20

java.awt.datatransfer Reference

20.1 Clipboard *

Description
The Clipboard class is a repository for a Transferable object and can be used for cut, copy, and paste operations. The system clipboard can be accessed by calling Toolkit.getDefaultToolkit().getSystemClipboard(). You can use this technique if you are interested in exchanging data between your application and other applications (Java or non-Java) running on the system. In addition, Clipboard can be instantiated directly, if “private” clipboards are needed.

Class Definition
```java
public class java.awt.datatransfer.Clipboard
    extends java.lang.Object {

    // Variables
    protected Transferable contents;
    protected ClipboardOwner owner;

    // Constructors
    public Clipboard (String name);

    // Instance Methods
    public synchronized Transferable getContents (Object requestor);
    public String getName();
    public synchronized void setContents (Transferable contents, ClipboardOwner owner);
}
```


### Variables

**contents**

protected Transferable contents

The object that the Clipboard contains, i.e., the object that has been cut or copied.

**owner**

protected ClipboardOwner owner

The object that owns the contents. When something else is placed on the clipboard, owner is notified via lostOwnership().

### Constructors

**Clipboard**

```java
public Clipboard (String name)
```

Parameters

- **name**
  
  The name for this Clipboard.

Description

Constructs a Clipboard object with the given name.

### Instance Methods

**getContents**

```java
public synchronized Transferable getContents (Object requestor)
```

Parameters

- **requestor**
  
  The object asking for the contents.

Returns

An object that implements the Transferable interface.

Description

Returns the current contents of the Clipboard. You could use this method to paste data from the clipboard into your application.

**getName**

```java
public String getName()
```

Returns

Clipboard’s name.

Description

Returns the name used when this clipboard was constructed.

Toolkit.getSystemClipboard() returns a Clipboard named “System”.

**setContents**
public synchronized void setContents (Transferable contents, ClipboardOwner owner)

Parameters  contents  New contents.
            owner    Owner of the new contents.

Description  Changes the contents of the Clipboard. You could use this method to cut or copy data from your application to the clipboard.

See Also  ClipboardOwner, Toolkit, Transferable

20.2  ClipboardOwner *

Description  ClipboardOwner is implemented by classes that want to be notified when someone else sets the contents of a clipboard.

Interface Definition  

```java
public abstract interface java.awt.datatransfer.ClipboardOwner {

    // Interface Methods
    public abstract void lostOwnership (Clipboard clipboard, Transferable contents);
}
```

Interface Methods  

lostOwnership  

```java
public abstract void lostOwnership (Clipboard clipboard, Transferable contents)
```

Parameters  clipboard  The clipboard whose contents have changed.
            contents  The contents that this owner originally put on the clipboard.

Description  Tells the ClipboardOwner that the contents it placed on the given clipboard are no longer there.

See Also  Clipboard, StringSelection, Transferable
20.3 DataFlavor ★

Description
The DataFlavor class encapsulates information about data formats.

Class Definition
```java
public class java.awt.datatransfer.DataFlavor
    extends java.lang.Object {

    // Class Variables
    public static DataFlavor plainTextFlavor;
    public static DataFlavor stringFlavor;

    // Constructors
    public DataFlavor (Class representationClass, String humanPresentableName);
    public DataFlavor (String MIMEType, String humanPresentableName);

    // Instance Methods
    public boolean equals (DataFlavor dataFlavor);
    public String getHumanPresentableName();
    public String getMIMEType();
    public Class getRepresentationClass();
    public boolean isMIMETypeEqual (String MIMEType);
    public final boolean isMIMETypeEqual (DataFlavor dataFlavor);
    public void setHumanPresentableName (String humanPresentableName);

    // Protected Instance Methods
    protected String normalizeMIMEType (String MIMEType);
    protected String normalizeMIMETypeParameter (String parameterName, String parameterValue);
}
```

Class Variables
plainTextFlavor
```java
public static DataFlavor plainTextFlavor
A preset DataFlavor object representing plain text.
```

stringFlavor
```java
public static DataFlavor stringFlavor
A preset DataFlavor object representing a Java String.
```
**Constructors**

**DataFlavor**

```java
public DataFlavor (Class representationClass, String humanPresentableName)
```

- **Parameters**
  - `representationClass`  
    The Java class that represents data in this flavor.
  - `humanPresentableName`  
    A name for this flavor that humans will recognize.

- **Description**  
  Constructs a `DataFlavor` object with the given characteristics.

```java
public DataFlavor (String MIMEType, String humanPresentableName)
```

- **Parameters**
  - `MIMEType`  
    The MIME type string this `DataFlavor` represents.
  - `humanPresentableName`  
    A name for this flavor that humans will recognize.

- **Description**  
  Constructs a `DataFlavor` object with the given characteristics.

  The representation class used for this `DataFlavor` is `java.io.InputStream`.

**Instance Methods**

**equals**

```java
public boolean equals (DataFlavor dataFlavor)
```

- **Parameters**
  - `dataFlavor`  
    The flavor to compare.

- **Returns**
  - `true` if `dataFlavor` is equivalent to this `DataFlavor`, `false` otherwise.

- **Description**  
  Compares two different `DataFlavor` instances for equivalence.

**getHumanPresentableName**

```java
public String getHumanPresentableName()
```

- **Returns**  
  The name of this flavor.

---

* The type name changed to `x-java-serialized-object` in the 1.1.1 release.
getMIMEType
public String getMIMEType()
Returns The MIME type string for this flavor.

getRepresentationClass
public Class getRepresentationClass()
Returns The Java class that will be used to represent data in this flavor.

isMIMETypeEqual
public boolean isMIMETypeEqual (String MIMEType)
Parameters MIMEType The type to compare.
Returns true if the given MIME type is the same as this DataFlavor’s MIME type; false otherwise.
Description Compares two different DataFlavor MIME types for equivalence.

public final boolean isMIMETypeEqual (DataFlavor dataFlavor)
Parameters dataFlavor The flavor to compare.
Returns true if DataFlavor’s MIME type is the same as this DataFlavor’s MIME type; false otherwise.
Description Compares two different DataFlavor MIME types for equivalence.

setHumanPresentableName
public void setHumanPresentableName (String humanPresentableName)
Parameters humanPresentableName A name for this flavor that humans will recognize.
Description Changes the name of the DataFlavor.

Protected Instance Methods
normalizeMIMEType
protected String normalizeMIMEType (String MIMEType)
Parameters MIMEType The MIME type string to normalize.
Returns Normalized MIME type string.
This method is called for each MIME type string. Subclasses can override this method to add default parameter/value pairs to MIME strings.

**normalizeMIMETypeParameter**

```java
protected String normalizeMIMETypeParameter (String parameterName, String parameterValue)
```

**Parameters**
- `parameterName` : The MIME type parameter to normalize.
- `parameterValue` : The corresponding value.

**Returns** Normalized MIME type parameter string.

This method is called for each MIME type parameter string. Subclasses can override this method to handle special parameters, such as those that are case-insensitive.

**See Also**
Class, String

## 20.4 StringSelection ★

**Description**
StringSelection is a “convenience” class that can be used for copy and paste operations on Unicode text strings. For example, you could place a string on the system’s clipboard with the following code:

```java
Clipboard c =
    Toolkit.getDefaultToolkit().getSystemClipboard();
StringSelection s = new StringSelection("Be safe when you cut and paste.");
    c.setContents(s, s);
```

**Class Definition**

```java
public class java.awt.datatransfer.StringSelection
    extends java.lang.Object
    implements java.awt.datatransfer.ClipboardOwner,
        java.awt.datatransfer.Transferable {

    // Constructor
    public StringSelection(String data);

    // Instance Methods
```
public synchronized Object getTransferData (DataFlavor flavor) throws UnsupportedFlavorException, IOException;
public synchronized DataFlavor[] getTransferDataFlavors();
public boolean isDataFlavorSupported (DataFlavor flavor);
public void lostOwnership (Clipboard clipboard, Transferable contents);
}

Constructors
StringSelection
public StringSelection (String data)
Parameters  data  The string to be placed in a clipboard.
Description  Constructs a StringSelection object from the given string.

Instance Methods
getTransferData
public synchronized Object getTransferData (DataFlavor flavor) throws UnsupportedFlavorException, IOException
Parameters  flavor  The requested flavor for the returned data, which can be either DataFlavor.stringFlavor or DataFlavor.plainTextFlavor.
Returns  The string that the StringSelection was constructed with. This is returned either as a String object or a Reader object, depending on the flavor requested.
Throws  UnsupportedFlavorException  If the requested flavor is not supported.
        IOException  If a Reader representing the string could not be created.
Implements  Transferable.getTransferData(DataFlavor)
Description  Returns the string this StringSelection represents. This is returned either as a String object or a Reader object, depending on the flavor requested.

getTransferDataFlavors
public synchronized DataFlavor[] getTransferDataFlavors()
Returns  An array of the data flavors the StringSelection supports.
Implements  Transferable.getTransferDataFlavors()
Description  DataFlavor.stringFlavor and DataFlavor.plainTextFlavor are returned.
**isDataFlavorSupported**

```java
public boolean isDataFlavorSupported (DataFlavor flavor)
```

**Parameters**
- `flavor` The flavor in question.

**Returns**
- `true` if `flavor` is supported; `false` otherwise.

**Implements**
- Transferable.isDataFlavorSupported(DataFlavor)

**lostOwnership**

```java
public void lostOwnership (Clipboard clipboard, Transferable contents)
```

**Parameters**
- `clipboard` The clipboard whose contents are changing.
- `contents` The contents that were on the clipboard.

**Implements**
- ClipboardOwner.lostOwnership(Clipboard, Transferable)

**Description**
- Does nothing.

**See Also**
- Clipboard, ClipboardOwner, DataFlavor, String, Transferable

---

### 20.5 Transferable ★

**Description**

The `Transferable` interface is implemented by objects that can be placed on Clipboards.

**Interface Definition**

```java
public abstract interface Transferable {

    // Instance Methods
    public abstract Object getTransferData (DataFlavor flavor)
        throws UnsupportedFlavorException, IOException;
    public abstract DataFlavor[] getTransferDataFlavors();
    public abstract boolean isDataFlavorSupported (DataFlavor flavor);
}
```

**Interface Methods**

**getTransferData**

```java
```
public abstract Object getTransferData (DataFlavor flavor)
throws UnsupportedFlavorException, IOException

Parameters
flavor The requested flavor for the returned data.

Returns
The data represented by this Transferable object, in the requested flavor.

Throws
UnsupportedFlavorException If the requested flavor is not supported.
IOException If a Reader representing the data could not be created.

Description
Returns the data this Transferable object represents. The class of object returned depends on the flavor requested.

getTransferDataFlavors
public abstract DataFlavor[] getTransferDataFlavors()

Returns
An array of the supported data flavors.

Description
The data flavors should be returned in order, sorted from most to least descriptive.

isDataFlavorSupported
public abstract boolean isDataFlavorSupported (DataFlavor flavor)

Parameters
flavor The flavor in question.

Returns
true if flavor is supported; false otherwise.

See Also
Clipboard, DataFlavor, Reader, StringSelection, Transferable

20.6 UnsupportedFlavorException ★

Description
This exception is thrown from Transferable.getTransferData(DataFlavor) to indicate that the DataFlavor requested is not available.

Class Definition
public class java.awt.datatransfer.UnsupportedFlavorException
extends java.lang.Exception {

    // Constructor
    public UnsupportedFlavorException (DataFlavor flavor);
}
Constructors

UnsupportedFlavorException

    public UnsupportedFlavorException (DataFlavor flavor)

Parameters

flavor The flavor that caused the exception.

See Also

DataFlavor, Exception, Transferable