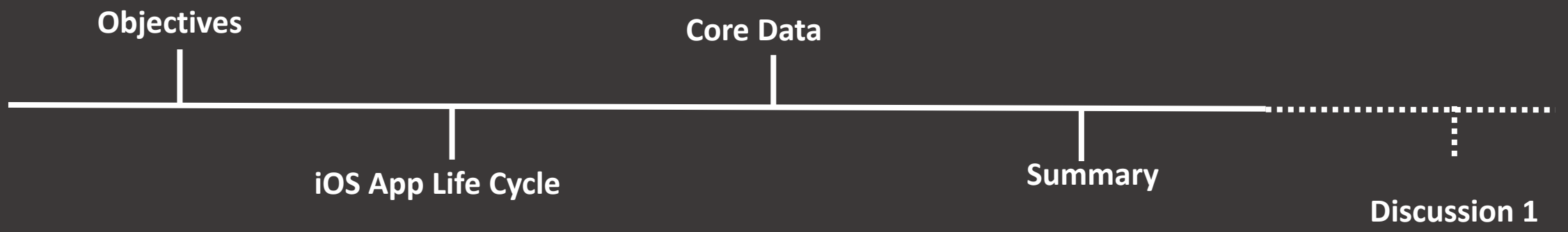
Mobile Applications Development 2 (MAD2)

Diploma in IT

Teaching Team:
Dr Joel Yang
Mr Charles Keck

NGEE ANN
SCHOOL OF INFOCOMM TECHNOLOGY

Chapter 5
iOS App Life Cycle and Core Data



Objectives

To be able to understand:

- The App Life Cycle
  - Managing your app's Life Cycle
- Core Data

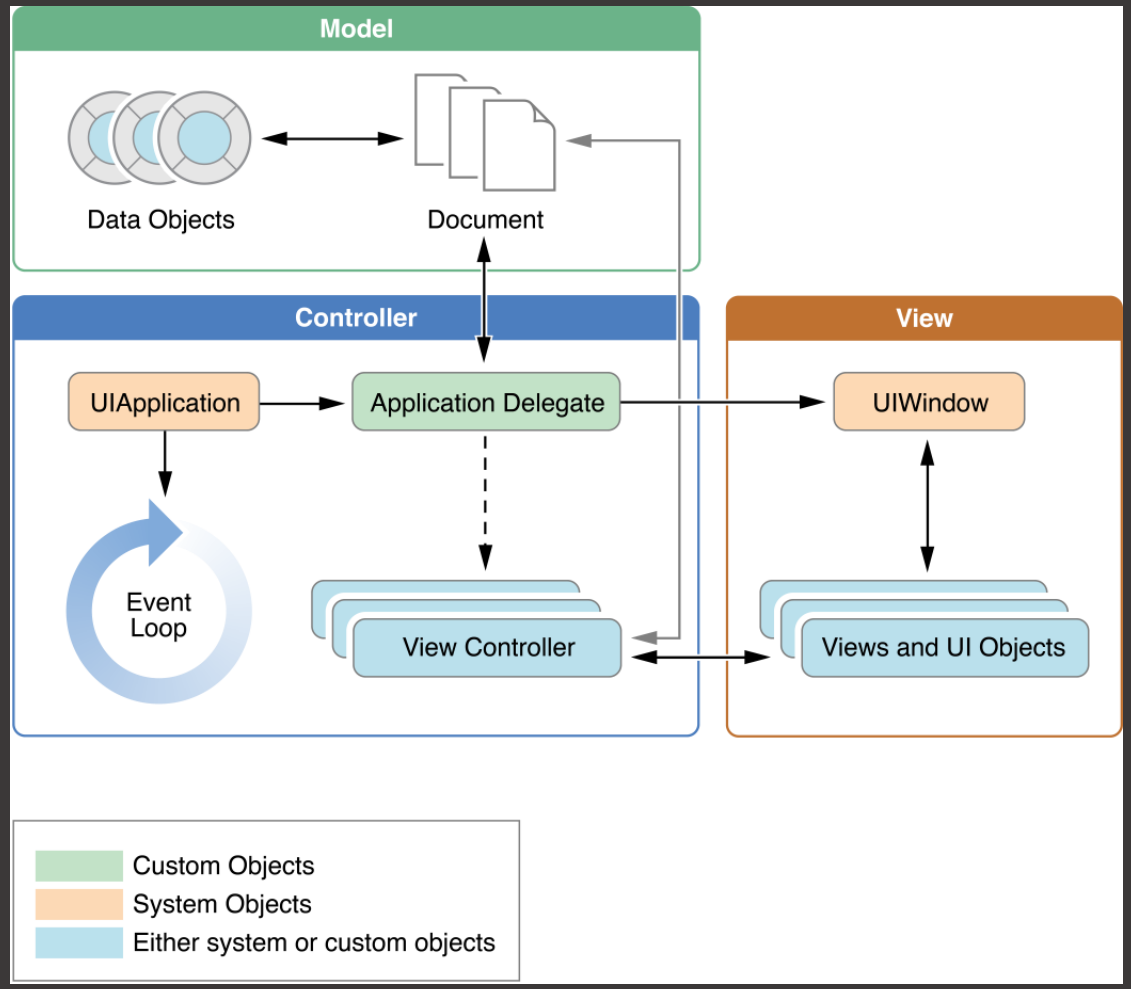
App Life Cycle

MAD2 Oct 2018

Structure of an App

- During startup, the UIApplicationMain function sets up several key objects and starts the app running. At the heart of every iOS app is the UIApplication object, whose job is to facilitate the interactions between the system and other objects in the app.

Structure of an App (Key Objects)

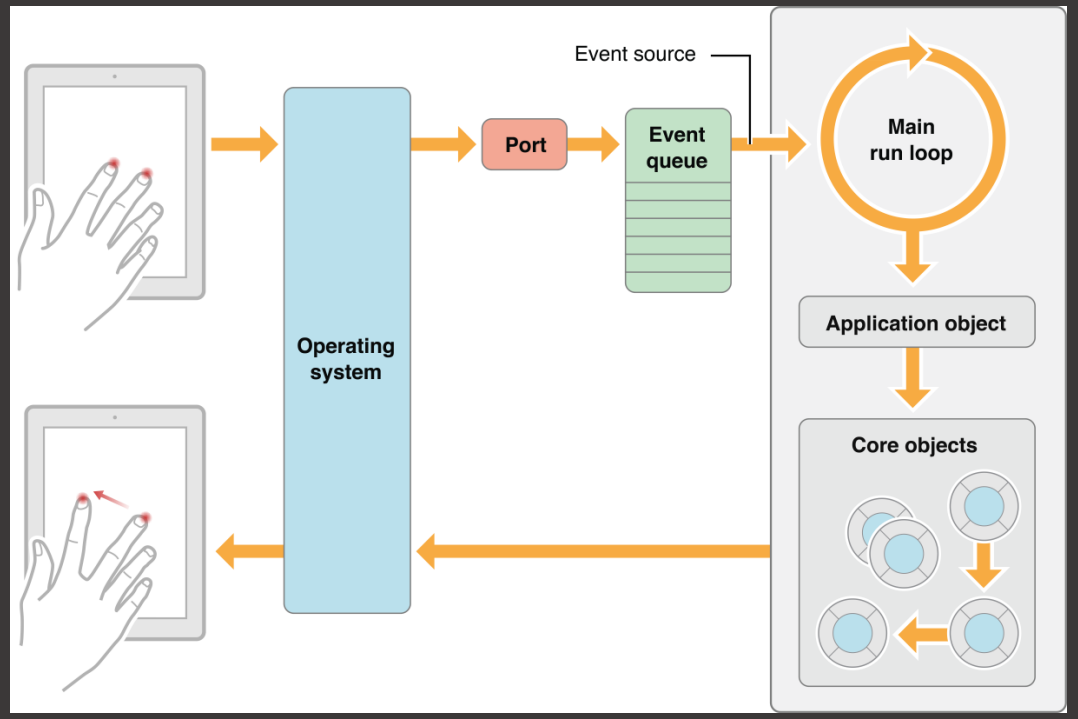


The Main Run Loop

- An app's main run loop processes all user-related events. The UIApplication object sets up the main run loop at launch time and uses it to process events and handle updates to view-based interfaces.

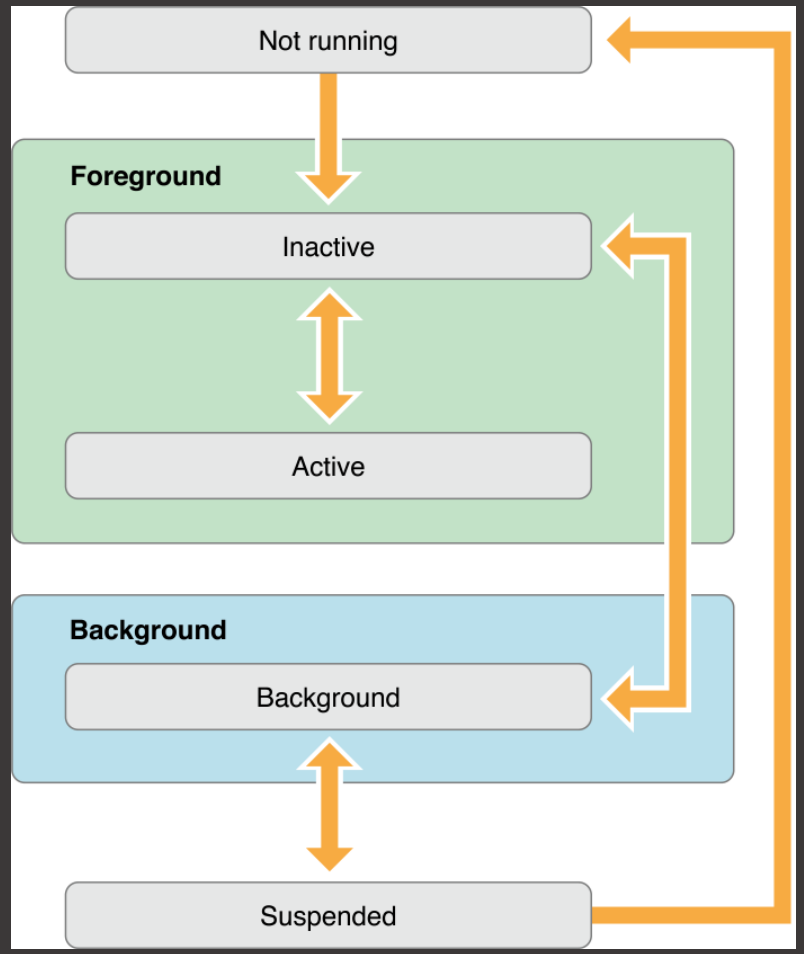
The Main Run Loop

- As the user interacts with a device, events related to those interactions are generated by the system and delivered to the app via a special port set up by UIKit.
- Events are queued internally by the app and dispatched one-by-one to the main run loop for execution.
- The UIApplication object is the first object to receive the event and make the decision about what needs to be done.

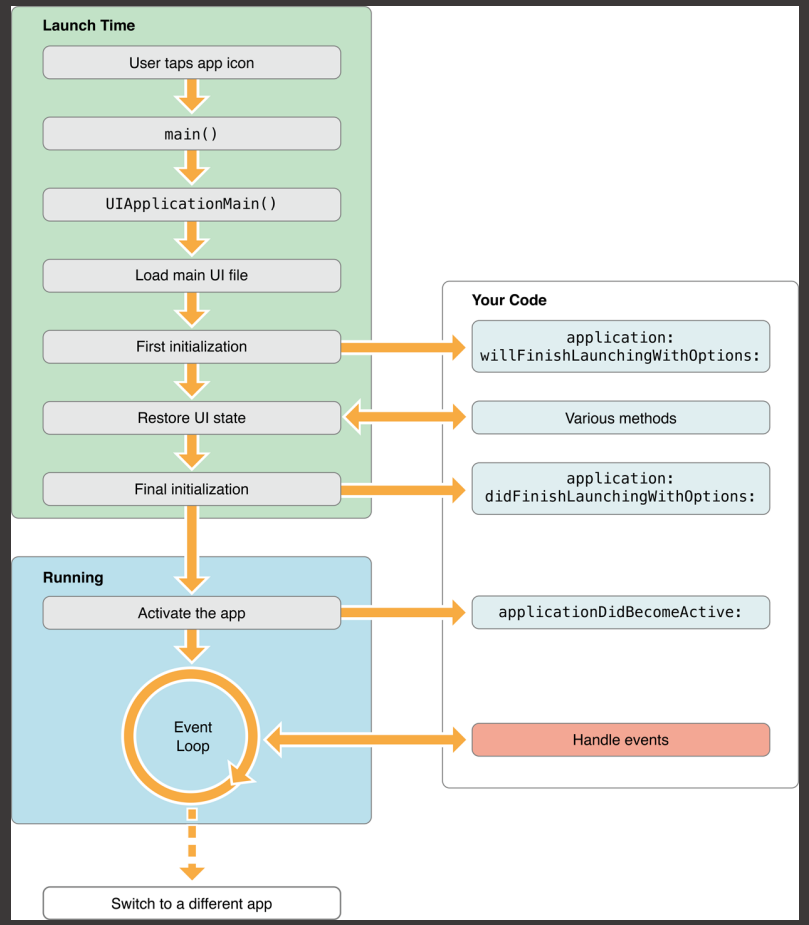


Execution States for Apps

- The system moves your app from state to state in response to actions happening throughout the system.
- For example, when the user presses the Home button, a phone call comes in, or any of several other interruptions occurs, the currently running apps change state in response.

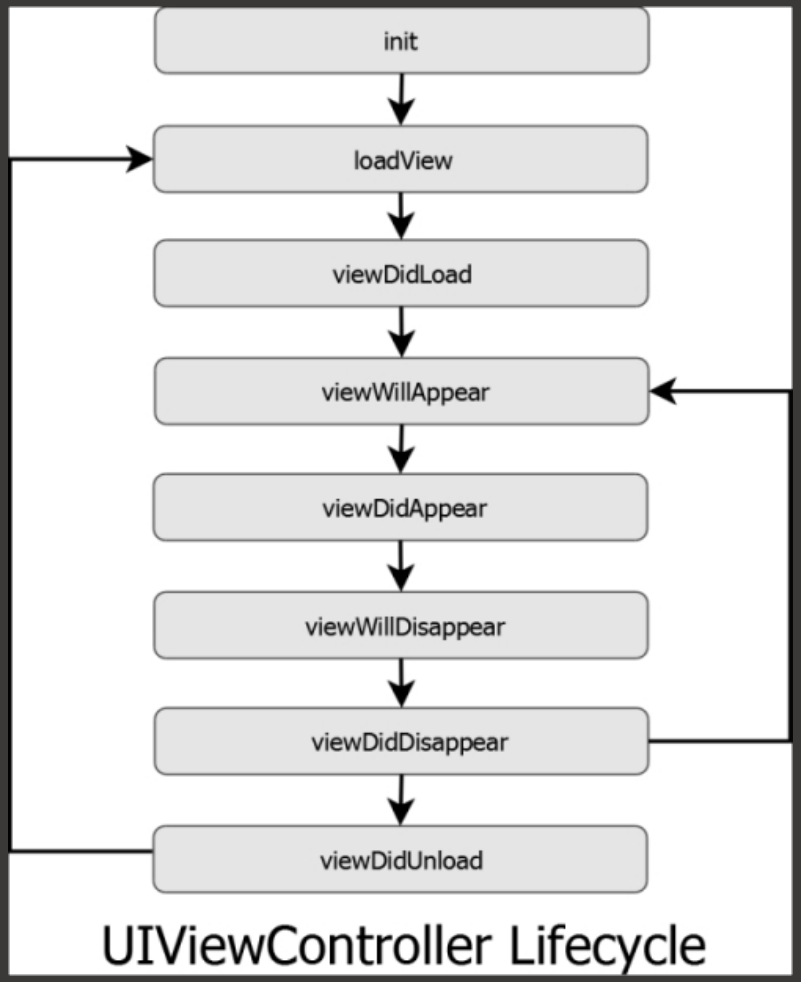


AppDelegate.swift



UIViewController Life Cycle Methods

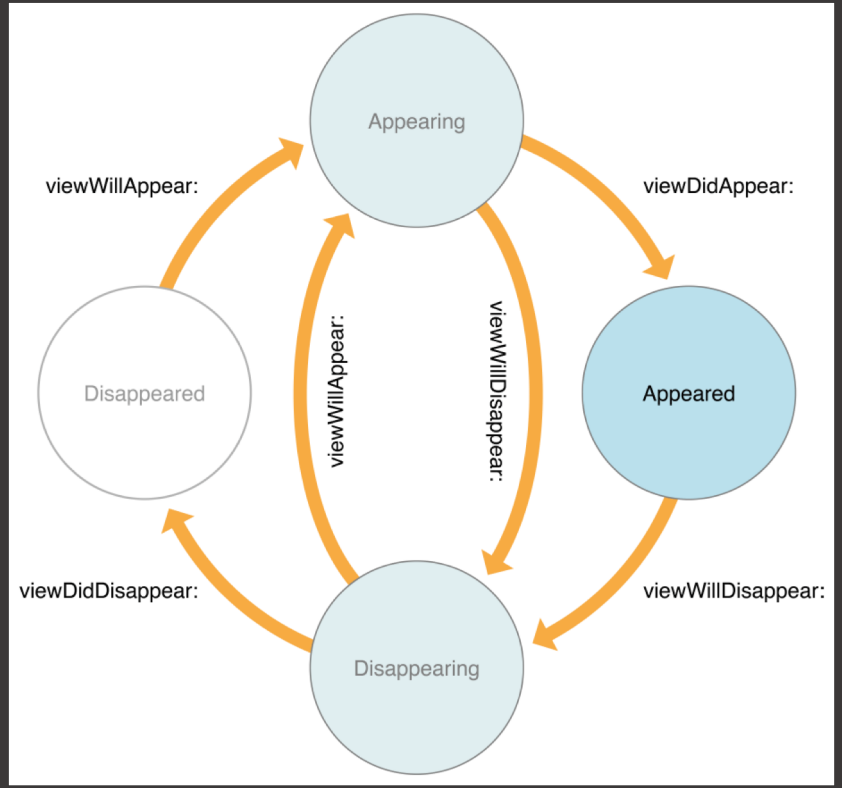
- The UIViewController class is the base class responsible for managing the communication between model and view classes in an iOS.



Handling View-Related Notifications

- When the visibility of its views changes, a view controller automatically calls its own methods so that subclasses can respond to the change.

- Use a method like viewWillAppear(_:) to prepare your views to appear onscreen, and use the viewWillDisappear(_:) to save changes or other state information.



Core Data

MAD2 Oct 2018

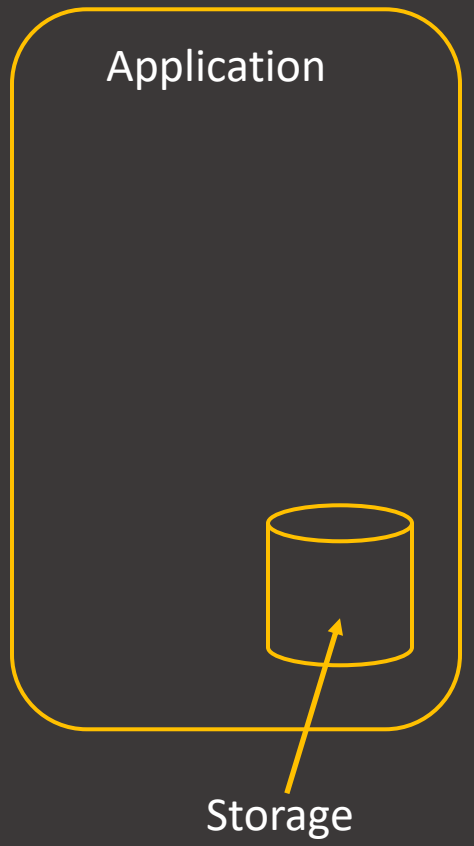
What is Core Data?

- Core Data is a framework that you use to manage the model layer objects in your application.

- It provides generalized and automated solutions to common tasks associated with object life cycle and object graph management, including persistence.

Persistent Storage

- Persistent storage is any data storage device that retains data after power to that device is shut off. It is also sometimes referred to as non-volatile storage.

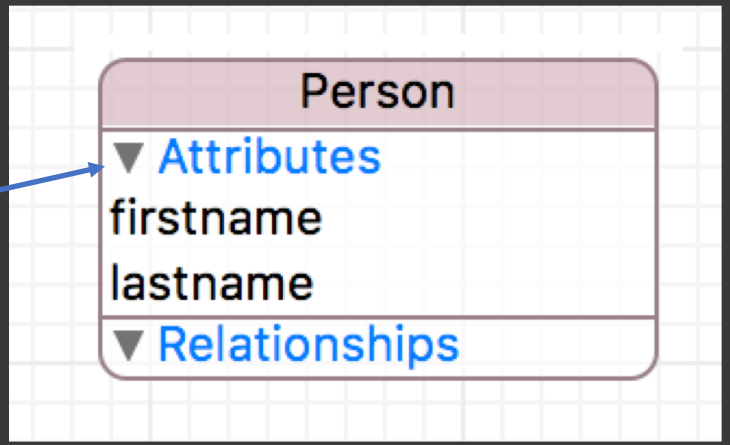


Entity Class

Person
firstname lastname : :

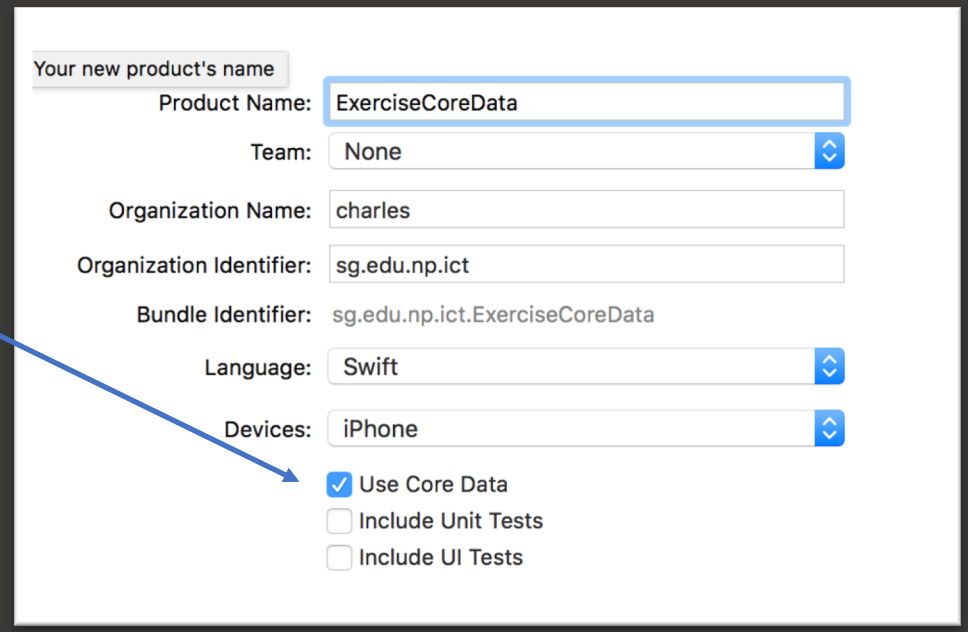
```
import CoreData


class Person: NSManagedObject {
    @NSManaged var firstname : NSString
    @NSManaged var lastname : NSString
}
```

Properties/Attributes?



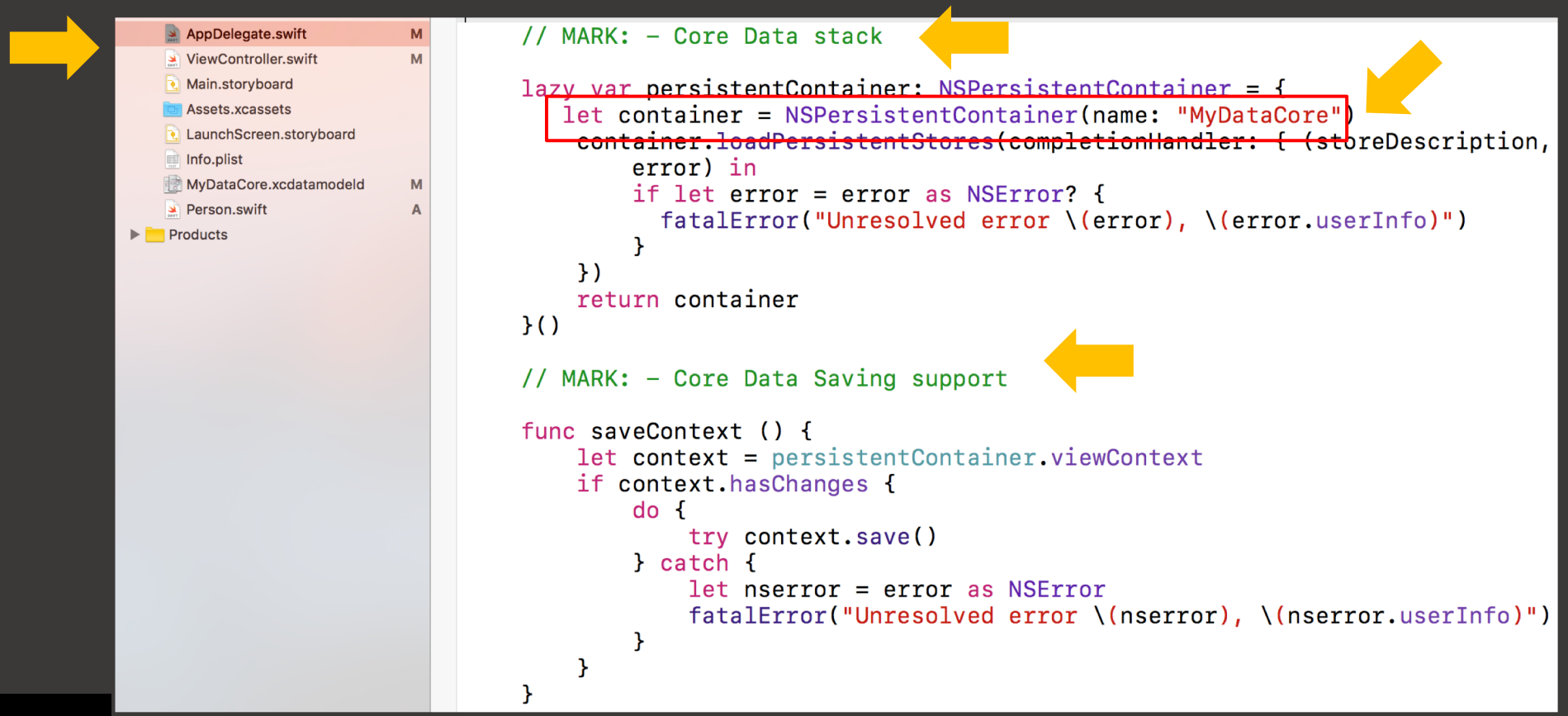
Xcode – Use Core Data

- Checking the "Use Core Data" box will cause Xcode to generate boilerplate code for what's known as a Core Data stack in AppDelegate.swift.

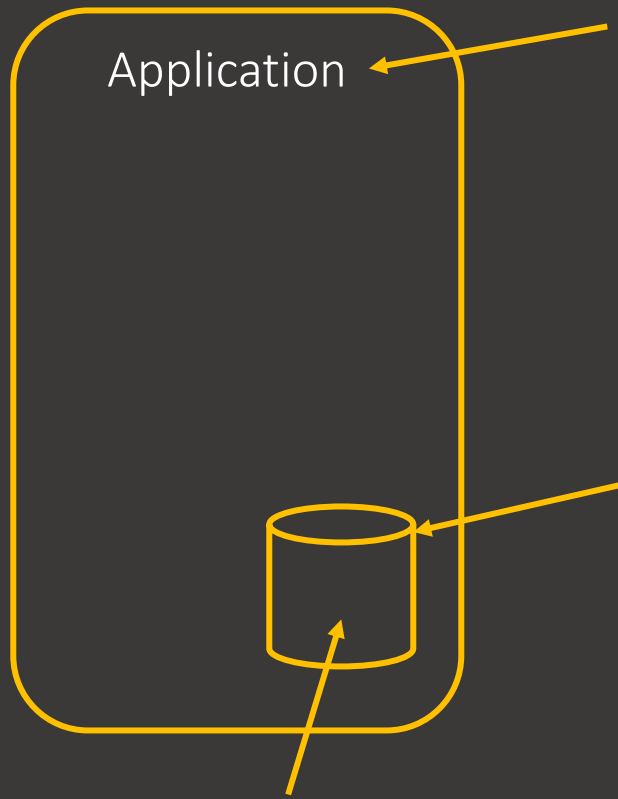


Core Data BoilerPlate

- AppDelegate.swift



AppDelegate and Context



AppDelegate - UIApplication is the class that represents the current running application
i.e. It sends messages to its delegate when things that affect the application happen (e.g. when the application launches, quits, gains or loses focus, and so on.)

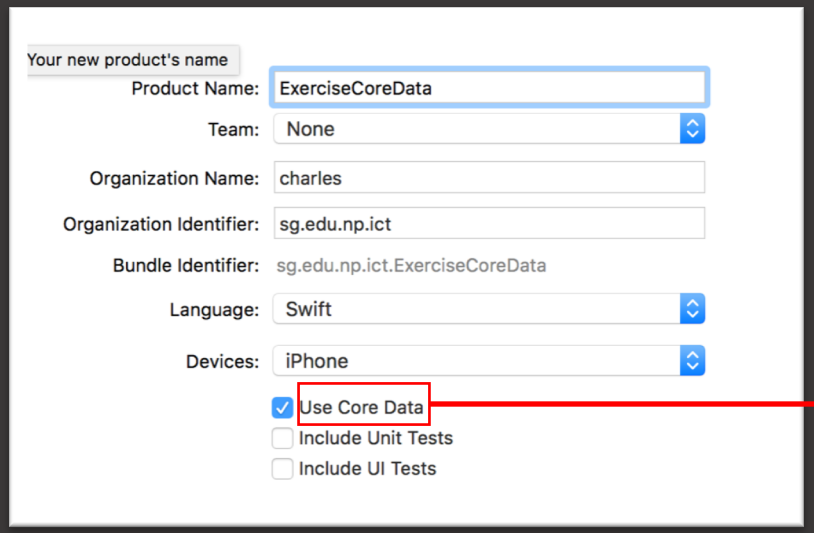
Accessible by a Context in the Application - An object that carries state information about an event, is a "Context" object.

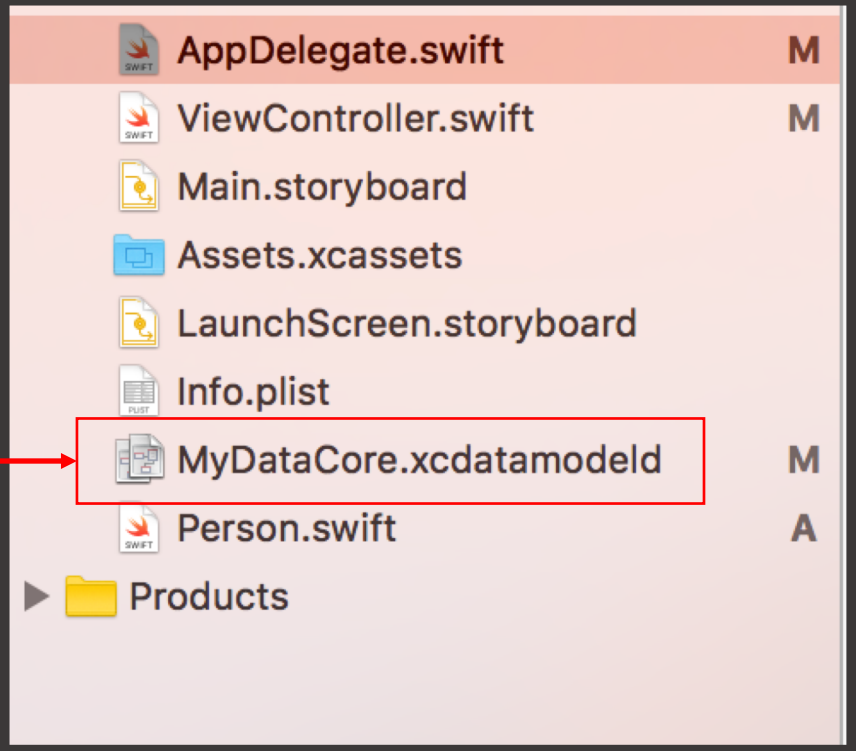
Swift Code

```
let appDelegate = (UIApplication.shared.delegate) as! AppDelegate

let context = appDelegate.persistentContainer.viewContext



//or write a getContext method in the appDelegate
let context = appDelegate.getContext()
```

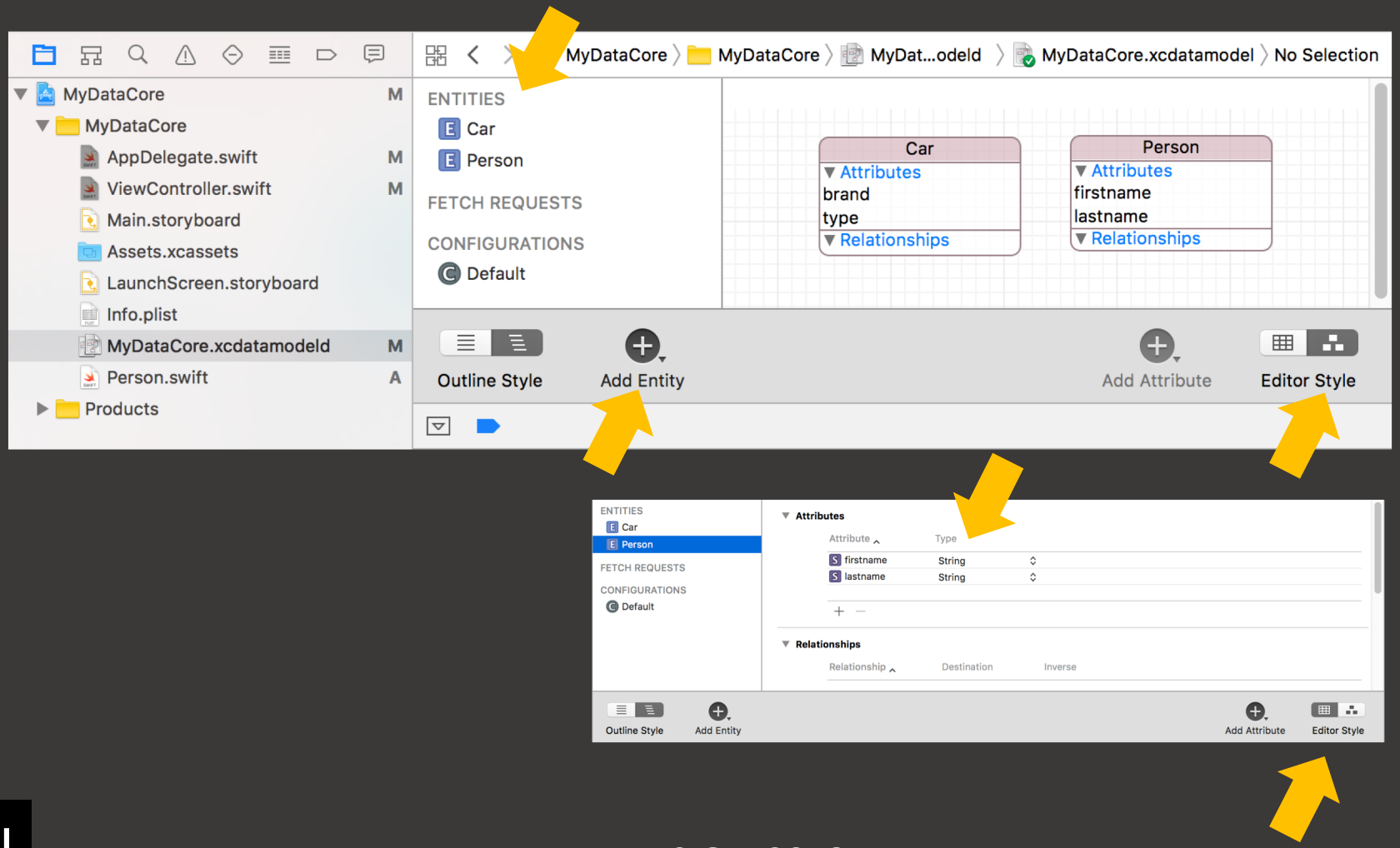
Data Model

- Extension : .xcdatamodeld

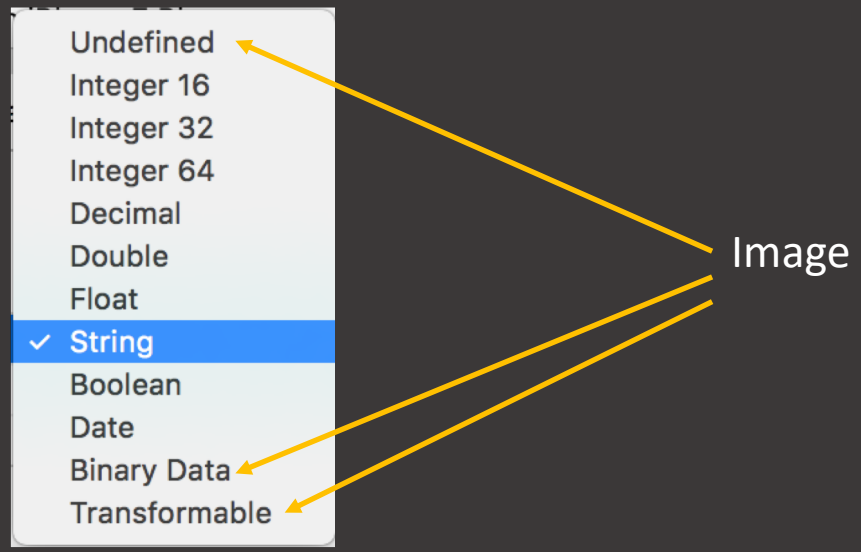




Data Model View

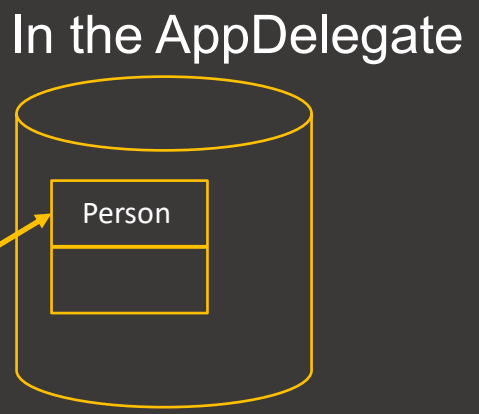


Data Type


Undefined
Integer 16
Integer 32
Integer 64
Decimal
Double
Float
String
Boolean
Date
Binary Data
Transformable
Image

Add Data in Swift

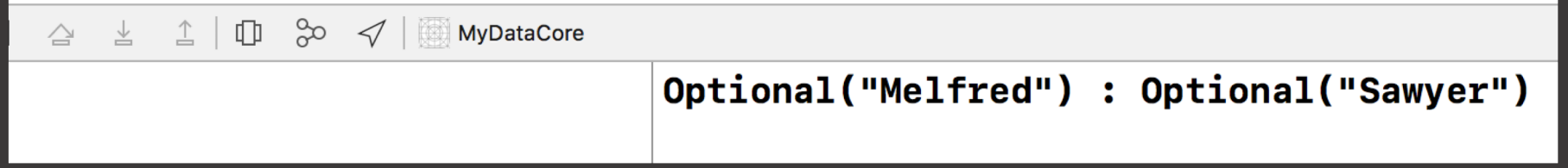
```
let p = Person(context: context)

p.firstname = "Melfred"
p.lastname = "Sawyer"

appDelegate.saveContext()
```



Retrieve Data

- Fetch Request and Exception Handling

```
do {
    let result = try context.fetch(Person.fetchRequest())
    let persons = result as! [Person]
    for person in persons {
        print("\(person.firstname) : \(person.lastname)")
    }
}
catch {
    print ("Error")
}
```

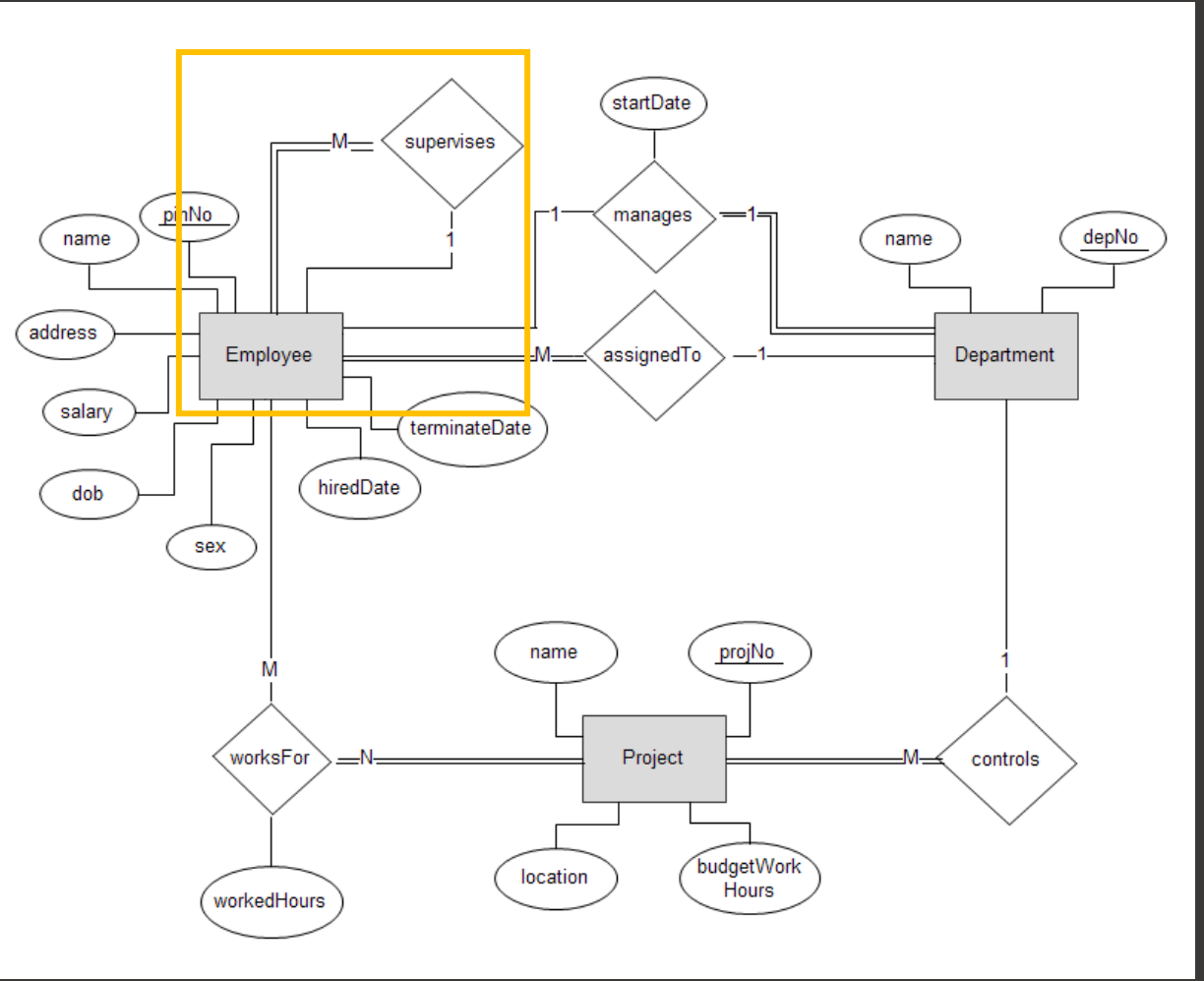


Relationships

- A relationship is a link between multiple entities. In Core Data, relationships between two entities are called to-one relationships, while those between one and many entities are called to-many relationships.

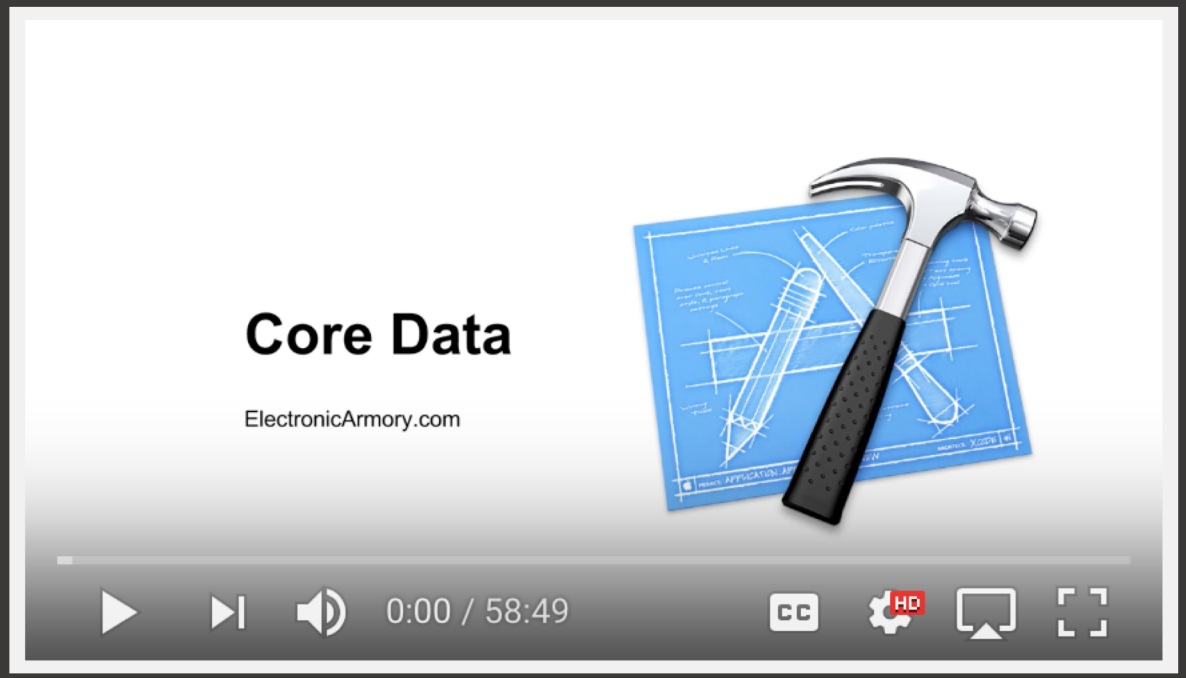
- For example, a Manager can have a to-many relationship with a set of employees, whereas an individual Employee will have a to-one relationship with his manager.

Relationships



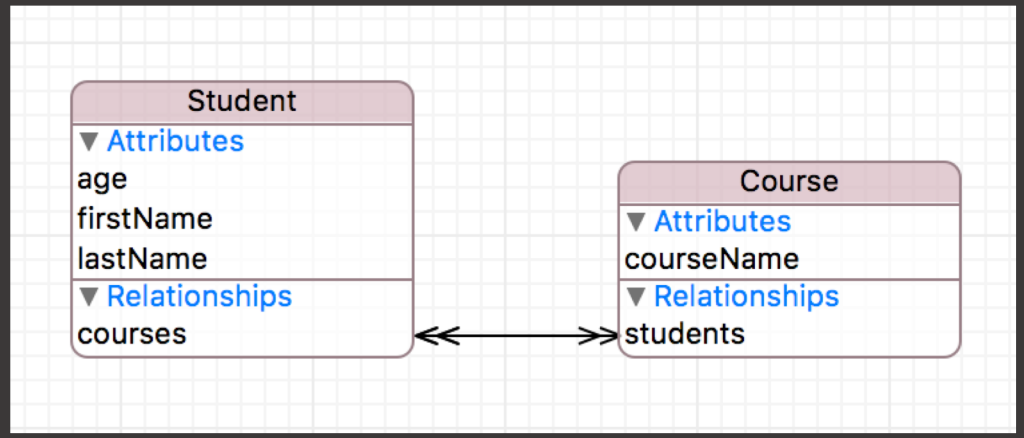
More Information

- https://www.youtube.com/watch?v=da6W7wDh0Dw
- Stop at 17:50



Quick Exercise

- Without creating NSManagedObject Subclass



XCode created the derived classes for developer

For advanced developer, you can generate the Classes (re-declaration error may occurs)

Summary

Understand the App Life Cycle and Core Data