AppKit / NSTreeController

Class

NSTreeController

A bindings-compatible controller that manages a tree of objects.

macOS

```
class NSTreeController
```

Overview

The `NSTreeController` class provides selection and sort management. Its primary purpose is to act as the controller when binding `NSOutlineView` and `NSBrowser` instances to a hierarchical collection of objects. The root content object of the tree can be a single object, or an array of objects.

An `NSTreeController` object requires that you describe how the tree of objects is traversed by specifying the key-path for child objects specified by `childrenKeyPath`. All child objects for the tree must be key-value coding compliant for the same child key path. If necessary, you should add properties to your model classes that map the child key name to the appropriate class-specific property name.

Child objects can implement a count method (specified to the tree controller using `countKeyPath`) that, if provided, returns the number of child objects available. Your model objects are expected to update the value of the count key path in a key-value observing compliant method. Optionally, you can also provide a leaf key path using `leafKeyPath` that specifies a key in your model object that returns `true` if the object is a leaf node, and `false` if it is not. Changes to the leaf node value of the child object should be made in a key-value observing compliant manner. Providing the leaf node key path can improve performance, because it prevents the `NSTreeController` from having to examine the child object to determine if it is a leaf node.

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

Topics

Managing Sort Descriptors

`var sortDescriptors: [NSSortDescriptor]`

An array containing the sort descriptors used to arrange the tree controller’s content.

Setting the content

`var content: Any?`

The tree controller’s content object.

Arranging Objects

`var arrangedObjects: NSTreeNode`

The tree controller’s sorted content objects.

`func rearrangeObjects()`

Use this method to trigger reordering of the tree controller’s content.

Getting the current selection

`func setSelectionIndexPath(IndexPath?) -> Bool`

Sets the tree controller’s current selection.

`var selectionIndexPath: IndexPath?`

The index path of the first selected object.

`func setSelectionIndexPaths([IndexPath]) -> Bool`

Sets the tree controller’s current selection to the specified index paths.

`var selectionIndexPaths: [IndexPath]`

An array containing the index paths of the currently selected objects.

`var selectedObjects: [Any]`

An array containing the currently selected objects in the tree controller’s content.

`var selectedNodes: [NSTreeNode]`

An array containing the tree controller's selected tree nodes.

Managing Selections

`var selectsInsertedObjects: Bool`

A Boolean value that indicates whether the tree controller automatically selects objects as they are inserted.

`func addSelectionIndexPaths([IndexPath]) -> Bool`

Adds the objects at the specified `indexPaths` in the tree controller's content to the current selection.

`func removeSelectionIndexPaths([IndexPath]) -> Bool`

Removes the objects at the specified index paths from the tree controller's current selection.

`var avoidsEmptySelection: Bool`

A Boolean value that indicates whether the tree controller requires the content array to attempt to maintain a selection at all times, avoiding an empty selection.

`var preservesSelection: Bool`

A Boolean value that indicates whether the tree controller will attempt to preserve the current selection when the content changes.

`var alwaysUsesMultipleValuesMarker: Bool`

A Boolean value that indicates whether the tree controller always returns the multiple values marker when multiple objects are selected, even if the selected items have the same value.

Adding, inserting and removing objects

`func add(Any?)`

Adds an object to the tree controller's content after the current selection.

`func addChild(Any?)`

Adds a child object to the currently selected item.

`var canAddChild: Bool`

A Boolean value that indicates if a child object can be added to the tree controller's content.

`var canInsert: Bool`

A Boolean value that indicates if an object can be inserted into the tree controller's content.

`var canInsertChild: Bool`

A Boolean value that indicates if a child object can be inserted into the tree controller's content.

`func insert(Any?)`

Creates a new object of the class specified by `objectClass` and inserts it into the tree controller's content.

`func insertChild(Any?)`

Creates a new object of the class specified by `objectClass` and inserts it into the tree controller's content as a child of the current selection.

`func insert(Any?, atArrangedObjectIndexPath: IndexPath)`

Inserts `object` into the tree controller's arranged objects array at the location specified by `indexPath`, and adds it to the tree controller's content.

`func insert([Any], atArrangedObjectIndexPaths: [IndexPath])`

Inserts `objects` into the tree controller's arranged objects array at the locations specified in `indexPaths`, and adds them to the tree controller's content.

`func remove(Any?)`

Removes the tree controller's selected objects from the content.

`func removeObject(atArrangedObjectIndexPath: IndexPath)`

Removes the object at the specified `indexPath` in the tree controller's arranged objects from the tree controller's content.

`func removeObjects(atArrangedObjectIndexPaths: [IndexPath])`

Removes the objects at the specified `indexPaths` in the tree controller's arranged objects from the tree controller's content.

`func move(NSTreeNode, to: IndexPath)`

Moves the specified tree node to the new index path.

`func move([NSTreeNode], to: IndexPath)`

Moves the specified tree nodes to the new index path.

Specifying model attributes

`var childrenKeyPath: String?`

The key path used to find the children in the tree controller's objects.

`func childrenKeyPath(for: NSTreeNode) -> String?`

Returns the key path used to find the children in the specified tree node.

`var countKeyPath: String?`

The key path used to find the number of children for a node.

`func countKeyPath(for: NSTreeNode) -> String?`

Returns the key path that provides the number of children for a specified node.

`var leafKeyPath: String?`

The key path used by the tree controller to determine if a node is a leaf key.

`func leafKeyPath(for: NSTreeNode) -> String?`

Returns the key path that specifies whether the node is a leaf node.

Relationships

Inherits From

`NSObjectController`

Conforms To

`CVarArg`
`CustomDebugStringConvertible`
`CustomStringConvertible`
`Equatable`
`Hashable`
`NSCoding`
`NSEditor`
`NSEditorRegistration`
`NSObjectProtocol`
`Sendable`
`SendableMetatype`

See Also

Tree-Based Data

{} Navigating Hierarchical Data Using Outline and Split Views

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

`class NSTreeNode`

A node in a tree of nodes.