| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/DataFlavor.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/datatransfer/ClipboardOwner.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/datatransfer/FlavorEvent.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/datatransfer/DataFlavor.html)    [**NO FRAMES**](http://docs.google.com/DataFlavor.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#1ksv4uv) | [METHOD](#4i7ojhp) |

## **java.awt.datatransfer**

Class DataFlavor

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

**All Implemented Interfaces:** [Externalizable](http://docs.google.com/java/io/Externalizable.html), [Serializable](http://docs.google.com/java/io/Serializable.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html) **Direct Known Subclasses:** [ActivationDataFlavor](http://docs.google.com/javax/activation/ActivationDataFlavor.html)

public class **DataFlavor**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Externalizable](http://docs.google.com/java/io/Externalizable.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html)

A DataFlavor provides meta information about data. DataFlavor is typically used to access data on the clipboard, or during a drag and drop operation.

An instance of DataFlavor encapsulates a content type as defined in [RFC 2045](http://www.ietf.org/rfc/rfc2045.txt) and [RFC 2046](http://www.ietf.org/rfc/rfc2046.txt). A content type is typically referred to as a MIME type.

A content type consists of a media type (referred to as the primary type), a subtype, and optional parameters. See [RFC 2045](http://www.ietf.org/rfc/rfc2045.txt) for details on the syntax of a MIME type.

The JRE data transfer implementation interprets the parameter "class" of a MIME type as **a representation class**. The representation class reflects the class of the object being transferred. In other words, the representation class is the type of object returned by [Transferable.getTransferData(java.awt.datatransfer.DataFlavor)](http://docs.google.com/java/awt/datatransfer/Transferable.html#getTransferData(java.awt.datatransfer.DataFlavor)). For example, the MIME type of [imageFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#imageFlavor) is "image/x-java-image;class=java.awt.Image", the primary type is image, the subtype is x-java-image, and the representation class is java.awt.Image. When getTransferData is invoked with a DataFlavor of imageFlavor, an instance of java.awt.Image is returned. It's important to note that DataFlavor does no error checking against the representation class. It is up to consumers of DataFlavor, such as Transferable, to honor the representation class.

Note, if you do not specify a representation class when creating a DataFlavor, the default representation class is used. See appropriate documentation for DataFlavor's constructors.

Also, DataFlavor instances with the "text" primary MIME type may have a "charset" parameter. Refer to [RFC 2046](http://www.ietf.org/rfc/rfc2046.txt) and [selectBestTextFlavor(java.awt.datatransfer.DataFlavor[])](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#selectBestTextFlavor(java.awt.datatransfer.DataFlavor%5B%5D)) for details on "text" MIME types and the "charset" parameter.

Equality of DataFlavors is determined by the primary type, subtype, and representation class. Refer to [equals(DataFlavor)](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#equals(java.awt.datatransfer.DataFlavor)) for details. When determining equality, any optional parameters are ignored. For example, the following produces two DataFlavors that are considered identical:

DataFlavor flavor1 = new DataFlavor(Object.class, "X-test/test; class=<java.lang.Object>; foo=bar");  
 DataFlavor flavor2 = new DataFlavor(Object.class, "X-test/test; class=<java.lang.Object>; x=y");  
 // The following returns true.  
 flavor1.equals(flavor2);

As mentioned, flavor1 and flavor2 are considered identical. As such, asking a Transferable for either DataFlavor returns the same results.

For more information on the using data transfer with Swing see the  [How to Use Drag and Drop and Data Transfer](http://java.sun.com/docs/books/tutorial/uiswing/misc/dnd.html), section in *Java Tutorial*.

**See Also:**[Serialized Form](http://docs.google.com/serialized-form.html#java.awt.datatransfer.DataFlavor)

| **Field Summary** | |
| --- | --- |
| static [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) | [**imageFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#imageFlavor)            The DataFlavor representing a Java Image class, where: |
| static [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) | [**javaFileListFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#javaFileListFlavor)            To transfer a list of files to/from Java (and the underlying platform) a DataFlavor of this type/subtype and representation class of java.util.List is used. |
| static [String](http://docs.google.com/java/lang/String.html) | [**javaJVMLocalObjectMimeType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#javaJVMLocalObjectMimeType)            To transfer a reference to an arbitrary Java object reference that has no associated MIME Content-type, across a Transferable interface WITHIN THE SAME JVM, a DataFlavor with this type/subtype is used, with a representationClass equal to the type of the class/interface being passed across the Transferable. |
| static [String](http://docs.google.com/java/lang/String.html) | [**javaRemoteObjectMimeType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#javaRemoteObjectMimeType)            In order to pass a live link to a Remote object via a Drag and Drop ACTION\_LINK operation a Mime Content Type of application/x-java-remote-object should be used, where the representation class of the DataFlavor represents the type of the Remote interface to be transferred. |
| static [String](http://docs.google.com/java/lang/String.html) | [**javaSerializedObjectMimeType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#javaSerializedObjectMimeType)            A MIME Content-Type of application/x-java-serialized-object represents a graph of Java object(s) that have been made persistent. |
| static [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) | [**plainTextFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#plainTextFlavor)  **Deprecated.** *as of 1.3. Use DataFlavor.getReaderForText(Transferable) instead of Transferable.getTransferData(DataFlavor.plainTextFlavor).* |
| static [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) | [**stringFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#stringFlavor)            The DataFlavor representing a Java Unicode String class, where: |

| **Constructor Summary** | |
| --- | --- |
| [**DataFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#DataFlavor())()            Constructs a new DataFlavor. |
| [**DataFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#DataFlavor(java.lang.Class,%20java.lang.String))([Class](http://docs.google.com/java/lang/Class.html)<?> representationClass, [String](http://docs.google.com/java/lang/String.html) humanPresentableName)            Constructs a DataFlavor that represents a Java class. |
| [**DataFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#DataFlavor(java.lang.String))([String](http://docs.google.com/java/lang/String.html) mimeType)            Constructs a DataFlavor from a mimeType string. |
| [**DataFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#DataFlavor(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) mimeType, [String](http://docs.google.com/java/lang/String.html) humanPresentableName)            Constructs a DataFlavor that represents a MimeType. |
| [**DataFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#DataFlavor(java.lang.String,%20java.lang.String,%20java.lang.ClassLoader))([String](http://docs.google.com/java/lang/String.html) mimeType, [String](http://docs.google.com/java/lang/String.html) humanPresentableName, [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) classLoader)            Constructs a DataFlavor that represents a MimeType. |

| **Method Summary** | |
| --- | --- |
| [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#clone())()            Returns a clone of this DataFlavor. |
| boolean | [**equals**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#equals(java.awt.datatransfer.DataFlavor))([DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) that)            Tests a DataFlavor to this DataFlavor for equality. |
| boolean | [**equals**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) o)            Tests an arbitrary Object to this DataFlavor for equality. |
| boolean | [**equals**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#equals(java.lang.String))([String](http://docs.google.com/java/lang/String.html) s)  **Deprecated.** *As inconsistent with hashCode() contract, use isMimeTypeEqual(String) instead.* |
| [Class](http://docs.google.com/java/lang/Class.html)<?> | [**getDefaultRepresentationClass**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getDefaultRepresentationClass())() |
| [String](http://docs.google.com/java/lang/String.html) | [**getDefaultRepresentationClassAsString**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getDefaultRepresentationClassAsString())() |
| [String](http://docs.google.com/java/lang/String.html) | [**getHumanPresentableName**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getHumanPresentableName())()            Returns the human presentable name for the data format that this DataFlavor represents. |
| [String](http://docs.google.com/java/lang/String.html) | [**getMimeType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getMimeType())()            Returns the MIME type string for this DataFlavor. |
| [String](http://docs.google.com/java/lang/String.html) | [**getParameter**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getParameter(java.lang.String))([String](http://docs.google.com/java/lang/String.html) paramName)            Returns the human presentable name for this DataFlavor if paramName equals "humanPresentableName". |
| [String](http://docs.google.com/java/lang/String.html) | [**getPrimaryType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getPrimaryType())()            Returns the primary MIME type for this DataFlavor. |
| [Reader](http://docs.google.com/java/io/Reader.html) | [**getReaderForText**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getReaderForText(java.awt.datatransfer.Transferable))([Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable)            Gets a Reader for a text flavor, decoded, if necessary, for the expected charset (encoding). |
| [Class](http://docs.google.com/java/lang/Class.html)<?> | [**getRepresentationClass**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getRepresentationClass())()            Returns the Class which objects supporting this DataFlavor will return when this DataFlavor is requested. |
| [String](http://docs.google.com/java/lang/String.html) | [**getSubType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getSubType())()            Returns the sub MIME type of this DataFlavor. |
| static [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) | [**getTextPlainUnicodeFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#getTextPlainUnicodeFlavor())()            Returns a DataFlavor representing plain text with Unicode encoding, where: |
| int | [**hashCode**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#hashCode())()            Returns hash code for this DataFlavor. |
| boolean | [**isFlavorJavaFileListType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isFlavorJavaFileListType())()            Returns true if the DataFlavor specified represents a list of file objects. |
| boolean | [**isFlavorRemoteObjectType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isFlavorRemoteObjectType())()            Returns true if the DataFlavor specified represents a remote object. |
| boolean | [**isFlavorSerializedObjectType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isFlavorSerializedObjectType())()            Returns true if the DataFlavor specified represents a serialized object. |
| boolean | [**isFlavorTextType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isFlavorTextType())()            Returns whether this DataFlavor is a valid text flavor for this implementation of the Java platform. |
| boolean | [**isMimeTypeEqual**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isMimeTypeEqual(java.awt.datatransfer.DataFlavor))([DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) dataFlavor)            Compares the mimeType of two DataFlavor objects. |
| boolean | [**isMimeTypeEqual**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isMimeTypeEqual(java.lang.String))([String](http://docs.google.com/java/lang/String.html) mimeType)            Returns whether the string representation of the MIME type passed in is equivalent to the MIME type of this DataFlavor. |
| boolean | [**isMimeTypeSerializedObject**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isMimeTypeSerializedObject())()            Does the DataFlavor represent a serialized object? |
| boolean | [**isRepresentationClassByteBuffer**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isRepresentationClassByteBuffer())()            Returns whether the representation class for this DataFlavor is java.nio.ByteBuffer or a subclass thereof. |
| boolean | [**isRepresentationClassCharBuffer**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isRepresentationClassCharBuffer())()            Returns whether the representation class for this DataFlavor is java.nio.CharBuffer or a subclass thereof. |
| boolean | [**isRepresentationClassInputStream**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isRepresentationClassInputStream())()            Does the DataFlavor represent a java.io.InputStream? |
| boolean | [**isRepresentationClassReader**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isRepresentationClassReader())()            Returns whether the representation class for this DataFlavor is java.io.Reader or a subclass thereof. |
| boolean | [**isRepresentationClassRemote**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isRepresentationClassRemote())()            Returns true if the representation class is Remote. |
| boolean | [**isRepresentationClassSerializable**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#isRepresentationClassSerializable())()            Returns true if the representation class can be serialized. |
| boolean | [**match**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#match(java.awt.datatransfer.DataFlavor))([DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) that)            Tests a DataFlavor to this DataFlavor for equality. |
| protected  [String](http://docs.google.com/java/lang/String.html) | [**normalizeMimeType**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#normalizeMimeType(java.lang.String))([String](http://docs.google.com/java/lang/String.html) mimeType)  **Deprecated.** |
| protected  [String](http://docs.google.com/java/lang/String.html) | [**normalizeMimeTypeParameter**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#normalizeMimeTypeParameter(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) parameterName, [String](http://docs.google.com/java/lang/String.html) parameterValue)  **Deprecated.** |
| void | [**readExternal**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#readExternal(java.io.ObjectInput))([ObjectInput](http://docs.google.com/java/io/ObjectInput.html) is)            Restores this DataFlavor from a Serialized state. |
| static [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) | [**selectBestTextFlavor**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#selectBestTextFlavor(java.awt.datatransfer.DataFlavor%5B%5D))([DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html)[] availableFlavors)            Selects the best text DataFlavor from an array of DataFlavors. |
| void | [**setHumanPresentableName**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#setHumanPresentableName(java.lang.String))([String](http://docs.google.com/java/lang/String.html) humanPresentableName)            Sets the human presentable name for the data format that this DataFlavor represents. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#toString())()            String representation of this DataFlavor and its parameters. |
| protected static [Class](http://docs.google.com/java/lang/Class.html)<?> | [**tryToLoadClass**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#tryToLoadClass(java.lang.String,%20java.lang.ClassLoader))([String](http://docs.google.com/java/lang/String.html) className, [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) fallback)            Tries to load a class from: the bootstrap loader, the system loader, the context loader (if one is present) and finally the loader specified. |
| void | [**writeExternal**](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#writeExternal(java.io.ObjectOutput))([ObjectOutput](http://docs.google.com/java/io/ObjectOutput.html) os)            Serializes this DataFlavor. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [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** |
| --- |

### stringFlavor

public static final [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) **stringFlavor**

The DataFlavor representing a Java Unicode String class, where:

representationClass = java.lang.String  
 mimeType = "application/x-java-serialized-object"

### imageFlavor

public static final [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) **imageFlavor**

The DataFlavor representing a Java Image class, where:

representationClass = java.awt.Image  
 mimeType = "image/x-java-image"

### plainTextFlavor

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public static final [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) **plainTextFlavor**

**Deprecated.** *as of 1.3. Use DataFlavor.getReaderForText(Transferable) instead of Transferable.getTransferData(DataFlavor.plainTextFlavor).*The DataFlavor representing plain text with Unicode encoding, where:

representationClass = InputStream  
 mimeType = "text/plain; charset=unicode"

This DataFlavor has been **deprecated** because (1) Its representation is an InputStream, an 8-bit based representation, while Unicode is a 16-bit character set; and (2) The charset "unicode" is not well-defined. "unicode" implies a particular platform's implementation of Unicode, not a cross-platform implementation.

### javaSerializedObjectMimeType

public static final [String](http://docs.google.com/java/lang/String.html) **javaSerializedObjectMimeType**

A MIME Content-Type of application/x-java-serialized-object represents a graph of Java object(s) that have been made persistent. The representation class associated with this DataFlavor identifies the Java type of an object returned as a reference from an invocation java.awt.datatransfer.getTransferData.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.datatransfer.DataFlavor.javaSerializedObjectMimeType)

### javaFileListFlavor

public static final [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) **javaFileListFlavor**

To transfer a list of files to/from Java (and the underlying platform) a DataFlavor of this type/subtype and representation class of java.util.List is used. Each element of the list is required/guaranteed to be of type java.io.File.

### javaJVMLocalObjectMimeType

public static final [String](http://docs.google.com/java/lang/String.html) **javaJVMLocalObjectMimeType**

To transfer a reference to an arbitrary Java object reference that has no associated MIME Content-type, across a Transferable interface WITHIN THE SAME JVM, a DataFlavor with this type/subtype is used, with a representationClass equal to the type of the class/interface being passed across the Transferable.

The object reference returned from Transferable.getTransferData for a DataFlavor with this MIME Content-Type is required to be an instance of the representation Class of the DataFlavor.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.datatransfer.DataFlavor.javaJVMLocalObjectMimeType)

### javaRemoteObjectMimeType

public static final [String](http://docs.google.com/java/lang/String.html) **javaRemoteObjectMimeType**

In order to pass a live link to a Remote object via a Drag and Drop ACTION\_LINK operation a Mime Content Type of application/x-java-remote-object should be used, where the representation class of the DataFlavor represents the type of the Remote interface to be transferred.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.datatransfer.DataFlavor.javaRemoteObjectMimeType)

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

### DataFlavor

public **DataFlavor**()

Constructs a new DataFlavor. This constructor is provided only for the purpose of supporting the Externalizable interface. It is not intended for public (client) use.

**Since:** 1.2

### DataFlavor

public **DataFlavor**([Class](http://docs.google.com/java/lang/Class.html)<?> representationClass,  
 [String](http://docs.google.com/java/lang/String.html) humanPresentableName)

Constructs a DataFlavor that represents a Java class.

The returned DataFlavor will have the following characteristics:

representationClass = representationClass  
 mimeType = application/x-java-serialized-object

**Parameters:**representationClass - the class used to transfer data in this flavorhumanPresentableName - the human-readable string used to identify this flavor; if this parameter is null then the value of the the MIME Content Type is used **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if representationClass is null

### DataFlavor

public **DataFlavor**([String](http://docs.google.com/java/lang/String.html) mimeType,  
 [String](http://docs.google.com/java/lang/String.html) humanPresentableName)

Constructs a DataFlavor that represents a MimeType.

The returned DataFlavor will have the following characteristics:

If the mimeType is "application/x-java-serialized-object; class=<representation class>", the result is the same as calling new DataFlavor(Class:forName(<representation class>).

Otherwise:

representationClass = InputStream  
 mimeType = mimeType

**Parameters:**mimeType - the string used to identify the MIME type for this flavor; if the the mimeType does not specify a "class=" parameter, or if the class is not successfully loaded, then an IllegalArgumentException is thrownhumanPresentableName - the human-readable string used to identify this flavor; if this parameter is null then the value of the the MIME Content Type is used **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if mimeType is invalid or if the class is not successfully loaded [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if mimeType is null

### DataFlavor

public **DataFlavor**([String](http://docs.google.com/java/lang/String.html) mimeType,  
 [String](http://docs.google.com/java/lang/String.html) humanPresentableName,  
 [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) classLoader)  
 throws [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html)

Constructs a DataFlavor that represents a MimeType.

The returned DataFlavor will have the following characteristics:

If the mimeType is "application/x-java-serialized-object; class=<representation class>", the result is the same as calling new DataFlavor(Class:forName(<representation class>).

Otherwise:

representationClass = InputStream  
 mimeType = mimeType

**Parameters:**mimeType - the string used to identify the MIME type for this flavorhumanPresentableName - the human-readable string used to identify this flavorclassLoader - the class loader to use **Throws:** [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if the class is not loaded [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if mimeType is invalid [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if mimeType is null

### DataFlavor

public **DataFlavor**([String](http://docs.google.com/java/lang/String.html) mimeType)  
 throws [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html)

Constructs a DataFlavor from a mimeType string. The string can specify a "class=" parameter to create a DataFlavor with the desired representation class. If the string does not contain "class=" parameter, java.io.InputStream is used as default.

**Parameters:**mimeType - the string used to identify the MIME type for this flavor; if the class specified by "class=" parameter is not successfully loaded, then an ClassNotFoundException is thrown **Throws:** [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if the class is not loaded [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if mimeType is invalid [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if mimeType is null

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

### tryToLoadClass

protected static final [Class](http://docs.google.com/java/lang/Class.html)<?> **tryToLoadClass**([String](http://docs.google.com/java/lang/String.html) className,  
 [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) fallback)  
 throws [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html)

Tries to load a class from: the bootstrap loader, the system loader, the context loader (if one is present) and finally the loader specified.

**Parameters:**className - the name of the class to be loadedfallback - the fallback loader **Returns:**the class loaded **Throws:** [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if class is not found

### toString

public [String](http://docs.google.com/java/lang/String.html) **toString**()

String representation of this DataFlavor and its parameters. The resulting String contains the name of the DataFlavor class, this flavor's MIME type, and its representation class. If this flavor has a primary MIME type of "text", supports the charset parameter, and has an encoded representation, the flavor's charset is also included. See selectBestTextFlavor for a list of text flavors which support the charset parameter.

**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**string representation of this DataFlavor**See Also:**[selectBestTextFlavor(java.awt.datatransfer.DataFlavor[])](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#selectBestTextFlavor(java.awt.datatransfer.DataFlavor%5B%5D))

### getTextPlainUnicodeFlavor

public static final [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) **getTextPlainUnicodeFlavor**()

Returns a DataFlavor representing plain text with Unicode encoding, where:

representationClass = java.io.InputStream  
 mimeType = "text/plain;  
 charset=<platform default Unicode encoding>"

Sun's implementation for Microsoft Windows uses the encoding utf-16le. Sun's implementation for Solaris and Linux uses the encoding iso-10646-ucs-2.

**Returns:**a DataFlavor representing plain text with Unicode encoding**Since:** 1.3

### selectBestTextFlavor

public static final [DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) **selectBestTextFlavor**([DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html)[] availableFlavors)

Selects the best text DataFlavor from an array of DataFlavors. Only DataFlavor.stringFlavor, and equivalent flavors, and flavors that have a primary MIME type of "text", are considered for selection.

Flavors are first sorted by their MIME types in the following order:

* "text/sgml"
* "text/xml"
* "text/html"
* "text/rtf"
* "text/enriched"
* "text/richtext"
* "text/uri-list"
* "text/tab-separated-values"
* "text/t140"
* "text/rfc822-headers"
* "text/parityfec"
* "text/directory"
* "text/css"
* "text/calendar"
* "application/x-java-serialized-object"
* "text/plain"
* "text/<other>"

For example, "text/sgml" will be selected over "text/html", and DataFlavor.stringFlavor will be chosen over DataFlavor.plainTextFlavor.

If two or more flavors share the best MIME type in the array, then that MIME type will be checked to see if it supports the charset parameter.

The following MIME types support, or are treated as though they support, the charset parameter:

* "text/sgml"
* "text/xml"
* "text/html"
* "text/enriched"
* "text/richtext"
* "text/uri-list"
* "text/directory"
* "text/css"
* "text/calendar"
* "application/x-java-serialized-object"
* "text/plain"

The following MIME types do not support, or are treated as though they do not support, the charset parameter:

* "text/rtf"
* "text/tab-separated-values"
* "text/t140"
* "text/rfc822-headers"
* "text/parityfec"

For "text/<other>" MIME types, the first time the JRE needs to determine whether the MIME type supports the charset parameter, it will check whether the parameter is explicitly listed in an arbitrarily chosen DataFlavor which uses that MIME type. If so, the JRE will assume from that point on that the MIME type supports the charset parameter and will not check again. If the parameter is not explicitly listed, the JRE will assume from that point on that the MIME type does not support the charset parameter and will not check again. Because this check is performed on an arbitrarily chosen DataFlavor, developers must ensure that all DataFlavors with a "text/<other>" MIME type specify the charset parameter if it is supported by that MIME type. Developers should never rely on the JRE to substitute the platform's default charset for a "text/<other>" DataFlavor. Failure to adhere to this restriction will lead to undefined behavior.

If the best MIME type in the array does not support the charset parameter, the flavors which share that MIME type will then be sorted by their representation classes in the following order: java.io.InputStream, java.nio.ByteBuffer, [B, <all others>.

If two or more flavors share the best representation class, or if no flavor has one of the three specified representations, then one of those flavors will be chosen non-deterministically.

If the best MIME type in the array does support the charset parameter, the flavors which share that MIME type will then be sorted by their representation classes in the following order: java.io.Reader, java.lang.String, java.nio.CharBuffer, [C, <all others>.

If two or more flavors share the best representation class, and that representation is one of the four explicitly listed, then one of those flavors will be chosen non-deterministically. If, however, no flavor has one of the four specified representations, the flavors will then be sorted by their charsets. Unicode charsets, such as "UTF-16", "UTF-8", "UTF-16BE", "UTF-16LE", and their aliases, are considered best. After them, the platform default charset and its aliases are selected. "US-ASCII" and its aliases are worst. All other charsets are chosen in alphabetical order, but only charsets supported by this implementation of the Java platform will be considered.

If two or more flavors share the best charset, the flavors will then again be sorted by their representation classes in the following order: java.io.InputStream, java.nio.ByteBuffer, [B, <all others>.

If two or more flavors share the best representation class, or if no flavor has one of the three specified representations, then one of those flavors will be chosen non-deterministically.

**Parameters:**availableFlavors - an array of available DataFlavors **Returns:**the best (highest fidelity) flavor according to the rules specified above, or null, if availableFlavors is null, has zero length, or contains no text flavors**Since:** 1.3

### getReaderForText

public [Reader](http://docs.google.com/java/io/Reader.html) **getReaderForText**([Transferable](http://docs.google.com/java/awt/datatransfer/Transferable.html) transferable)  
 throws [UnsupportedFlavorException](http://docs.google.com/java/awt/datatransfer/UnsupportedFlavorException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html)

Gets a Reader for a text flavor, decoded, if necessary, for the expected charset (encoding). The supported representation classes are java.io.Reader, java.lang.String, java.nio.CharBuffer, [C, java.io.InputStream, java.nio.ByteBuffer, and [B.

Because text flavors which do not support the charset parameter are encoded in a non-standard format, this method should not be called for such flavors. However, in order to maintain backward-compatibility, if this method is called for such a flavor, this method will treat the flavor as though it supports the charset parameter and attempt to decode it accordingly. See selectBestTextFlavor for a list of text flavors which do not support the charset parameter.

**Parameters:**transferable - the Transferable whose data will be requested in this flavor **Returns:**a Reader to read the Transferable's data **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the representation class is not one of the seven listed above [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the Transferable has null data [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the Transferable is null [UnsupportedEncodingException](http://docs.google.com/java/io/UnsupportedEncodingException.html) - if this flavor's representation is java.io.InputStream, java.nio.ByteBuffer, or [B and this flavor's encoding is not supported by this implementation of the Java platform [UnsupportedFlavorException](http://docs.google.com/java/awt/datatransfer/UnsupportedFlavorException.html) - if the Transferable does not support this flavor [IOException](http://docs.google.com/java/io/IOException.html) - if the data cannot be read because of an I/O error**Since:** 1.3 **See Also:**[selectBestTextFlavor(java.awt.datatransfer.DataFlavor[])](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#selectBestTextFlavor(java.awt.datatransfer.DataFlavor%5B%5D))

### getMimeType

public [String](http://docs.google.com/java/lang/String.html) **getMimeType**()

Returns the MIME type string for this DataFlavor.

**Returns:**the MIME type string for this flavor

### getRepresentationClass

public [Class](http://docs.google.com/java/lang/Class.html)<?> **getRepresentationClass**()

Returns the Class which objects supporting this DataFlavor will return when this DataFlavor is requested.

**Returns:**the Class which objects supporting this DataFlavor will return when this DataFlavor is requested

### getHumanPresentableName

public [String](http://docs.google.com/java/lang/String.html) **getHumanPresentableName**()

Returns the human presentable name for the data format that this DataFlavor represents. This name would be localized for different countries.

**Returns:**the human presentable name for the data format that this DataFlavor represents

### getPrimaryType

public [String](http://docs.google.com/java/lang/String.html) **getPrimaryType**()

Returns the primary MIME type for this DataFlavor.

**Returns:**the primary MIME type of this DataFlavor

### getSubType

public [String](http://docs.google.com/java/lang/String.html) **getSubType**()

Returns the sub MIME type of this DataFlavor.

**Returns:**the Sub MIME type of this DataFlavor

### getParameter

public [String](http://docs.google.com/java/lang/String.html) **getParameter**([String](http://docs.google.com/java/lang/String.html) paramName)

Returns the human presentable name for this DataFlavor if paramName equals "humanPresentableName". Otherwise returns the MIME type value associated with paramName.

**Parameters:**paramName - the parameter name requested **Returns:**the value of the name parameter, or null if there is no associated value

### setHumanPresentableName

public void **setHumanPresentableName**([String](http://docs.google.com/java/lang/String.html) humanPresentableName)

Sets the human presentable name for the data format that this DataFlavor represents. This name would be localized for different countries.

**Parameters:**humanPresentableName - the new human presentable name

### equals

public boolean **equals**([Object](http://docs.google.com/java/lang/Object.html) o)

Tests an arbitrary Object to this DataFlavor for equality. Two DataFlavors are considered equal if and only if their MIME primary type and subtype and representation class are equal. Additionally, if the primary type is "text", the subtype denotes a text flavor which supports the charset parameter, and the representation class is not java.io.Reader, java.lang.String, java.nio.CharBuffer, or [C, the charset parameter must also be equal. If a charset is not explicitly specified for one or both DataFlavors, the platform default encoding is assumed. See selectBestTextFlavor for a list of text flavors which support the charset parameter.

**Overrides:**[equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)) in class [Object](http://docs.google.com/java/lang/Object.html) **Parameters:**o - the Object to compare with this **Returns:**true if that is equivalent to this DataFlavor; false otherwise**See Also:**[selectBestTextFlavor(java.awt.datatransfer.DataFlavor[])](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#selectBestTextFlavor(java.awt.datatransfer.DataFlavor%5B%5D))

### equals

public boolean **equals**([DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) that)

Tests a DataFlavor to this DataFlavor for equality. Two DataFlavors are considered equal if and only if their MIME primary type and subtype and representation class are equal. Additionally, if the primary type is "text", the subtype denotes a text flavor which supports the charset parameter, and the representation class is not java.io.Reader, java.lang.String, java.nio.CharBuffer, or [C, the charset parameter must also be equal. If a charset is not explicitly specified for one or both DataFlavors, the platform default encoding is assumed. See selectBestTextFlavor for a list of text flavors which support the charset parameter.

**Parameters:**that - the DataFlavor to compare with this **Returns:**true if that is equivalent to this DataFlavor; false otherwise**See Also:**[selectBestTextFlavor(java.awt.datatransfer.DataFlavor[])](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#selectBestTextFlavor(java.awt.datatransfer.DataFlavor%5B%5D))

### equals

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public boolean **equals**([String](http://docs.google.com/java/lang/String.html) s)

**Deprecated.** *As inconsistent with hashCode() contract, use isMimeTypeEqual(String) instead.*

Compares only the mimeType against the passed in String and representationClass is not considered in the comparison. If representationClass needs to be compared, then equals(new DataFlavor(s)) may be used.

**Returns:**true if the String (MimeType) is equal

### hashCode

public int **hashCode**()

Returns hash code for this DataFlavor. For two equal DataFlavors, hash codes are equal. For the String that matches DataFlavor.equals(String), it is not guaranteed that DataFlavor's hash code is equal to the hash code of the String.

**Overrides:**[hashCode](http://docs.google.com/java/lang/Object.html#hashCode()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a hash code for this DataFlavor**See Also:**[Object.equals(java.lang.Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### match

public boolean **match**([DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) that)

Tests a DataFlavor to this DataFlavor for equality. Two DataFlavors are considered equal if and only if their MIME primary type and subtype and representation class are equal. Additionally, if the primary type is "text", the subtype denotes a text flavor which supports the charset parameter, and the representation class is not java.io.Reader, java.lang.String, java.nio.CharBuffer, or [C, the charset parameter must also be equal. If a charset is not explicitly specified for one or both DataFlavors, the platform default encoding is assumed. See selectBestTextFlavor for a list of text flavors which support the charset parameter.

**Parameters:**that - the DataFlavor to compare with this **Returns:**true if that is equivalent to this DataFlavor; false otherwise**Since:** 1.3 **See Also:**[selectBestTextFlavor(java.awt.datatransfer.DataFlavor[])](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#selectBestTextFlavor(java.awt.datatransfer.DataFlavor%5B%5D))

### isMimeTypeEqual

public boolean **isMimeTypeEqual**([String](http://docs.google.com/java/lang/String.html) mimeType)

Returns whether the string representation of the MIME type passed in is equivalent to the MIME type of this DataFlavor. Parameters are not included in the comparison.

**Parameters:**mimeType - the string representation of the MIME type **Returns:**true if the string representation of the MIME type passed in is equivalent to the MIME type of this DataFlavor; false otherwise **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if mimeType is null

### isMimeTypeEqual

public final boolean **isMimeTypeEqual**([DataFlavor](http://docs.google.com/java/awt/datatransfer/DataFlavor.html) dataFlavor)

Compares the mimeType of two DataFlavor objects. No parameters are considered.

**Parameters:**dataFlavor - the DataFlavor to be compared **Returns:**true if the MimeTypes are equal, otherwise false

### isMimeTypeSerializedObject

public boolean **isMimeTypeSerializedObject**()

Does the DataFlavor represent a serialized object?

### getDefaultRepresentationClass

public final [Class](http://docs.google.com/java/lang/Class.html)<?> **getDefaultRepresentationClass**()

### getDefaultRepresentationClassAsString

public final [String](http://docs.google.com/java/lang/String.html) **getDefaultRepresentationClassAsString**()

### isRepresentationClassInputStream

public boolean **isRepresentationClassInputStream**()

Does the DataFlavor represent a java.io.InputStream?

### isRepresentationClassReader

public boolean **isRepresentationClassReader**()

Returns whether the representation class for this DataFlavor is java.io.Reader or a subclass thereof.

**Since:** 1.4

### isRepresentationClassCharBuffer

public boolean **isRepresentationClassCharBuffer**()

Returns whether the representation class for this DataFlavor is java.nio.CharBuffer or a subclass thereof.

**Since:** 1.4

### isRepresentationClassByteBuffer

public boolean **isRepresentationClassByteBuffer**()

Returns whether the representation class for this DataFlavor is java.nio.ByteBuffer or a subclass thereof.

**Since:** 1.4

### isRepresentationClassSerializable

public boolean **isRepresentationClassSerializable**()

Returns true if the representation class can be serialized.

**Returns:**true if the representation class can be serialized

### isRepresentationClassRemote

public boolean **isRepresentationClassRemote**()

Returns true if the representation class is Remote.

**Returns:**true if the representation class is Remote

### isFlavorSerializedObjectType

public boolean **isFlavorSerializedObjectType**()

Returns true if the DataFlavor specified represents a serialized object.

**Returns:**true if the DataFlavor specified represents a Serialized Object

### isFlavorRemoteObjectType

public boolean **isFlavorRemoteObjectType**()

Returns true if the DataFlavor specified represents a remote object.

**Returns:**true if the DataFlavor specified represents a Remote Object

### isFlavorJavaFileListType

public boolean **isFlavorJavaFileListType**()

Returns true if the DataFlavor specified represents a list of file objects.

**Returns:**true if the DataFlavor specified represents a List of File objects

### isFlavorTextType

public boolean **isFlavorTextType**()

Returns whether this DataFlavor is a valid text flavor for this implementation of the Java platform. Only flavors equivalent to DataFlavor.stringFlavor and DataFlavors with a primary MIME type of "text" can be valid text flavors.

If this flavor supports the charset parameter, it must be equivalent to DataFlavor.stringFlavor, or its representation must be java.io.Reader, java.lang.String, java.nio.CharBuffer, [C, java.io.InputStream, java.nio.ByteBuffer, or [B. If the representation is java.io.InputStream, java.nio.ByteBuffer, or [B, then this flavor's charset parameter must be supported by this implementation of the Java platform. If a charset is not specified, then the platform default charset, which is always supported, is assumed.

If this flavor does not support the charset parameter, its representation must be java.io.InputStream, java.nio.ByteBuffer, or [B.

See selectBestTextFlavor for a list of text flavors which support the charset parameter.

**Returns:**true if this DataFlavor is a valid text flavor as described above; false otherwise**Since:** 1.4 **See Also:**[selectBestTextFlavor(java.awt.datatransfer.DataFlavor[])](http://docs.google.com/java/awt/datatransfer/DataFlavor.html#selectBestTextFlavor(java.awt.datatransfer.DataFlavor%5B%5D))

### writeExternal

public void **writeExternal**([ObjectOutput](http://docs.google.com/java/io/ObjectOutput.html) os)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Serializes this DataFlavor.

**Specified by:**[writeExternal](http://docs.google.com/java/io/Externalizable.html#writeExternal(java.io.ObjectOutput)) in interface [Externalizable](http://docs.google.com/java/io/Externalizable.html) **Parameters:**os - the stream to write the object to **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - Includes any I/O exceptions that may occur

### readExternal

public void **readExternal**([ObjectInput](http://docs.google.com/java/io/ObjectInput.html) is)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html)

Restores this DataFlavor from a Serialized state.

**Specified by:**[readExternal](http://docs.google.com/java/io/Externalizable.html#readExternal(java.io.ObjectInput)) in interface [Externalizable](http://docs.google.com/java/io/Externalizable.html) **Parameters:**is - the stream to read data from in order to restore the object **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if I/O errors occur [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - If the class for an object being restored cannot be found.

### clone

public [Object](http://docs.google.com/java/lang/Object.html) **clone**()  
 throws [CloneNotSupportedException](http://docs.google.com/java/lang/CloneNotSupportedException.html)

Returns a clone of this DataFlavor.

**Overrides:**[clone](http://docs.google.com/java/lang/Object.html#clone()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a clone of this DataFlavor **Throws:** [CloneNotSupportedException](http://docs.google.com/java/lang/CloneNotSupportedException.html) - if the object's class does not support the Cloneable interface. Subclasses that override the clone method can also throw this exception to indicate that an instance cannot be cloned.**See Also:**[Cloneable](http://docs.google.com/java/lang/Cloneable.html)

### normalizeMimeTypeParameter

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
protected [String](http://docs.google.com/java/lang/String.html) **normalizeMimeTypeParameter**([String](http://docs.google.com/java/lang/String.html) parameterName,  
 [String](http://docs.google.com/java/lang/String.html) parameterValue)

**Deprecated.**

Called on DataFlavor for every MIME Type parameter to allow DataFlavor subclasses to handle special parameters like the text/plain charset parameters, whose values are case insensitive. (MIME type parameter values are supposed to be case sensitive.

This method is called for each parameter name/value pair and should return the normalized representation of the parameterValue. This method is never invoked by this implementation from 1.1 onwards.

### normalizeMimeType

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
protected [String](http://docs.google.com/java/lang/String.html) **normalizeMimeType**([String](http://docs.google.com/java/lang/String.html) mimeType)

**Deprecated.**

Called for each MIME type string to give DataFlavor subtypes the opportunity to change how the normalization of MIME types is accomplished. One possible use would be to add default parameter/value pairs in cases where none are present in the MIME type string passed in. This method is never invoked by this implementation from 1.1 onwards.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/DataFlavor.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/datatransfer/ClipboardOwner.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/datatransfer/FlavorEvent.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/datatransfer/DataFlavor.html)    [**NO FRAMES**](http://docs.google.com/DataFlavor.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#1ksv4uv) | [METHOD](#4i7ojhp) |

[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).