

[Foundation](#) / UndoManager

Class

UndoManager

A general-purpose recorder of operations that enables undo and redo.

iOS 3.0+ | iPadOS 3.0+ | Mac Catalyst 13.1+ | macOS 10.0+ | tvOS 9.0+ | visionOS 1.0+ | watchOS 2.0+

```
@MainActor
class UndoManager
```

Overview

You register an undo operation by calling one of the methods described in [Registering undo operations](#). You specify the name of the object that's changing (or the owner of that object) and provide a closure, method, or invocation to revert its state.

After you register an undo operation, you can call [undo\(\)](#) on the undo manager to revert to the state of the last undo operation. When undoing an action, [UndoManager](#) saves the operations you revert to so that you can call [redo\(\)](#) automatically.

Typically, apps with UI interactions work with [UndoManager](#). For example, UIKit implements undo and redo in its text view object, making it easy for you to undo and redo actions in objects along the responder chain. [UndoManager](#) also serves as a general-purpose state manager, which you can use to undo and redo many kinds of actions. For example, an interactive command-line utility can use this class to undo the last command run, or a networking library can undo a request by sending another request that invalidates the previous one.

Important

UndoManager is MainActor-isolated in Swift, making it safe to use in UI frameworks like AppKit and UIKit that expect to execute code on the main thread, queue, or actor. When registering an undoable action with registerUndo(withTarget:handler:), the handler closure is also MainActor-isolated to ensure safety and simplify ergonomics.

Topics

Registering undo operations

```
func registerUndo<TargetType>(withTarget: TargetType, handler: (TargetType) -> Void)
```

Registers the specified closure to implement a single undo operation that the target receives.

```
func registerUndo(withTarget: Any, selector: Selector, object: Any?)
```

Registers the selector of the specified target to implement a single undo operation that the target receives.

```
func prepare(withInvocationTarget: Any) -> Any
```

Prepares the undo manager for invocation-based undo with the given target as the subject of the next undo operation.

Checking undo ability

```
var canUndo: Bool
```

A Boolean value that indicates whether the manager has any actions to undo.

```
var canRedo: Bool
```

A Boolean value that indicates whether the manager has any actions to redo.

Performing undo and redo

```
func undo()
```

Closes the top-level undo group if necessary, and then performs undo operations on the group.

```
func 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.

```
func 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.

Managing undo and redo stack depth

```
var levelsOfUndo: Int
```

The maximum number of top-level undo groups the undo manager holds.

```
var undoCount: Int
```

The number of times you can invoke undo before there are no actions left to undo.

```
var redoCount: Int
```

The number of times you can invoke redo before there are no actions left to redo.

Creating undo groups

```
func beginUndoGrouping()
```

Marks the beginning of an undo group.

```
func endUndoGrouping()
```

Marks the end of an undo group.

```
var groupsByEvent: Bool
```

A Boolean value that indicates whether the manager automatically creates undo groups around each pass of the run loop.

```
var groupingLevel: Int
```

The number of nested undo groups (or redo groups, if redo is the most recent operation) in the current event loop.

Enabling and disabling undo

```
func disableUndoRegistration()
```

Disables the recording of undo operations.

```
func enableUndoRegistration()
```

Enables the recording of undo operations.

```
var isUndoRegistrationEnabled: Bool
```

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

Checking whether undo or redo is in process

```
var isUndoing: Bool
```

Returns a Boolean value that indicates whether the manager is in the process of performing an undo action.

```
var isRedoing: Bool
```

Returns a Boolean value that indicates whether the manager is in the process of performing a redo action.

Clearing undo operations

```
func removeAllActions()
```

Clears the undo and redo stacks and reenables the manager.

```
func removeAllActions(withTarget: Any)
```

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

Managing the action name

```
var undoActionName: String
```

The name identifying the undo action.

```
var redoActionName: String
```

The name identifying the redo action.

```
func setActionName(LocalizedStringResource?)
```

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

```
func setActionName(String)
```

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

Getting and localizing the menu item title

```
var undoMenuItemTitle: String
```

The title of the Undo menu command, such as Undo Paste.

```
var redoMenuItemTitle: String
```

The title of the Redo menu command, such as Redo Paste.

```
func undoMenuItemTitle(forUndoActionName: String) -> String
```

Returns the localized title of the Undo menu command for the identified action.

```
func redoMenuItemTitle(forUndoActionName: String) -> String
```

Returns the localized title of the Redo menu command for the identified action.

Working with user info

```
func setActionUserInfoValue(Any?, forKey: UndoManager.UserInfoKey)
```

Sets a user info value for an undo or redo action.

```
func undoActionUserInfoValue(forKey: UndoManager.UserInfoKey) -> Any?
```

Retrieves the undo action's user info value for the given key.

```
func redoActionUserInfoValue(forKey: UndoManager.UserInfoKey) -> Any?
```

Retrieves the redo action's user info value for the given key.

```
struct UserInfoKey
```

An extensible namespace for undo and redo user info keys.

Working with run loops

```
var runLoopModes: [RunLoop.Mode]
```

The modes governing the types of input to handle during a cycle of the run loop.

```
let NSUndoCloseGroupingRunLoopOrdering: Int
```

A priority to use when using a run loop to close an undo group.

Using discardable undo and redo actions

```
func setActionIsDiscardable(Bool)
```

Sets whether the next undo or redo action is discardable.

```
var undoActionIsDiscardable: Bool
```

A Boolean value that indicates whether the next undo action is discardable.

```
var redoActionIsDiscardable: Bool
```

A Boolean value that indicates whether the next redo action is discardable.

Working with notifications

```
static let NSUndoManagerWillUndoChange: NSNotification.Name
```

Posted just before an undo manager performs an undo operation.

```
static let NSUndoManagerDidUndoChange: NSNotification.Name
```

Posted just after an undo manager performs an undo operation.

```
static let NSUndoManagerWillRedoChange: NSNotification.Name
```

Posted just before an undo manager performs a redo operation.

```
static let NSUndoManagerDidRedoChange: NSNotification.Name
```

Posted just after an undo manager performs a redo operation.

```
static let NSUndoManagerCheckpoint: NSNotification.Name
```

Posted whenever an undo manager opens or closes an undo group (except when it opens a top-level group) and when checking the redo stack.

```
static let NSUndoManagerDidOpenUndoGroup: NSNotification.Name
```

Posted whenever an undo manager opens an undo group.

```
static let NSUndoManagerWillCloseUndoGroup: NSNotification.Name
```

Posted before an undo manager closes an undo group.

```
static let NSUndoManagerDidCloseUndoGroup: NSNotification.Name
```

Posted after an undo manager closes an undo group.

```
let NSUndoManagerGroupIsDiscardableKey: String
```

A key, used in a notification's user info, that indicates the undo group contains only discardable actions.

Working with notification messages

```
struct WillUndoChangeMessage
```

A message that an undo manager sends before undoing a change.

```
struct DidUndoChangeMessage
```

A message that an undo manager sends after undoing a change.

`struct WillRedoChangeMessage`

A message that an undo manager sends before redoing a change.

`struct DidRedoChangeMessage`

A message that an undo manager sends after redoing a change.

`struct CheckpointMessage`

A message that an undo manager sends at certain checkpoints.

`struct DidOpenUndoGroupMessage`

A message that an undo manager sends after opening an undo group.

`struct WillCloseUndoGroupMessage`

A message that an undo manager sends before closing an undo group.

`struct DidCloseUndoGroupMessage`

A message that an undo manager sends after closing an undo group.

Relationships

Inherits From

`NSObject`

Conforms To

`CVarArg`

`CustomDebugStringConvertible`

`CustomStringConvertible`

`Equatable`

`Hashable`

`NSObjectProtocol`

`Sendable`