

---

# NSPredicateEditorRowTemplate Class Reference

[Cocoa](#) > [User Experience](#)





Apple Inc.  
© 2007 Apple Inc.  
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

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

Apple, the Apple logo, Cocoa, Mac, and Mac OS are trademarks of Apple Inc., registered in the United States and other countries.

Simultaneously published in the United States and Canada.

**Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.**

**IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY**

**DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.**

**THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.**

**Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.**

# Contents

---

## **NSPredicateEditorRowTemplate Class Reference 5**

---

Overview	5
Tasks	6
Initializing a Template	6
Core Data Integration	6
Primitive Methods	6
Information About a Row Template	7
Class Methods	7
templatesWithAttributeKeyPaths:inEntityDescription:	7
Instance Methods	8
compoundTypes	8
displayableSubpredicatesOfPredicate:	8
initWithCompoundTypes:	8
initWithLeftExpressions:rightExpressionAttributeType:modifier:operators:options:	9
initWithLeftExpressions:rightExpressions:modifier:operators:options:	10
leftExpressions	10
matchForPredicate:	11
modifier	11
operators	11
options	12
predicateWithSubpredicates:	12
rightExpressionAttributeType	12
rightExpressions	13
setPredicate:	13
templateViews	13

---

## **Document Revision History 15**

---

## **Index 17**

---



# NSPredicateEditorRowTemplate Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSCoding NSCopying NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/AppKit.framework
<b>Availability</b>	Available in Mac OS X v10.5 and later.
<b>Declared in</b>	NSPredicateEditorRowTemplate.h
<b>Companion guides</b>	Control and Cell Programming Topics for Cocoa Predicate Programming Guide
<b>Related sample code</b>	PredicateEditorSample

## Overview

`NSPredicateEditorRowTemplate` describes available predicates and how to display them.

You can create instances of `NSPredicateEditorRowTemplate` programmatically or in Interface Builder. By default, a non-compound row template has three views: a popup (or static text field) on the left, a popup or static text field for operators, and either a popup or other view on the right. You can subclass `NSPredicateEditorRowTemplate` to create a row template with different numbers or types of views.

`NSPredicateEditorRowTemplate` is a concrete class, but it has five primitive methods which are called by `NSPredicateEditor`: [templateViews](#) (page 13), [matchForPredicate:](#) (page 11), [setPredicate:](#) (page 13), [displayableSubpredicatesOfPredicate:](#) (page 8), and [predicateWithSubpredicates:](#) (page 12). `NSPredicateEditorRowTemplate` implements all of them, but you can override them for custom templates. The primitive methods are used by an instance of `NSPredicateEditor` as follows.

First, an instance of `NSPredicateEditor` is created, and some row templates are set on it—either through a nib file or programmatically. The first thing predicate editor does is ask each of the templates for their views, using [templateViews](#) (page 13).

After setting up the predicate editor, you typically send it a `setObjectValue:` message to restore a saved predicate. `NSPredicateEditor` needs to determine which of its templates should display each predicate in the predicate tree. It does this by sending each of its row templates a [matchForPredicate:](#) (page 11) message and choosing the one that returns the highest value.

After finding the best match for a predicate, `NSPredicateEditor` copies that template to get fresh views, inserts them into the proper row, and then sets the predicate on the template using `setPredicate:` (page 13). Within that method, the `NSPredicateEditorRowTemplate` object must set its views' values to represent that predicate.

`NSPredicateEditorRowTemplate` next asks the template for the “displayable sub-predicates” of the predicate by sending a `displayableSubpredicatesOfPredicate:` (page 8) message. If a template represents a predicate in its entirety, or if the predicate has no subpredicates, it can return `nil` for this. Otherwise, it should return a list of predicates to be made into sub-rows of that template's row. The whole process repeats for each sub-predicate.

At this point, the user sees the predicate that was saved. If the user then makes some changes to the views of the templates, this causes `NSPredicateEditor` to recompute its predicate by asking each of the templates to return the predicate represented by the new view values, passing in the subpredicates represented by the sub-rows (an empty array if there are none, or `nil` if they aren't supported by that predicate type):

`predicateWithSubpredicates:` (page 12)

## Tasks

### Initializing a Template

- `initWithLeftExpressions:rightExpressions:modifier:operators:options:` (page 10)  
Initializes and returns a “pop-up-pop-up-pop-up”-style row template.
- `initWithLeftExpressions:rightExpressionAttributeType:modifier:operators:options:` (page 9)  
Initializes and returns a “pop-up-pop-up-view”-style row template.
- `initWithCompoundTypes:` (page 8)  
Initializes and returns a row template suitable for displaying compound predicates.

### Core Data Integration

- + `templatesWithAttributeKeyPaths:inEntityDescription:` (page 7)  
Returns an array of predicate templates for the given attribute key paths for a given entity.

### Primitive Methods

- `matchForPredicate:` (page 11)  
Returns a positive number if the receiver can represent a given predicate, and 0 if it cannot.
- `templateViews` (page 13)  
Returns the views for the receiver.
- `setPredicate:` (page 13)  
Sets the value of the views according to the given predicate.
- `displayableSubpredicatesOfPredicate:` (page 8)  
Returns the subpredicates that should be made sub-rows of a given predicate.

- [predicateWithSubpredicates:](#) (page 12)  
Returns the predicate represented by the receiver's views' values and the given sub-predicates.

## Information About a Row Template

- [leftExpressions](#) (page 10)  
Returns the left hand expressions for the receiver.
- [rightExpressions](#) (page 13)  
Returns the right hand expressions for the receiver.
- [compoundTypes](#) (page 8)  
Returns the compound predicate types for the receiver.
- [modifier](#) (page 11)  
Returns the comparison predicate modifier for the receiver.
- [operators](#) (page 11)  
Returns the array of operators for the receiver.
- [options](#) (page 12)  
Returns the comparison predicate options for the receiver.
- [rightExpressionAttributeType](#) (page 12)  
Returns the attribute type of the receiver's right expression.

## Class Methods

### templatesWithAttributeKeyPaths:inEntityDescription:

Returns an array of predicate templates for the given attribute key paths for a given entity.

```
+ (NSArray *)templatesWithAttributeKeyPaths:(NSArray *)keyPaths
inEntityDescription:(NSEntityDescription *)entityDescription
```

#### Parameters

*keyPaths*

An array of attribute key paths originating at *entityDescription*. The key paths may cross relationships but must terminate in attributes.

*entityDescription*

A Core Data entity description.

#### Return Value

An array of predicate templates for *keyPaths* originating at *entityDescription*.

#### Discussion

This method determines which key paths in the entity description can use the same views (that is, share the same attribute type). For each of these groups, it instantiates individual templates via [initWithLeftExpressions:rightExpressions:modifier:operators:options:](#) (page 10).

#### Availability

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

## Instance Methods

### compoundTypes

Returns the compound predicate types for the receiver.

- (NSArray \*)compoundTypes

**Return Value**An array of `NSNumber` objects specifying compound predicate types. See `Compound_Predicate_Types` for possible values.**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

### displayableSubpredicatesOfPredicate:

Returns the subpredicates that should be made sub-rows of a given predicate.

- (NSArray \*)displayableSubpredicatesOfPredicate:(NSPredicate \*)*predicate***Parameters***predicate*

A predicate object.

**Return Value**The subpredicates that should be made sub-rows of *predicate*. For compound predicates (instances of `NSCompoundPredicate`), the array of subpredicates; for other types of predicate, returns `nil`. If a template represents a predicate in its entirety, or if the predicate has no subpredicates, returns `nil`.**Discussion**

You can override this method to create custom templates that handle complicated compound predicates.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

### initWithCompoundTypes:

Initializes and returns a row template suitable for displaying compound predicates.

- (id)initWithCompoundTypes:(NSArray \*)*compoundTypes*



**Parameters***compoundTypes*

An array of `NSNumber` objects specifying compound predicate types. See `Compound_Predicate_Types` for possible values.

**Return Value**

A row template initialized for displaying compound predicates of the types specified by *compoundTypes*.

**Discussion**

`NSPredicateEditor` contains such a template by default.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

`NSPredicateEditorRowTemplate.h`

**`initWithLeftExpressions:rightExpressionAttributeType:modifier:operators:options:`**

Initializes and returns a “pop-up-pop-up-view”-style row template.

```
- (id)initWithLeftExpressions:(NSArray *)leftExpressions
    rightExpressionAttributeType:(NSAttributeType)attributeType
    modifier:(NSComparisonPredicateModifier)modifier operators:(NSArray *)operators
    options:(NSUInteger)options
```

**Parameters***leftExpressions*

An array of `NSExpression` objects that represent the left hand side of a predicate.

*attributeType*

An attribute type for the right hand side of a predicate. This value dictates the type of view created, and how the control's object value is coerced before putting it into a predicate.

*modifier*

A modifier for the predicate (see `NSComparisonPredicateModifier` for possible values).

*operators*

An array of `NSNumber` objects specifying the operator type (see `NSPredicateOperatorType` for possible values).

*options*

Options for the predicate (see `NSComparisonPredicate_Options` for possible values).

**Return Value**

A row template initialized using the given arguments.

**Discussion**

The type of *attributeType* dictates the type of view created. For example, `NSDateAttributeType` will create an `NSDatePicker` object, `NSInteger64AttributeType` will create a short text field, and `NSStringAttributeType` will produce a longer text field. You can resize the views as you want.

Predicates do not automatically coerce types for you. For example, comparing a number to a string will raise an exception. Therefore, the attribute type is also needed to determine how the control's object value must be coerced before putting it into a predicate.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

**initWithLeftExpressions:rightExpressions:modifier:operators:options:**

Initializes and returns a “pop-up-pop-up-pop-up”-style row template.

```
- (id)initWithLeftExpressions:(NSArray *)leftExpressions rightExpressions:(NSArray *)rightExpressions modifier:(NSComparisonPredicateModifier)modifier operators:(NSArray *)operators options:(NSUInteger)options
```

**Parameters**

*leftExpressions*

An array of `NSExpression` objects that represent the left hand side of a predicate.

*rightExpressions*

An array of `NSExpression` objects that represent the right hand side of a predicate.

*modifier*

A modifier for the predicate (see `NSComparisonPredicateModifier` for possible values).

*operators*

An array of `NSNumber` objects specifying the operator type (see `NSPredicateOperatorType` for possible values).

*options*

Options for the predicate (see `NSComparisonPredicate_Options` for possible values).

**Return Value**

A row template of the “pop-up-pop-up-pop-up”-form, with the left and right popups representing the left and right expression arrays `leftExpressions` and `rightExpressions`, and the center popup representing the operators.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

**leftExpressions**

Returns the left hand expressions for the receiver.

```
- (NSArray *)leftExpressions
```

**Return Value**

The left hand expressions for the receiver

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

## matchForPredicate:

Returns a positive number if the receiver can represent a given predicate, and 0 if it cannot.

- (double)matchForPredicate:(NSPredicate \*)*predicate*

### Return Value

A positive number if the template can represent *predicate*, and 0 if it cannot.

### Discussion

By default, returns values in the range 0 to 1.

The highest match among all the templates determines which template is responsible for displaying the predicate. You can override this to determine which predicates your custom template handles.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSPredicateEditorRowTemplate.h

## modifier

Returns the comparison predicate modifier for the receiver.

- (NSComparisonPredicateModifier)modifier

### Return Value

The comparison predicate modifier for the receiver.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSPredicateEditorRowTemplate.h

## operators

Returns the array of operators for the receiver.

- (NSArray \*)operators

### Return Value

The array of operators for the receiver.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSPredicateEditorRowTemplate.h

## options

Returns the comparison predicate options for the receiver.

- (NSInteger)options

### Return Value

The comparison predicate options for the receiver. See `NSComparisonPredicate_Options` for possible values. Returns 0 if this does not apply (for example, for a compound template initialized with `initWithCompoundTypes:` (page 8)).

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

`NSPredicateEditorRowTemplate.h`

## predicateWithSubpredicates:

Returns the predicate represented by the receiver's views' values and the given sub-predicates.

- (NSPredicate \*)predicateWithSubpredicates:(NSArray \*)subpredicates

### Parameters

*subpredicates*

An array of predicates.

### Return Value

The predicate represented by the values of the template's views and the given subpredicates. You can override this method to return the predicate represented by your custom views.

### Discussion

This method is only called if `matchForPredicate:` (page 11) returned a positive value for the receiver.

You can override this method to return the predicate represented by a custom view.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

`NSPredicateEditorRowTemplate.h`

## rightExpressionAttributeType

Returns the attribute type of the receiver's right expression.

- (NSAttributeType)rightExpressionAttributeType

### Return Value

The attribute type of the receiver's right expression.

### Availability

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

**rightExpressions**

Returns the right hand expressions for the receiver.

- (NSArray \*)rightExpressions

**Return Value**

The right hand expressions for the receiver

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

**setPredicate:**

Sets the value of the views according to the given predicate.

- (void)setPredicate:(NSPredicate \*)*predicate***Parameters***predicate*

The predicate value for the receiver.

**Discussion**This method is only called if [matchForPredicate:](#) (page 11) returned a positive value for the receiver.

You can override this to set the values of custom views.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

**templateViews**

Returns the views for the receiver.

- (NSArray \*)templateViews

**Return Value**

The views for the receiver.

**Discussion**Instances of `NSPopUpButton` are treated specially by `NSPredicateEditor`; their menu items are merged into a single popup button, and matching menu item titles are combined. In this way, a single tree is built from the separate templates.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPredicateEditorRowTemplate.h

# Document Revision History

---

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

Date	Notes
2007-01-12	New document that describes the class that specifies, for a predicate editor view, predicates and how to display them.

## REVISION HISTORY

### Document Revision History



# Index

---

## C

---

compoundTypes [instance method 8](#)

## D

---

displayableSubpredicatesOfPredicate: [instance method 8](#)

## I

---

initWithCompoundTypes: [instance method 8](#)  
initWithLeftExpressions:  
    rightExpressionAttributeType:modifier:operators:  
    options: [instance method 9](#)  
initWithLeftExpressions:rightExpressions:modifier:  
    operators:options: [instance method 10](#)

## L

---

leftExpressions [instance method 10](#)

## M

---

matchForPredicate: [instance method 11](#)  
modifier [instance method 11](#)

## O

---

operators [instance method 11](#)  
options [instance method 12](#)

## P

---

predicateWithSubpredicates: [instance method 12](#)

## R

---

rightExpressionAttributeType [instance method 12](#)  
rightExpressions [instance method 13](#)

## S

---

setPredicate: [instance method 13](#)

## T

---

templatesWithAttributeKeyPaths:  
    inEntityDescription: [class method 7](#)  
templateViews [instance method 13](#)