



Introducción a Core Data

Pablo Ezequiel Romero (Nextive)

Que vamos a ver hoy

- Mecanismos de persistencia
- Core Data Stack
- Operaciones básicas
 - Inicialización, ABMs, consultas
- Data Model
- Subclases fuertemente tipeados
- Relaciones / Fetched properties
- Temas para seguir leyendo
- Preguntas

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Mecanismos de persistencia

- Property List / NSUserDefaults
 - Arrays, dictionaries, strings, dates, numbers
 - Facil de usar
 - XML o datos binarios
 - Poco volumen de datos
- SQLite
 - Es mas complejo
 - Maneja mayor volumen de datos

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



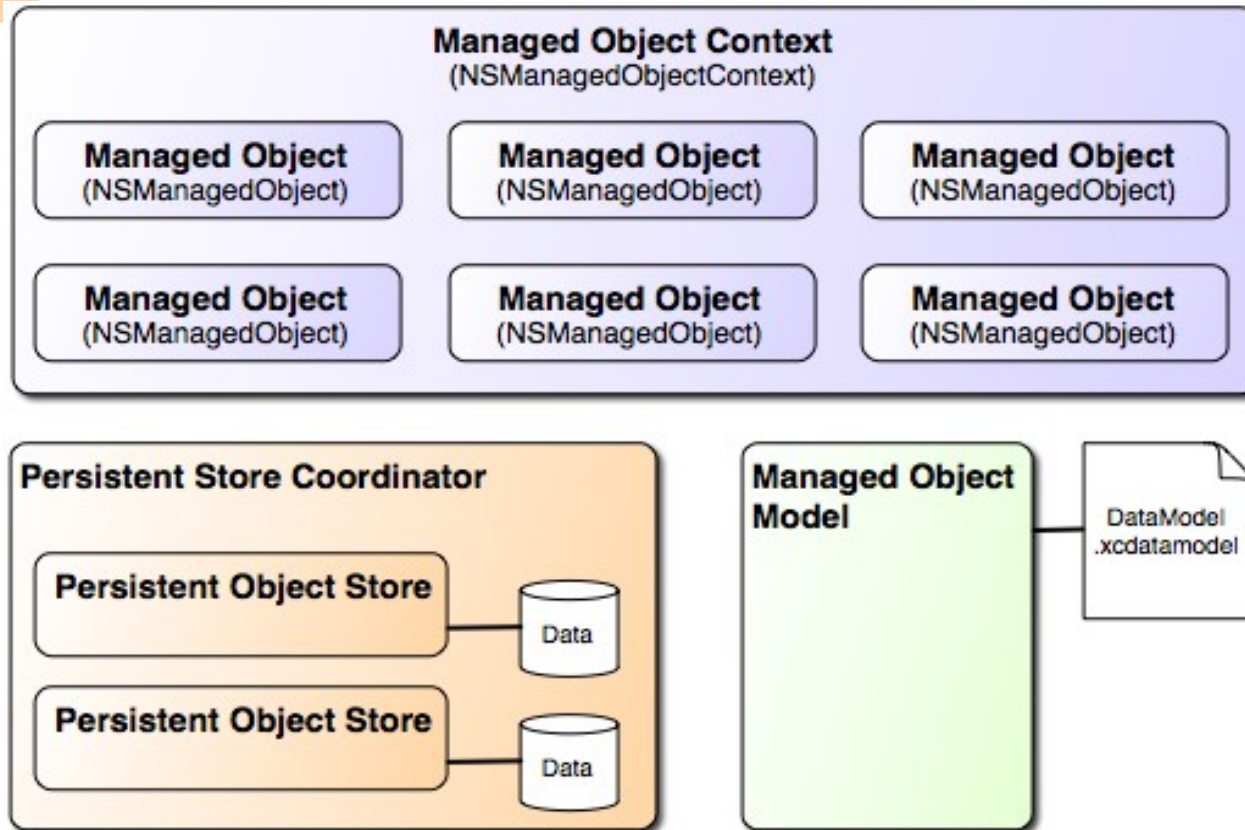
Core Data

- Core Data
 - Basicamente es un ORM
 - Alto nivel de abstraccion
 - Por lo general emplea SQLite
 - iPhone/iPod 3.0 (y iPad)

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Core Data Stack



EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Core Data Stack

Managed Object

Representa una instancia de una entidad

Managed Object Context

Contexto en donde viven los Managed Objects

Managed Object Model

Representa el modelo de datos usado para hacer el mapeo

Persistent Store Coordinator

Coleccion de Persistent Objects

Persistent Object Store

Representa un medio de almacenamiento

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Inicialización

1. Agregamos el framework **CoreData.framework**

2. Creamos un **Data Model**

3. Generamos las clases de datos (subclases de **NSManagedObject**)

4. Creamos un **Managed Object Model** con el **Data Model** que creamos

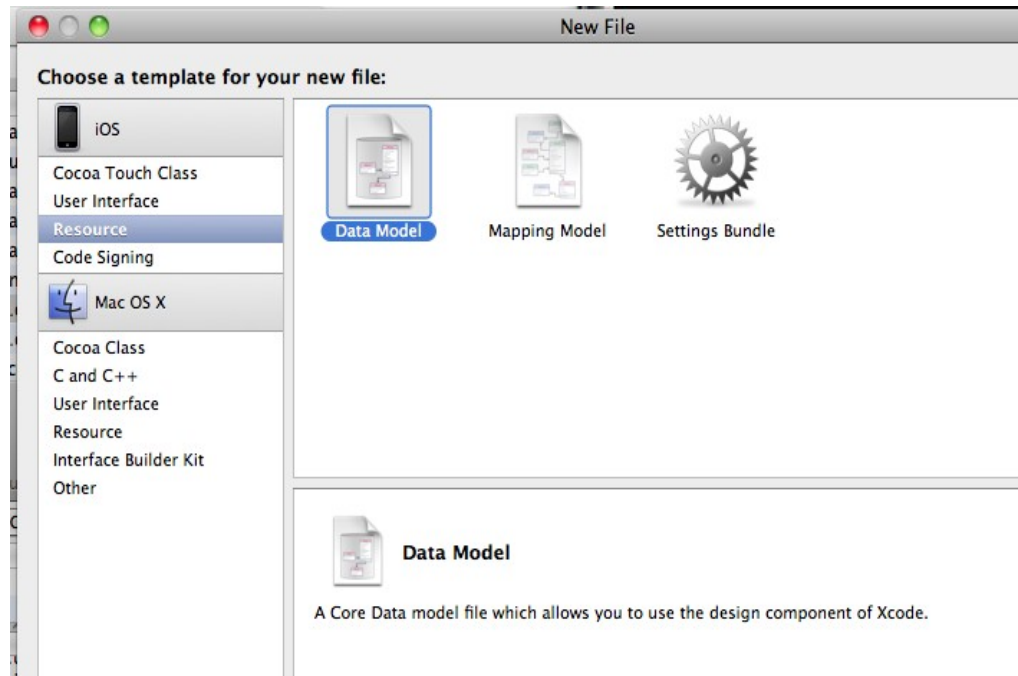
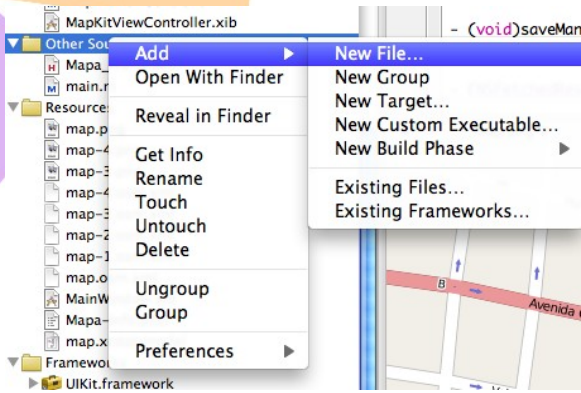
5. Creamos el **Persistent Store Coordinator** con el **Managed Object Model** que creamos y le agregamos un **Persisten Object Store**

6. Creamos un **Managed Object Context** y le seteamos el **Persistent Store Coordinator**

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



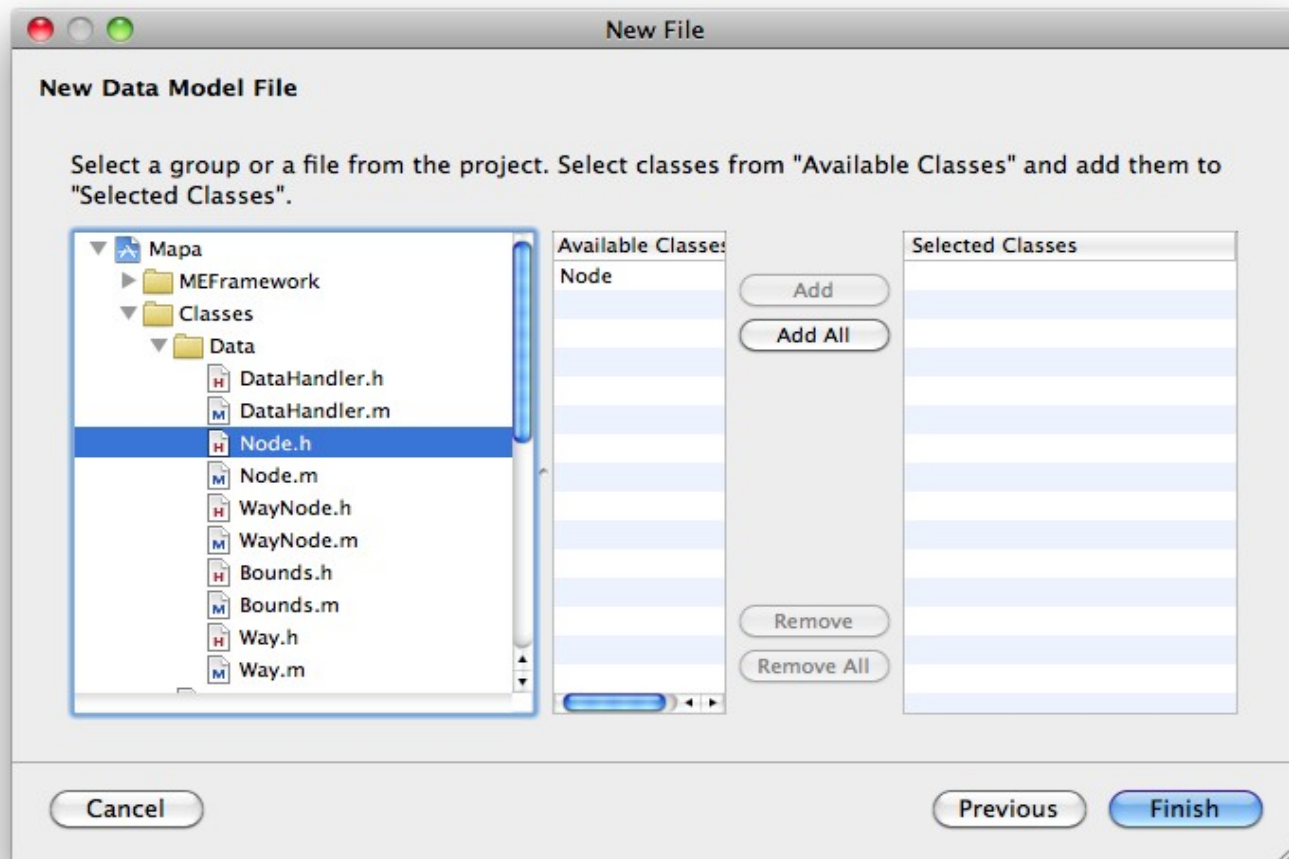
2. Creamos un Data Model



EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



2. Creamos un Data Model



EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



2. Creamos un Data Model

The screenshot shows the Xcode IDE with a data model named 'data.xccdatamodel'. The interface includes a menu bar, a toolbar with 'Breakpoints', 'Build and Debug', and 'Tasks', and a 'Project' panel. The main area is divided into a 'Table' view and a 'Diagram' view.

Table View:

Entity	Property	Kind	Type or Destination
Equipo	nombre	Attribute	String
Equipo	partidosLocal	Relationship	Partido
Equipo	partidosVisitante	Relationship	Partido
Partido	equipoLocal	Relationship	Equipo
Partido	equipoVisitante	Relationship	Equipo
Partido	fecha	Attribute	Date
Partido	golesEquipoLocal	Attribute	Int 16
Partido	golesEquipoVisitante	Attribute	Int 16

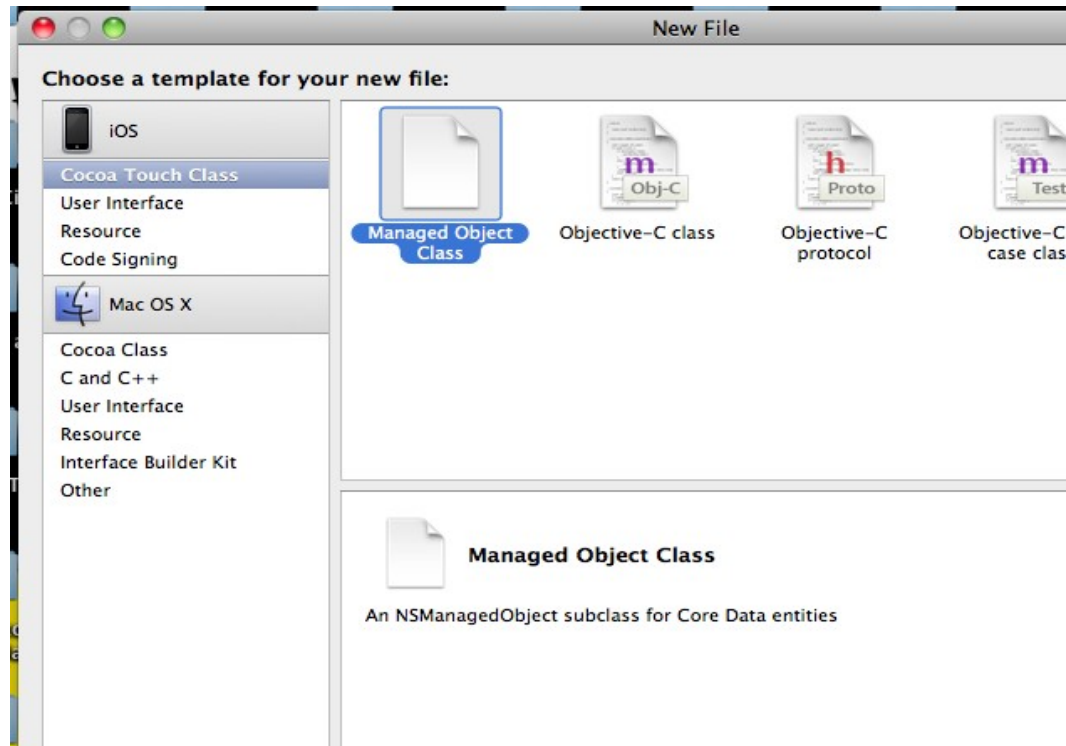
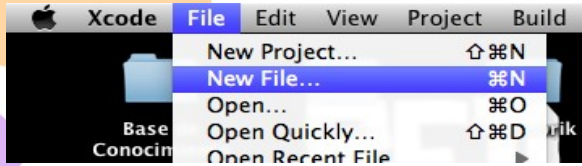
Diagram View:

The diagram shows two entities: 'Equipo' and 'Partido'. 'Equipo' has attributes 'nombre' and relationships 'partidosLocal' and 'partidosVisitante'. 'Partido' has attributes 'fecha', 'golesEquipoLocal', and 'golesEquipoVisitante', and relationships 'equipoLocal' and 'equipoVisitante'. Bidirectional arrows connect 'partidosLocal' to 'equipoLocal' and 'partidosVisitante' to 'equipoVisitante'.

Attribute Panel:

Name: nombre
 Optional Transient Indexed
Type: String
Min Length: Max Length:
Reg. Ex:
Default Value:

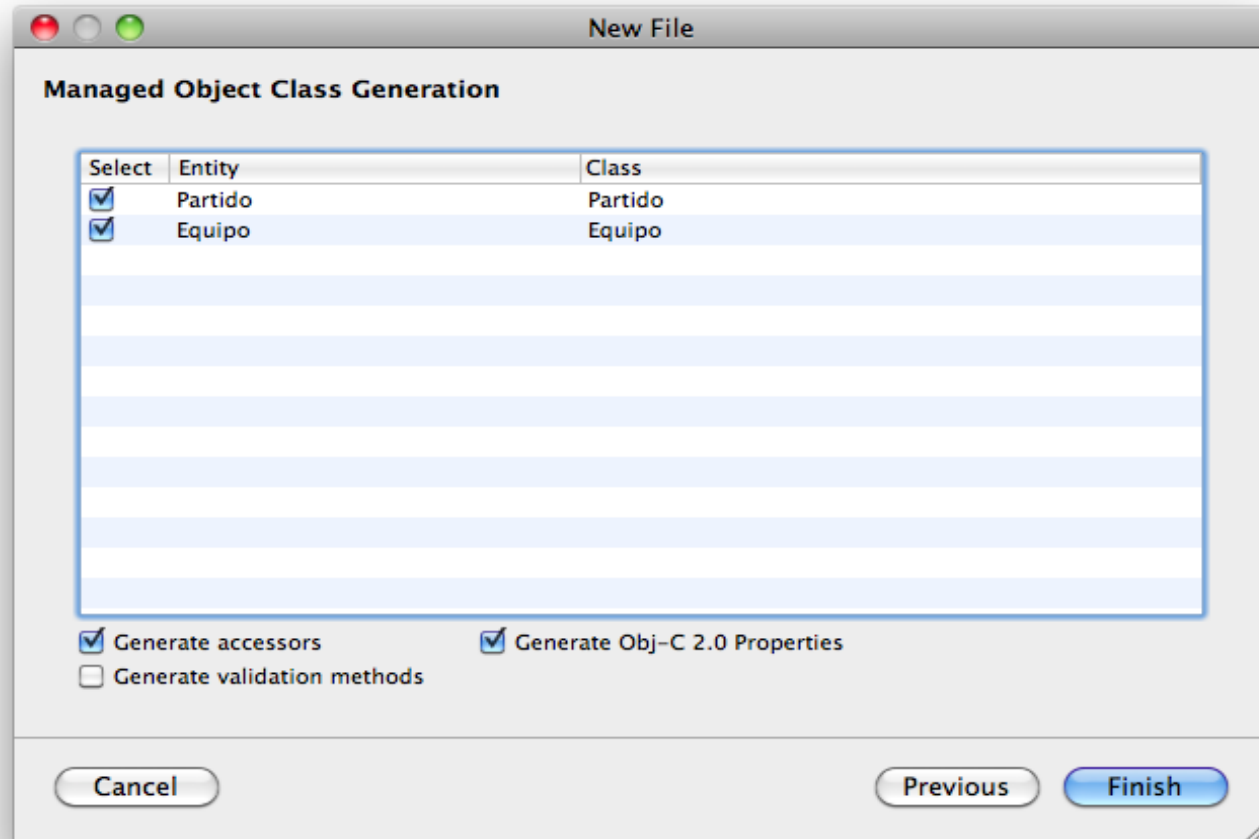
3. Generamos las clases de datos



EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



3. Generamos las clases de datos



3. Generamos las clases de datos

```
// Partido.h

#import <CoreData/CoreData.h>

@class Equipo;

@interface Partido : NSObject
{
}

@property (nonatomic, retain) NSDate * fecha;
@property (nonatomic, retain) NSNumber * golesEquipoVisitante;
@property (nonatomic, retain) NSNumber * golesEquipoLocal;
@property (nonatomic, retain) Equipo * equipoVisitante;
@property (nonatomic, retain) Equipo * equipoLocal;

@end
```

```
// Partido.m

#import "Partido.h"

#import "Equipo.h"

@implementation Partido

@dynamic golesEquipoVisitante;
@dynamic fecha;
@dynamic golesEquipoLocal;
@dynamic equipoVisitante;
@dynamic equipoLocal;

@end
```

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



3. Generamos las clases de datos

```
// Equipo.h

#import <CoreData/CoreData.h>

@class Partido;

@interface Equipo : NSObject
{
}

@property (nonatomic, retain) NSString * nombre;
@property (nonatomic, retain) NSSet* partidosLocal;
@property (nonatomic, retain) NSSet* partidosVisitante;

@end

@interface Equipo (CoreDataGeneratedAccessors)
- (void)addPartidosLocalObject:(Partido *)value;
- (void)removePartidosLocalObject:(Partido *)value;
- (void)addPartidosLocal:(NSSet *)value;
- (void)removePartidosLocal:(NSSet *)value;

- (void)addPartidosVisitanteObject:(Partido *)value;
- (void)removePartidosVisitanteObject:(Partido *)value;
- (void)addPartidosVisitante:(NSSet *)value;
- (void)removePartidosVisitante:(NSSet *)value;

@end
```

```
// Equipo.m

#import "Equipo.h"
#import "Partido.h"

@implementation Equipo

@dynamic nombre;
@dynamic partidosLocal;
@dynamic partidosVisitante;

@end
```

Inicialización

1. Agregamos el framework **CoreData.framework**
2. Creamos un **Data Model**
3. Generamos las clases de datos (subclases de **NSManagedObject**)
4. Creamos un **Managed Object Model** con el **Data Model** que creamos
5. Creamos el **Persistent Store Coordinator** con el **Managed Object Model** que creamos y le agregamos un **Persisten Object Store**
6. Creamos un **Managed Object Context** y le seteamos el **Persistent Store Coordinator**

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Inicialización (pasos 4, 5 y 6)

```
// 4. Creamos el Managed Object Model
NSManagedObjectModel *managedObjectModel =
[NSManagedObjectModel mergedModelFromBundles:nil];

// 5. Creamos el Persistent Store Coordinator y seteamos un Persistent Object Store
NSError *error = nil;

NSPersistentStoreCoordinator *persistentStoreCoordinator =
[[NSPersistentStoreCoordinator alloc] initWithManagedObjectModel:managedObjectModel];

if (![persistentStoreCoordinator addPersistentStoreWithType:NSSQLiteStoreType
                                configuration:nil
                                URL:storeURL
                                options:nil
                                error:&error]) {
    NSLog(@"Unresolved error %@, %@", error, [error userInfo]);
    abort();
}

// 6. Creamos un Managed Object Context y le seteamos el Persistent Store Coordinator
NSManagedObjectContext *managedObjectContext =
[[NSManagedObjectContext alloc] init];

[managedObjectContext setPersistentStoreCoordinator:persistentStoreCoordinator];
```

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Altas

1. Creamos una instancia de un Managed Object a través de la clase `NSEntityDescription`

2. Seteamos los valores de la instancia.

3. Salvamos el Managed Object a través del `Managed Object Context`

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Altas

```
// Creamos los equipos
Equipo *equipoLocal = (Equipo*)[NSEntityDescription insertNewObjectForEntityForName:@"Equipo"
                                inManagedObjectContext:managedObjectContext];

equipoLocal.nombre = @"Boca";

Equipo *equipoVisitante = (Equipo*)[NSEntityDescription insertNewObjectForEntityForName:@"Equipo"
                                        inManagedObjectContext:managedObjectContext];

equipoVisitante.nombre = @"River";

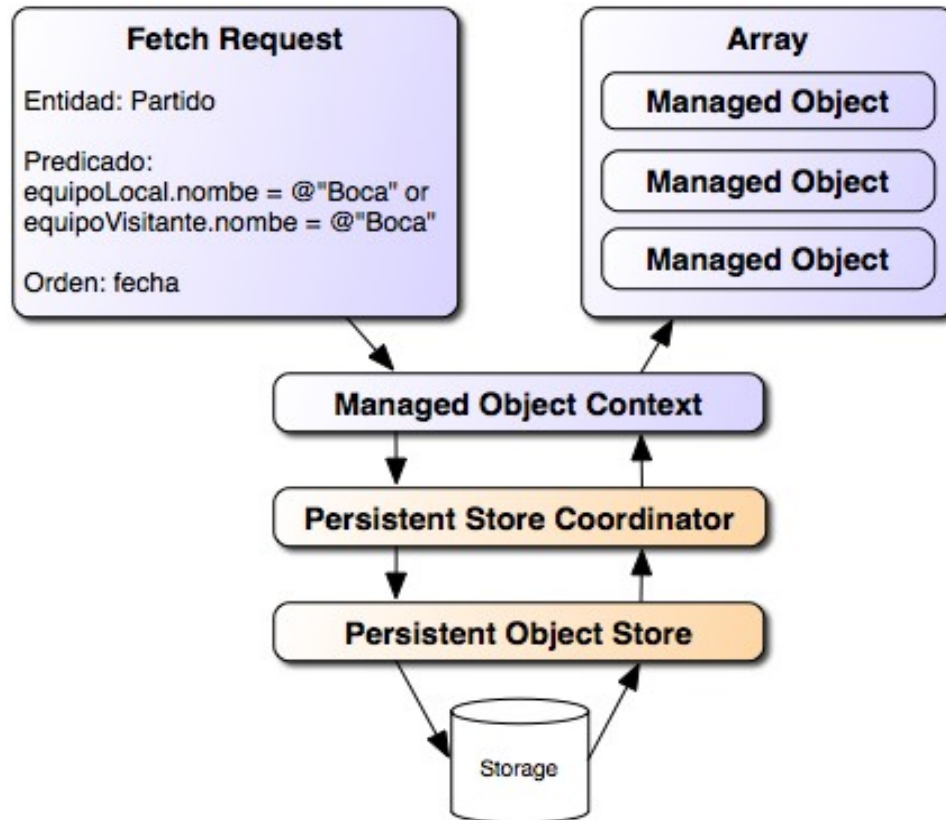
// Creamos el partido
Partido *partido = (Partido*)[NSEntityDescription insertNewObjectForEntityForName:@"Partido"
                                inManagedObjectContext:managedObjectContext];

srandom(time(NULL));

partido.golesEquipoVisitante = [NSNumber numberWithInt:random()%5];
partido.golesEquipoLocal = [NSNumber numberWithInt:random()%5];
partido.fecha = [NSDate date];
partido.equipoVisitante = equipoVisitante;
partido.equipoLocal = equipoLocal;

// Salvamos los datos a traves del managed object context
NSError *error = nil;
if (managedObjectContext != nil) {
    if (![managedObjectContext save:&error]) {
        NSLog(@"Unresolved error %@, %@", error, [error userInfo]);
        abort();
    }
}
```

Consultas



EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Consultas

```
// 1. Creamos el fetch request.
NSFetchRequest *fetchRequest = [[[NSFetchRequest alloc] init] autorelease];

// 2. Seteamos la entidad.
NSEntityDescription *entity = [NSEntityDescription entityForName:@"Partido"
                                inManagedObjectContext:managedObjectContext];
[fetchRequest setEntity:entity];

// 3. Seteamos un predicado
NSPredicate *predicate = [NSPredicate predicateWithFormat:@"equipoLocal.nombre = 'Boca' or
equipoVisitante.nombre = 'Boca'"];
[fetchRequest setPredicate:predicate];

// 4. Seteamos le orden
NSSortDescriptor *sortDescriptor = [[[NSSortDescriptor alloc] initWithKey:@"fecha"
                                ascending:YES] autorelease];
[fetchRequest setSortDescriptors:[NSArray arrayWithObject:sortDescriptor]];

// 5. Finalmente realizamos la consulta
NSError *error = nil;
NSArray *results = [self.managedObjectContext executeFetchRequest:fetchRequest error:&error];
```

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Modificaciones

1. Obtenemos una instancia de un **Managed Object** (como resultado de una búsqueda, un alta, etc)

2. Seteamos los valores de la instancia.

3. Salvamos el **Managed Object** a través del **Managed Object Context**

Modificaciones

```
// 1. Obtenemos el NSManagedObject
Partido *partido = ...;

// 2. Actualizamos las propiedades
partido.fecha = [NSDate date];

// 3. Salvamos el contexto
NSError *error = nil;
if (![managedObjectContext save:&error]) {

    NSLog(@"Unresolved error %@, %@", error, [error userInfo]);
    abort();
}
```

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Bajas

1. Obtenemos una instancia de un **Managed Object** (como resultado de una búsqueda, un alta, etc)

2. Marcamos la instancia del **Managed Object** para borrado a través del **Managed Object Context**

3. Salvamos el **Managed Object Context**

Bajas

```
// 1. Obtenemos el NSManagedObject
Partido *partido = ...;

// 2. Lo borramos del contexto
[managedObjectContext deleteObject:partido];

// 3. Salvamos el contexto
NSError *error = nil;
if (![managedObjectContext save:&error]) {

    NSLog(@"Unresolved error %@, %@", error, [error userInfo]);
    abort();
}
```

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



NSManagedObject metodos utiles

-entity	Devuelve la Entidad asociada al objeto.
-objectID	Devuelve el Managed Object ID .
-valueForKey:	Devuelve el valor correspondiente al nombre de la Propiedad .
-setValue: forKey:	Setea un valor a la Propiedad .

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



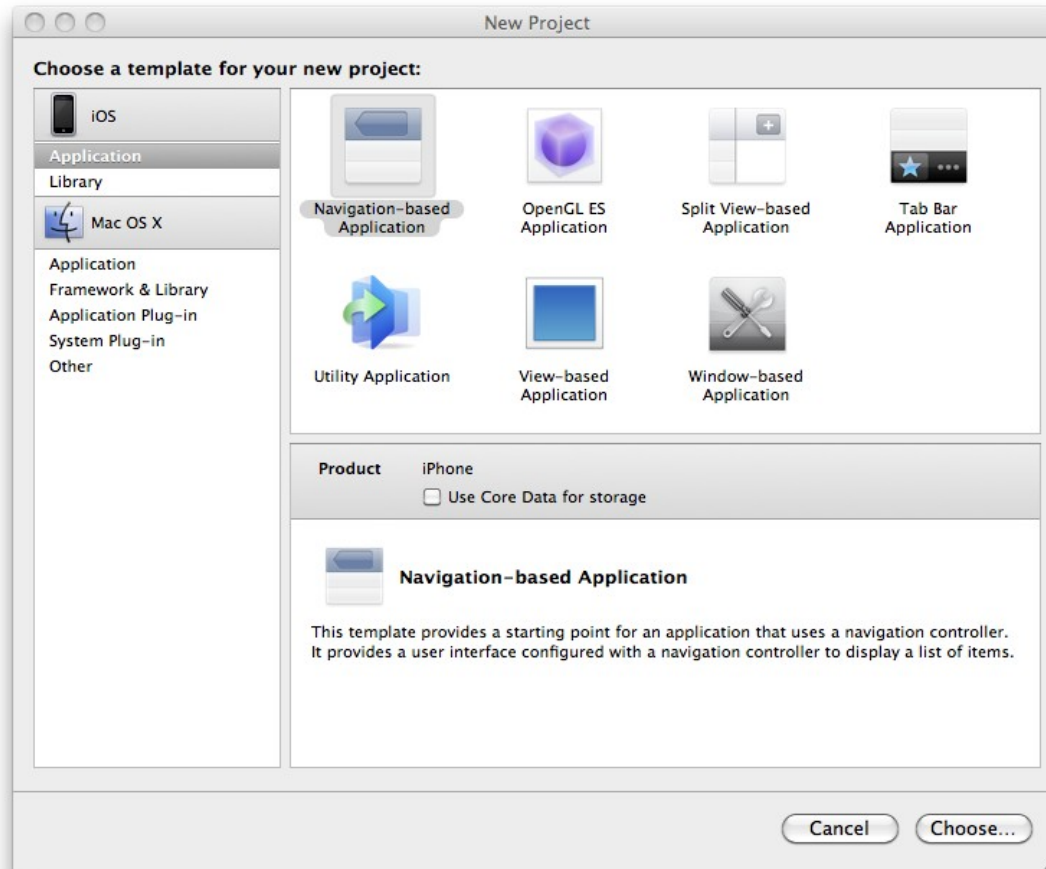
NSManagedObjectContext metodos utiles

-save:	Graba los cambios físicamente.
-objectWithID:	Devuelve un Managed Object usando el Managed Object ID como referencia.
-deleteObject:	Marca el Managed Object para borrarlo.
-undo	Vuelve atrás la ultima acción. El método -redo se encuentra disponible.
-lock	Util para múltiples threads. Los metodos -unlock y -tryLock se encuentran disponibles.
-rollback	Revierte todos los cambios.
-reset	Limpia todos los Managed Object cacheados.
-undoManager	Devuelve una instancia de NSUndoManager usada en el contexto actual.
-executeFetchRequest: error:	Corre un Fetch Request y devuelve un array de Managed Objects .

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Xcode template



EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Para seguir leyendo

- NSFetchedResultsController
- Migraciones y versionado
- Threading
- Imágenes
- Precarga de datos
- Transient Attribute (full name)
- Transformable Attributes (NSData)

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Demo



- Mismo modelo de datos (relaciones, fetched properties)
- Alta, baja, consulta
- No usa NSFetchedResultsController
- Disponible en:

<https://microedition.svn.beanstalkapp.com/public/>

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Referencias

- Core Data Tutorial for iOS:
<http://developer.apple.com/library/ios/#documentation/DataManagement/Conceptual/iPhoneCoreData01/Introduction/Introduction.html>
- Introduction to Core Data Programming Guide:
<http://developer.apple.com/library/mac/#documentation/Cocoa/Conceptual/CoreData/cdProgrammingGuide.html>
- Los primeros 7 capítulos del libro More iPhone 3 Development de Dave Mark y Jeff LaMarche
- Mi Blog: <http://www.microedition.biz/blog>
- Twitter: @microeditionbiz
- Email: pablo@nextive.com

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Referencias

- Core Data Tutorial for iOS:
<http://developer.apple.com/library/ios/#documentation/DataManagement/Conceptual/iPhoneCoreData01/Introduction/Introduction.html>
- Introduction to Core Data Programming Guide:
<http://developer.apple.com/library/mac/#documentation/Cocoa/Conceptual/CoreData/cdProgrammingGuide.html>
- Los primeros 7 capítulos del libro More iPhone 3 Development de Dave Mark y Jeff LaMarche
- Mi Blog: <http://www.microedition.biz/blog>
- Twitter: @microeditionbiz
- Email: pablo@nextive.com

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010



Preguntas

EXPOSICIÓN DE VIDEOJUEGOS ARGENTINA 2010

