| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/UndoManager.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/javax/swing/undo/UndoableEditSupport.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?javax/swing/undo/UndoManager.html)    [**NO FRAMES**](http://docs.google.com/UndoManager.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#4d34og8) | [CONSTR](#17dp8vu) | [METHOD](#3rdcrjn) | DETAIL: FIELD | [CONSTR](#1ksv4uv) | [METHOD](#2jxsxqh) |

## **javax.swing.undo**

Class UndoManager

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [javax.swing.undo.AbstractUndoableEdit](http://docs.google.com/javax/swing/undo/AbstractUndoableEdit.html)  
 [javax.swing.undo.CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html)  
 **javax.swing.undo.UndoManager**

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [EventListener](http://docs.google.com/java/util/EventListener.html), [UndoableEditListener](http://docs.google.com/javax/swing/event/UndoableEditListener.html), [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html)

public class **UndoManager**extends [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html)implements [UndoableEditListener](http://docs.google.com/javax/swing/event/UndoableEditListener.html)

UndoManager manages a list of UndoableEdits, providing a way to undo or redo the appropriate edits. There are two ways to add edits to an UndoManager. Add the edit directly using the addEdit method, or add the UndoManager to a bean that supports UndoableEditListener. The following examples creates an UndoManager and adds it as an UndoableEditListener to a JTextField:

UndoManager undoManager = new UndoManager();  
 JTextField tf = ...;  
 tf.getDocument().addUndoableEditListener(undoManager);

UndoManager maintains an ordered list of edits and the index of the next edit in that list. The index of the next edit is either the size of the current list of edits, or if undo has been invoked it corresponds to the index of the last significant edit that was undone. When undo is invoked all edits from the index of the next edit to the last significant edit are undone, in reverse order. For example, consider an UndoManager consisting of the following edits: **A** *b* *c* **D**. Edits with a upper-case letter in bold are significant, those in lower-case and italicized are insignificant.

|  |
| --- |
| Figure 1 |

As shown in [figure 1](#3znysh7), if **D** was just added, the index of the next edit will be 4. Invoking undo results in invoking undo on **D** and setting the index of the next edit to 3 (edit *c*), as shown in the following figure.

|  |
| --- |
| Figure 2 |

The last significant edit is **A**, so that invoking undo again invokes undo on *c*, *b*, and **A**, in that order, setting the index of the next edit to 0, as shown in the following figure.

|  |
| --- |
| Figure 3 |

Invoking redo results in invoking redo on all edits between the index of the next edit and the next significant edit (or the end of the list). Continuing with the previous example if redo were invoked, redo would in turn be invoked on **A**, *b* and *c*. In addition the index of the next edit is set to 3 (as shown in [figure 2](#2et92p0)).

Adding an edit to an UndoManager results in removing all edits from the index of the next edit to the end of the list. Continuing with the previous example, if a new edit, *e*, is added the edit **D** is removed from the list (after having die invoked on it). If *c* is not incorporated by the next edit (*c*.addEdit(*e*) returns true), or replaced by it (*e*.replaceEdit(*c*) returns true), the new edit is added after *c*, as shown in the following figure.

|  |
| --- |
| Figure 4 |

Once end has been invoked on an UndoManager the superclass behavior is used for all UndoableEdit methods. Refer to CompoundEdit for more details on its behavior.

Unlike the rest of Swing, this class is thread safe.

**Warning:** Serialized objects of this class will not be compatible with future Swing releases. The current serialization support is appropriate for short term storage or RMI between applications running the same version of Swing. As of 1.4, support for long term storage of all JavaBeansTM has been added to the java.beans package. Please see [XMLEncoder](http://docs.google.com/java/beans/XMLEncoder.html).

| **Field Summary** | |
| --- | --- |

| **Fields inherited from class javax.swing.undo.**[**CompoundEdit**](http://docs.google.com/javax/swing/undo/CompoundEdit.html) |
| --- |
| [edits](http://docs.google.com/javax/swing/undo/CompoundEdit.html#edits) |

| **Fields inherited from class javax.swing.undo.**[**AbstractUndoableEdit**](http://docs.google.com/javax/swing/undo/AbstractUndoableEdit.html) |
| --- |
| [RedoName](http://docs.google.com/javax/swing/undo/AbstractUndoableEdit.html#RedoName), [UndoName](http://docs.google.com/javax/swing/undo/AbstractUndoableEdit.html#UndoName) |

| **Constructor Summary** | |
| --- | --- |
| [**UndoManager**](http://docs.google.com/javax/swing/undo/UndoManager.html#UndoManager())()            Creates a new UndoManager. |

| **Method Summary** | |
| --- | --- |
| boolean | [**addEdit**](http://docs.google.com/javax/swing/undo/UndoManager.html#addEdit(javax.swing.undo.UndoableEdit))([UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) anEdit)            Adds an UndoableEdit to this UndoManager, if it's possible. |
| boolean | [**canRedo**](http://docs.google.com/javax/swing/undo/UndoManager.html#canRedo())()            Returns true if edits may be redone. |
| boolean | [**canUndo**](http://docs.google.com/javax/swing/undo/UndoManager.html#canUndo())()            Returns true if edits may be undone. |
| boolean | [**canUndoOrRedo**](http://docs.google.com/javax/swing/undo/UndoManager.html#canUndoOrRedo())()            Returns true if it is possible to invoke undo or redo. |
| void | [**discardAllEdits**](http://docs.google.com/javax/swing/undo/UndoManager.html#discardAllEdits())()            Empties the undo manager sending each edit a die message in the process. |
| protected  [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) | [**editToBeRedone**](http://docs.google.com/javax/swing/undo/UndoManager.html#editToBeRedone())()            Returns the the next significant edit to be redone if redo is invoked. |
| protected  [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) | [**editToBeUndone**](http://docs.google.com/javax/swing/undo/UndoManager.html#editToBeUndone())()            Returns the the next significant edit to be undone if undo is invoked. |
| void | [**end**](http://docs.google.com/javax/swing/undo/UndoManager.html#end())()            Turns this UndoManager into a normal CompoundEdit. |
| int | [**getLimit**](http://docs.google.com/javax/swing/undo/UndoManager.html#getLimit())()            Returns the maximum number of edits this UndoManager holds. |
| [String](http://docs.google.com/java/lang/String.html) | [**getRedoPresentationName**](http://docs.google.com/javax/swing/undo/UndoManager.html#getRedoPresentationName())()            Returns a description of the redoable form of this edit. |
| [String](http://docs.google.com/java/lang/String.html) | [**getUndoOrRedoPresentationName**](http://docs.google.com/javax/swing/undo/UndoManager.html#getUndoOrRedoPresentationName())()            Convenience method that returns either getUndoPresentationName or getRedoPresentationName. |
| [String](http://docs.google.com/java/lang/String.html) | [**getUndoPresentationName**](http://docs.google.com/javax/swing/undo/UndoManager.html#getUndoPresentationName())()            Returns a description of the undoable form of this edit. |
| void | [**redo**](http://docs.google.com/javax/swing/undo/UndoManager.html#redo())()            Redoes the appropriate edits. |
| protected  void | [**redoTo**](http://docs.google.com/javax/swing/undo/UndoManager.html#redoTo(javax.swing.undo.UndoableEdit))([UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) edit)            Redoes all changes from the index of the next edit to edit, updating the index of the next edit appropriately. |
| void | [**setLimit**](http://docs.google.com/javax/swing/undo/UndoManager.html#setLimit(int))(int l)            Sets the maximum number of edits this UndoManager holds. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/javax/swing/undo/UndoManager.html#toString())()            Returns a string that displays and identifies this object's properties. |
| protected  void | [**trimEdits**](http://docs.google.com/javax/swing/undo/UndoManager.html#trimEdits(int,%20int))(int from, int to)            Removes edits in the specified range. |
| protected  void | [**trimForLimit**](http://docs.google.com/javax/swing/undo/UndoManager.html#trimForLimit())()            Reduces the number of queued edits to a range of size limit, centered on the index of the next edit. |
| void | [**undo**](http://docs.google.com/javax/swing/undo/UndoManager.html#undo())()            Undoes the appropriate edits. |
| void | [**undoableEditHappened**](http://docs.google.com/javax/swing/undo/UndoManager.html#undoableEditHappened(javax.swing.event.UndoableEditEvent))([UndoableEditEvent](http://docs.google.com/javax/swing/event/UndoableEditEvent.html) e)            An UndoableEditListener method. |
| void | [**undoOrRedo**](http://docs.google.com/javax/swing/undo/UndoManager.html#undoOrRedo())()            Convenience method that invokes one of undo or redo. |
| protected  void | [**undoTo**](http://docs.google.com/javax/swing/undo/UndoManager.html#undoTo(javax.swing.undo.UndoableEdit))([UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) edit)            Undoes all changes from the index of the next edit to edit, updating the index of the next edit appropriately. |

| **Methods inherited from class javax.swing.undo.**[**CompoundEdit**](http://docs.google.com/javax/swing/undo/CompoundEdit.html) |
| --- |
| [die](http://docs.google.com/javax/swing/undo/CompoundEdit.html#die()), [getPresentationName](http://docs.google.com/javax/swing/undo/CompoundEdit.html#getPresentationName()), [isInProgress](http://docs.google.com/javax/swing/undo/CompoundEdit.html#isInProgress()), [isSignificant](http://docs.google.com/javax/swing/undo/CompoundEdit.html#isSignificant()), [lastEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html#lastEdit()) |

| **Methods inherited from class javax.swing.undo.**[**AbstractUndoableEdit**](http://docs.google.com/javax/swing/undo/AbstractUndoableEdit.html) |
| --- |
| [replaceEdit](http://docs.google.com/javax/swing/undo/AbstractUndoableEdit.html#replaceEdit(javax.swing.undo.UndoableEdit)) |

| **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()), [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)) |

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

### UndoManager

public **UndoManager**()

Creates a new UndoManager.

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

### getLimit

public int **getLimit**()

Returns the maximum number of edits this UndoManager holds. A value less than 0 indicates the number of edits is not limited.

**Returns:**the maximum number of edits this UndoManager holds**See Also:**[addEdit(javax.swing.undo.UndoableEdit)](http://docs.google.com/javax/swing/undo/UndoManager.html#addEdit(javax.swing.undo.UndoableEdit)), [setLimit(int)](http://docs.google.com/javax/swing/undo/UndoManager.html#setLimit(int))

### discardAllEdits

public void **discardAllEdits**()

Empties the undo manager sending each edit a die message in the process.

**See Also:**[AbstractUndoableEdit.die()](http://docs.google.com/javax/swing/undo/AbstractUndoableEdit.html#die())

### trimForLimit

protected void **trimForLimit**()

Reduces the number of queued edits to a range of size limit, centered on the index of the next edit.

### trimEdits

protected void **trimEdits**(int from,  
 int to)

Removes edits in the specified range. All edits in the given range (inclusive, and in reverse order) will have die invoked on them and are removed from the list of edits. This has no effect if from > to.

**Parameters:**from - the minimum index to removeto - the maximum index to remove

### setLimit

public void **setLimit**(int l)

Sets the maximum number of edits this UndoManager holds. A value less than 0 indicates the number of edits is not limited. If edits need to be discarded to shrink the limit, die will be invoked on them in the reverse order they were added. The default is 100.

**Parameters:**l - the new limit **Throws:** [RuntimeException](http://docs.google.com/java/lang/RuntimeException.html) - if this UndoManager is not in progress (end has been invoked)**See Also:**[CompoundEdit.isInProgress()](http://docs.google.com/javax/swing/undo/CompoundEdit.html#isInProgress()), [end()](http://docs.google.com/javax/swing/undo/UndoManager.html#end()), [addEdit(javax.swing.undo.UndoableEdit)](http://docs.google.com/javax/swing/undo/UndoManager.html#addEdit(javax.swing.undo.UndoableEdit)), [getLimit()](http://docs.google.com/javax/swing/undo/UndoManager.html#getLimit())

### editToBeUndone

protected [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) **editToBeUndone**()

Returns the the next significant edit to be undone if undo is invoked. This returns null if there are no edits to be undone.

**Returns:**the next significant edit to be undone

### editToBeRedone

protected [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) **editToBeRedone**()

Returns the the next significant edit to be redone if redo is invoked. This returns null if there are no edits to be redone.

**Returns:**the next significant edit to be redone

### undoTo

protected void **undoTo**([UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) edit)  
 throws [CannotUndoException](http://docs.google.com/javax/swing/undo/CannotUndoException.html)

Undoes all changes from the index of the next edit to edit, updating the index of the next edit appropriately.

**Throws:** [CannotUndoException](http://docs.google.com/javax/swing/undo/CannotUndoException.html) - if one of the edits throws CannotUndoException

### redoTo

protected void **redoTo**([UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) edit)  
 throws [CannotRedoException](http://docs.google.com/javax/swing/undo/CannotRedoException.html)

Redoes all changes from the index of the next edit to edit, updating the index of the next edit appropriately.

**Throws:** [CannotRedoException](http://docs.google.com/javax/swing/undo/CannotRedoException.html) - if one of the edits throws CannotRedoException

### undoOrRedo

public void **undoOrRedo**()  
 throws [CannotRedoException](http://docs.google.com/javax/swing/undo/CannotRedoException.html),  
 [CannotUndoException](http://docs.google.com/javax/swing/undo/CannotUndoException.html)

Convenience method that invokes one of undo or redo. If any edits have been undone (the index of the next edit is less than the length of the edits list) this invokes redo, otherwise it invokes undo.

**Throws:** [CannotUndoException](http://docs.google.com/javax/swing/undo/CannotUndoException.html) - if one of the edits throws CannotUndoException [CannotRedoException](http://docs.google.com/javax/swing/undo/CannotRedoException.html) - if one of the edits throws CannotRedoException**See Also:**[canUndoOrRedo()](http://docs.google.com/javax/swing/undo/UndoManager.html#canUndoOrRedo()), [getUndoOrRedoPresentationName()](http://docs.google.com/javax/swing/undo/UndoManager.html#getUndoOrRedoPresentationName())

### canUndoOrRedo

public boolean **canUndoOrRedo**()

Returns true if it is possible to invoke undo or redo.

**Returns:**true if invoking canUndoOrRedo is valid**See Also:**[undoOrRedo()](http://docs.google.com/javax/swing/undo/UndoManager.html#undoOrRedo())

### undo

public void **undo**()  
 throws [CannotUndoException](http://docs.google.com/javax/swing/undo/CannotUndoException.html)

Undoes the appropriate edits. If end has been invoked this calls through to the superclass, otherwise this invokes undo on all edits between the index of the next edit and the last significant edit, updating the index of the next edit appropriately.

**Specified by:**[undo](http://docs.google.com/javax/swing/undo/UndoableEdit.html#undo()) in interface [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html)**Overrides:**[undo](http://docs.google.com/javax/swing/undo/CompoundEdit.html#undo()) in class [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html) **Throws:** [CannotUndoException](http://docs.google.com/javax/swing/undo/CannotUndoException.html) - if one of the edits throws CannotUndoException or there are no edits to be undone**See Also:**[CompoundEdit.end()](http://docs.google.com/javax/swing/undo/CompoundEdit.html#end()), [canUndo()](http://docs.google.com/javax/swing/undo/UndoManager.html#canUndo()), [editToBeUndone()](http://docs.google.com/javax/swing/undo/UndoManager.html#editToBeUndone())

### canUndo

public boolean **canUndo**()

Returns true if edits may be undone. If end has been invoked, this returns the value from super. Otherwise this returns true if there are any edits to be undone (editToBeUndone returns non-null).

**Specified by:**[canUndo](http://docs.google.com/javax/swing/undo/UndoableEdit.html#canUndo()) in interface [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html)**Overrides:**[canUndo](http://docs.google.com/javax/swing/undo/CompoundEdit.html#canUndo()) in class [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html) **Returns:**true if there are edits to be undone**See Also:**[CompoundEdit.canUndo()](http://docs.google.com/javax/swing/undo/CompoundEdit.html#canUndo()), [editToBeUndone()](http://docs.google.com/javax/swing/undo/UndoManager.html#editToBeUndone())

### redo

public void **redo**()  
 throws [CannotRedoException](http://docs.google.com/javax/swing/undo/CannotRedoException.html)

Redoes the appropriate edits. If end has been invoked this calls through to the superclass. Otherwise this invokes redo on all edits between the index of the next edit and the next significant edit, updating the index of the next edit appropriately.

**Specified by:**[redo](http://docs.google.com/javax/swing/undo/UndoableEdit.html#redo()) in interface [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html)**Overrides:**[redo](http://docs.google.com/javax/swing/undo/CompoundEdit.html#redo()) in class [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html) **Throws:** [CannotRedoException](http://docs.google.com/javax/swing/undo/CannotRedoException.html) - if one of the edits throws CannotRedoException or there are no edits to be redone**See Also:**[CompoundEdit.end()](http://docs.google.com/javax/swing/undo/CompoundEdit.html#end()), [canRedo()](http://docs.google.com/javax/swing/undo/UndoManager.html#canRedo()), [editToBeRedone()](http://docs.google.com/javax/swing/undo/UndoManager.html#editToBeRedone())

### canRedo

public boolean **canRedo**()

Returns true if edits may be redone. If end has been invoked, this returns the value from super. Otherwise, this returns true if there are any edits to be redone (editToBeRedone returns non-null).

**Specified by:**[canRedo](http://docs.google.com/javax/swing/undo/UndoableEdit.html#canRedo()) in interface [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html)**Overrides:**[canRedo](http://docs.google.com/javax/swing/undo/CompoundEdit.html#canRedo()) in class [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html) **Returns:**true if there are edits to be redone**See Also:**[CompoundEdit.canRedo()](http://docs.google.com/javax/swing/undo/CompoundEdit.html#canRedo()), [editToBeRedone()](http://docs.google.com/javax/swing/undo/UndoManager.html#editToBeRedone())

### addEdit

public boolean **addEdit**([UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) anEdit)

Adds an UndoableEdit to this UndoManager, if it's possible. This removes all edits from the index of the next edit to the end of the edits list. If end has been invoked the edit is not added and false is returned. If end hasn't been invoked this returns true.

**Specified by:**[addEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html#addEdit(javax.swing.undo.UndoableEdit)) in interface [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html)**Overrides:**[addEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html#addEdit(javax.swing.undo.UndoableEdit)) in class [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html) **Parameters:**anEdit - the edit to be added **Returns:**true if anEdit can be incorporated into this edit**See Also:**[CompoundEdit.end()](http://docs.google.com/javax/swing/undo/CompoundEdit.html#end()), [CompoundEdit.addEdit(javax.swing.undo.UndoableEdit)](http://docs.google.com/javax/swing/undo/CompoundEdit.html#addEdit(javax.swing.undo.UndoableEdit))

### end

public void **end**()

Turns this UndoManager into a normal CompoundEdit. This removes all edits that have been undone.

**Overrides:**[end](http://docs.google.com/javax/swing/undo/CompoundEdit.html#end()) in class [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html) **See Also:**[CompoundEdit.end()](http://docs.google.com/javax/swing/undo/CompoundEdit.html#end())

### getUndoOrRedoPresentationName

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

Convenience method that returns either getUndoPresentationName or getRedoPresentationName. If the index of the next edit equals the size of the edits list, getUndoPresentationName is returned, otherwise getRedoPresentationName is returned.

**Returns:**undo or redo name

### getUndoPresentationName

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

Returns a description of the undoable form of this edit. If end has been invoked this calls into super. Otherwise if there are edits to be undone, this returns the value from the next significant edit that will be undone. If there are no edits to be undone and end has not been invoked this returns the value from the UIManager property "AbstractUndoableEdit.undoText".

**Specified by:**[getUndoPresentationName](http://docs.google.com/javax/swing/undo/UndoableEdit.html#getUndoPresentationName()) in interface [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html)**Overrides:**[getUndoPresentationName](http://docs.google.com/javax/swing/undo/CompoundEdit.html#getUndoPresentationName()) in class [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html) **Returns:**a description of the undoable form of this edit**See Also:**[undo()](http://docs.google.com/javax/swing/undo/UndoManager.html#undo()), [CompoundEdit.getUndoPresentationName()](http://docs.google.com/javax/swing/undo/CompoundEdit.html#getUndoPresentationName())

### getRedoPresentationName

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

Returns a description of the redoable form of this edit. If end has been invoked this calls into super. Otherwise if there are edits to be redone, this returns the value from the next significant edit that will be redone. If there are no edits to be redone and end has not been invoked this returns the value from the UIManager property "AbstractUndoableEdit.redoText".

**Specified by:**[getRedoPresentationName](http://docs.google.com/javax/swing/undo/UndoableEdit.html#getRedoPresentationName()) in interface [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html)**Overrides:**[getRedoPresentationName](http://docs.google.com/javax/swing/undo/CompoundEdit.html#getRedoPresentationName()) in class [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html) **Returns:**a description of the redoable form of this edit**See Also:**[redo()](http://docs.google.com/javax/swing/undo/UndoManager.html#redo()), [CompoundEdit.getRedoPresentationName()](http://docs.google.com/javax/swing/undo/CompoundEdit.html#getRedoPresentationName())

### undoableEditHappened

public void **undoableEditHappened**([UndoableEditEvent](http://docs.google.com/javax/swing/event/UndoableEditEvent.html) e)

An UndoableEditListener method. This invokes addEdit with e.getEdit().

**Specified by:**[undoableEditHappened](http://docs.google.com/javax/swing/event/UndoableEditListener.html#undoableEditHappened(javax.swing.event.UndoableEditEvent)) in interface [UndoableEditListener](http://docs.google.com/javax/swing/event/UndoableEditListener.html) **Parameters:**e - the UndoableEditEvent the UndoableEditEvent will be added from**See Also:**[addEdit(javax.swing.undo.UndoableEdit)](http://docs.google.com/javax/swing/undo/UndoManager.html#addEdit(javax.swing.undo.UndoableEdit))

### toString

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

Returns a string that displays and identifies this object's properties.

**Overrides:**[toString](http://docs.google.com/javax/swing/undo/CompoundEdit.html#toString()) in class [CompoundEdit](http://docs.google.com/javax/swing/undo/CompoundEdit.html) **Returns:**a String representation of this object

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/UndoManager.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/javax/swing/undo/UndoableEditSupport.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?javax/swing/undo/UndoManager.html)    [**NO FRAMES**](http://docs.google.com/UndoManager.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#4d34og8) | [CONSTR](#17dp8vu) | [METHOD](#3rdcrjn) | DETAIL: FIELD | [CONSTR](#1ksv4uv) | [METHOD](#2jxsxqh) |

[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).