Class

NSRuleEditor

An interface for configuring a rule-based list of options.

macOS

```
@MainActor
class NSRuleEditor
```

Overview

A rule editor lets the user visually create and configure a list of options that are expressed as a predicate (as described in Predicate Programming Guide). Each row displayed by the rule editor represents a particular path down a tree of choices. The rule editor's delegate provides the tree of choices to be displayed. The rule editor presents those choices to the user as a row of popup buttons, static text fields, and custom views.

`NSRuleEditor` exposes one binding, `rows`. You can bind `rows` to an ordered collection (such as an instance of `NSMutableArray`). Each object in the collection should have the following properties:

@"rowType"
: An integer representing the type of the row (`NSRuleEditorRowType`).

@"subrows"
: An ordered to-many relation (such as an instance of `NSMutableArray`) containing the directly nested subrows for the given row.

@"displayValues"
: An ordered to-many relation containing the display values for the row.

@"criteria"
: An ordered to-many relation containing the criteria for the row.

> **Note**
>
> If you override `viewDidMoveToWindow()` in a subclass of `NSRuleEditor`, you must invoke super's implementation.

Topics

Configuring the Delegate

`var delegate: (any NSRuleEditorDelegate)?`

The rule editor's delegate.

`protocol NSRuleEditorDelegate`

The `NSRuleEditorDelegate` protocol defines the optional methods implemented by delegates of `NSRuleEditor` objects.

Configuring a Rule Editor

`var isEditable: Bool`

A Boolean value that determines whether the rule editor is editable.

`var nestingMode: NSRuleEditor.NestingMode`

The rule editor's nesting mode.

`enum NestingMode`

Specifies a type for nesting modes.

`var canRemoveAllRows: Bool`

A Boolean value that indicates whether all the rows can be removed.

`var rowHeight: CGFloat`

The rule editor's row height.

Working with Formatting

`var formattingDictionary: [String : String]?`

The formatting dictionary for the rule editor.

`var formattingStringsFilename: String?`

The name of the rule editor's strings file.

Providing Data

```
func reloadCriteria()
```

Instructs the receiver to refetch criteria from its delegate.

```
func setCriteria([Any], andDisplayValues: [Any], forRowAt: Int)
```

Modifies the row at a given index to contain the given items and values.

```
func criteria(forRow: Int) -> [Any]
```

Returns the currently chosen items for a given row.

```
func displayValues(forRow: Int) -> [Any]
```

Returns the chosen values for a given row.

Obtaining Row Information

```
var numberOfRows: Int
```

The number of rows in the rule editor.

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

Returns the index of the parent of a given row.

```
func row(forDisplayValue: Any) -> Int
```

Returns the index of the row containing a given value.

```
func rowType(forRow: Int) -> NSRuleEditor.RowType
```

Returns the type of a given row.

```
enum RowType
```

Specifies a type for row types.

```
func subrowIndexes(forRow: Int) -> IndexSet
```

Returns the immediate subrows of a given row.

Working with the Selection

```
var selectedRowIndexes: IndexSet
```

The indexes of the rule editor's selected rows.

`func selectRowIndexes(IndexSet, byExtendingSelection: Bool)`

Sets in the receiver the indexes of rows that are selected.

Manipulating Rows

`func addRow(Any?)`

Adds a row to the receiver.

`func insertRow(at: Int, with: NSRuleEditor.RowType, asSubrowOfRow: Int, animate: Bool)`

Adds a new row of a given type at a given location.

`func removeRow(at: Int)`

Removes the row at a given index.

`func removeRows(at: IndexSet, includeSubrows: Bool)`

Removes the rows at given indexes.

Working with Predicates

`var predicate: NSPredicate?`

The rule editor's predicate.

`func reloadPredicate()`

Instructs the receiver to regenerate its predicate by invoking the corresponding delegate method.

`func predicate(forRow: Int) -> NSPredicate?`

Returns the predicate for a given row.

Supporting Bindings

`var rowClass: AnyClass`

The class used to create a new row in the "rows" binding.

`var rowTypeKeyPath: String`

The key path for the row type.

`var subrowsKeyPath: String`

The key path for the subrows.

`var criteriaKeyPath: String`

The criteria key path.

`var displayValuesKeyPath: String`

The display values key path.

Notifications

`class let rowsDidChangeNotification: NSNotification.Name`

This notification is posted to the default notification center whenever the view's rows change.

Relationships

Inherits From

`NSControl`

Inherited By

`NSPredicateEditor`

Conforms To

`CVarArg`
`CustomDebugStringConvertible`
`CustomStringConvertible`
`Equatable`
`Hashable`
`NSAccessibilityElementProtocol`
`NSAccessibilityProtocol`
`NSAnimatablePropertyContainer`
`NSAppearanceCustomization`
`NSCoding`
`NSDraggingDestination`
`NSObjectProtocol`
`NSStandardKeyBindingResponding`
`NSTouchBarProvider`
`NSUserActivityRestoring`

`NSUserInterfaceItemIdentification`
`Sendable`
`SendableMetatype`

See Also

Controls

Responding to control-based events using target-action

Handle user input by connecting buttons, sliders, and other controls to your app's code using the target-action design pattern.

`class` `NSButton`

A control that defines an area on the screen that a user clicks to trigger an action.

`class` `NSColorWell`

A control that displays a color value and lets the user change that color value.

Combo Box

Display a list of values in a pop-up menu that lets the user select a value or type in a custom value.

`class` `NSComboButton`

A button with a pull-down menu and a default action.

Date Picker

Display a calendar date and provide controls for editing the date value.

`class` `NSImageView`

A display of image data in a frame.

`class` `NSLevelIndicator`

A visual representation of a level or quantity, using discrete values.

Path Control

A display of a file system path or virtual path information.

`class` `NSPopUpButton`

A control for selecting an item from a list.

`class` NSProgressIndicator

An interface that provides visual feedback to the user about the status of an ongoing task.

`class` NSPredicateEditor

A defined set of rules that allows the editing of predicate objects.

Search Field

Provide a text field that is optimized for text-based search interfaces.

`class` NSSegmentedControl

Display one or more buttons in a single horizontal group.

Slider

Display a range of values from which the user selects a single value.