Class

NSOutlineView

A view that uses a row-and-column format to display hierarchical data like directories and files that can be expanded and collapsed.

macOS

```
@MainActor
class NSOutlineView
```

Overview

Like a table view, an outline view does not store its own data, instead it retrieves data values as needed from a data source to which it has a weak reference (see Delegates and Data Sources). See `NSOutlineViewDataSource`, which declares the methods that an `NSOutlineView` object uses to access the contents of its data source object.

An outline view has the following features:

- A user can expand and collapse rows, edit values, and resize and rearrange columns.
- Each item in the outline view must be unique. In order for the collapsed state to remain consistent between reloads the item's pointer must remain the same and the item must maintain `isEqual(_:)` sameness.
- The view gets data from a data source (see `NSOutlineViewDataSource`).
- The view retrieves only the data that needs to be displayed.

Important

It is possible that your data source methods for populating the outline view may be called before `awakeFromNib()` if the data source is specified in Interface Builder. You should defend against this by having the data source's `outlineView(_:numberOfChildrenOfItem:)` method return `0` for the number of items when the data source has not yet been configured. In `awakeFromNib()`, when the data source is initialized you should always call `reloadData()`.

For more information about using NSOutlineView in your app, see Navigating Hierarchical Data Using Outline and Split Views.

Subclassing

Subclassing `NSOutlineView` is not recommended. Customization can be accomplished in your data source class implementation (conforming to `NSOutlineViewDataSource`) or your delegate class implementation (conforming to `NSOutlineViewDelegate`).

Topics

Accessing the Data Source

`var dataSource: (any NSOutlineViewDataSource)?`

The object that provides the data displayed by the receiver.

`var stronglyReferencesItems: Bool`

A Boolean value that indicates whether the outline view retains and releases the objects returned from its data source.

Working with Expandability

`func isExpandable(Any?) -> Bool`

Returns a Boolean value that indicates whether a given item is expandable.

`func isItemExpanded(Any?) -> Bool`

Returns a Boolean value that indicates whether a given item is expanded.

Expanding and Collapsing the Outline

`func expandItem(Any?)`

Expands a given item.

`func expandItem(Any?, expandChildren: Bool)`

Expands a specified item and, optionally, its children.

`func collapseItem(Any?)`

Collapses a given item.

`func collapseItem(Any?, collapseChildren: Bool)`

Collapses a given item and, optionally, its children.

Redisplaying Information

`func reloadItem(Any?)`

Reloads and redisplays the data for the given item.

`func reloadItem(Any?, reloadChildren: Bool)`

Reloads a given item and, optionally, its children.

Converting Between Items and Rows

`func item(atRow: Int) -> Any?`

Returns the item associated with a given row.

`func row(forItem: Any?) -> Int`

Returns the row associated with a given item.

Working with the Outline Column

`var outlineTableColumn: NSTableColumn?`

The table column in which hierarchical data is displayed.

`var autoresizesOutlineColumn: Bool`

A Boolean value that indicates whether the outline view resizes its outline column when the user expands or collapses items.

Working with Indentation

`func level(forItem: Any?) -> Int`

Returns the indentation level for a given item.

```
func level(forRow: Int) -> Int
```

Returns the indentation level for a given row.

```
var indentationPerLevel: CGFloat
```

The per-level indentation, measured in points.

```
var indentationMarkerFollowsCell: Bool
```

A Boolean value indicating whether the indentation marker symbol displayed in the outline column should be indented along with the cell contents.

Working with Persistence

```
var autosaveExpandedItems: Bool
```

A Boolean value indicating whether the expanded items are automatically saved across launches of the app.

Supporting Drag and Drop

```
func setDropItem(Any?, dropChildIndex: Int)
```

Used to “retarget” a proposed drop.

```
func shouldCollapseAutoExpandedItems(forDeposited: Bool) -> Bool
```

Returns a Boolean value that indicates whether auto-expanded items should return to their original collapsed state.

Getting Related Items

```
func parent(forItem: Any?) -> Any?
```

Returns the parent for a given item.

```
func childIndex(forItem: Any) -> Int
```

Returns the child index of the specified item within its parent.

```
func child(Int, ofItem: Any?) -> Any?
```

Returns the specified child of an item.

```
func numberOfChildren(ofItem: Any?) -> Int
```

Returns the number of children for the specified parent item.

Getting the Frame for a Cell

```
func frameOfOutlineCell(atRow: Int) -> NSRect
```

Returns the frame of the outline cell for a given row.

Accessing the Delegate

```
var delegate: (any NSOutlineViewDelegate)?
```

The outline view's delegate.

Manipulating Items

```
func insertItems(at: IndexSet, inParent: Any?, withAnimation: NSTableView.AnimationOptions)
```

Inserts new items at the given indexes in the given parent with the specified optional animations.

```
func moveItem(at: Int, inParent: Any?, to: Int, inParent: Any?)
```

Moves an item at a given index in the given parent to a new index in a new parent.

```
func removeItems(at: IndexSet, inParent: Any?, withAnimation: NSTableView.AnimationOptions)
```

Removes items at the given indexes in the given parent with the specified optional animations.

User Interface Layout Direction

```
var userInterfaceLayoutDirection: NSUserInterfaceLayoutDirection
```

The user interface layout direction.

Constants

- Drop on Item Index

  This constant defines an index that allows you to drop an item directly on a target.

- Outline View Button Keys

  These keys are used by the outline view to create disclosure buttons that collapse and expand items.

Notifications

`class let` **columnDidMoveNotification**: `NSNotification.Name`

Posted whenever a column is moved by user action in an `NSOutlineView` object.

`class let` **columnDidResizeNotification**: `NSNotification.Name`

Posted whenever a column is resized in an `NSOutlineView` object.

`class let` **itemDidCollapseNotification**: `NSNotification.Name`

Posted whenever an item is collapsed in an `NSOutlineView` object.

`class let` **itemDidExpandNotification**: `NSNotification.Name`

Posted whenever an item is expanded in an `NSOutlineView` object.

`class let` **itemWillCollapseNotification**: `NSNotification.Name`

Posted before an item is collapsed (after the user clicks the arrow but before the item is collapsed).

`class let` **itemWillExpandNotification**: `NSNotification.Name`

Posted before an item is expanded (after the user clicks the arrow but before the item is collapsed).

`class let` **selectionDidChangeNotification**: `NSNotification.Name`

Posted after the outline view's selection changes.

`class let` **selectionIsChangingNotification**: `NSNotification.Name`

Posted as the outline view's selection changes (while the mouse button is still down).

Relationships

Inherits From

`NSTableView`

Conforms To

`CVarArg`
`CustomDebugStringConvertible`

CustomStringConvertible
Equatable
Hashable
NSAccessibilityElementProtocol
NSAccessibilityGroup
NSAccessibilityOutline
NSAccessibilityProtocol
NSAccessibilityTable
NSAnimatablePropertyContainer
NSAppearanceCustomization
NSCoding
NSDraggingDestination
NSDraggingSource
NSObjectProtocol
NSStandardKeyBindingResponding
NSTextDelegate
NSTextViewDelegate
NSTouchBarProvider
NSUserActivityRestoring
NSUserInterfaceItemIdentification
NSUserInterfaceValidations
Sendable
SendableMetatype

See Also

View

{} Navigating Hierarchical Data Using Outline and Split Views

Build a structured user interface that simplifies navigation in your app.