForDragOperation:** message is sent to the destination, depending on the value returned by the most recent invocation of **draggingEntered:** or **draggingUpdated:**.
- If the **prepareForDragOperation:** message returned YES, a **performDragOperation:** message is sent.
- Finally, if **performDragOperation:** returned YES, **concludeDragOperation:** is sent.

Method Types

Before the image is released	– draggingEntered: – draggingUpdated: – draggingExited:
After the image is released	– prepareForDragOperation: – performDragOperation: – concludeDragOperation:

Instance Methods

concludeDragOperation:

– (void)**concludeDragOperation:**(id <NSDraggingInfo>)*sender*

Invoked when the dragging operation is complete and the previous **performDragOperation:** returned YES. The destination implements this method to perform any tidying up that it needs to do, such as updating its visual representation now that it has incorporated the dragged data. This is the last message that's sent from *sender* to the destination during a dragging session.

draggingEntered:

– (unsigned int)**draggingEntered:**(id <NSDraggingInfo>)*sender*

Invoked when a dragged image enters the destination. Specifically, this method is invoked when the mouse pointer enters the destination's bounds rectangle (if it's a view object) or its frame rectangle (if it's a window object).

This method must return a value that indicates which dragging operation the destination will perform when the image is released. In deciding which dragging operation to return, the method should evaluate the overlap between both the dragging operations allowed by the source (accessible through the **draggingSourceOperationMask** method) and the dragging operations and pasteboard data types that the destination itself supports. The returned value should be exactly one of the following:

Option	Meaning
NSDragOperationCopy	The data represented by the image will be copied.
NSDragOperationLink	The data will be shared.
NSDragOperationGeneric	The operation will be defined by the destination.
NSDragOperationPrivate	The operation is negotiated privately between the source and the destination.
NSDragOperationAll	Combines all the above.

If none of the operations is appropriate, this method should return NSDragOperationNone (this is the default response if the method isn't implemented by the destination).

The code below is a simple example of a method that responds distinctly when one of two different types of data is dragged into the destination view or window. If the dragged data is a color and the source object permits copying, the return value indicates that the destination will permit copying of the color data on the pasteboard. If the dragged data is an RTF file and the source object permits linking, the return value indicates that the destination will permit linking of the RTF file on the pasteboard. Otherwise the code returns NSDragOperationNone, indicating that the destination will not permit any dragging operations with the data on pastboard.

```
- (unsigned int)draggingEntered:(id <NSDraggingInfo>)sender
```

```
{
    NSPasteboard *pboard;
    NSDragOperation sourceDragMask;

    sourceDragMask = [sender draggingSourceOperationMask];
    pboard = [sender draggingPasteboard];

    if ([[pboard types] indexOfObject:NSColorPboardType] != NSNotFound) {
        if (sourceDragMask & NSDragOperationCopy) {
            return NSDragOperationCopy;
        }
    }
    if ([[pboard types] indexOfObject:NSRTFPboardType] != NSNotFound) {
        if (sourceDragMask & NSDragOperationLink) {
            return NSDragOperationLink;
        }
    }
    return NSDragOperationNone;
}
```

See also: – **draggingUpdated:**, – **draggingExited:**, – **prepareForDragOperation:**

draggingExited:

– (void)**draggingExited:**(id <NSDraggingInfo>)*sender*

Invoked when the dragged image exits the destination's bounds rectangle (in the case of a view object) or its frame rectangle (in the case of a window object).

draggingUpdated:

– (unsigned int)**draggingUpdated:**(id <NSDraggingInfo>)*sender*

Invoked periodically as the image is held within the destination. The messages continue until the image is either released or dragged out of the window or view. The return value should be one of the dragging operation options listed under the **draggingEntered:** method. The default return value (if this method isn't implemented by the destination) is the value returned by the previous **draggingEntered:** message.

This method provides the destination with an opportunity to modify the dragging operation depending on the position of the mouse pointer inside of the destination view or window object. For example, you may have several graphics or areas of text contained within the same view and wish to tailor the dragging operation, or to ignore the drag event completely, depending upon which object is underneath the mouse pointer at the time when the user releases the dragged image and the **performDragOperation:** method is invoked.

You typically examine the contents of the pasteboard in the **draggingEntered:** method, where this examination is performed only once, rather than in the **draggingUpdate:** method which is invoked multiple times.

Only one destination at a time receives a sequence of **draggingUpdated:** messages. If the mouse pointer is within the bounds of two overlapping views that are both valid destinations, the uppermost view receives these messages until the image is either released or dragged out.

See also: – **draggingExited:**, – **prepareForDragOperation:**

performDragOperation:

– (BOOL)**performDragOperation:**(id <NSDraggingInfo>)*sender*

Invoked after the released image has been removed from the screen and the previous **prepareForDragOperation:** message has returned YES. The destination should implement this method to do the real work of importing the pasteboard data represented by the image. If the destination accepts the data, it returns YES, otherwise it returns NO. The default is to return NO.

See also: – **concludeDragOperation:**

prepareForDragOperation:

– (BOOL)**prepareForDragOperation:**(id <NSDraggingInfo>)*sender*

Invoked when the image is released, if the most recent **draggingEntered:** or **draggingUpdated:** message returned an acceptable drag-operation value. Returns YES if the receiver agrees to perform the drag operation and NO if not.

See also: **– performDragOperation:**