NSRelationshipDescription Class Reference

Cocoa > Objective-C Language

2006-07-11

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

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

NSRelationshipDescription Class Reference

Inherits from	NSPropertyDescription : NSObject
Conforms to	NSCoding (NSPropertyDescription) NSCopying (NSPropertyDescription) NSObject (NSObject)
Framework	/System/Library/Frameworks/CoreData.framework
Availability	Available in Mac OS X v10.4 and later.
Companion guide	Core Data Programming Guide
Declared in	NSRelationshipDescription.h
Related sample code	Core Data HTML Store CoreRecipes CustomAtomicStoreSubclass

Overview

The `NSRelationshipDescription` class is used to describe relationships of an entity in an `NSEntityDescription` object.

`NSRelationshipDescription` extends `NSPropertyDescription` to describe features appropriate to relationships, including cardinality (the number of objects allowed in the relationship), the destination entity, and delete rules.

Cardinality

The maximum and minimum counts for a relationship indicate the number of objects referenced (`1` for a to-one relationship, `-1` means undefined). Note that the counts are only enforced if the relationship value in the containing object is not `nil`. That is, provided that the relationship value is optional, there may be zero objects in the relationship, which might be less than the minimum count.

Editing Relationship Descriptions

Relationship descriptions are editable until they are used by an object graph manager. This allows you to create or modify them dynamically. However, once a description is used (when the managed object model to which it belongs is associated with a persistent store coordinator), it *must not* (indeed cannot) be changed.

This is enforced at runtime: any attempt to mutate a model or any of its sub-objects after the model is associated with a persistent store coordinator causes an exception to be thrown. If you need to modify a model that is in use, create a copy, modify the copy, and then discard the objects with the old model.

Tasks

Getting and Setting Type Information

`– destinationEntity` (page 7)
Returns the entity description of the receiver's destination.

`– setDestinationEntity:` (page 10)
Sets the entity description for the receiver's destination.

`– inverseRelationship` (page 7)
Returns the relationship that represents the inverse of the receiver.

`– setInverseRelationship:` (page 10)
Sets the inverse relationship of the receiver.

Getting and Setting Delete Rules

`– deleteRule` (page 7)
Returns the delete rule of the receiver.

`– setDeleteRule:` (page 9)
Sets the delete rule of the receiver.

Cardinality

`– maxCount` (page 8)
Returns the maximum count of the receiver.

`– setMaxCount:` (page 10)
Sets the maximum count of the receiver.

`– minCount` (page 9)
Returns the minimum count of the receiver.

`– setMinCount:` (page 11)
Sets the minimum count of the receiver.

`– isToMany` (page 8)
Returns a Boolean value that indicates whether the receiver represents a to-many relationship.

Versioning Support

`– versionHash` (page 11)
Returns the version hash for the receiver.

Instance Methods

deleteRule

Returns the delete rule of the receiver.

```
- (NSDeleteRule)deleteRule
```

Return Value
The receiver’s delete rule.

Availability
Available in Mac OS X v10.4 and later.

See Also
- setDeleteRule: (page 9)

Declared In
`NSRelationshipDescription.h`

destinationEntity

Returns the entity description of the receiver's destination.

```
- (NSEntityDescription *)destinationEntity
```

Return Value
The entity description for the receiver's destination.

Availability
Available in Mac OS X v10.4 and later.

See Also
- setDestinationEntity: (page 10)

Related Sample Code
Core Data HTML Store

CoreRecipes

Declared In
`NSRelationshipDescription.h`

inverseRelationship

Returns the relationship that represents the inverse of the receiver.

```
- (NSRelationshipDescription *)inverseRelationship
```

Return Value
The relationship that represents the inverse of the receiver.

Discussion

Given a to-many relationship “employees” between a Department entity and an Employee entity (a department may have many employees), and a to-one relationship “department” between an Employee entity and a Department entity (an employee may belong to only one department), the inverse of the “department” relationship is the “employees” relationship.

Availability

Available in Mac OS X v10.4 and later.

See Also

`- setInverseRelationship:` (page 10)

Related Sample Code

CoreRecipes

Declared In

`NSRelationshipDescription.h`

isToMany

Returns a Boolean value that indicates whether the receiver represents a to-many relationship.

```
- (BOOL)isToMany
```

Return Value

`YES` if the receiver represents a to-many relationship (its `maxCount` is greater than `1`) otherwise `NO`.

Availability

Available in Mac OS X v10.4 and later.

See Also

`- maxCount` (page 8)

`- setMaxCount:` (page 10)

Related Sample Code

CoreRecipes

Declared In

`NSRelationshipDescription.h`

maxCount

Returns the maximum count of the receiver.

```
- (NSUInteger)maxCount
```

Return Value

The maximum count of the receiver.

Availability

Available in Mac OS X v10.4 and later.

See Also

- `isToMany` (page 8)
- `minCount` (page 9)
- `setMaxCount:` (page 10)
- `setMinCount:` (page 11)

Declared In

`NSRelationshipDescription.h`

minCount

Returns the minimum count of the receiver.

```
- (NSUInteger)minCount
```

Return Value

The minimum count of the receiver.

Availability

Available in Mac OS X v10.4 and later.

See Also

- `maxCount` (page 8)
- `setMaxCount:` (page 10)
- `setMinCount:` (page 11)

Declared In

`NSRelationshipDescription.h`

setDeleteRule:

Sets the delete rule of the receiver.

```
- (void)setDeleteRule:(NSDeleteRule)rule
```

Parameters

rule

The delete rule for the receiver.

Special Considerations

This method raises an exception if the receiver’s model has been used by an object graph manager.

Availability

Available in Mac OS X v10.4 and later.

See Also

- `deleteRule` (page 7)

Declared In

`NSRelationshipDescription.h`

setDestinationEntity:

Sets the entity description for the receiver's destination.

```
- (void)setDestinationEntity:(NSEntityDescription *)entity
```

Parameters

entity

The destination entity for the receiver.

Special Considerations

This method raises an exception if the receiver's model has been used by an object graph manager.

Availability

Available in Mac OS X v10.4 and later.

See Also

`- destinationEntity` (page 7)

Declared In

`NSRelationshipDescription.h`

setInverseRelationship:

Sets the inverse relationship of the receiver.

```
- (void)setInverseRelationship:(NSRelationshipDescription *)relationship
```

Parameters

relationship

The inverse relationship for the receiver.

Special Considerations

This method raises an exception if the receiver's model has been used by an object graph manager.

Availability

Available in Mac OS X v10.4 and later.

See Also

`- inverseRelationship` (page 7)

Declared In

`NSRelationshipDescription.h`

setMaxCount:

Sets the maximum count of the receiver.

```
- (void)setMaxCount:(NSUInteger)maxCount
```

Parameters

maxCount

The maximum count of the receiver.

Special Considerations

This method raises an exception if the receiver's model has been used by an object graph manager.

Availability

Available in Mac OS X v10.4 and later.

See Also

- `isToMany` (page 8)
- `maxCount` (page 8)
- `minCount` (page 9)
- `setMinCount:` (page 11)

Declared In

`NSRelationshipDescription.h`

setMinCount:

Sets the minimum count of the receiver.

```
- (void)setMinCount:(NSUInteger)minCount
```

Parameters

`minCount`

The minimum count of the receiver.

Special Considerations

This method raises an exception if the receiver's model has been used by an object graph manager.

Availability

Available in Mac OS X v10.4 and later.

See Also

- `maxCount` (page 8)
- `minCount` (page 9)
- `setMaxCount:` (page 10)

Declared In

`NSRelationshipDescription.h`

versionHash

Returns the version hash for the receiver.

```
- (NSData *)versionHash
```

Return Value

The version hash for the receiver.

Discussion

The version hash is used to uniquely identify an attribute based on its configuration. This value includes the `versionHash` information from `NSPropertyDescription`, the name of the destination entity and the inverse relationship, and the min and max count.

Availability
Available in Mac OS X v10.5 and later.

See Also
`– versionHash` (`NSPropertyDescription`)

Declared In
`NSRelationshipDescription.h`

Constants

NSDeleteRule

These constants define what happens to relationships when an object is deleted.

```
typedef enum {
    NSNoActionDeleteRule,
    NSNullifyDeleteRule,
    NSCascadeDeleteRule,
    NSDenyDeleteRule
} NSDeleteRule;
```

Constants

`NSNoActionDeleteRule`

If the object is deleted, no modifications are made to objects at the destination of the relationship.

If you use this rule, you are responsible for maintaining the integrity of the object graph. This rule is strongly discouraged for all but advanced users. You should normally use `NSNullifyDeleteRule` instead.

Available in Mac OS X v10.4 and later.

Declared in `NSRelationshipDescription.h`.

`NSNullifyDeleteRule`

If the object is deleted, back pointers from the objects to which it is related are nullified.

Available in Mac OS X v10.4 and later.

Declared in `NSRelationshipDescription.h`.

`NSCascadeDeleteRule`

If the object is deleted, the destination object or objects of this relationship are also deleted.

Available in Mac OS X v10.4 and later.

Declared in `NSRelationshipDescription.h`.

`NSDenyDeleteRule`

If the destination of this relationship is not `nil`, the delete creates a validation error.

Available in Mac OS X v10.4 and later.

Declared in `NSRelationshipDescription.h`.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`NSRelationshipDescription.h`

Document Revision History

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

Date	Notes
2006-07-11	Updated for Mac OS X v10.5.
2006-05-23	First publication of this content as a separate document.

Index