NSUndoManager Class Reference

Cocoa > Data Management

2007-01-18

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

NSUndoManager Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/Foundation.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	Undo Architecture
Declared in	NSUndoManager.h
Related sample code	CoreRecipes iSpend Quartz Composer WWDC 2005 TextEdit Sketch-112 TextEditPlus

Overview

`NSUndoManager` is a general-purpose recorder of operations for undo and redo.

You register an undo operation by specifying the object that's changing (or the owner of that object), along with a method to invoke to revert its state, and the arguments for that method. When performing undo an `NSUndoManager` saves the operations reverted so that you can redo the undos. If used in a Cocoa Application Kit-based application, `NSUndoManager` groups all operations within a single cycle of the run loop, so that performing an undo reverts all changes that occurred during the cycle.

`NSUndoManager` is implemented as a class of the Foundation framework because executables other than applications might want to revert changes to their states. For example, you might have an interactive command-line tool with undo and redo commands, or there could be distributed object implementations that can revert operations "over the wire." However, users typically see undo and redo as application features. The Application Kit implements undo and redo in its NSTextView object and makes it easy to implement it in objects along the responder chain.

Tasks

Registering Undo Operations

- `registerUndoWithTarget:selector:object:` (page 16)
  Records a single undo operation for a given target, so that when an undo is performed it is sent a specified selector with a given object as the sole argument.
- `prepareWithInvocationTarget:` (page 14)
  Prepares the receiver for invocation-based undo with the given target as the subject of the next undo operation and returns `self`.
- `forwardInvocation:` (page 11)
  Overrides `NSObject`'s implementation to record the given invocation as an undo operation.

Checking Undo Ability

- `canUndo` (page 9)
  Returns a Boolean value that indicates whether the receiver has any actions to undo.
- `canRedo` (page 9)
  Returns a Boolean value that indicates whether the receiver has any actions to redo.

Performing Undo and Redo

- `undo` (page 20)
  Closes the top-level undo group if necessary and invokes `undoNestedGroup` (page 22).
- `undoNestedGroup` (page 22)
  Performs the undo operations in the last undo group (whether top-level or nested), recording the operations on the redo stack as a single group.
- `redo` (page 15)
  Performs the operations in the last group on the redo stack, if there are any, recording them on the undo stack as a single group.

Limiting the Undo Stack

- `setLevelsOfUndo:` (page 19)
  Sets the maximum number of top-level undo groups the receiver holds.
- `levelsOfUndo` (page 14)
  Returns the maximum number of top-level undo groups the receiver holds.

Creating Undo Groups

- `beginUndoGrouping` (page 8)
  Marks the beginning of an undo group.

- `endUndoGrouping` (page 10)
  Marks the end of an undo group.
- `enableUndoRegistration` (page 10)
  Enables the recording of undo operations.
- `groupsByEvent` (page 12)
  Returns a Boolean value that indicates whether the receiver automatically creates undo groups around each pass of the run loop.
- `setGroupsByEvent:` (page 19)
  Sets a Boolean value that specifies whether the receiver automatically groups undo operations during the run loop.
- `groupingLevel` (page 12)
  Returns the number of nested undo groups (or redo groups, if Redo was invoked last) in the current event loop.

Disabling Undo

- `disableUndoRegistration` (page 10)
  Disables the recording of undo operations, whether by `registerUndoWithTarget:selector:object:` (page 16) or by invocation-based undo.
- `isUndoRegistrationEnabled` (page 13)
  Returns a Boolean value that indicates whether the recording of undo operations is enabled.

Checking Whether Undo or Redo Is Being Performed

- `isUndoing` (page 13)
  Returns a Boolean value that indicates whether the receiver is in the process of performing its `undo` (page 20) or `undoNestedGroup` (page 22) method.
- `isRedoing` (page 12)
  Returns a Boolean value that indicates whether the receiver is in the process of performing its `redo` (page 15) method.

Clearing Undo Operations

- `removeAllActions` (page 17)
  Clears the undo and redo stacks and re-enables the receiver.
- `removeAllActionsWithTarget:` (page 17)
  Clears the undo and redo stacks of all operations involving the specified target as the recipient of the undo message.

Managing the Action Name

- `setActionName:` (page 18)
  Sets the name of the action associated with the Undo or Redo command.

- `redoActionName` (page 15)
  Returns the name identifying the redo action.
- `undoActionName` (page 21)
  Returns the name identifying the undo action.

Getting and Localizing the Menu Item Title

- `redoMenuItemTitle` (page 15)
  Returns the complete title of the Redo menu command, for example, “Redo Paste.”
- `undoMenuItemTitle` (page 21)
  Returns the complete title of the Undo menu command, for example, “Undo Paste.”
- `redoMenuTitleForUndoActionName:` (page 16)
  Returns the complete, localized title of the Redo menu command for the action identified by the given name.
- `undoMenuTitleForUndoActionName:` (page 22)
  Returns the complete, localized title of the Undo menu command for the action identified by the given name.

Working with Run Loops

- `runLoopModes` (page 18)
  Returns the modes governing the types of input handled during a cycle of the run loop.
- `setRunLoopModes:` (page 20)
  Sets the modes that determine the types of input handled during a cycle of the run loop.

Instance Methods

beginUndoGrouping

Marks the beginning of an undo group.

```
- (void)beginUndoGrouping
```

Discussion
All individual undo operations before a subsequent `endUndoGrouping` (page 10) message are grouped together and reversed by a later `undo` (page 20) message. By default undo groups are begun automatically at the start of the event loop, but you can begin your own undo groups with this method, and nest them within other groups.

This method posts an `NSUndoManagerCheckpointNotification` (page 23) unless a top-level undo is in progress. It posts an `NSUndoManagerDidOpenUndoGroupNotification` (page 23) if a new group was successfully created.

Availability
Available in Mac OS X v10.0 and later.

Declared In
NSUndoManager.h

canRedo

Returns a Boolean value that indicates whether the receiver has any actions to redo.

```
- (BOOL)canRedo
```

Return Value
YES if the receiver has any actions to redo, otherwise NO.

Discussion
Because any undo operation registered clears the redo stack, this method posts an NSUndoManagerCheckpointNotification (page 23) to allow clients to apply their pending operations before testing the redo stack.

Availability
Available in Mac OS X v10.0 and later.

See Also
- canUndo (page 9)
- redo (page 15)

Declared In
NSUndoManager.h

canUndo

Returns a Boolean value that indicates whether the receiver has any actions to undo.

```
- (BOOL)canUndo
```

Return Value
YES if the receiver has any actions to undo, otherwise NO.

Discussion
The return value does not mean you can safely invoke undo (page 20) or undoNestedGroup (page 22)—you may have to close open undo groups first.

Availability
Available in Mac OS X v10.0 and later.

See Also
- canRedo (page 9)
- enableUndoRegistration (page 10)
- registerUndoWithTarget:selector:object: (page 16)

Declared In
NSUndoManager.h

disableUndoRegistration

Disables the recording of undo operations, whether by `registerUndoWithTarget:selector:object:` (page 16) or by invocation-based undo.

```
- (void)disableUndoRegistration
```

Discussion
This method can be invoked multiple times by multiple clients. The `enableUndoRegistration` (page 10) method must be invoked an equal number of times to re-enable undo registration.

Availability
Available in Mac OS X v10.0 and later.

Related Sample Code
Departments and Employees
File Wrappers with Core Data Documents

Declared In
`NSUndoManager.h`

enableUndoRegistration

Enables the recording of undo operations.

```
- (void)enableUndoRegistration
```

Discussion
Because undo registration is enabled by default, it is often used to balance a prior `disableUndoRegistration` (page 10) message. Undo registration isn't actually re-enabled until an enable message balances the last disable message in effect. Raises an `NSInternalInconsistencyException` if invoked while no `disableUndoRegistration` (page 10) message is in effect.

Availability
Available in Mac OS X v10.0 and later.

Related Sample Code
Departments and Employees
File Wrappers with Core Data Documents

Declared In
`NSUndoManager.h`

endUndoGrouping

Marks the end of an undo group.

```
- (void)endUndoGrouping
```

Discussion
All individual undo operations back to the matching `beginUndoGrouping` (page 8) message are grouped together and reversed by a later `undo` (page 20) or `undoNestedGroup` (page 22) message. Undo groups can be nested, thus providing functionality similar to nested transactions. Raises an `NSInternalInconsistencyException` if there's no `beginUndoGrouping` (page 8) message in effect.

This method posts an `NSUndoManagerCheckpointNotification` (page 23) and an `NSUndoManagerWillCloseUndoGroupNotification` (page 24) just before the group is closed.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `levelsOfUndo` (page 14)

Declared In
`NSUndoManager.h`

forwardInvocation:

Overrides `NSObject`'s implementation to record the given invocation as an undo operation.

```
- (void)forwardInvocation:(NSInvocation *)anInvocation
```

Parameters
anInvocation
The invocation to record.

Discussion
Also clears the redo stack. *anInvocation* and its arguments that are objects are retained. You can override this method if you want different or supplementary invocation-based behavior. See "Registering Undo Operations" for more information.

Raises an `NSInternalInconsistencyException` if `prepareWithInvocationTarget:` (page 14) was not invoked before this method. This method then clears the prepared invocation target. Also raises an `NSInternalInconsistencyException` if invoked when no undo group has been established using `beginUndoGrouping` (page 8). Undo groups are normally set by default, so you should rarely need to begin a top-level undo group explicitly.

Availability
Available in Mac OS X v10.0 through Mac OS X v10.4.

See Also
- `undoNestedGroup` (page 22)
- `registerUndoWithTarget:selector:object:` (page 16)
- `groupingLevel` (page 12)

Declared In
`NSUndoManager.h`

groupingLevel

Returns the number of nested undo groups (or redo groups, if Redo was invoked last) in the current event loop.

```
- (NSInteger)groupingLevel
```

Return Value
An integer indicating the number of nested groups. If 0 is returned, there is no open undo or redo group.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `levelsOfUndo` (page 14)
- `setLevelsOfUndo:` (page 19)

Declared In
`NSUndoManager.h`

groupsByEvent

Returns a Boolean value that indicates whether the receiver automatically creates undo groups around each pass of the run loop.

```
- (BOOL)groupsByEvent
```

Return Value
`YES` if the receiver automatically creates undo groups around each pass of the run loop, otherwise `NO`.

Discussion
The default is `YES`.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `beginUndoGrouping` (page 8)
- `setGroupsByEvent:` (page 19)

Declared In
`NSUndoManager.h`

isRedoing

Returns a Boolean value that indicates whether the receiver is in the process of performing its `redo` (page 15) method.

```
- (BOOL)isRedoing
```

Return Value
`YES` if the method is being performed, otherwise `NO`.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `isUndoing` (page 13)

Declared In
`NSUndoManager.h`

isUndoing

Returns a Boolean value that indicates whether the receiver is in the process of performing its `undo` (page 20) or `undoNestedGroup` (page 22) method.

```
- (BOOL)isUndoing
```

Return Value
`YES` if the method is being performed, otherwise `NO`.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `isRedoing` (page 12)

Declared In
`NSUndoManager.h`

isUndoRegistrationEnabled

Returns a Boolean value that indicates whether the recording of undo operations is enabled.

```
- (BOOL)isUndoRegistrationEnabled
```

Return Value
`YES` if registration is enabled; otherwise, `NO`.

Discussion
Undo registration is enabled by default.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `disableUndoRegistration` (page 10)
- `enableUndoRegistration` (page 10)

Declared In
`NSUndoManager.h`

levelsOfUndo

Returns the maximum number of top-level undo groups the receiver holds.

```
- (NSUInteger)levelsOfUndo
```

Return Value
An integer specifying the number of undo groups. A limit of `0` indicates no limit, so old undo groups are never dropped.

Discussion
When ending an undo group results in the number of groups exceeding this limit, the oldest groups are dropped from the stack. The default is `0`.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `enableUndoRegistration` (page 10)
- `setLevelsOfUndo:` (page 19)

Declared In
`NSUndoManager.h`

prepareWithInvocationTarget:

Prepares the receiver for invocation-based undo with the given target as the subject of the next undo operation and returns `self`.

```
- (id)prepareWithInvocationTarget:(id)target
```

Parameters
target
The target of the undo operation.

Return Value
`self`.

Discussion
See “Registering Undo Operations” for more information.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `forwardInvocation:` (page 11)

Related Sample Code
Squiggles

Declared In
`NSUndoManager.h`

redo

Performs the operations in the last group on the redo stack, if there are any, recording them on the undo stack as a single group.

```
- (void)redo
```

Discussion
Raises an `NSInternalInconsistencyException` if the method is invoked during an undo operation.

This method posts an `NSUndoManagerCheckpointNotification` (page 23) and `NSUndoManagerWillRedoChangeNotification` (page 24) before it performs the redo operation, and it posts the `NSUndoManagerDidRedoChangeNotification` (page 24) after it performs the redo operation.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `registerUndoWithTarget:selector:object:` (page 16)

Declared In
`NSUndoManager.h`

redoActionName

Returns the name identifying the redo action.

```
- (NSString *)redoActionName
```

Return Value
The redo action name. Returns an empty string (@"") if no action name has been assigned or if there is nothing to redo.

Discussion
For example, if the menu title is “Redo Delete,” the string returned is “Delete.”

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setActionName:` (page 18)
- `undoActionName` (page 21)

Declared In
`NSUndoManager.h`

redoMenuItemTitle

Returns the complete title of the Redo menu command, for example, “Redo Paste.”

```
- (NSString *)redoMenuItemTitle
```

Return Value
The menu item title.

Discussion
Returns “Redo” if no action name has been assigned or `nil` if there is nothing to redo.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- undoMenuItemTitle` (page 21)

Declared In
`NSUndoManager.h`

redoMenuTitleForUndoActionName:

Returns the complete, localized title of the Redo menu command for the action identified by the given name.

```
- (NSString *)redoMenuTitleForUndoActionName:(NSString *)actionName
```

Parameters
actionName
The name of the undo action.

Return Value
The localized title of the redo menu item.

Discussion
Override this method if you want to customize the localization behavior. This method is invoked by `redoMenuItemTitle` (page 15).

Availability
Available in Mac OS X v10.0 and later.

See Also
`- undoMenuTitleForUndoActionName:` (page 22)

Declared In
`NSUndoManager.h`

registerUndoWithTarget:selector:object:

Records a single undo operation for a given target, so that when an undo is performed it is sent a specified selector with a given object as the sole argument.

```
- (void)registerUndoWithTarget:(id)target selector:(SEL)aSelector object:(id)anObject
```

Parameters
target
The target of the undo operation.
aSelector
The selector for the undo operation.
anObject
The argument sent with the selector.

Discussion
Also clears the redo stack. Does not retain *target*, but does retain *anObject*. See "Registering Undo Operations" for more information.

Raises an `NSInternalInconsistencyException` if invoked when no undo group has been established using `beginUndoGrouping` (page 8). Undo groups are normally set by default, so you should rarely need to begin a top-level undo group explicitly.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `undoNestedGroup` (page 22)
- `forwardInvocation:` (page 11)
- `groupingLevel` (page 12)

Related Sample Code
File Wrappers with Core Data Documents

Declared In
`NSUndoManager.h`

removeAllActions

Clears the undo and redo stacks and re-enables the receiver.

```
- (void)removeAllActions
```

Availability
Available in Mac OS X v10.0 and later.

See Also
- `enableUndoRegistration` (page 10)
- `removeAllActionsWithTarget:` (page 17)

Related Sample Code
Departments and Employees

Declared In
`NSUndoManager.h`

removeAllActionsWithTarget:

Clears the undo and redo stacks of all operations involving the specified target as the recipient of the undo message.

```
- (void)removeAllActionsWithTarget:(id)target
```

Parameters
target
The recipient of the undo messages to be removed.

Discussion
Doesn't re-enable the receiver if it's disabled. An object that shares an `NSUndoManager` with other clients should invoke this message in its implementation of `dealloc`.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `enableUndoRegistration` (page 10)
- `removeAllActions` (page 17)

Declared In
`NSUndoManager.h`

runLoopModes

Returns the modes governing the types of input handled during a cycle of the run loop.

```
- (NSArray *)runLoopModes
```

Return Value
An array of string constants specifying the current run-loop modes.

Discussion
By default, the sole run-loop mode is `NSDefaultRunLoopMode` (which excludes data from `NSConnection` objects).

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setRunLoopModes:` (page 20)
- `performSelector:target:argument:order:modes:` (`NSRunLoop`)

Declared In
`NSUndoManager.h`

setActionName:

Sets the name of the action associated with the Undo or Redo command.

```
- (void)setActionName:(NSString *)actionName
```

Parameters
`actionName`
The name of the action.

Discussion
If *`actionName`* is an empty string, the action name currently associated with the menu command is removed. There is no effect if *`actionName`* is `nil`.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `redoActionName` (page 15)
- `undoActionName` (page 21)

Related Sample Code
Sketch-112

Declared In
`NSUndoManager.h`

setGroupsByEvent:

Sets a Boolean value that specifies whether the receiver automatically groups undo operations during the run loop.

```
- (void)setGroupsByEvent:(BOOL)flag
```

Parameters

flag

If `YES`, the receiver creates undo groups around each pass through the run loop; if `NO` it doesn't.

Discussion
The default is `YES`. If you turn automatic grouping off, you must close groups explicitly before invoking either `undo` (page 20) or `undoNestedGroup` (page 22).

Availability
Available in Mac OS X v10.0 and later.

See Also
- `groupingLevel` (page 12)
- `groupsByEvent` (page 12)

Declared In
`NSUndoManager.h`

setLevelsOfUndo:

Sets the maximum number of top-level undo groups the receiver holds.

```
- (void)setLevelsOfUndo:(NSUInteger)anInt
```

Parameters

anInt

The maximum number of undo groups. A limit of `0` indicates no limit, so that old undo groups are never dropped.

Discussion
When ending an undo group results in the number of groups exceeding this limit, the oldest groups are dropped from the stack. The default is `0`.

If invoked with a limit below the prior limit, old undo groups are immediately dropped.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `enableUndoRegistration` (page 10)
- `levelsOfUndo` (page 14)

Declared In
`NSUndoManager.h`

setRunLoopModes:

Sets the modes that determine the types of input handled during a cycle of the run loop.

```
- (void)setRunLoopModes:(NSArray *)modes
```

Parameters

modes

An array of string constants specifying the run-loop modes to set.

Discussion
By default, the sole run-loop mode is `NSDefaultRunLoopMode` (which excludes data from `NSConnection` objects). With this method, you could limit the input to data received during a mouse-tracking session by setting the mode to `NSEventTrackingRunLoopMode`, or you could limit it to data received from a modal panel with `NSModalPanelRunLoopMode`.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `runLoopModes` (page 18)
- `performSelector:target:argument:order:modes:` (`NSRunLoop`)

Declared In
`NSUndoManager.h`

undo

Closes the top-level undo group if necessary and invokes `undoNestedGroup` (page 22).

```
- (void)undo
```

Discussion
This method also invokes `endUndoGrouping` (page 10) if the nesting level is 1. Raises an `NSInternalInconsistencyException` if more than one undo group is open (that is, if the last group isn't at the top level).

This method posts an `NSUndoManagerCheckpointNotification` (page 23).

Availability
Available in Mac OS X v10.0 and later.

See Also
- `enableUndoRegistration` (page 10)
- `groupingLevel` (page 12)

Declared In
`NSUndoManager.h`

undoActionName

Returns the name identifying the undo action.

```
- (NSString *)undoActionName
```

Return Value
The undo action name. Returns an empty string (`@""`) if no action name has been assigned or if there is nothing to undo.

Discussion
For example, if the menu title is “Undo Delete,” the string returned is “Delete.”

Availability
Available in Mac OS X v10.0 and later.

See Also
- `redoActionName` (page 15)
- `setActionName:` (page 18)

Declared In
`NSUndoManager.h`

undoMenuItemTitle

Returns the complete title of the Undo menu command, for example, “Undo Paste.”

```
- (NSString *)undoMenuItemTitle
```

Return Value
The menu item title.

Discussion
Returns “Undo” if no action name has been assigned or `nil` if there is nothing to undo.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `redoMenuItemTitle` (page 15)

Declared In
`NSUndoManager.h`

undoMenuTitleForUndoActionName:

Returns the complete, localized title of the Undo menu command for the action identified by the given name.

```
- (NSString *)undoMenuTitleForUndoActionName:(NSString *)actionName
```

Parameters

actionName

The name of the undo action.

Return Value

The localized title of the undo menu item.

Discussion

Override this method if you want to customize the localization behavior. This method is invoked by `undoMenuItemTitle` (page 21).

Availability

Available in Mac OS X v10.0 and later.

See Also

`- redoMenuTitleForUndoActionName:` (page 16)

Declared In

`NSUndoManager.h`

undoNestedGroup

Performs the undo operations in the last undo group (whether top-level or nested), recording the operations on the redo stack as a single group.

```
- (void)undoNestedGroup
```

Discussion

Raises an `NSInternalInconsistencyException` if any undo operations have been registered since the last `enableUndoRegistration` (page 10) message.

This method posts an `NSUndoManagerCheckpointNotification` (page 23) and `NSUndoManagerWillUndoChangeNotification` (page 24) before it performs the undo operation, and it posts an `NSUndoManagerDidUndoChangeNotification` (page 24) after it performs the undo operation.

Availability

Available in Mac OS X v10.0 and later.

See Also

`- undo` (page 20)

Declared In

`NSUndoManager.h`

Constants

NSUndoCloseGroupingRunLoopOrdering

`NSUndoManager` provides this constant as a convenience; you can use it to compare to values returned by some `NSUndoManager` methods.

```
enum {
   NSUndoCloseGroupingRunLoopOrdering = 350000
};
```

Constants

`NSUndoCloseGroupingRunLoopOrdering`

Used with `NSRunLoop`'s `performSelector:target:argument:order:modes:`.

Available in Mac OS X v10.0 and later.

Declared in `NSUndoManager.h`.

Declared In

`NSUndoManager.h`

Notifications

NSUndoManagerCheckpointNotification

Posted whenever an `NSUndoManager` object opens or closes an undo group (except when it opens a top-level group) and when checking the redo stack in `canRedo` (page 9). The notification object is the `NSUndoManager` object. This notification does not contain a *`userInfo`* dictionary.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSUndoManager.h`

NSUndoManagerDidOpenUndoGroupNotification

Posted whenever an `NSUndoManager` object opens an undo group, which occurs in the implementation of the `beginUndoGrouping` (page 8) method. The notification object is the `NSUndoManager` object. This notification does not contain a *`userInfo`* dictionary.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSUndoManager.h`

NSUndoManagerDidRedoChangeNotification

Posted just after an `NSUndoManager` object performs a redo operation (`redo` (page 15)). The notification object is the `NSUndoManager` object. This notification does not contain a *`userInfo`* dictionary.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`NSUndoManager.h`

NSUndoManagerDidUndoChangeNotification

Posted just after an `NSUndoManager` object performs an undo operation. If you invoke `undo` (page 20) or `undoNestedGroup` (page 22), this notification is posted. The notification object is the `NSUndoManager` object. This notification does not contain a *`userInfo`* dictionary.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`NSUndoManager.h`

NSUndoManagerWillCloseUndoGroupNotification

Posted before an `NSUndoManager` object closes an undo group, which occurs in the implementation of the `endUndoGrouping` (page 10) method. The notification object is the `NSUndoManager` object. This notification does not contain a *`userInfo`* dictionary.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`NSUndoManager.h`

NSUndoManagerWillRedoChangeNotification

Posted just before an `NSUndoManager` object performs a redo operation (`redo` (page 15)). The notification object is the `NSUndoManager` object. This notification does not contain a *`userInfo`* dictionary.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`NSUndoManager.h`

NSUndoManagerWillUndoChangeNotification

Posted just before an `NSUndoManager` object performs an undo operation. If you invoke `undo` (page 20) or `undoNestedGroup` (page 22), this notification is posted. The notification object is the `NSUndoManager` object. This notification does not contain a *`userInfo`* dictionary.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`NSUndoManager.h`

Document Revision History

This table describes the changes to *NSUndoManager Class Reference*.

Date	Notes
2007-01-18	Corrected description of return value for undoActionName: and redoActionName: if no action is registered.
2007-02-08	Corrected the return-value descriptions for undoActionName: and redoActionName: when no action is registered.
2006-05-23	First publication of this content as a separate document.

Document Revision History

Index

S

U