Introducción a core data

Core Data es un motor integrado a una aplicación ios, y tiene un diseñador muy potente para trabajar bae de datos embebidos

Puntos de revisiòn

Estos son los **puntos más importantes** en los que nos centraremos:

- Familiarizarte con el concepto de Persistencia
- ¿En qué consiste Core Data?
- Conceptos Fundamentales que debes comprender
- Crear el Modelo de Datos de tu app utilizando el Editor de Xcode
- Guardar información en Core Data
- Obtener información de Core Data
- Mostrar esa información obtenida en una Table View
- ¿Cómo integrar Core Data en una Aplicación iOS?

Echando un vistazo a nuestra App

Vamos a activar el app con core data de esta forma, podremos **guardar** y **recuperar** la lista de tareas cuando queramos.

El Concepto de Persistencia

Hemos mencionado antes que Core Data es un **sistema de persistencia**. Por tanto, lo primero que deberíamos ver, es en que consiste la **Persistencia**.
Una posible **definición** podría ser esta:
Persistencia es la acción de **preservar** la información de un objeto de forma permanente (**guardarlo**), unido a la posibilidad de poder **recuperar** la información del mismo (**leerlo**) para que pueda ser nuevamente utilizado.
Dicho de otro modo, la persistencia nos permite **guardar los datos** de nuestras aplicaciones para poder **utilizarlos**cuando el usuario **vuelva a ejecutar** nuestra app.
Si desarrollamos una aplicación que no integre **ningún sistema de persistencia**, todos los datos que vayamos utilizando durante la ejecución de la misma simplemente se **escribirán en memoria**. ¿Qué quiere decir esto? Quiere decir que al estar únicamente almacenados en memoria y no en disco, en la siguiente ejecución de la app, **todos estos datos se perderán**.
En Desarrollo iOS existen varios **sistemas de persistencia**, diferentes "**herramientas**" que podemos utilizar como desarrolladores para guardar los **datos** de nuestras aplicaciones.
Aquí tienes algunos de ellos:

- NSUserDefaults
- Property Lists
- NSFileManager
- SQLite
- Core Data

Nosotros, en este tutorial nos vamos a centrar en **Core Data**. Es probable que en próximos tutoriales veamos **otros sistemas** de persistencia que puedes utilizar.

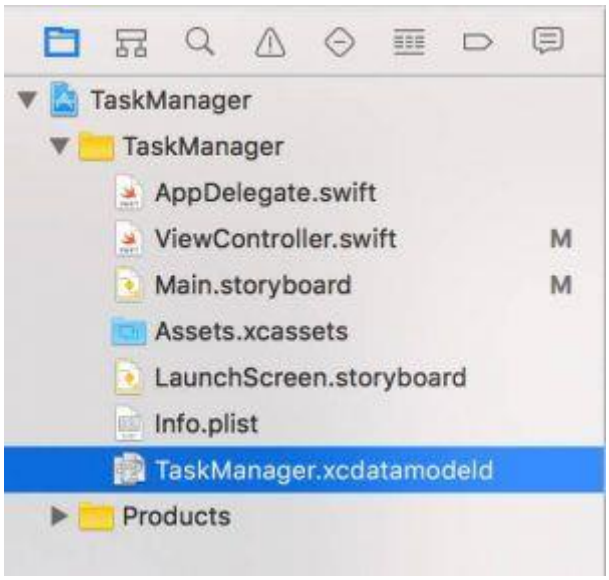
¿En qué consiste Core Data?

Core Data es un **Framework de Persistencia** desarrollado por Apple que nos permite simplificar la gestión del **modelo de datos** de nuestras aplicaciones.
Core Data suele utilizarse a través de una **base de datos** de tipo **SQLite**.
Pero el valor que realmente nos aporta, son una **serie de herramientas** que nos facilitan tanto la **creación** del modelo de nuestra app como la **gestión** posterior desde nuestro código.
Además, esta **capa** que envuelve nuestra **base de datos SQLite** nos permite que nosotros como desarrolladores **no** tengamos que trabajar directamente **con sentencias SQL**.
Una vez que sabemos en que consiste Core Data, veamos los **conceptos** más importantes que debes dominar.

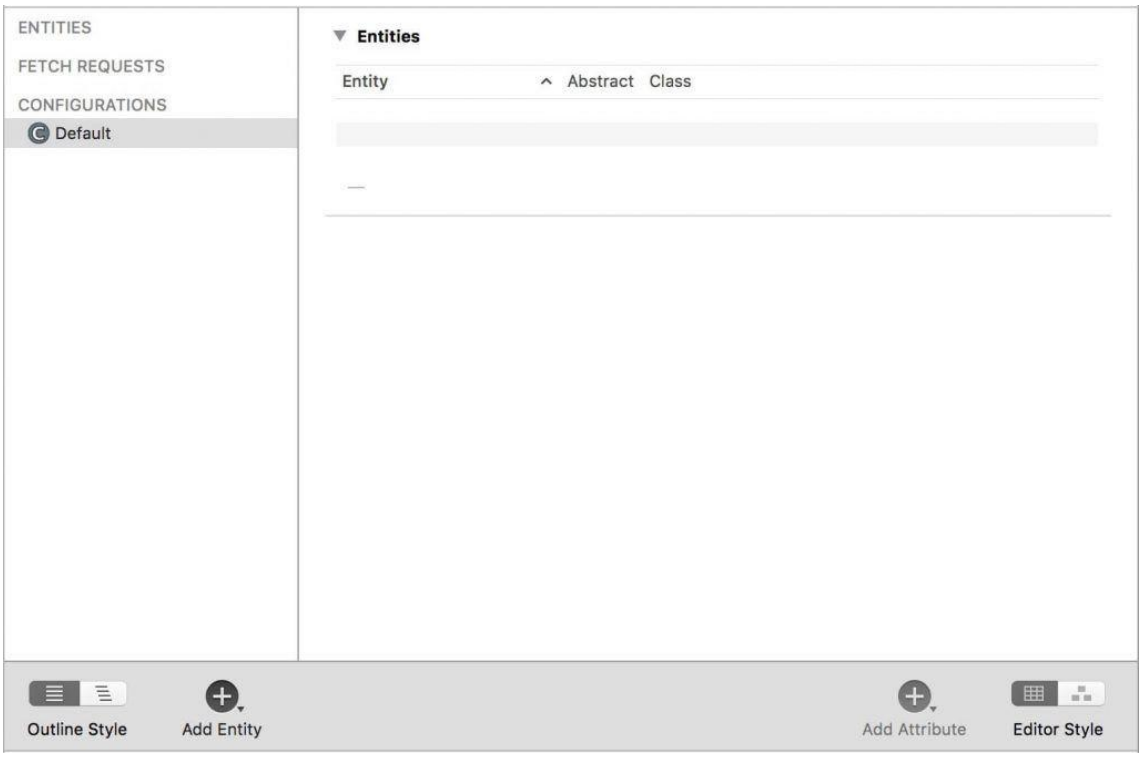
Conceptos Fundamentales

NSManagedObjectModel
Es la **representación** de nuestro **Modelo** en disco.
NSManagedObject
El objeto NSManagedObject representa un **objeto único** almacenado en Core Data.
NSManagedObjectContext
NSManagedObjectContext representa algo parecido a un "**espacio de memoria temporal**" donde poder trabajar antes de guardar los datos.
Si piensas en **como guardar** un objeto con Core Data, podríamos decir que se trata de un **proceso** de dos pasos. Primero insertas el objeto en el **managed object context** y una vez que estás seguro puedes **confirmar el guardado** del objeto almacenándolo en disco.
Xcode genera **automáticamente** un managed object context en nuestras aplicaciones siempre que activemos la opción **Use Core Data** al crear el proyecto.
Concretamente el managed object context se almacena en una propiedad del **appDelegate**, por lo que cuando necesites utilizarlo lo primero que tendrás que hacer será **crear una referencia** al appDelegate de tu aplicación.
NSEntityDescription
El objeto **NSEntityDescription** describe una **Entidad** en Core Data. Una instancia de NSEntityDescription determina el nombre de la entidad, sus atributos y relaciones y la clase por la que está representada.
NSFetchRequest
NSFetchRequest es la clase responsable de **recuperar datos** de Core Data. Para recuperar estos datos, utilizaremos **peticiones** a las que especificaremos una serie de criterios. Estas peticiones son bastante potentes. Puedes utilizar **fetchRequest** para recuperar un **conjunto de objetos** que cumplan unas determinadas condiciones. Por ejemplo: "Dame todos los usuarios que se hayan dado de alta en el último mes y que hayan realizado alguna publicación en nuestra app". NSFetchRequest utiliza **calificadores** para filtrar los resultados que queremos obtener.

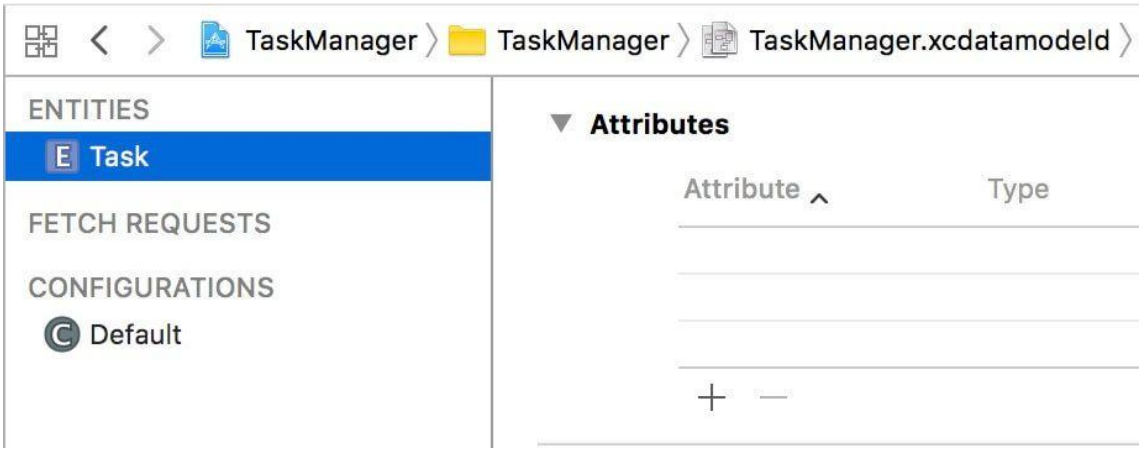
Creando el Modelo de nuestra aplicación
Antes de entrar directamente a **modificar** el código de la app, vamos a **crear el modelo** de nuestra aplicación.
El primer paso será crear un **Managed Object Model**, que representa a nuestro **modelo** en Base de Datos.
Como, al crear nuestro proyecto, activamos la **opción Core Data**, Xcode **automáticamente** ha creado un **fichero**que representa nuestro modelo de datos y que se llama **TaskManager.xcdatamodeld**.



Haz clic en **TaskManager.xcdatamodeld** para abrirlo. Verás el **editor** que Xcode te ofrece para crear nuestro modelo.



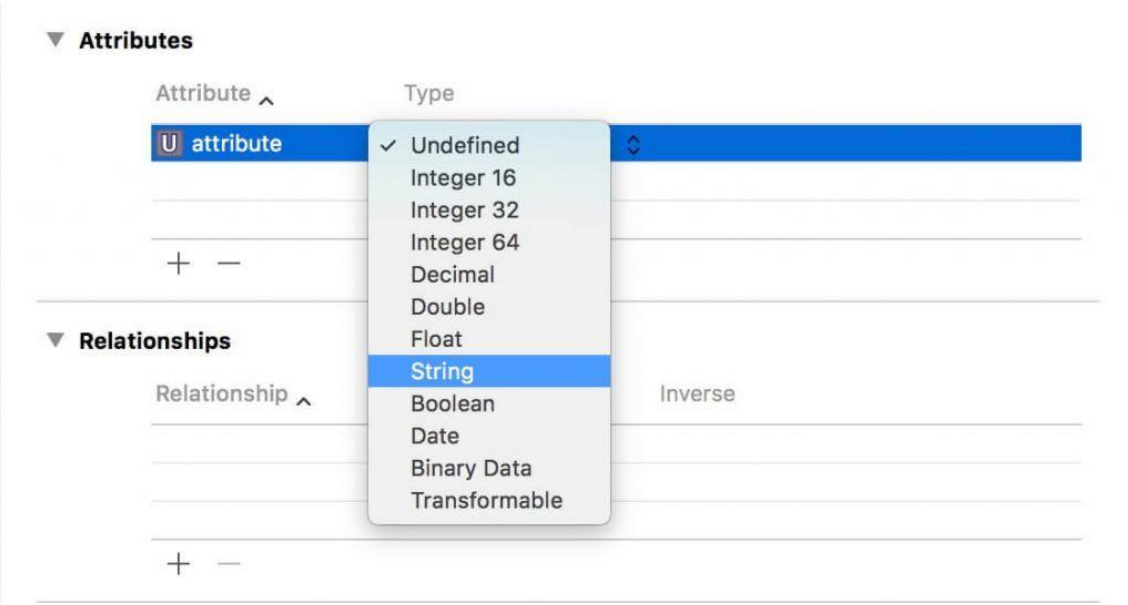
Este **editor** ofrece un gran número de opciones. Por ahora nos centraremos en **crear una nueva Entidad**.
Haz clic en el botón **Add Entity** situado en la parte inferior izquierda para crear una **Entidad**. Haz doble clic en la Entidad que acabas de crear para cambiarle el nombre.
Dale el nombre **Task**.



Le damos este nombre porque va a representar a **cada una de las tareas** que vamos a ir creando en nuestra aplicación.
Si no tienes claro a que nos referimos cuando hablamos de una **Entidad**, aquí tienes una serie de **conceptos**relacionados con **Core Data** que deberías de conocer:

- Una **Entidad** es la representación de una Clase en Core Data. En una base de datos relacional correspondería a una Tabla. Por ejemplo, si quisiéramos crear una app como Instagram, podríamos tener una entidad llamada Usuario que representaría toda la información relativa a los usuarios de nuestra aplicación.
- Un **Atributo** es un tipo de dato concreto, asociado a una Entidad específica. Por ejemplo, en el caso de que tuviéramos una entidad llamada Usuario, podríamos tener como atributos: username, email, name, etc. En una base de datos relacional, un atributo de una entidad se correspondería con un campo de una tabla.
- Una **Relación** es un enlace entre diferentes entidades. Las relaciones pueden ser de 1-1 o de 1-N. Una relación 1-1 sería por ejemplo: Un usuario tiene una única imagen de perfil en Instagram. Una relación 1-N sería por ejemplo: Un usuario tiene asociadas varias publicaciones en Instagram.

Ahora que ya sabes lo que es un **atributo**, podemos crear uno nuevo en la entidad Task. Desde el Editor del Modelo de Xcode, selecciona la entidad **Task** y pulsa en el botón situado en la parte inferior derecha **Add Attribute**.
Dale el nombre **name** y cambia su tipo a **String**, pulsando en el desplegable situado bajo **Type**.



Como has visto en el **menú desplegable**, podemos especificar **diferentes tipos** para un atributo. Esto puede ser muy útil cuando crees modelos **más completos**.