Mobile Applications Development 2 (MAD2)

Diploma in IT

Teaching Team:
Mr Ng Poh Oon
Mr Charles Keck

NGEE ANN
SCHOOL OF INFOCOMM TECHNOLOGY

Chapter 5
Core Data

Objectives

To be able to understand:

- Core Data Framework
  - Persist or cache data and support undo on a single device

https://developer.apple.com/documentation/coredata

Core Data Framework

MAD2 Oct 2019

NGEE ANN POLYTECHNIC
SCHOOL OF INFOCOMM TECHNOLOGY

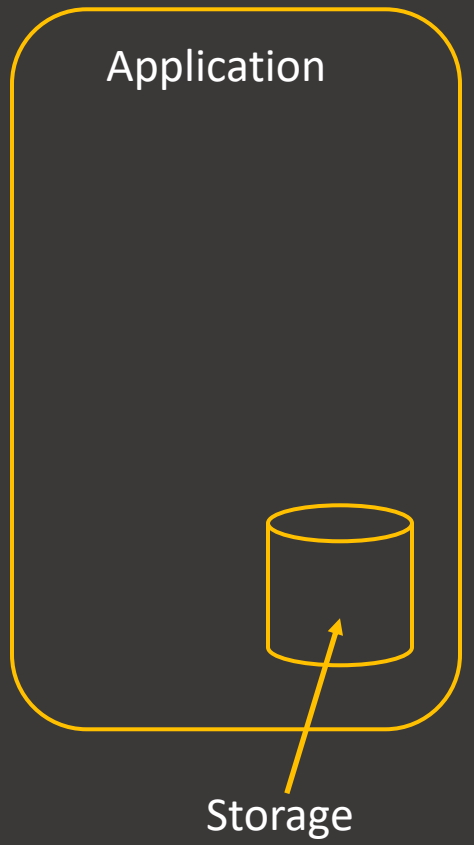
What is Core Data?

- 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.

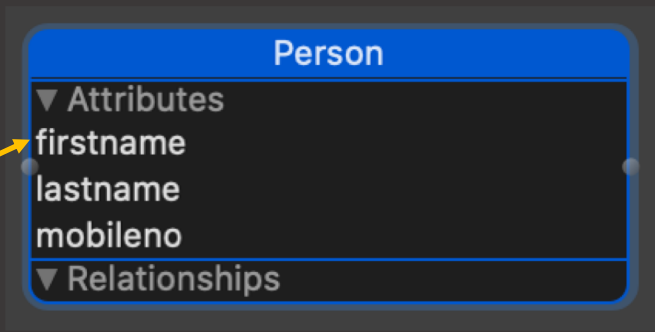


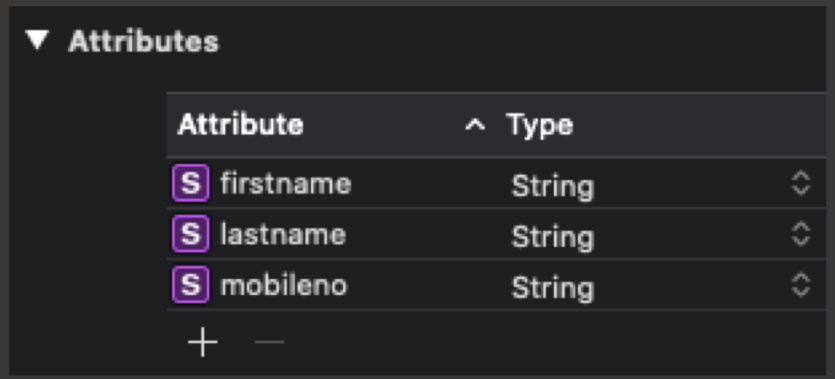
Entity Class

Person
firstname lastname : :

```
import CoreData


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

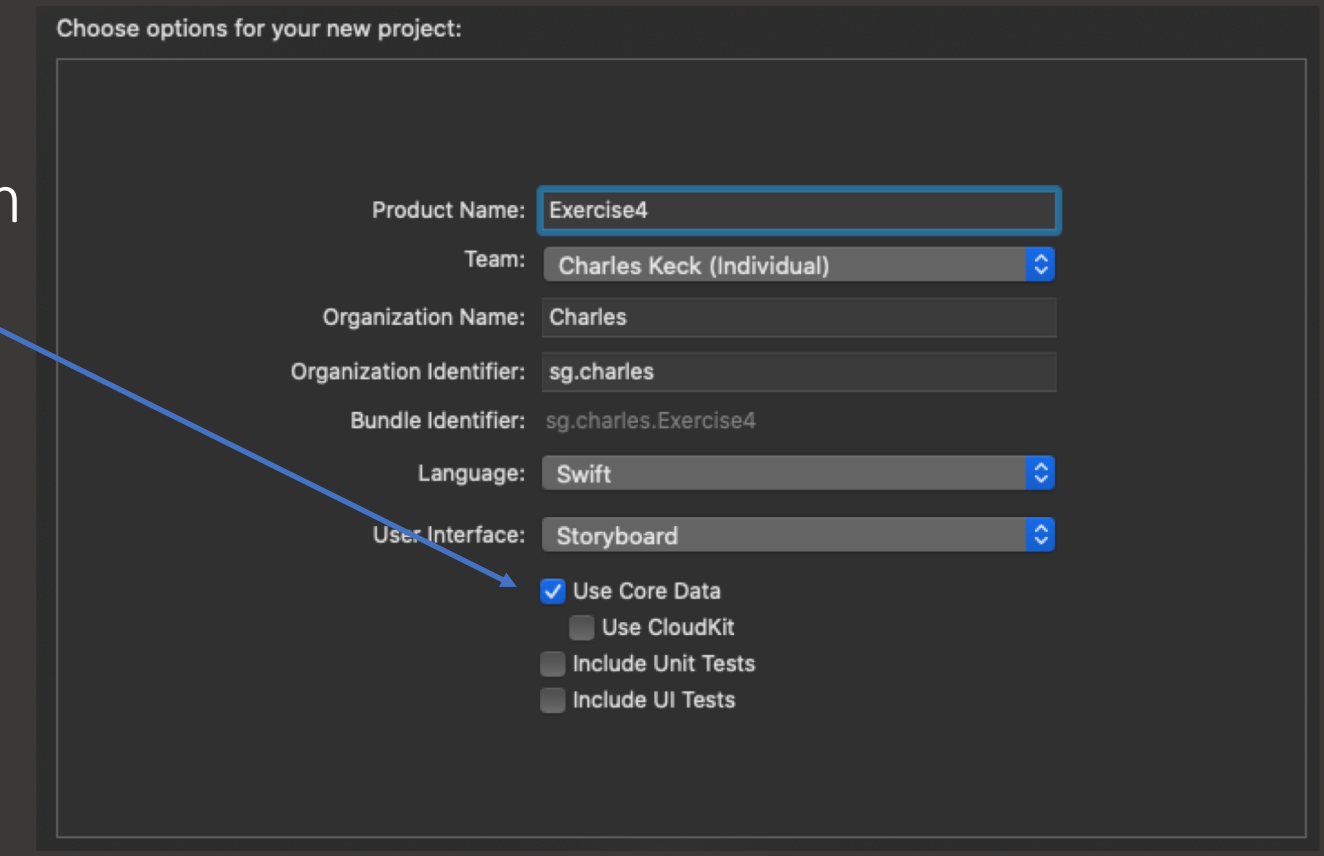
Properties/Attributes?





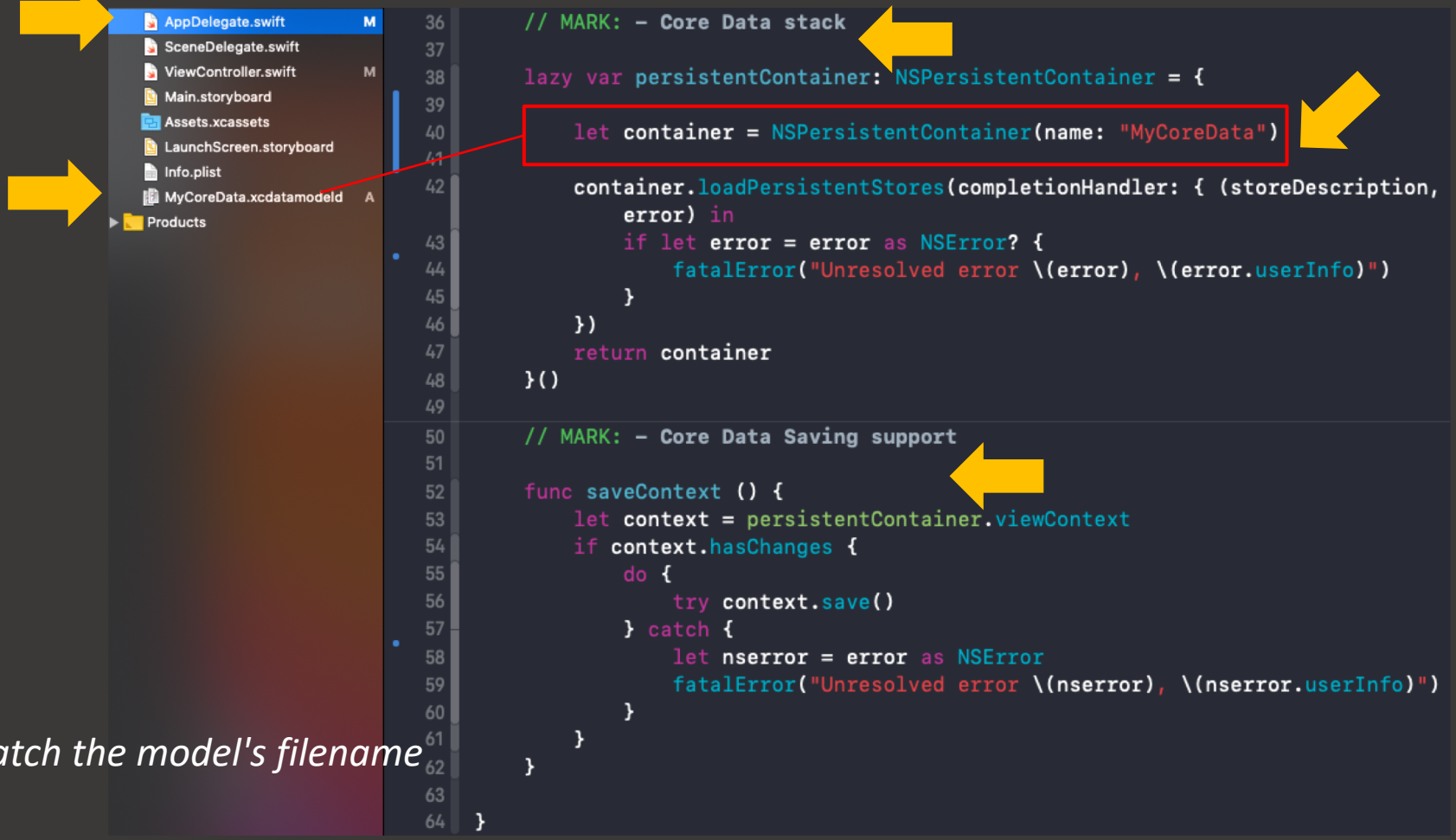
Xcode – Use Core Data

- 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.



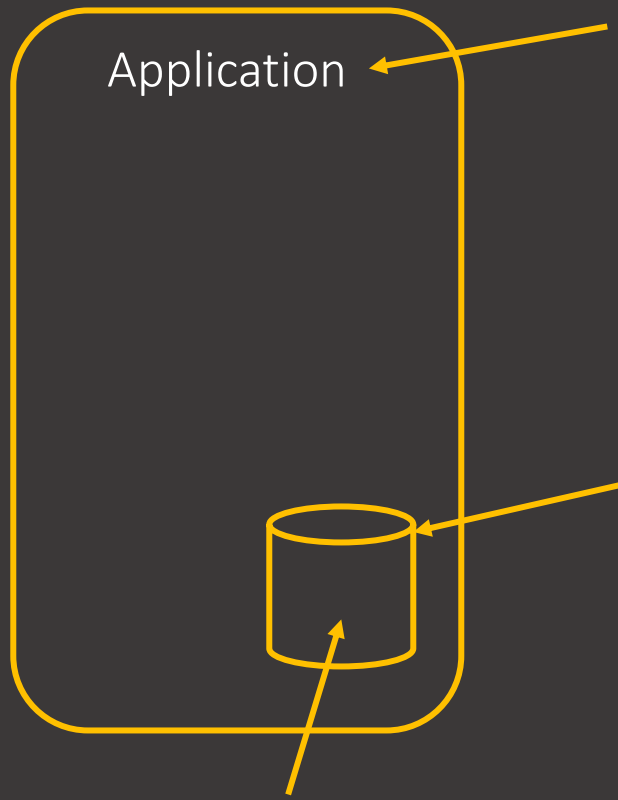
Core Data BoilerPlate

- AppDelegate.swift



The name must match the model's filename

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.)

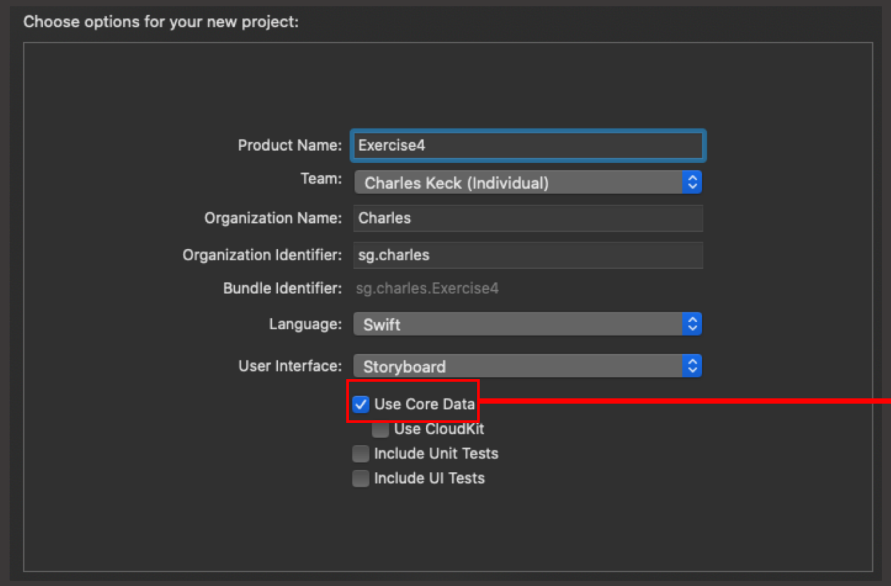
Accessible by a Context in the Application - An object that carries state information about an event, is a "Context" object.

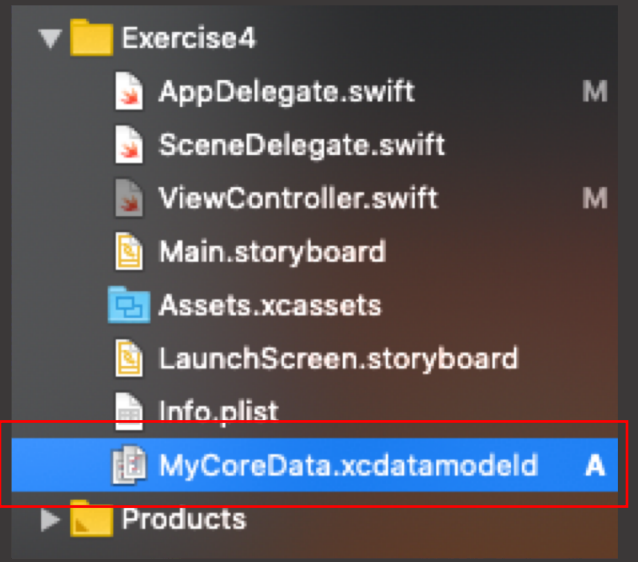
Swift Code (View Controller)

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

let context = appDelegate.persistentContainer.viewContext

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

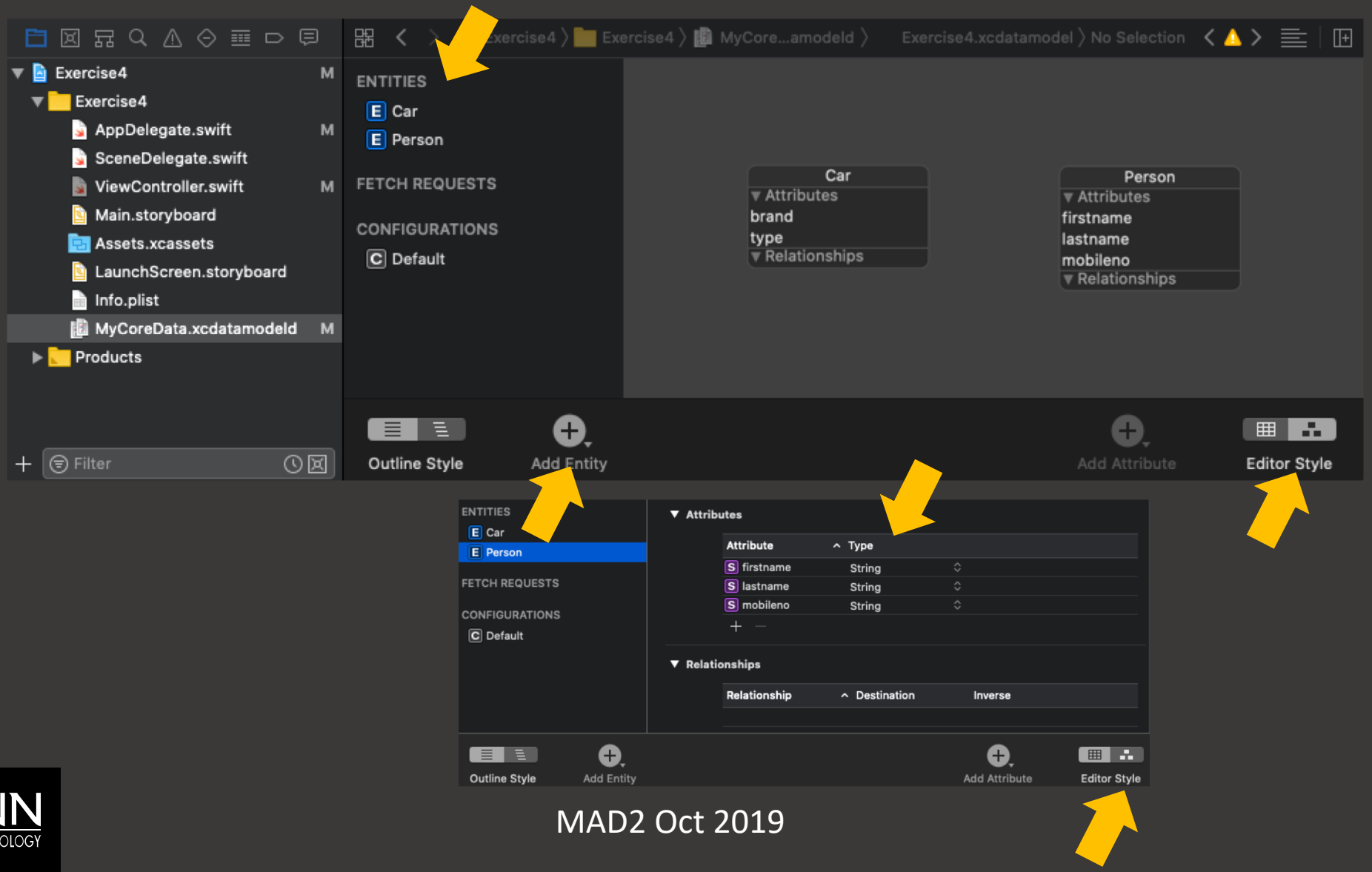
Data Model

- Extension : .xcdatamodeld

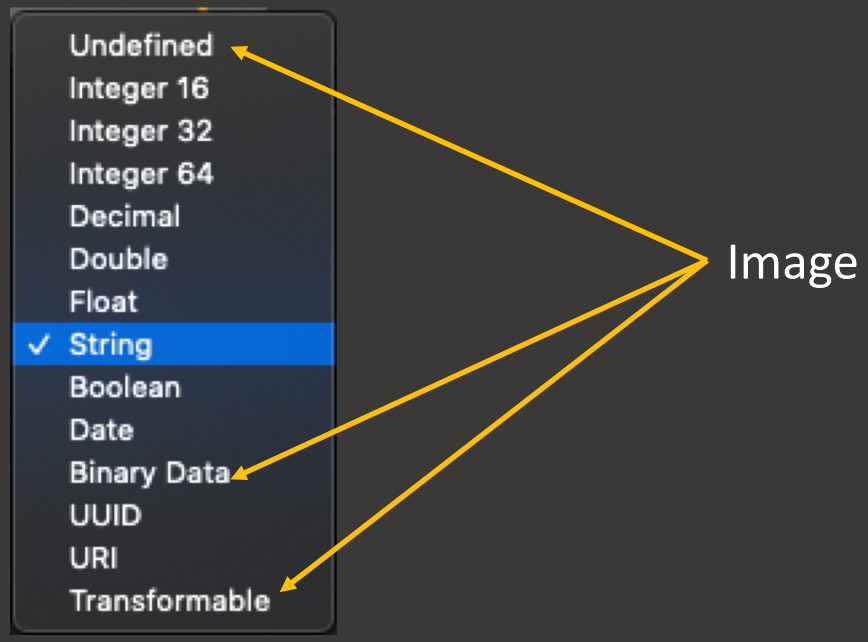




Data Model View



Data Type


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

Add Data to Core Data

```
func AddData(){
    let appDelegate = (UIApplication.shared.delegate) as! AppDelegate
    let context = appDelegate.persistentContainer.viewContext

    let entity = NSEntityDescription.entity(forEntityName: "Person", in: context)!

    let person = NSManagedObject(entity: entity, insertInto: context)
    person.setValue("Melfred", forKeyPath: "firstname")
    person.setValue("Sawyer", forKeyPath: "lastname")
    person.setValue("91111222", forKeyPath: "mobileno")
                    Data                    Key
    do {
        try context.save()

    } catch let error as NSError {
        print("Could not save. \(error), \(error.userInfo)")
    }

}
```

Fetching Data from Core Data

- Fetch Request and Exception Handling

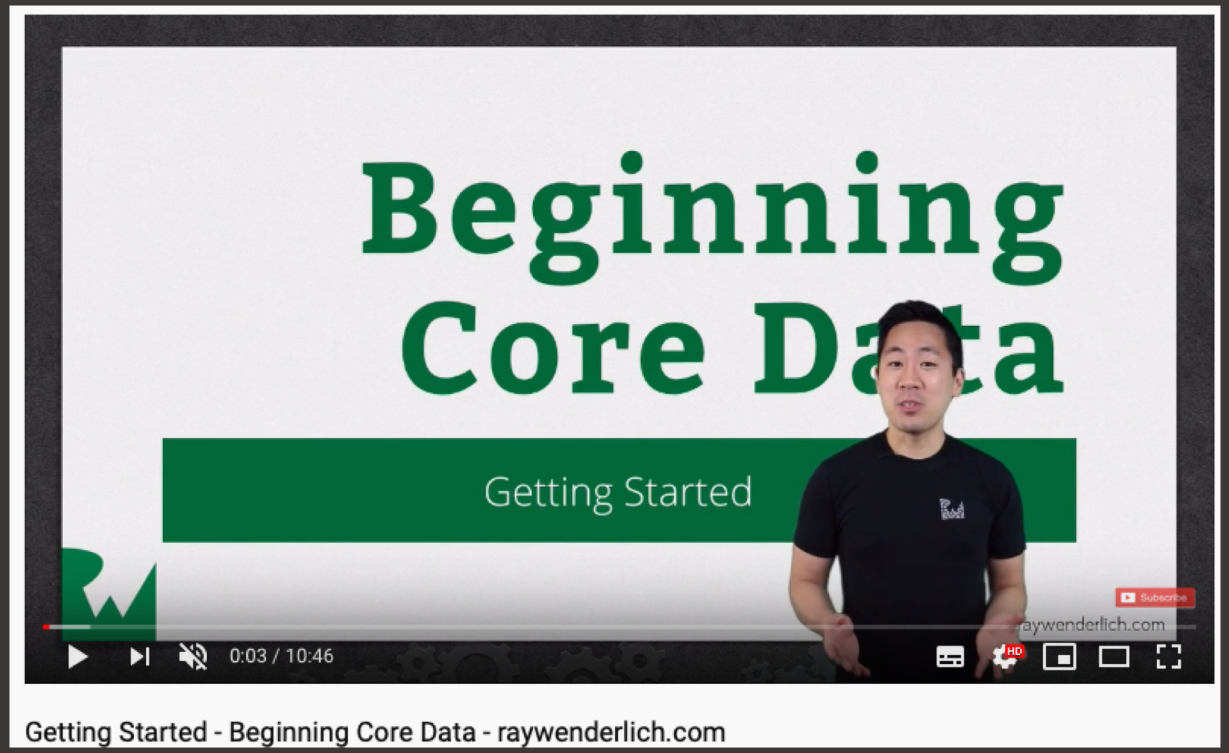
```
func FetchData(){

    var people:[NSManagedObject] = []

    let appDelegate = (UIApplication.shared.delegate) as! AppDelegate
    let context = appDelegate.persistentContainer.viewContext

    let fetchRequest = NSFetchRequest<NSManagedObject>(entityName: "Person")
    do {
        people = try context.fetch(fetchRequest)

        for p in people {
            let firstname = p.value(forKeyPath: "firstname") as? String
            let lastname = p.value(forKeyPath: "lastname") as? String
            let mobileno = p.value(forKeyPath: "mobileno") as? String
            print("\(firstname!) \(lastname!), \(mobileno!)")
        }

    } catch let error as NSError {
      print("Could not fetch. \(error), \(error.userInfo)")
    }
}
```

Empty List

```
Melfred Sawyer, 91111222
Simon Wood, 92222111

All Output ⌃   Filter
```

More Information

- https://youtu.be/PMze4tT2w_U



Getting Started - Beginning Core Data - raywenderlich.com

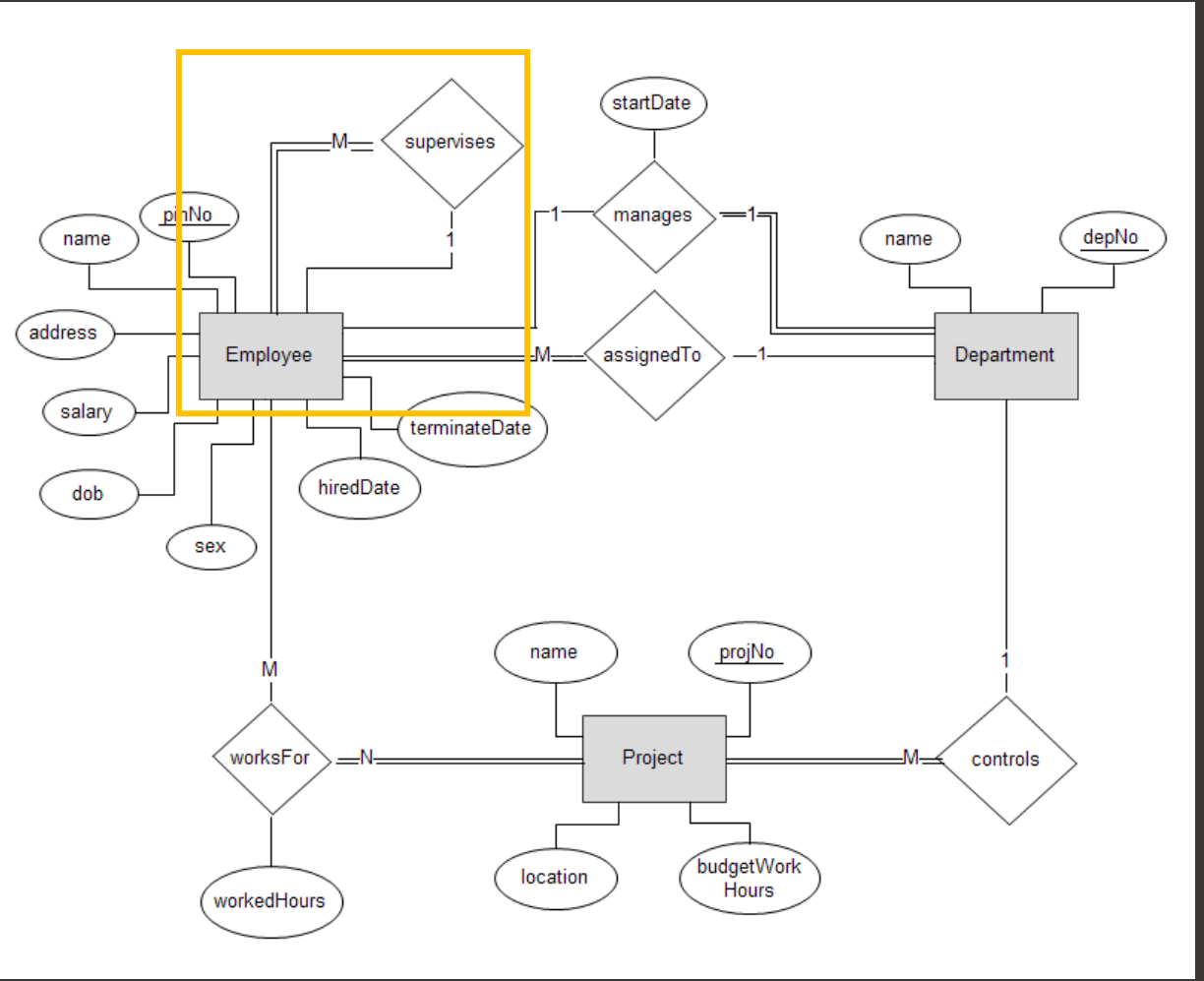
https://www.raywenderlich.com/7569-getting-started-with-core-data-tutorial#toc-anchor-003

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.

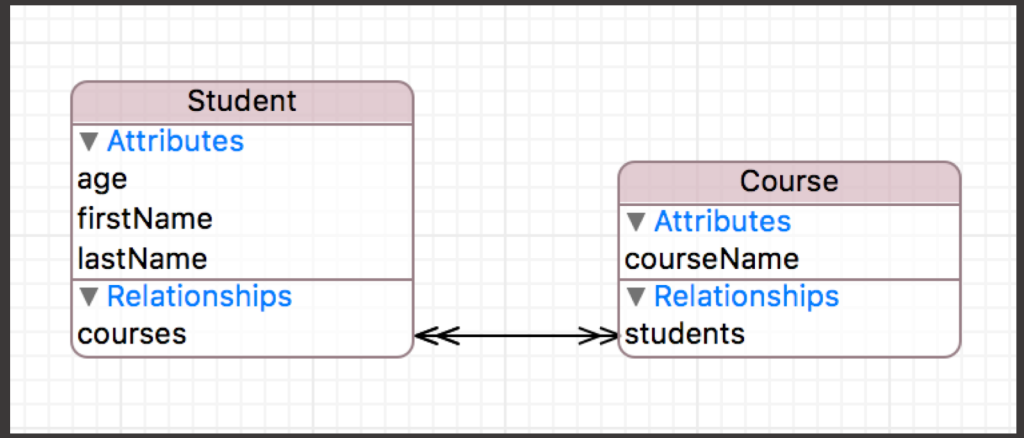
- For example, a Manager can have a to-many relationship with a set of employees, whereas an individual Employee will have a to-one relationship with his manager.

Relationships



Quick Exercise

- Without creating NSManagedObject Subclass

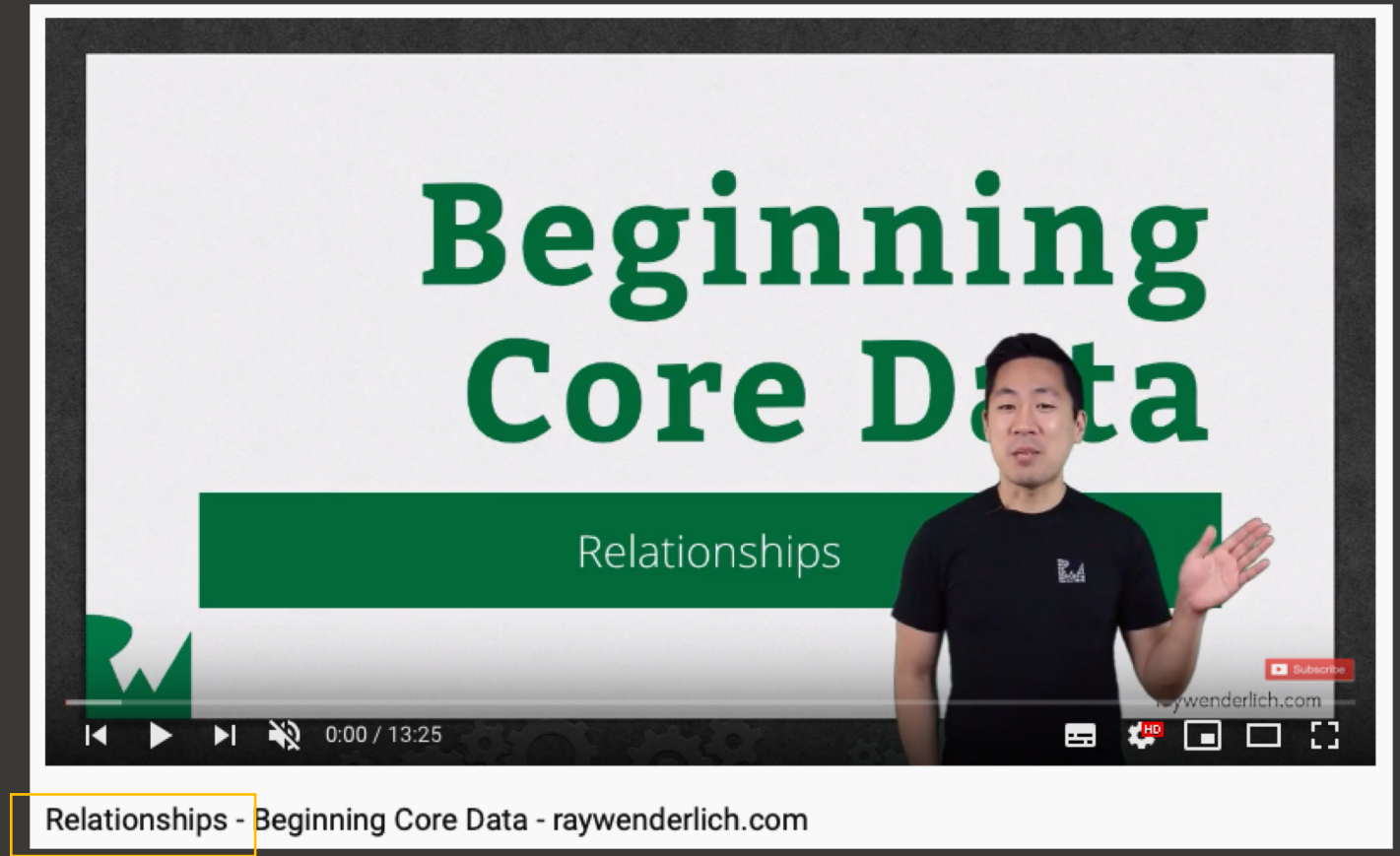


XCode created the derived classes for developer

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

More Information

- https://youtu.be/uJuLk1niBYA



Summary

Understand the Core Data framework