| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/DragSource.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/awt/dnd/DragGestureRecognizer.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/dnd/DragSourceAdapter.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/dnd/DragSource.html)    [**NO FRAMES**](http://docs.google.com/DragSource.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#35nkun2) | [METHOD](#44sinio) |

## **java.awt.dnd**

Class DragSource

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **java.awt.dnd.DragSource**

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html)

public class **DragSource**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Serializable](http://docs.google.com/java/io/Serializable.html)

The DragSource is the entity responsible for the initiation of the Drag and Drop operation, and may be used in a number of scenarios:

* 1 default instance per JVM for the lifetime of that JVM.
* 1 instance per class of potential Drag Initiator object (e.g TextField). [implementation dependent]
* 1 per instance of a particular Component, or application specific object associated with a Component instance in the GUI. [implementation dependent]
* Some other arbitrary association. [implementation dependent]

Once the DragSource is obtained, a DragGestureRecognizer should also be obtained to associate the DragSource with a particular Component.

The initial interpretation of the user's gesture, and the subsequent starting of the drag operation are the responsibility of the implementing Component, which is usually implemented by a DragGestureRecognizer.

When a drag gesture occurs, the DragSource's startDrag() method shall be invoked in order to cause processing of the user's navigational gestures and delivery of Drag and Drop protocol notifications. A DragSource shall only permit a single Drag and Drop operation to be current at any one time, and shall reject any further startDrag() requests by throwing an IllegalDnDOperationException until such time as the extant operation is complete.

The startDrag() method invokes the createDragSourceContext() method to instantiate an appropriate DragSourceContext and associate the DragSourceContextPeer with that.

If the Drag and Drop System is unable to initiate a drag operation for some reason, the startDrag() method throws a java.awt.dnd.InvalidDnDOperationException to signal such a condition. Typically this exception is thrown when the underlying platform system is either not in a state to initiate a drag, or the parameters specified are invalid.

Note that during the drag, the set of operations exposed by the source at the start of the drag operation may not change until the operation is complete. The operation(s) are constant for the duration of the operation with respect to the DragSource.

**Since:** 1.2 **See Also:**[Serialized Form](http://docs.google.com/serialized-form.html#java.awt.dnd.DragSource)

| **Field Summary** | |
| --- | --- |
| static [Cursor](http://docs.google.com/java/awt/Cursor.html) | [**DefaultCopyDrop**](http://docs.google.com/java/awt/dnd/DragSource.html#DefaultCopyDrop)            The default Cursor to use with a copy operation indicating that a drop is currently allowed. |
| static [Cursor](http://docs.google.com/java/awt/Cursor.html) | [**DefaultCopyNoDrop**](http://docs.google.com/java/awt/dnd/DragSource.html#DefaultCopyNoDrop)            The default Cursor to use with a copy operation indicating that a drop is currently not allowed. |
| static [Cursor](http://docs.google.com/java/awt/Cursor.html) | [**DefaultLinkDrop**](http://docs.google.com/java/awt/dnd/DragSource.html#DefaultLinkDrop)            The default Cursor to use with a link operation indicating that a drop is currently allowed. |
| static [Cursor](http://docs.google.com/java/awt/Cursor.html) | [**DefaultLinkNoDrop**](http://docs.google.com/java/awt/dnd/DragSource.html#DefaultLinkNoDrop)            The default Cursor to use with a link operation indicating that a drop is currently not allowed. |
| static [Cursor](http://docs.google.com/java/awt/Cursor.html) | [**DefaultMoveDrop**](http://docs.google.com/java/awt/dnd/DragSource.html#DefaultMoveDrop)            The default Cursor to use with a move operation indicating that a drop is currently allowed. |
| static [Cursor](http://docs.google.com/java/awt/Cursor.html) | [**DefaultMoveNoDrop**](http://docs.google.com/java/awt/dnd/DragSource.html#DefaultMoveNoDrop)            The default Cursor to use with a move operation indicating that a drop is currently not allowed. |

| **Constructor Summary** | |
| --- | --- |
| [**DragSource**](http://docs.google.com/java/awt/dnd/DragSource.html#DragSource())()            Creates a new DragSource. |

| **Method Summary** | |
| --- | --- |
| void | [**addDragSourceListener**](http://docs.google.com/java/awt/dnd/DragSource.html#addDragSourceListener(java.awt.dnd.DragSourceListener))([DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)            Adds the specified DragSourceListener to this DragSource to receive drag source events during drag operations intiated with this DragSource. |
| void | [**addDragSourceMotionListener**](http://docs.google.com/java/awt/dnd/DragSource.html#addDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener))([DragSourceMotionListener](http://docs.google.com/java/awt/dnd/DragSourceMotionListener.html) dsml)            Adds the specified DragSourceMotionListener to this DragSource to receive drag motion events during drag operations intiated with this DragSource. |
| [DragGestureRecognizer](http://docs.google.com/java/awt/dnd/DragGestureRecognizer.html) | [**createDefaultDragGestureRecognizer**](http://docs.google.com/java/awt/dnd/DragSource.html#createDefaultDragGestureRecognizer(java.awt.Component,%20int,%20java.awt.dnd.DragGestureListener))([Component](http://docs.google.com/java/awt/Component.html) c, int actions, [DragGestureListener](http://docs.google.com/java/awt/dnd/DragGestureListener.html) dgl)            Creates a new DragGestureRecognizer that implements the default abstract subclass of DragGestureRecognizer for this DragSource, and sets the specified Component and DragGestureListener on the newly created object. |
| | <T extends [DragGestureRecognizer](http://docs.google.com/java/awt/dnd/DragGestureRecognizer.html)>  T | | --- | | [**createDragGestureRecognizer**](http://docs.google.com/java/awt/dnd/DragSource.html#createDragGestureRecognizer(java.lang.Class,%20java.awt.Component,%20int,%20java.awt.dnd.DragGestureListener))([Class](http://docs.google.com/java/lang/Class.html)<T> recognizerAbstractClass, [Component](http://docs.google.com/java/awt/Component.html) c, int actions, [DragGestureListener](http://docs.google.com/java/awt/dnd/DragGestureListener.html) dgl)            Creates a new DragGestureRecognizer that implements the specified abstract subclass of DragGestureRecognizer, and sets the specified Component and DragGestureListener on the newly created object. |
| protected  [DragSourceContext](http://docs.google.com/java/awt/dnd/DragSourceContext.html) | [**createDragSourceContext**](http://docs.google.com/java/awt/dnd/DragSource.html#createDragSourceContext(java.awt.dnd.peer.DragSourceContextPeer,%20java.awt.dnd.DragGestureEvent,%20java.awt.Cursor,%20java.awt.Image,%20java.awt.Point,%20java.awt.datatransfer.Transferable,%20java.awt.dnd.DragSourceListener))(java.awt.dnd.peer.DragSourceContextPeer dscp, [DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) dgl, [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor, [Image](http://docs.google.com/java/awt/Image.html) dragImage, [Point](http://docs.google.com/java/awt/Point.html) imageOffset, [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) t, [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)            Creates the DragSourceContext to handle this drag. |
| static [DragSource](http://docs.google.com/java/awt/dnd/DragSource.html) | [**getDefaultDragSource**](http://docs.google.com/java/awt/dnd/DragSource.html#getDefaultDragSource())()            Gets the DragSource object associated with the underlying platform. |
| [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html)[] | [**getDragSourceListeners**](http://docs.google.com/java/awt/dnd/DragSource.html#getDragSourceListeners())()            Gets all the DragSourceListeners registered with this DragSource. |
| [DragSourceMotionListener](http://docs.google.com/java/awt/dnd/DragSourceMotionListener.html)[] | [**getDragSourceMotionListeners**](http://docs.google.com/java/awt/dnd/DragSource.html#getDragSourceMotionListeners())()            Gets all of the DragSourceMotionListeners registered with this DragSource. |
| static int | [**getDragThreshold**](http://docs.google.com/java/awt/dnd/DragSource.html#getDragThreshold())()            Returns the drag gesture motion threshold. |
| [FlavorMap](http://docs.google.com/java/awt/datatransfer/FlavorMap.html) | [**getFlavorMap**](http://docs.google.com/java/awt/dnd/DragSource.html#getFlavorMap())()            This method returns the FlavorMap for this DragSource. |
| | <T extends [EventListener](http://docs.google.com/java/util/EventListener.html)>  T[] | | --- | | [**getListeners**](http://docs.google.com/java/awt/dnd/DragSource.html#getListeners(java.lang.Class))([Class](http://docs.google.com/java/lang/Class.html)<T> listenerType)            Gets all the objects currently registered as *Foo*Listeners upon this DragSource. |
| static boolean | [**isDragImageSupported**](http://docs.google.com/java/awt/dnd/DragSource.html#isDragImageSupported())()            Reports whether or not drag Image support is available on the underlying platform. |
| void | [**removeDragSourceListener**](http://docs.google.com/java/awt/dnd/DragSource.html#removeDragSourceListener(java.awt.dnd.DragSourceListener))([DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)            Removes the specified DragSourceListener from this DragSource. |
| void | [**removeDragSourceMotionListener**](http://docs.google.com/java/awt/dnd/DragSource.html#removeDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener))([DragSourceMotionListener](http://docs.google.com/java/awt/dnd/DragSourceMotionListener.html) dsml)            Removes the specified DragSourceMotionListener from this DragSource. |
| void | [**startDrag**](http://docs.google.com/java/awt/dnd/DragSource.html#startDrag(java.awt.dnd.DragGestureEvent,%20java.awt.Cursor,%20java.awt.Image,%20java.awt.Point,%20java.awt.datatransfer.Transferable,%20java.awt.dnd.DragSourceListener))([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) trigger, [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor, [Image](http://docs.google.com/java/awt/Image.html) dragImage, [Point](http://docs.google.com/java/awt/Point.html) dragOffset, [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable, [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)            Start a drag, given the DragGestureEvent that initiated the drag, the initial Cursor to use, the Image to drag, the offset of the Image origin from the hotspot of the Cursor at the instant of the trigger, the subject data of the drag, and the DragSourceListener. |
| void | [**startDrag**](http://docs.google.com/java/awt/dnd/DragSource.html#startDrag(java.awt.dnd.DragGestureEvent,%20java.awt.Cursor,%20java.awt.Image,%20java.awt.Point,%20java.awt.datatransfer.Transferable,%20java.awt.dnd.DragSourceListener,%20java.awt.datatransfer.FlavorMap))([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) trigger, [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor, [Image](http://docs.google.com/java/awt/Image.html) dragImage, [Point](http://docs.google.com/java/awt/Point.html) imageOffset, [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable, [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl, [FlavorMap](http://docs.google.com/java/awt/datatransfer/FlavorMap.html) flavorMap)            Start a drag, given the DragGestureEvent that initiated the drag, the initial Cursor to use, the Image to drag, the offset of the Image origin from the hotspot of the Cursor at the instant of the trigger, the Transferable subject data of the drag, the DragSourceListener, and the FlavorMap. |
| void | [**startDrag**](http://docs.google.com/java/awt/dnd/DragSource.html#startDrag(java.awt.dnd.DragGestureEvent,%20java.awt.Cursor,%20java.awt.datatransfer.Transferable,%20java.awt.dnd.DragSourceListener))([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) trigger, [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor, [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable, [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)            Start a drag, given the DragGestureEvent that initiated the drag, the initial Cursor to use, the Transferable subject data of the drag, and the DragSourceListener. |
| void | [**startDrag**](http://docs.google.com/java/awt/dnd/DragSource.html#startDrag(java.awt.dnd.DragGestureEvent,%20java.awt.Cursor,%20java.awt.datatransfer.Transferable,%20java.awt.dnd.DragSourceListener,%20java.awt.datatransfer.FlavorMap))([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) trigger, [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor, [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable, [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl, [FlavorMap](http://docs.google.com/java/awt/datatransfer/FlavorMap.html) flavorMap)            Start a drag, given the DragGestureEvent that initiated the drag, the initial Cursor to use, the Transferable subject data of the drag, the DragSourceListener, and the FlavorMap. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Field Detail** |
| --- |

### DefaultCopyDrop

public static final [Cursor](http://docs.google.com/java/awt/Cursor.html) **DefaultCopyDrop**

The default Cursor to use with a copy operation indicating that a drop is currently allowed. null if GraphicsEnvironment.isHeadless() returns true.

**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### DefaultMoveDrop

public static final [Cursor](http://docs.google.com/java/awt/Cursor.html) **DefaultMoveDrop**

The default Cursor to use with a move operation indicating that a drop is currently allowed. null if GraphicsEnvironment.isHeadless() returns true.

**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### DefaultLinkDrop

public static final [Cursor](http://docs.google.com/java/awt/Cursor.html) **DefaultLinkDrop**

The default Cursor to use with a link operation indicating that a drop is currently allowed. null if GraphicsEnvironment.isHeadless() returns true.

**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### DefaultCopyNoDrop

public static final [Cursor](http://docs.google.com/java/awt/Cursor.html) **DefaultCopyNoDrop**

The default Cursor to use with a copy operation indicating that a drop is currently not allowed. null if GraphicsEnvironment.isHeadless() returns true.

**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### DefaultMoveNoDrop

public static final [Cursor](http://docs.google.com/java/awt/Cursor.html) **DefaultMoveNoDrop**

The default Cursor to use with a move operation indicating that a drop is currently not allowed. null if GraphicsEnvironment.isHeadless() returns true.

**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### DefaultLinkNoDrop

public static final [Cursor](http://docs.google.com/java/awt/Cursor.html) **DefaultLinkNoDrop**

The default Cursor to use with a link operation indicating that a drop is currently not allowed. null if GraphicsEnvironment.isHeadless() returns true.

**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

| **Constructor Detail** |
| --- |

### DragSource

public **DragSource**()  
 throws [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html)

Creates a new DragSource.

**Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

| **Method Detail** |
| --- |

### getDefaultDragSource

public static [DragSource](http://docs.google.com/java/awt/dnd/DragSource.html) **getDefaultDragSource**()

Gets the DragSource object associated with the underlying platform.

**Returns:**the platform DragSource **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if GraphicsEnvironment.isHeadless() returns true**See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless())

### isDragImageSupported

public static boolean **isDragImageSupported**()

Reports whether or not drag Image support is available on the underlying platform.

**Returns:**if the Drag Image support is available on this platform

### startDrag

public void **startDrag**([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) trigger,  
 [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor,  
 [Image](http://docs.google.com/java/awt/Image.html) dragImage,  
 [Point](http://docs.google.com/java/awt/Point.html) imageOffset,  
 [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable,  
 [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl,  
 [FlavorMap](http://docs.google.com/java/awt/datatransfer/FlavorMap.html) flavorMap)  
 throws [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)

Start a drag, given the DragGestureEvent that initiated the drag, the initial Cursor to use, the Image to drag, the offset of the Image origin from the hotspot of the Cursor at the instant of the trigger, the Transferable subject data of the drag, the DragSourceListener, and the FlavorMap.

**Parameters:**trigger - the DragGestureEvent that initiated the dragdragCursor - the initial Cursor or null for defaultsdragImage - the image to drag or null,imageOffset - the offset of the Image origin from the hotspot of the Cursor at the instant of the triggertransferable - the subject data of the dragdsl - the DragSourceListenerflavorMap - the FlavorMap to use, or null

**Throws:** java.awt.dnd.InvalidDnDOperationException - if the Drag and Drop system is unable to initiate a drag operation, or if the user attempts to start a drag while an existing drag operation is still executing [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)

### startDrag

public void **startDrag**([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) trigger,  
 [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor,  
 [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable,  
 [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl,  
 [FlavorMap](http://docs.google.com/java/awt/datatransfer/FlavorMap.html) flavorMap)  
 throws [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)

Start a drag, given the DragGestureEvent that initiated the drag, the initial Cursor to use, the Transferable subject data of the drag, the DragSourceListener, and the FlavorMap.

**Parameters:**trigger - the DragGestureEvent that initiated the dragdragCursor - the initial Cursor or null for defaultstransferable - the subject data of the dragdsl - the DragSourceListenerflavorMap - the FlavorMap to use or null

**Throws:** java.awt.dnd.InvalidDnDOperationException - if the Drag and Drop system is unable to initiate a drag operation, or if the user attempts to start a drag while an existing drag operation is still executing [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)

### startDrag

public void **startDrag**([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) trigger,  
 [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor,  
 [Image](http://docs.google.com/java/awt/Image.html) dragImage,  
 [Point](http://docs.google.com/java/awt/Point.html) dragOffset,  
 [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable,  
 [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)  
 throws [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)

Start a drag, given the DragGestureEvent that initiated the drag, the initial Cursor to use, the Image to drag, the offset of the Image origin from the hotspot of the Cursor at the instant of the trigger, the subject data of the drag, and the DragSourceListener.

**Parameters:**trigger - the DragGestureEvent that initiated the dragdragCursor - the initial Cursor or null for defaultsdragImage - the Image to drag or nulldragOffset - the offset of the Image origin from the hotspot of the Cursor at the instant of the triggertransferable - the subject data of the dragdsl - the DragSourceListener

**Throws:** java.awt.dnd.InvalidDnDOperationException - if the Drag and Drop system is unable to initiate a drag operation, or if the user attempts to start a drag while an existing drag operation is still executing [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)

### startDrag

public void **startDrag**([DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) trigger,  
 [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor,  
 [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable,  
 [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)  
 throws [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)

Start a drag, given the DragGestureEvent that initiated the drag, the initial Cursor to use, the Transferable subject data of the drag, and the DragSourceListener.

**Parameters:**trigger - the DragGestureEvent that initiated the dragdragCursor - the initial Cursor or null for defaultstransferable - the subject data of the dragdsl - the DragSourceListener

**Throws:** java.awt.dnd.InvalidDnDOperationException - if the Drag and Drop system is unable to initiate a drag operation, or if the user attempts to start a drag while an existing drag operation is still executing [InvalidDnDOperationException](http://docs.google.com/java/awt/dnd/InvalidDnDOperationException.html)

### createDragSourceContext

protected [DragSourceContext](http://docs.google.com/java/awt/dnd/DragSourceContext.html) **createDragSourceContext**(java.awt.dnd.peer.DragSourceContextPeer dscp,  
 [DragGestureEvent](http://docs.google.com/java/awt/dnd/DragGestureEvent.html) dgl,  
 [Cursor](http://docs.google.com/java/awt/Cursor.html) dragCursor,  
 [Image](http://docs.google.com/java/awt/Image.html) dragImage,  
 [Point](http://docs.google.com/java/awt/Point.html) imageOffset,  
 [Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) t,  
 [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)

Creates the DragSourceContext to handle this drag.

To incorporate a new DragSourceContext subclass, subclass DragSource and override this method.

If dragImage is null, no image is used to represent the drag over feedback for this drag operation, but NullPointerException is not thrown.

If dsl is null, no drag source listener is registered with the created DragSourceContext, but NullPointerException is not thrown.

If dragCursor is null, the default drag cursors are used for this drag operation. NullPointerException is not thrown.

**Parameters:**dscp - The DragSourceContextPeer for this dragdgl - The DragGestureEvent that triggered the dragdragCursor - The initial Cursor to displaydragImage - The Image to drag or nullimageOffset - The offset of the Image origin from the hotspot of the cursor at the instant of the triggert - The subject data of the dragdsl - The DragSourceListener **Returns:**the DragSourceContext **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if dscp is null [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if dgl is null [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if dragImage is not null and imageOffset is null [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if t is null [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the Component associated with the trigger event is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the DragSource for the trigger event is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the drag action for the trigger event is DnDConstants.ACTION\_NONE. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the source actions for the DragGestureRecognizer associated with the trigger event are equal to DnDConstants.ACTION\_NONE.

### getFlavorMap

public [FlavorMap](http://docs.google.com/java/awt/datatransfer/FlavorMap.html) **getFlavorMap**()

This method returns the FlavorMap for this DragSource.

**Returns:**the FlavorMap for this DragSource

### createDragGestureRecognizer

public <T extends [DragGestureRecognizer](http://docs.google.com/java/awt/dnd/DragGestureRecognizer.html)> T **createDragGestureRecognizer**([Class](http://docs.google.com/java/lang/Class.html)<T> recognizerAbstractClass,  
 [Component](http://docs.google.com/java/awt/Component.html) c,  
 int actions,  
 [DragGestureListener](http://docs.google.com/java/awt/dnd/DragGestureListener.html) dgl)

Creates a new DragGestureRecognizer that implements the specified abstract subclass of DragGestureRecognizer, and sets the specified Component and DragGestureListener on the newly created object.

**Parameters:**recognizerAbstractClass - the requested abstract typeactions - the permitted source drag actionsc - the Component targetdgl - the DragGestureListener to notify

**Returns:**the new DragGestureRecognizer or null if the Toolkit.createDragGestureRecognizer method has no implementation available for the requested DragGestureRecognizer subclass and returns null

### createDefaultDragGestureRecognizer

public [DragGestureRecognizer](http://docs.google.com/java/awt/dnd/DragGestureRecognizer.html) **createDefaultDragGestureRecognizer**([Component](http://docs.google.com/java/awt/Component.html) c,  
 int actions,  
 [DragGestureListener](http://docs.google.com/java/awt/dnd/DragGestureListener.html) dgl)

Creates a new DragGestureRecognizer that implements the default abstract subclass of DragGestureRecognizer for this DragSource, and sets the specified Component and DragGestureListener on the newly created object. For this DragSource the default is MouseDragGestureRecognizer.

**Parameters:**c - the Component target for the recognizeractions - the permitted source actionsdgl - the DragGestureListener to notify

**Returns:**the new DragGestureRecognizer or null if the Toolkit.createDragGestureRecognizer method has no implementation available for the requested DragGestureRecognizer subclass and returns null

### addDragSourceListener

public void **addDragSourceListener**([DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)

Adds the specified DragSourceListener to this DragSource to receive drag source events during drag operations intiated with this DragSource. If a null listener is specified, no action is taken and no exception is thrown.

**Parameters:**dsl - the DragSourceListener to add**Since:** 1.4 **See Also:**[removeDragSourceListener(java.awt.dnd.DragSourceListener)](http://docs.google.com/java/awt/dnd/DragSource.html#removeDragSourceListener(java.awt.dnd.DragSourceListener)), [getDragSourceListeners()](http://docs.google.com/java/awt/dnd/DragSource.html#getDragSourceListeners())

### removeDragSourceListener

public void **removeDragSourceListener**([DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html) dsl)

Removes the specified DragSourceListener from this DragSource. If a null listener is specified, no action is taken and no exception is thrown. If the listener specified by the argument was not previously added to this DragSource, no action is taken and no exception is thrown.

**Parameters:**dsl - the DragSourceListener to remove**Since:** 1.4 **See Also:**[addDragSourceListener(java.awt.dnd.DragSourceListener)](http://docs.google.com/java/awt/dnd/DragSource.html#addDragSourceListener(java.awt.dnd.DragSourceListener)), [getDragSourceListeners()](http://docs.google.com/java/awt/dnd/DragSource.html#getDragSourceListeners())

### getDragSourceListeners

public [DragSourceListener](http://docs.google.com/java/awt/dnd/DragSourceListener.html)[] **getDragSourceListeners**()

Gets all the DragSourceListeners registered with this DragSource.

**Returns:**all of this DragSource's DragSourceListeners or an empty array if no such listeners are currently registered**Since:** 1.4 **See Also:**[addDragSourceListener(java.awt.dnd.DragSourceListener)](http://docs.google.com/java/awt/dnd/DragSource.html#addDragSourceListener(java.awt.dnd.DragSourceListener)), [removeDragSourceListener(java.awt.dnd.DragSourceListener)](http://docs.google.com/java/awt/dnd/DragSource.html#removeDragSourceListener(java.awt.dnd.DragSourceListener))

### addDragSourceMotionListener

public void **addDragSourceMotionListener**([DragSourceMotionListener](http://docs.google.com/java/awt/dnd/DragSourceMotionListener.html) dsml)

Adds the specified DragSourceMotionListener to this DragSource to receive drag motion events during drag operations intiated with this DragSource. If a null listener is specified, no action is taken and no exception is thrown.

**Parameters:**dsml - the DragSourceMotionListener to add**Since:** 1.4 **See Also:**[removeDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener)](http://docs.google.com/java/awt/dnd/DragSource.html#removeDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener)), [getDragSourceMotionListeners()](http://docs.google.com/java/awt/dnd/DragSource.html#getDragSourceMotionListeners())

### removeDragSourceMotionListener

public void **removeDragSourceMotionListener**([DragSourceMotionListener](http://docs.google.com/java/awt/dnd/DragSourceMotionListener.html) dsml)

Removes the specified DragSourceMotionListener from this DragSource. If a null listener is specified, no action is taken and no exception is thrown. If the listener specified by the argument was not previously added to this DragSource, no action is taken and no exception is thrown.

**Parameters:**dsml - the DragSourceMotionListener to remove**Since:** 1.4 **See Also:**[addDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener)](http://docs.google.com/java/awt/dnd/DragSource.html#addDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener)), [getDragSourceMotionListeners()](http://docs.google.com/java/awt/dnd/DragSource.html#getDragSourceMotionListeners())

### getDragSourceMotionListeners

public [DragSourceMotionListener](http://docs.google.com/java/awt/dnd/DragSourceMotionListener.html)[] **getDragSourceMotionListeners**()

Gets all of the DragSourceMotionListeners registered with this DragSource.

**Returns:**all of this DragSource's DragSourceMotionListeners or an empty array if no such listeners are currently registered**Since:** 1.4 **See Also:**[addDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener)](http://docs.google.com/java/awt/dnd/DragSource.html#addDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener)), [removeDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener)](http://docs.google.com/java/awt/dnd/DragSource.html#removeDragSourceMotionListener(java.awt.dnd.DragSourceMotionListener))

### getListeners

public <T extends [EventListener](http://docs.google.com/java/util/EventListener.html)> T[] **getListeners**([Class](http://docs.google.com/java/lang/Class.html)<T> listenerType)

Gets all the objects currently registered as *Foo*Listeners upon this DragSource. *Foo*Listeners are registered using the add*Foo*Listener method.

**Parameters:**listenerType - the type of listeners requested; this parameter should specify an interface that descends from java.util.EventListener **Returns:**an array of all objects registered as *Foo*Listeners on this DragSource, or an empty array if no such listeners have been added **Throws:** ClassCastException - if listenerType doesn't specify a class or interface that implements java.util.EventListener**Since:** 1.4 **See Also:**[getDragSourceListeners()](http://docs.google.com/java/awt/dnd/DragSource.html#getDragSourceListeners()), [getDragSourceMotionListeners()](http://docs.google.com/java/awt/dnd/DragSource.html#getDragSourceMotionListeners())

### getDragThreshold

public static int **getDragThreshold**()

Returns the drag gesture motion threshold. The drag gesture motion threshold defines the recommended behavior for [MouseDragGestureRecognizer](http://docs.google.com/java/awt/dnd/MouseDragGestureRecognizer.html)s.

If the system property awt.dnd.drag.threshold is set to a positive integer, this method returns the value of the system property; otherwise if a pertinent desktop property is available and supported by the implementation of the Java platform, this method returns the value of that property; otherwise this method returns some default value. The pertinent desktop property can be queried using java.awt.Toolkit.getDesktopProperty("DnD.gestureMotionThreshold").

**Returns:**the drag gesture motion threshold**Since:** 1.5 **See Also:**[MouseDragGestureRecognizer](http://docs.google.com/java/awt/dnd/MouseDragGestureRecognizer.html)

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/DragSource.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/awt/dnd/DragGestureRecognizer.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/dnd/DragSourceAdapter.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/dnd/DragSource.html)    [**NO FRAMES**](http://docs.google.com/DragSource.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#35nkun2) | [METHOD](#44sinio) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).