NSPersistentStore Class Reference

Data Management

2009-05-01

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

Simultaneously published in the United States and Canada.

Contents

NSPersistentStore Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/CoreData.framework
Availability	Available in iOS 3.0 and later.
Declared in	NSPersistentStore.h
Companion guides	Core Data Programming Guide Atomic Store Programming Topics

Overview

This class is the abstract base class for all Core Data persistent stores.

Core Data provides four store types—SQLite, Binary, XML, and In-Memory (the XML store is not available on iOS); these are described in Persistent Stores. Core Data also provides a subclass of `NSPersistentStore`, `NSAtomicStore`. The Binary and XML stores are examples of atomic stores that inherit functionality from `NSAtomicStore`.

Subclassing Notes

You should not subclass `NSPersistentStore` directly. Core Data currently only supports subclassing of `NSAtomicStore`.

The designated initializer is `initWithPersistentStoreCoordinator:configurationName:URL:options:` (page 10). When you implement the initializer, you must ensure you load metadata during initialization and set it using `setMetadata:` (page 12).

You must override these methods:

- `type` (page 13)
- `metadata` (page 11)
- `metadataForPersistentStoreWithURL:error:` (page 7)
- `setMetadata:forPersistentStoreWithURL:error:` (page 8)

Tasks

Initializing a Persistent Store

- `initWithPersistentStoreCoordinator:configurationName:URL:options:` (page 10)
  Returns a store initialized with the given arguments.

Working with State Information

- `type` (page 13)
  Returns the type string of the receiver.
- `persistentStoreCoordinator` (page 12)
  Returns the persistent store coordinator which loaded the receiver.
- `configurationName` (page 8)
  Returns the name of the managed object model configuration used to create the receiver.
- `options` (page 11)
  Returns the options with which the receiver was created.
- `URL` (page 14)
  Returns the URL for the receiver.
- `setURL:` (page 13)
  Sets the URL for the receiver.
- `identifier` (page 9)
  Returns the unique identifier for the receiver.
- `setIdentifier:` (page 12)
  Sets the unique identifier for the receiver.
- `isReadOnly` (page 10)
  Returns a Boolean value that indicates whether the receiver is read-only.
- `setReadOnly:` (page 13)
  Sets whether the receiver is read-only.

Managing Metadata

+ `metadataForPersistentStoreWithURL:error:` (page 7)
  Returns the metadata from the persistent store at the given URL.
+ `setMetadata:forPersistentStoreWithURL:error:` (page 8)
  Sets the metadata for the store at a given URL.
- `metadata` (page 11)
  Returns the metadata for the receiver.
- `loadMetadata:` (page 11)
  Instructs the receiver to load its metadata.
- `setMetadata:` (page 12)
  Sets the metadata for the receiver.

Setup and Teardown

`- didAddToPersistentStoreCoordinator:` (page 9)
Invoked after the receiver has been added to the persistent store coordinator.

`- willRemoveFromPersistentStoreCoordinator:` (page 14)
Invoked before the receiver is removed from the persistent store coordinator.

Supporting Migration

`+ migrationManagerClass` (page 7)
Returns the `NSMigrationManager` class for this store class.

Class Methods

metadataForPersistentStoreWithURL:error:

Returns the metadata from the persistent store at the given URL.

```
+ (NSDictionary *)metadataForPersistentStoreWithURL:(NSURL *)url error:(NSError
    **)error
```

Parameters

url
The location of the store.

error
If an error occurs, upon return contains an `NSError` object that describes the problem.

Return Value
The metadata from the persistent store at *url*. Returns `nil` if there is an error.

Special Considerations

Subclasses must override this method.

Availability
Available in iOS 3.0 and later.

Declared In
`NSPersistentStore.h`

migrationManagerClass

Returns the `NSMigrationManager` class for this store class.

```
+ (Class)migrationManagerClass
```

Return Value
The `NSMigrationManager` class for this store class

Discussion
In a subclass of `NSPersistentStore`, you can override this to provide a custom migration manager subclass (for example, to take advantage of store-specific functionality to improve migration performance).

Availability
Available in iOS 3.0 and later.

Declared In
`NSPersistentStore.h`

setMetadata:forPersistentStoreWithURL:error:

Sets the metadata for the store at a given URL.

```
+ (BOOL)setMetadata:(NSDictionary *)metadata forPersistentStoreWithURL:(NSURL *)url
    error:(NSError **)error
```

Parameters

metadata
The metadata for the store at *url*.

url
The location of the store.

error
If an error occurs, upon return contains an `NSError` object that describes the problem.

Return Value
`YES` if the metadata was written correctly, otherwise `NO`.

Special Considerations

Subclasses must override this method to set metadata appropriately.

Availability
Available in iOS 3.0 and later.

Declared In
`NSPersistentStore.h`

Instance Methods

configurationName

Returns the name of the managed object model configuration used to create the receiver.

```
- (NSString *)configurationName
```

Return Value
The name of the managed object model configuration used to create the receiver.

Availability
Available in iOS 3.0 and later.

Declared In
NSPersistentStore.h

didAddToPersistentStoreCoordinator:

Invoked after the receiver has been added to the persistent store coordinator.

```
- (void)didAddToPersistentStoreCoordinator:(NSPersistentStoreCoordinator
    *)coordinator
```

Parameters

coordinator
The persistent store coordinator to which the receiver was added.

Discussion
The default implementation does nothing. You can override this method in a subclass in order to perform any kind of setup necessary before the load method is invoked.

Availability
Available in iOS 3.0 and later.

Declared In
NSPersistentStore.h

identifier

Returns the unique identifier for the receiver.

```
- (NSString *)identifier
```

Return Value
The unique identifier for the receiver.

Discussion
The identifier is used as part of the managed object IDs for each object in the store.

Special Considerations

`NSPersistentStore` provides a default implementation to provide a globally unique identifier for the store instance.

Availability
Available in iOS 3.0 and later.

See Also
- `setIdentifier:` (page 12)
- `setMetadata:` (page 12)

Declared In
NSPersistentStore.h

initWithPersistentStoreCoordinator:configurationName:URL:options:

Returns a store initialized with the given arguments.

```
- (id)initWithPersistentStoreCoordinator:(NSPersistentStoreCoordinator *)root
    configurationName:(NSString *)name URL:(NSURL *)url options:(NSDictionary
    *)options
```

Parameters

coordinator

A persistent store coordinator.

configurationName

The name of the managed object model configuration to use. Pass `nil` if you do not want to specify a configuration.

url

The URL of the store to load.

options

A dictionary containing configuration options.

Return Value

A new store object, associated with *coordinator*, that represents a persistent store at url using the options in *options* and—if it is not `nil`—the managed object model configuration *configurationName*.

Discussion

You must ensure that you load metadata during initialization and set it using `setMetadata:` (page 12).

Special Considerations

This is the designated initializer for persistent stores.

Availability

Available in iOS 3.0 and later.

See Also

`- setMetadata:` (page 12)

Declared In

`NSPersistentStore.h`

isReadOnly

Returns a Boolean value that indicates whether the receiver is read-only.

```
- (BOOL)isReadOnly
```

Return Value

`YES` if the receiver is read-only, otherwise `NO`.

Availability

Available in iOS 3.0 and later.

Declared In

`NSPersistentStore.h`

loadMetadata:

Instructs the receiver to load its metadata.

```
- (BOOL)loadMetadata:(NSError **)error
```

Parameters

error

If an error occurs, upon return contains an `NSError` object that describes the problem.

Return Value

`YES` if the metadata was loaded correctly, otherwise `NO`.

Special Considerations

There is no way to return an error if the store is invalid.

Availability

Available in iOS 3.0 and later.

Declared In

`NSPersistentStore.h`

metadata

Returns the metadata for the receiver.

```
- (NSDictionary *)metadata
```

Return Value

The metadata for the receiver. The dictionary must include the store type (`NSStoreTypeKey`) and UUID (`NSStoreUUIDKey`).

Special Considerations

Subclasses must override this method to provide storage and persistence for the store metadata.

Availability

Available in iOS 3.0 and later.

Declared In

`NSPersistentStore.h`

options

Returns the options with which the receiver was created.

```
- (NSDictionary *)options
```

Return Value

The options with which the receiver was created.

Discussion

See `NSPersistentStoreCoordinator` for a list of key names for options in this dictionary.

Availability
Available in iOS 3.0 and later.

Declared In
NSPersistentStore.h

persistentStoreCoordinator

Returns the persistent store coordinator which loaded the receiver.

```
- (NSPersistentStoreCoordinator *)persistentStoreCoordinator
```

Return Value
The persistent store coordinator which loaded the receiver.

Availability
Available in iOS 3.0 and later.

Declared In
NSPersistentStore.h

setIdentifier:

Sets the unique identifier for the receiver.

```
- (void)setIdentifier:(NSString *)identifier
```

Parameters
identifier
The unique identifier for the receiver.

Availability
Available in iOS 3.0 and later.

See Also
- identifier (page 9)
- metadata (page 11)

Declared In
NSPersistentStore.h

setMetadata:

Sets the metadata for the receiver.

```
- (void)setMetadata:(NSDictionary *)storeMetadata
```

Parameters
storeMetadata
The metadata for the receiver.

Availability
Available in iOS 3.0 and later.

Declared In
NSPersistentStore.h

setReadOnly:

Sets whether the receiver is read-only.

```
- (void)setReadOnly:(BOOL)flag
```

Parameters

flag

YES if the receiver is read-only, otherwise NO.

Availability
Available in iOS 3.0 and later.

Declared In
NSPersistentStore.h

setURL:

Sets the URL for the receiver.

```
- (void)setURL:(NSURL *)url
```

Parameters

url

The URL for the receiver.

Discussion
To alter the location of a store, send the persistent store coordinator a `setURL:forPersistentStore:` message.

Availability
Available in iOS 3.0 and later.

See Also
- URL (page 14)

Declared In
NSPersistentStore.h

type

Returns the type string of the receiver.

```
- (NSString *)type
```

Return Value
The type string of the receiver.

Discussion
This string is used when specifying the type of store to add to a persistent store coordinator.

Special Considerations

Subclasses must override this method to provide a unique type.

Availability
Available in iOS 3.0 and later.

Declared In
NSPersistentStore.h

URL

Returns the URL for the receiver.

```
- (NSURL *)URL
```

Return Value
The URL for the receiver.

Availability
Available in iOS 3.0 and later.

See Also
- setURL: (page 13)

Declared In
NSPersistentStore.h

willRemoveFromPersistentStoreCoordinator:

Invoked before the receiver is removed from the persistent store coordinator.

```
- (void)willRemoveFromPersistentStoreCoordinator:(NSPersistentStoreCoordinator
    *)coordinator
```

Parameters

coordinator
The persistent store coordinator from which the receiver was removed.

Discussion
The default implementation does nothing. You can override this method in a subclass in order to perform any clean-up before the store is removed from the coordinator (and deallocated).

Availability
Available in iOS 3.0 and later.

Declared In
NSPersistentStore.h

Document Revision History

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

Date	Notes
2009-05-01	Corrected typographical errors.
2009-02-03	Updated for iOS 3.0.
2008-06-02	Updated for Mac OS X v10.6.
2007-07-22	New document that describes the abstract Core Data class that represents a persistent store.

Document Revision History