CORE DATA

LEXCODE 2013

¿QUE ES CORE DATA ?

Core Data se confunde erróneamente con con una base de datos y no es así. Aunque se puede usar SQLite, esto no significa que sea un envolvente de SQLite, ni que su diseño sea para uso como base de datos. Si lo que necesitas es una base de datos quizá Core Data no sea la solución apropiada.
Si buscas también una interfaz visual creativa o como presentar los datos de forma asombrosa, busca en otro lado.

CORE DATA es una estructura unificada (framework) para guardar y recuperar datos del modelo de una aplicación.

ESTRUCTURA

Aplicación

Managed Object Context
Managed Objects

Persistent Store Coordinator

Persistent Object Store

Data File

Managed Object Model
Entity Descriptions

MANAGED OBJECT CONTEXT

- Las app basadas en core data nunca interactuan directamente con la capa de persistencia.

- El código interactua con los managedObjects contenidos en la capa del contexto.

- El contexto mantiene el estado de los objetos en relación con los datos almacenados y maneja las relaciones entre Managed Objects definidos por el modelo de datos.

- Todas las operaciones son mantenidas temporalmente hasta que el contexto graba los cambios, que es cuando se pasan a la capa de persistencia.

MANAGED OBJECTS

- Para poder insertar una fila , o devolverla se debe crear un Managed Object.
- Son instancias de la clase ManagedObject.class
- Son mantenidas y contenidas por Managed Object Context

MANAGED OBJECT MODEL

- Define el concepto de entidades.
- Las entidades definen el modelo de datos de los Managed Objects.
- Las entidades son el análogo de una tabla en las bases de datos relacionales.
- Cada entidad tiene unos atributos asociados.
- ej: Un contacto tiene nombre, apellidos, teléfono... estos serian sus atributos

MANAGED OBJECT MODEL

- **Relaciones**: Lo mismo que en el modelo relacional (1-1, 1-M, M-M).

- **Fecth Property**:Alternativa a las relaciones. Permiten acceder a la propiedad de un objeto desde otro objeto, como si fuera una relación definida entre ambas entidades.Carecen de la flexibilidad de las relaciones. Apple las ve como relaciones débiles.

- **Fetch Request**:Una consulta predefinida que puede ser referenciada para obtener data objects basados en predicados definidos. ejem: Para obtener todos los contactos con nombre pepe.

PERSISTENT STORE COORDINATOR

- Es el responsable de coordinar el acceso a múltiples almacenes de objetos persistentes.

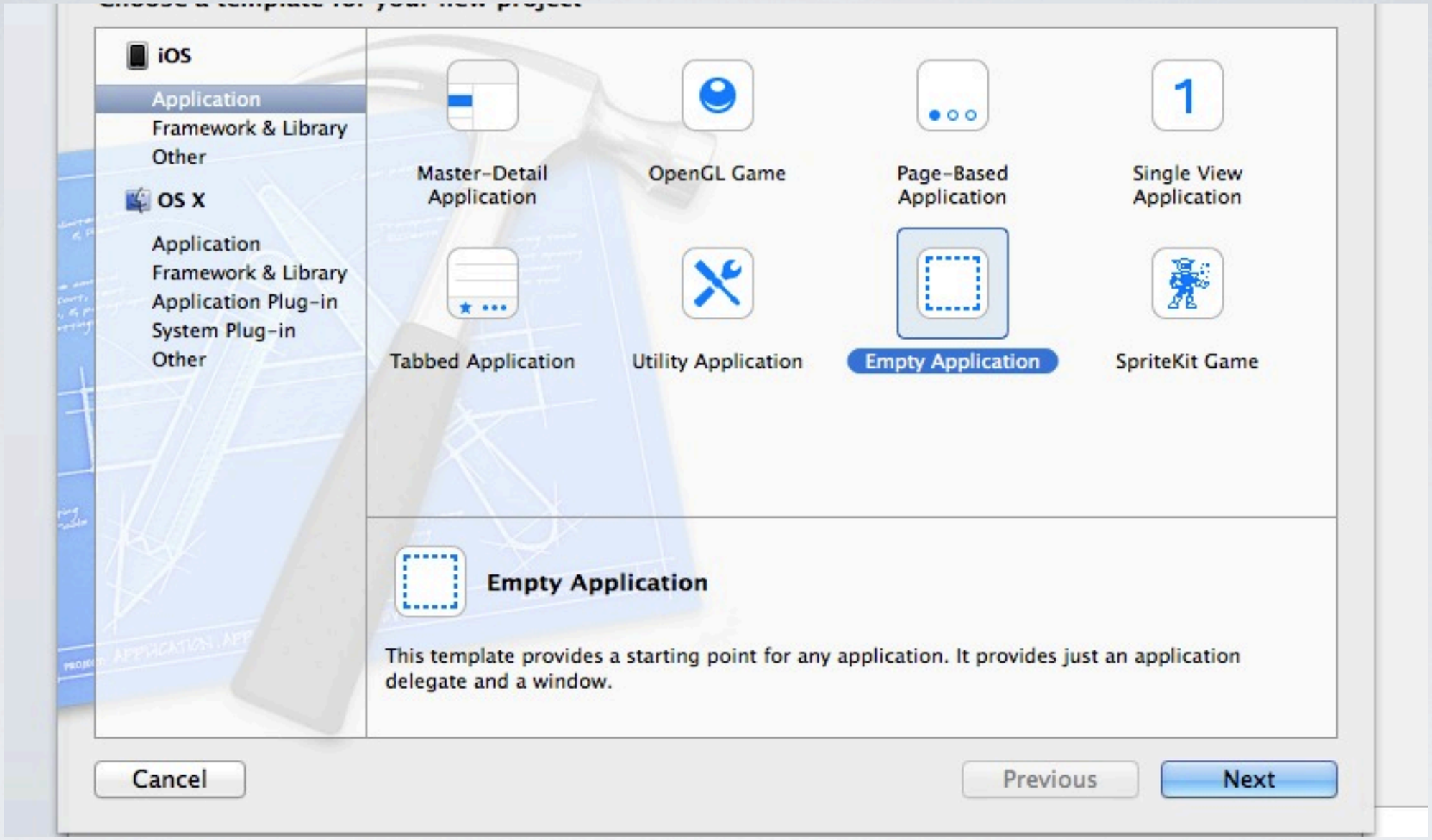
- Como desarrollador iOS no se interactua con el, de echo rara vez necesitamos mas de un almacén persistente abierto, de echo si si necesitaran se presentan a las capas superiores como uno solo.

PERSISTENT OBJECT STORE

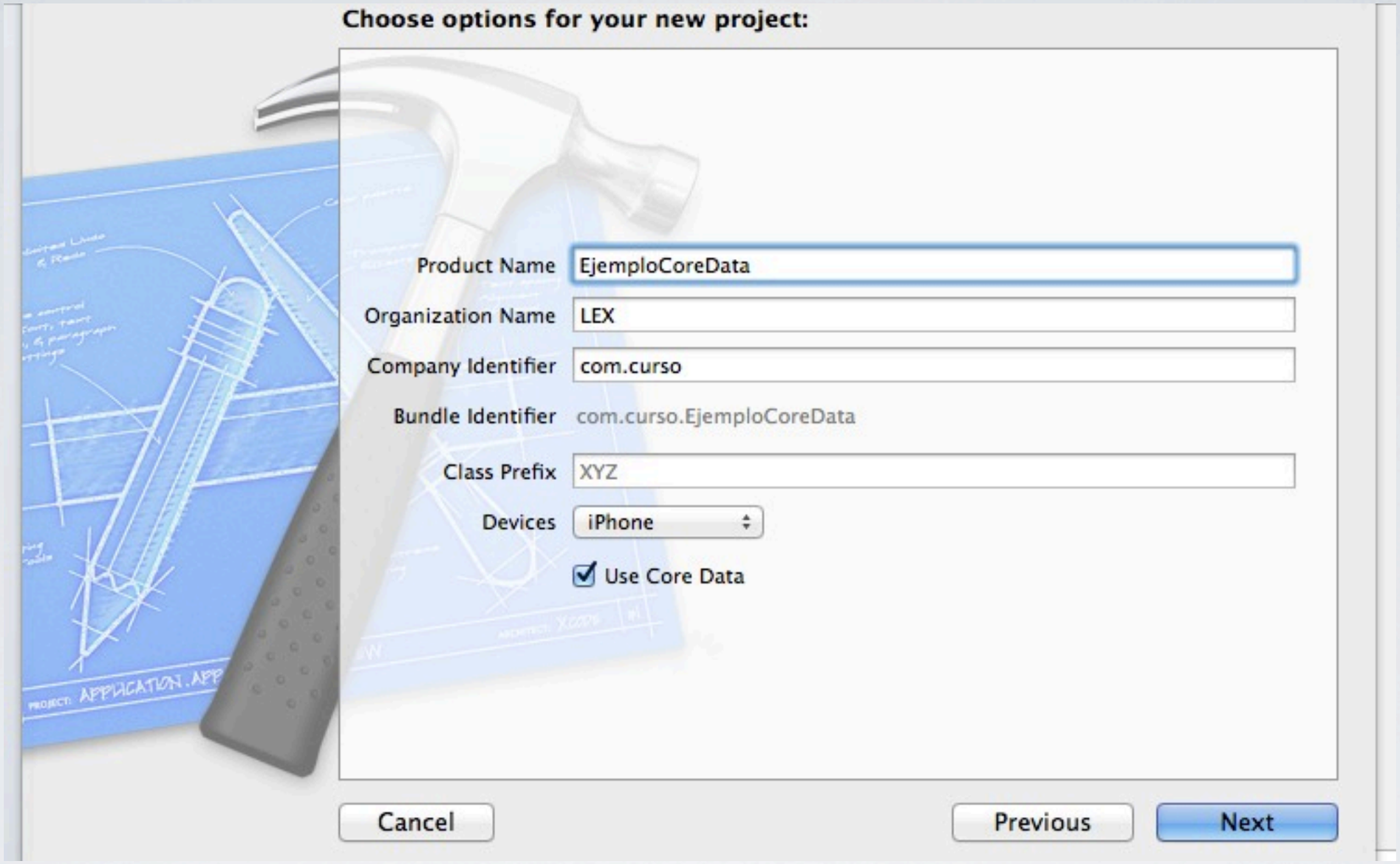
- Es la capa de almacenamiento cuando usamos CORE DATA.
- Core data soporta, sql-lite (por defecto), xml y binario.
- El tipo de almacenamiento es transparente para el desarrollador. Independientemente de la elegida se hacen las misma llamadas para obtener los mismo datos.

EJEMPLO BASICO CORE DATA

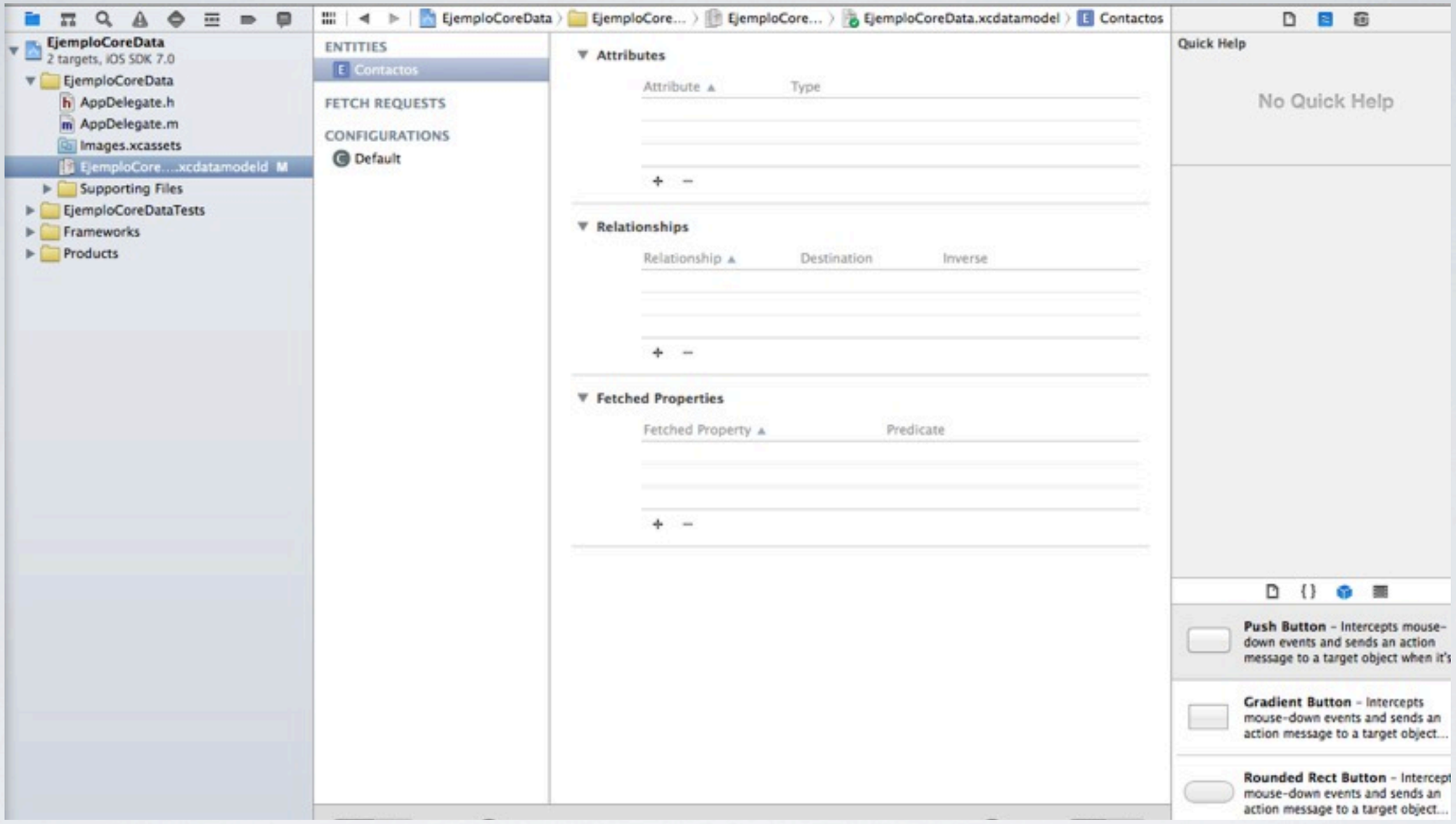
CON IOS 7



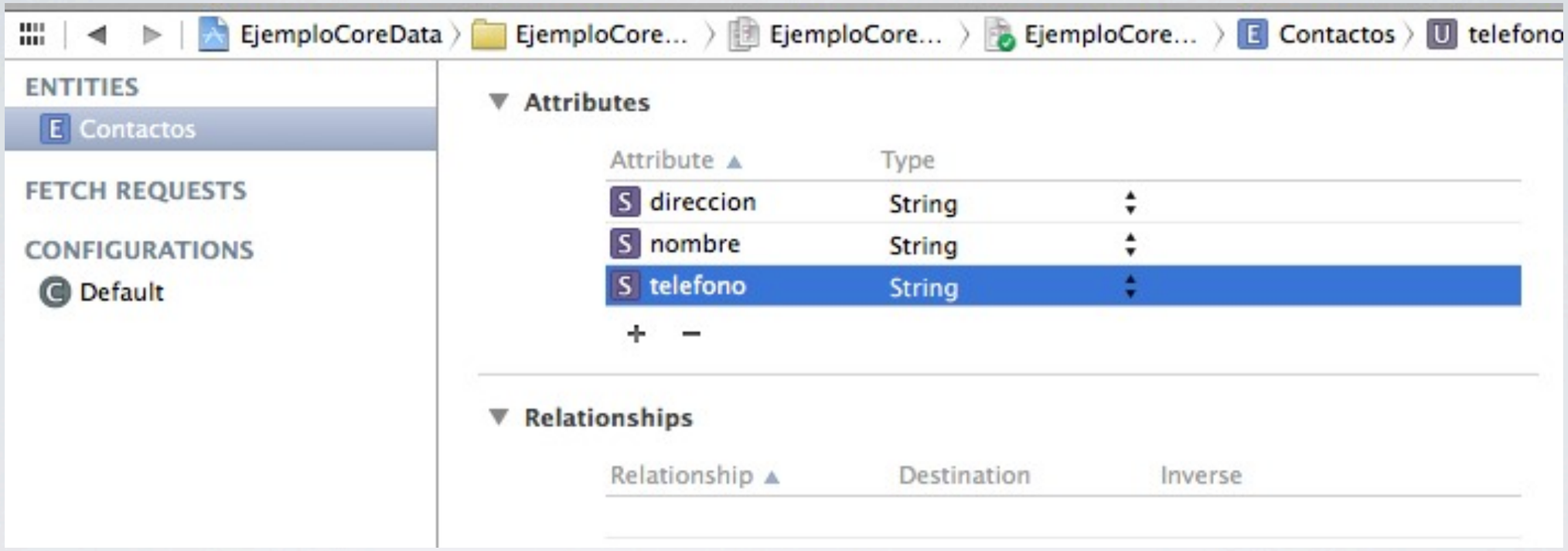
Creamos un proyecto vacío.



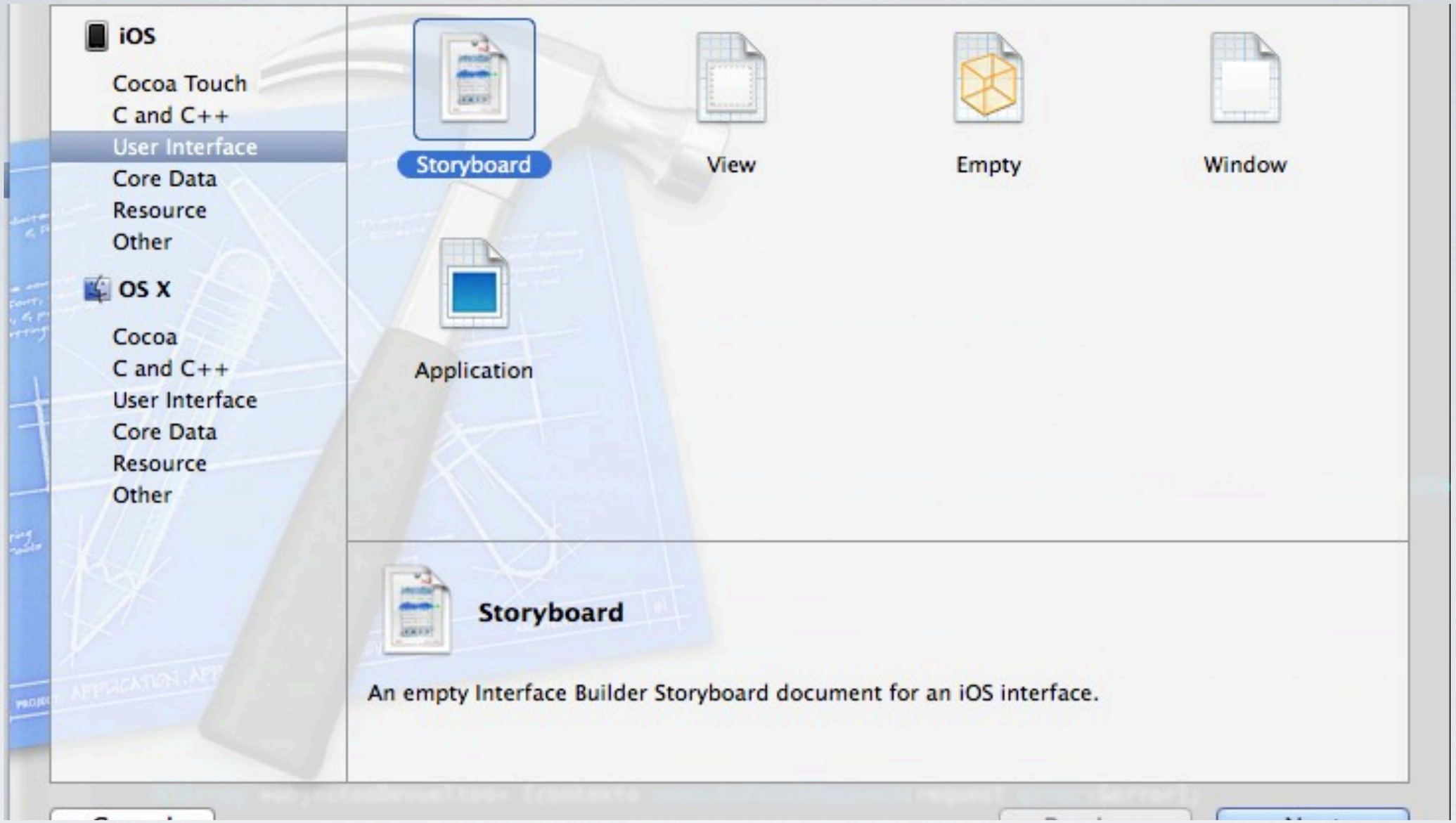
Le ponemos el nombre e importante marcar check box de Usar Core Data



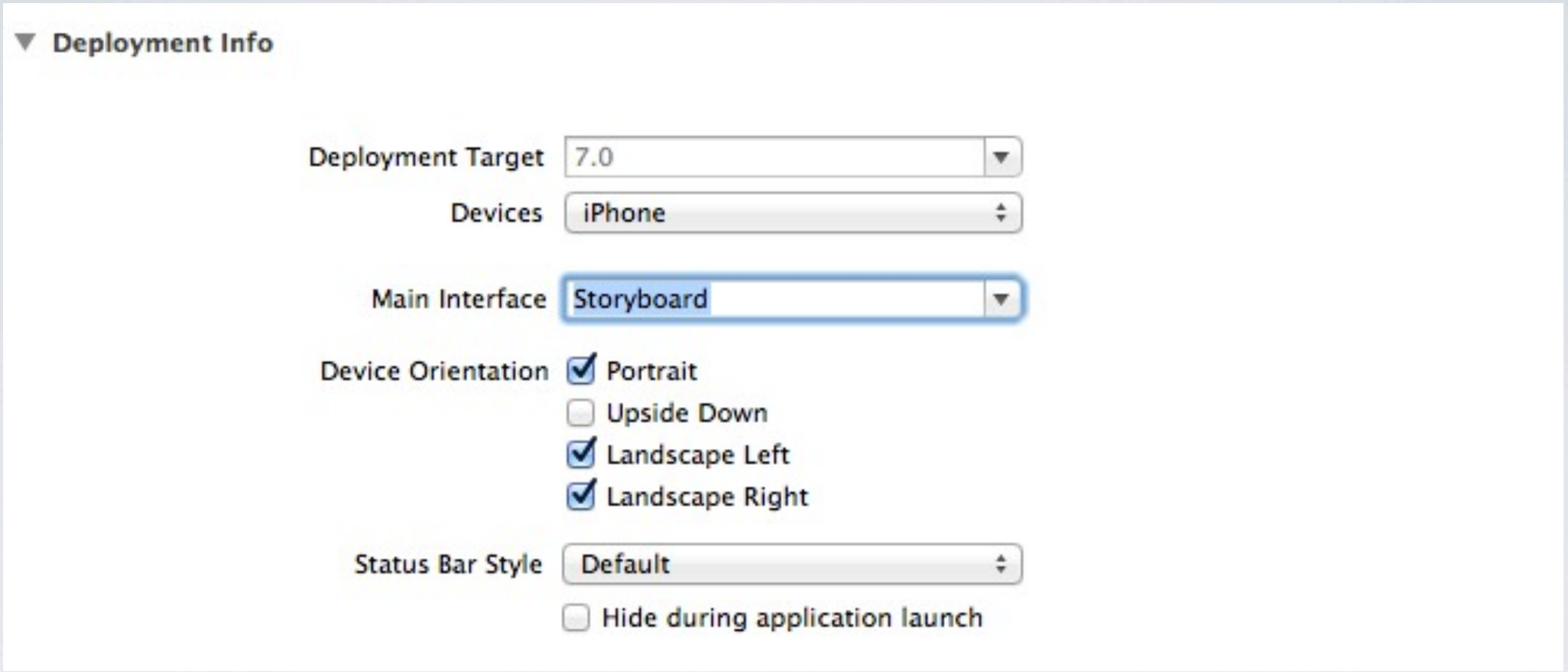
Pulsamos en el archivo .xcdatamodel y cambiamos el nombre de la entidad.



Añadimos los atributos con sus tipos correspondientes.



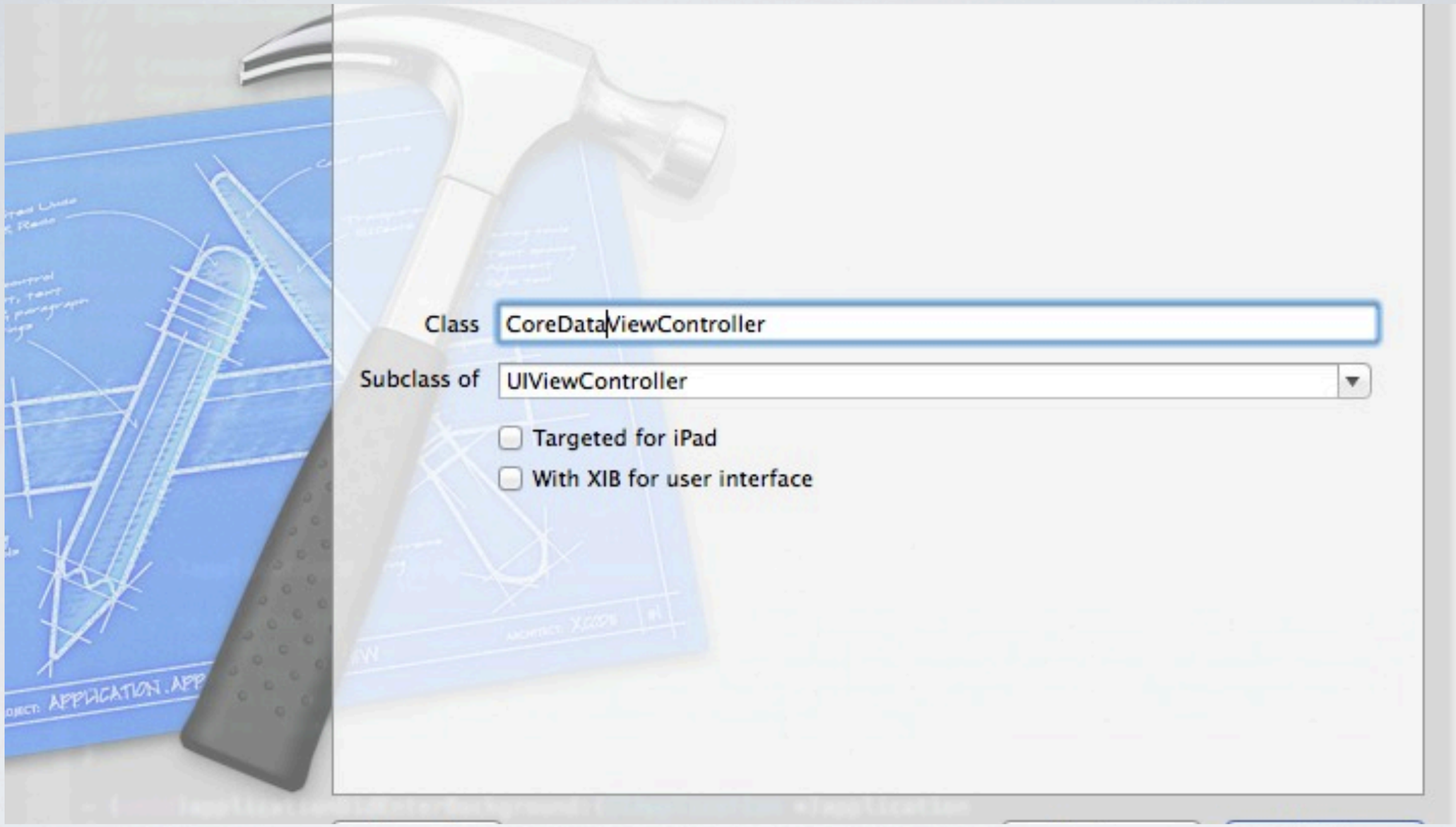
Añadimos al proyecto un storyboard.



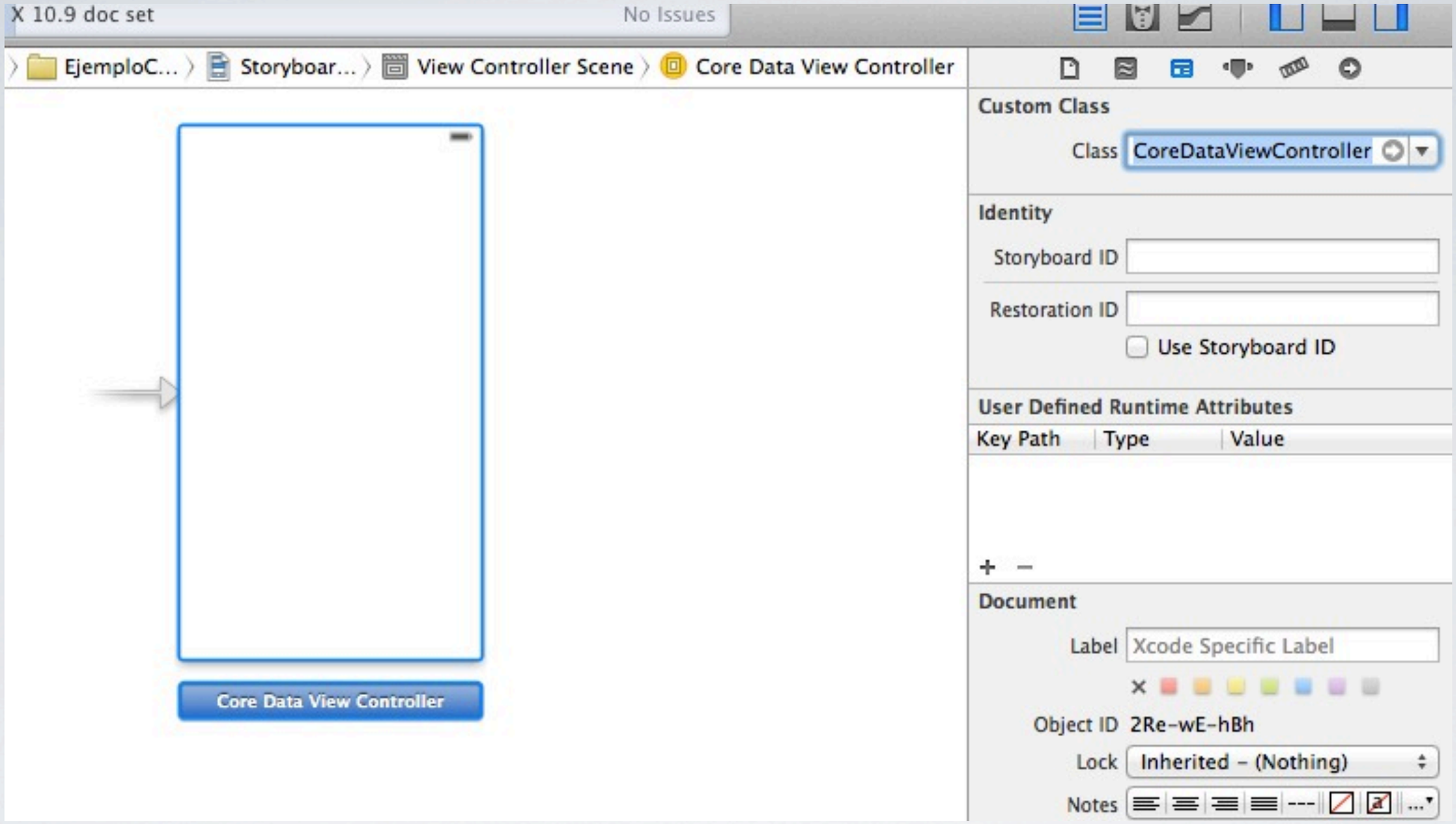
Acordarnos de especificar el storyboard en las propiedades del proyecto.

```
16
17  - (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
18  {
19
20      return YES;
21  }
22
23  - (void)applicationWillResignActive:(UIApplication *)application
```

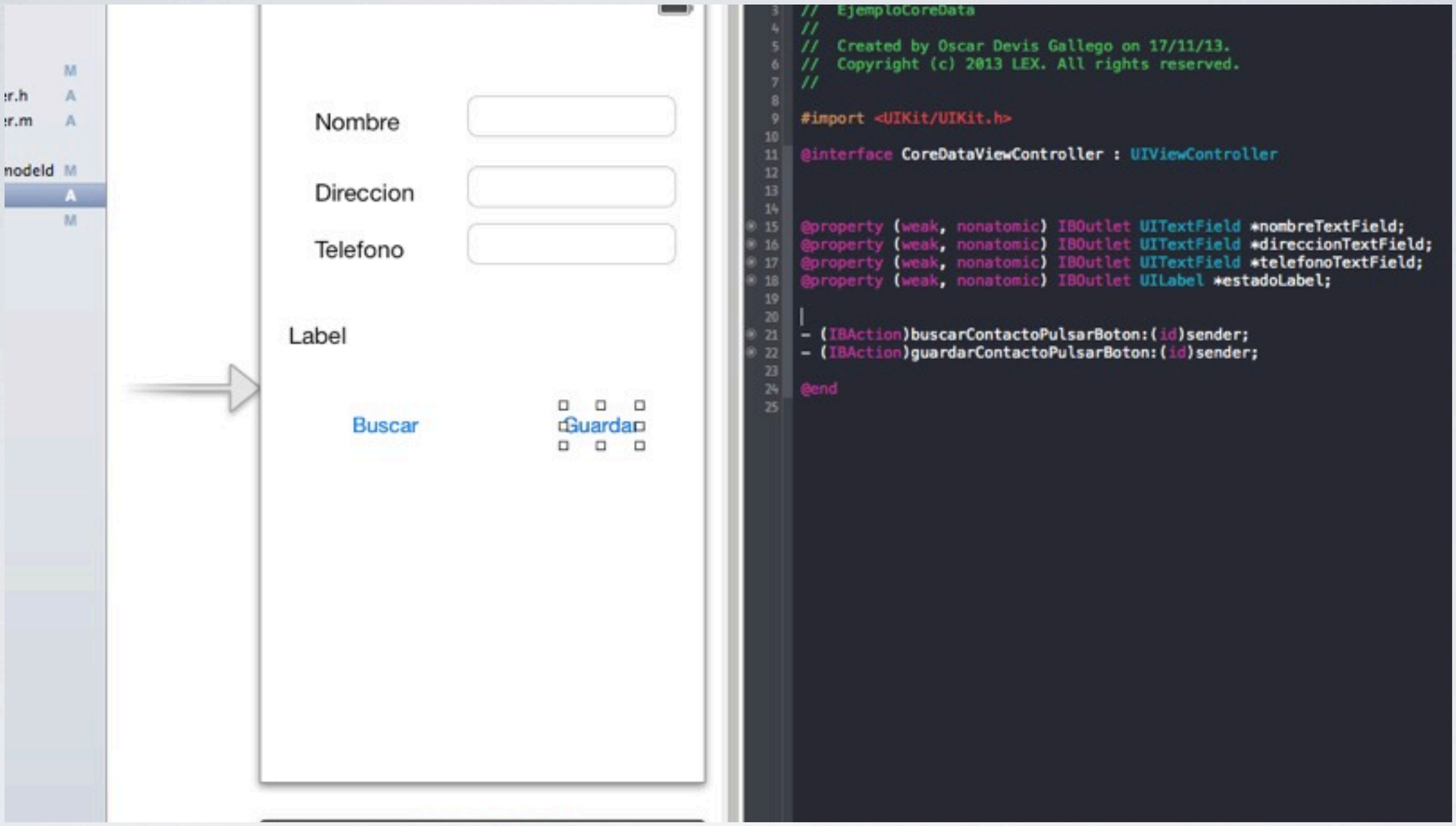
Ir a appDelegate y en el método didFinishLaunchingWithOptions borrar el código y dejar solo el return yes.Ya que al crear el proyecto vacío carga una vista en blanco



Añadimos un controlador, que herede de UIViewController, para manejar la vista.



Al añadir al storyboard el ViewController, asegurarnos que su clase sea el controlador que hemos añadido.



Creamos la vista y la enlazamos al fichero .h correspondiente las labels, los textfields y los botones.

```
IBAction)guardarContactoPulsarBoton:(id)sender {

 //Creamos el delegado
 AppDelegate * appDelegate =[[UIApplication sharedApplication]delegate];

 //Creamos el Managed Object Context
 NSManagedObjectContext *contexto = [appDelegate managedObjectContext];

 //Creamos el nuevo contacto
 NSManagedObject *nuevoContacto;
 nuevoContacto=[NSEntityDescription insertNewObjectForEntityForName:@"Contactos" inManagedObjectContext:contex

 //Le damos los valores de la vista
 [nuevoContacto setValue:self.nombreTextField.text forKey:@"nombre"];
 [nuevoContacto setValue:self.direccionTextField.text forKey:@"direccion"];
 [nuevoContacto setValue:self.telefonoTextField.text forKey:@"telefono"];

 //Limpiamos la vista
 self.nombreTextField.text=@"";
 self.direccionTextField.text=@"";
 self.telefonoTextField.text=@"";

 //Guardamos
 NSError *error;
 [contexto save:&error];

 //Informamos en la vista con la etiqueta estado
 self.estadoLabel.text=@"Contacto Guardado";
```

Completamos el código del método guardar contacto.

```
38 - (IBAction)buscarContactoPulsarBoton:(id)sender {
39
40     //Creamos el delegado
41      AppDelegate * appDelegate =[[UIApplication sharedApplication]delegate];
42
43     //Creamos el Managed Object Context
44     NSManagedObjectContext *contexto = [appDelegate managedObjectContext];
45
46     //Creamos el entity description
47     NSEntityDescription *entityDescription= [NSEntityDescription entityForName:@"Contactos" inManagedObjectContext:contexto];
48
49     //Creamos el fecth request
50     NSFetchRequest *request=[[NSFetchRequest alloc]init];
51     [request setEntity:entityDescription];
52
53     //Creamos la consulta
54     NSPredicate *consulta=[NSPredicate predicateWithFormat:@"nombre =%@",self.nombreTextField.text];
55     [request setPredicate:consulta];
56
57     //Realizamos la consulta
58     NSError *error;
59     NSArray *objectosDevueltos= [contexto executeFetchRequest:request error:&error];
60
61     //Mostramos las coincidencias si ha habido
62     NSManagedObject *resultado=nil;
63
64     if (objectosDevueltos.count==0){
65         self.estadoLabel.text=@"Valor no encontrado";
66     }
67     else{
68         resultado=objectosDevueltos[0];
69
70         self.direccionTextField.text=[resultado valueForKey:@"direccion"];
71         self.telefonoTextField.text=[resultado valueForKey:@"telefono"];
72
73         self.estadoLabel.text= [NSString stringWithFormat:@"%lu coincidencias encontradas", (unsigned long)[objectosDevueltos count]];
74
75     }
76
77 }
```

Completamos el código del método buscar contacto.

YA PODEMOS EJECUTAR LA APLICACIÓN