lecture 7

Data persistance

cs198-001 : fall 2017

Announcements

- Custom App Proposal due tonight at 11:59pm. Google form submission
- Snapchat Clone Part 2 due 11/6
- lab this week - meet with your assigned TA to go over your proposal

  you may need to attend a different lab (will send out room assignments Tuesday via email)
- grading

Overview : Today's Lecture

Future lecture topic preview

NSUserDefaults

Core Data

Data Persistence - motivation

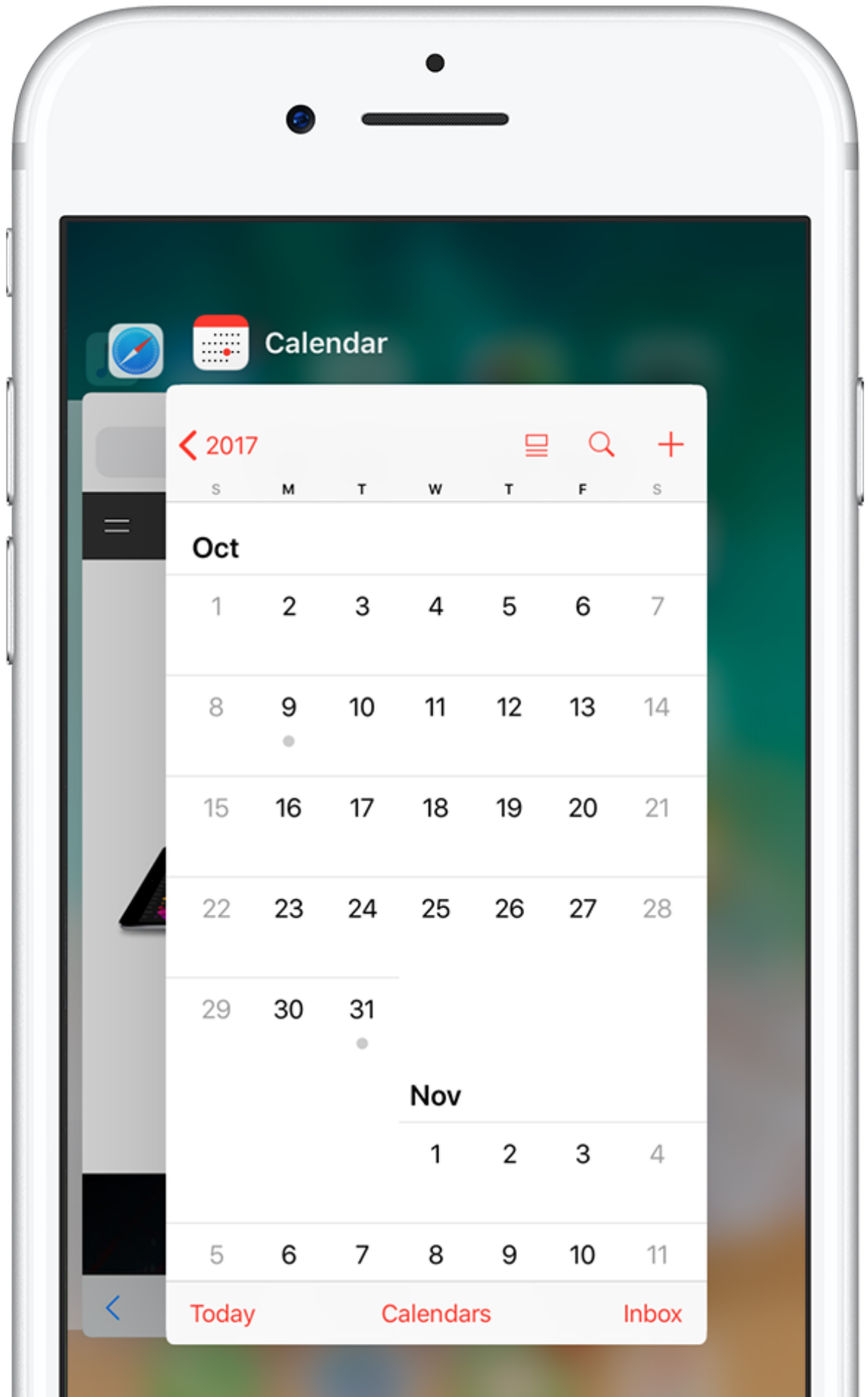
Allow users to store data in your application that will persist between application launches

Example: store high scores for a game, app preferences, background images, cached images fetched from the network, etc.

To check if data actually persists between app launches, force close your app (double click home button - swipe app off of the screen)

On simulator:

Home: cmd + Shift + H



Data Persistence - options

Store data in some remote database

i.e. Firebase. Requires internet + adds networking delays

Store data on phone disk

UserDefaults - store key value pairs (dictionary). Only to be used for small amounts of data. Very easy to use

CoreData - store objects and define relationships between them. Can be used for large amounts of data (proportional to amount of memory on phone). Much more complicated than UserDefaults

UserDefaults

UserDefaults

Simply a dictionary that persists between app launches.

To store (in this example, a string):

```
UserDefaults.standard.set("Dan",
                  forKey: "preferred name”)
```

To retrieve (again, a string):

```
if let savedName = UserDefaults.standard.
        string(forKey: "preferred name") {
    // do something with value
}
```

UserDefaults

```
func object(forKey: String)
```
Returns the object associated with the specified key.

```
func url(forKey: String)
```
Returns the URL associated with the specified key.

```
func array(forKey: String)
```
Returns the array associated with the specified key.

```
func dictionary(forKey: String)
```
Returns the dictionary object associated with the specified key.

```
func string(forKey: String)
```
Returns the string associated with the specified key.

```
func stringArray(forKey: String)
```
Returns the array of strings associated with the specified key.

```
func data(forKey: String)
```
Returns the data object associated with the specified key.

storing other data types

(non-inclusive list)

Core Data

...but first

App Delegate

App Delegate

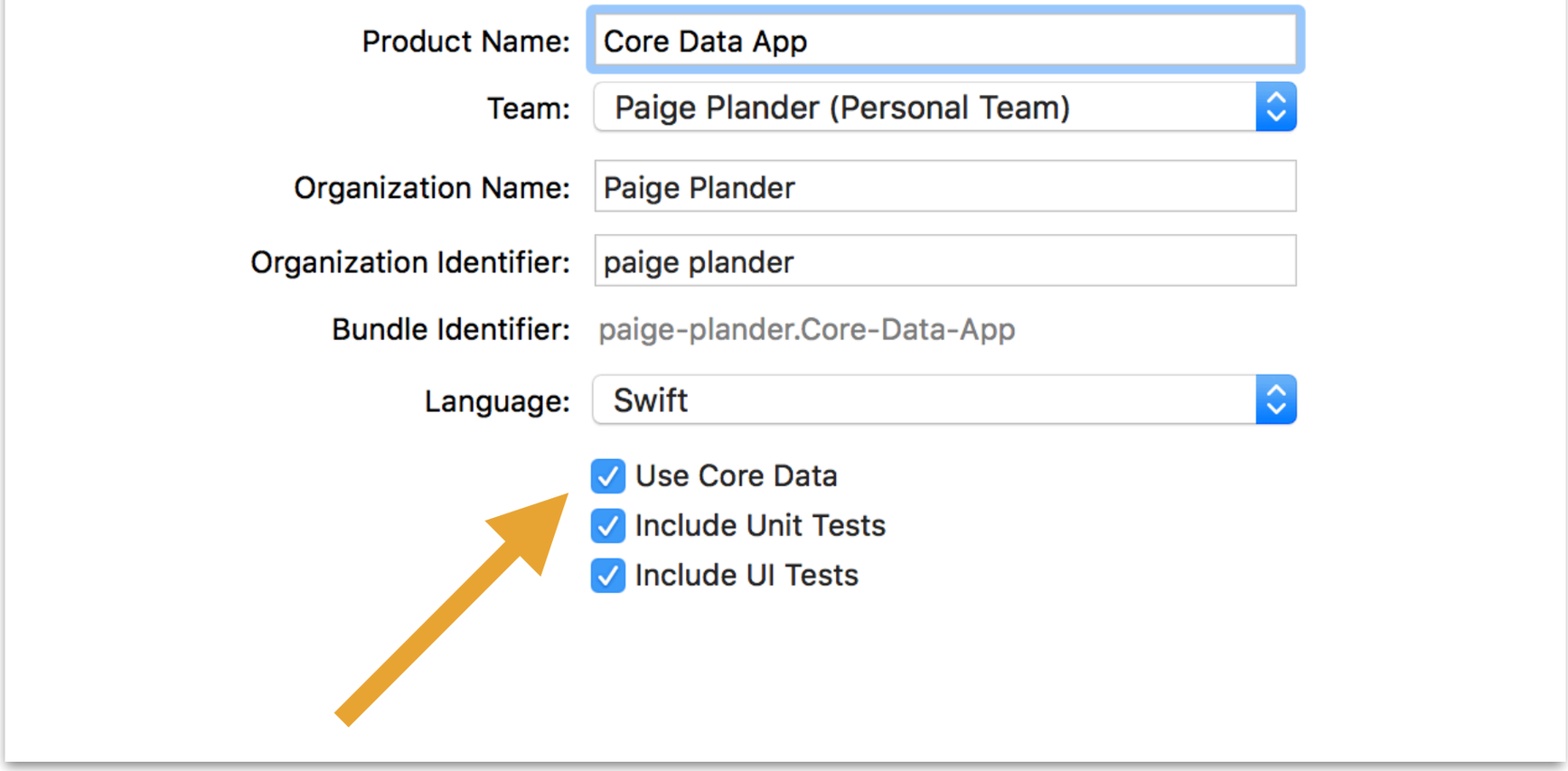
file name: AppDelegate.swift

methods within this class are repeatedly called throughout your app's lifecycle (they are called by a UIApplication singleton object)

When you want to use Core Data, you'll need to interact with functions within the App Delegate

App Delegate

```
// grab a reference to singleton
// instance of your appDelegate
let appDelegate =
    UIApplication.shared.delegate
    as! AppDelegate
```



Why do we care?

checking this icon generates some boilerplate Core Data related code for you within AppDelegate.swift

Core Data

Core Data - What is it?

Framework that allows you to store and retrieve data from a database (SQL) in an OOP way

Allows **data persistence** for large data sets / lists

Use it to create **data models** that can be added to and queried throughout your project

Core Data - What is it?

When do you want to use Core Data?

Examples

Allow users to save specific podcasts to their phone

Notes application that stores text + photos to your phone

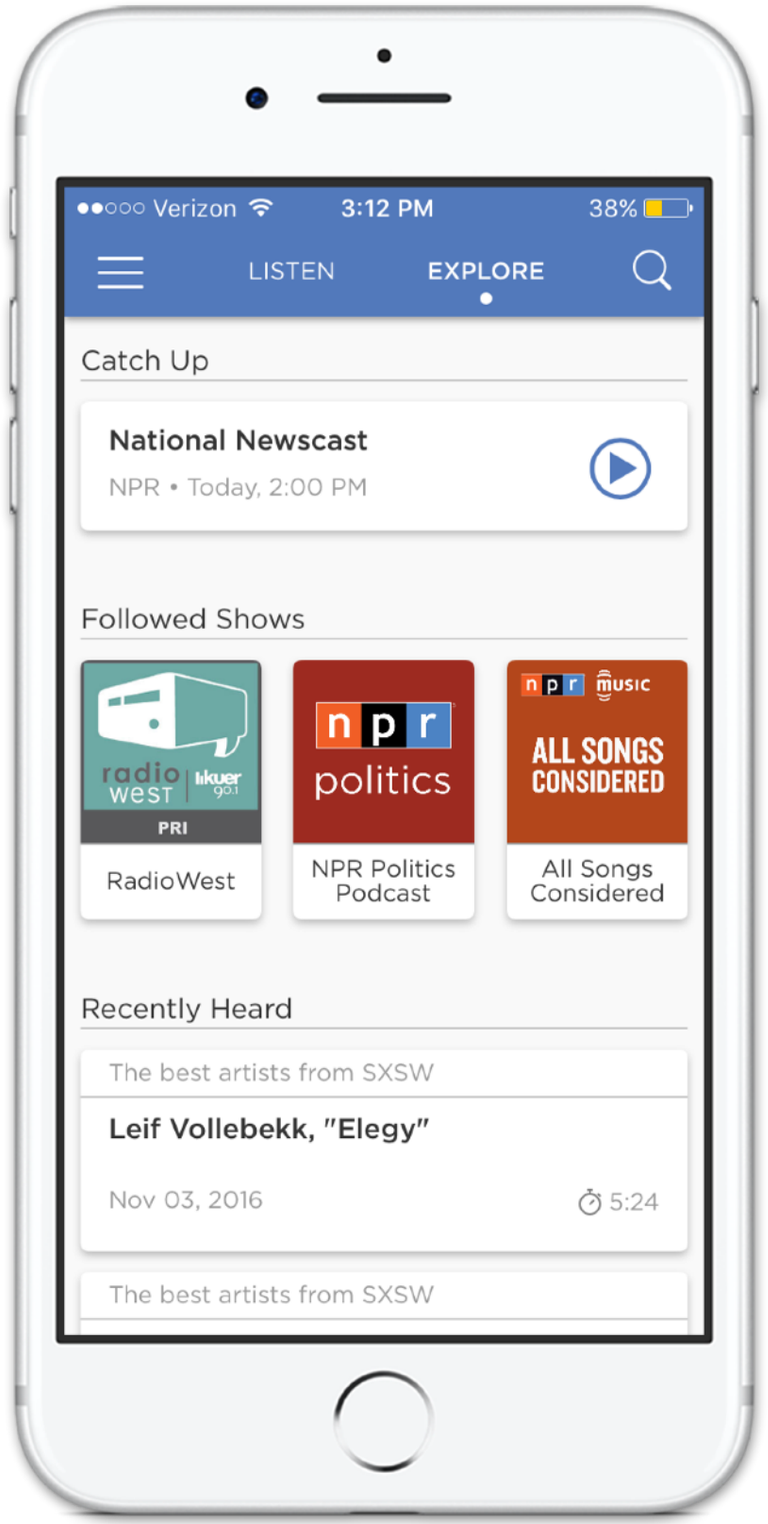


Image : NPR One

Core Data Vocabulary

Managed Object Model (Data Model/Object Graph)

.**xcdatamodeled** file

defines how your data is structured

Persistent Store

contains the actual data stored using Core Data

Core Data Vocabulary

Managed Object

an object (data) in the persistent store

Managed Object Context

the “space” (allocated area of memory) or scratch pad used for editing and saving managed objects.

Core Data Vocabulary

Persistent Store Coordinator

mediator between persistent stores and managed object contexts

Entities and Attributes

Think of Entities as Classes or Objects, and attributes are the properties of those objects.

Example: For an app that stores a list of dog profiles, entities are a array of Dog objects, and Attributes are name, age, fur color, etc.

Core Data Vocabulary

NSPersistentContainer

encapsulates the whole core data stack, which includes:

- NSManagedObjectModel
- NSPersistentStoreCoordinator
- NSManagedObjectContext

Fetch Request

grab a managed object from your persistent store using your context

Core Data Vocabulary (summary)

Data Model - NSManagedObjectModel

PS Coordinator. - NSPersistentStoreCoordinator

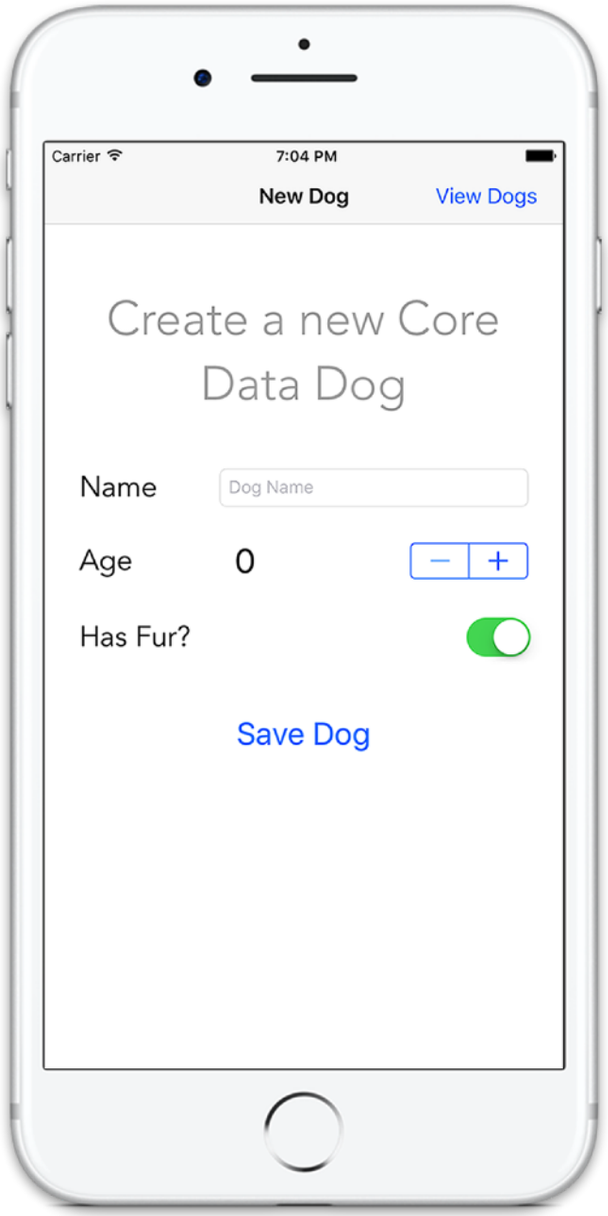
Context - NSManagedObjectContext

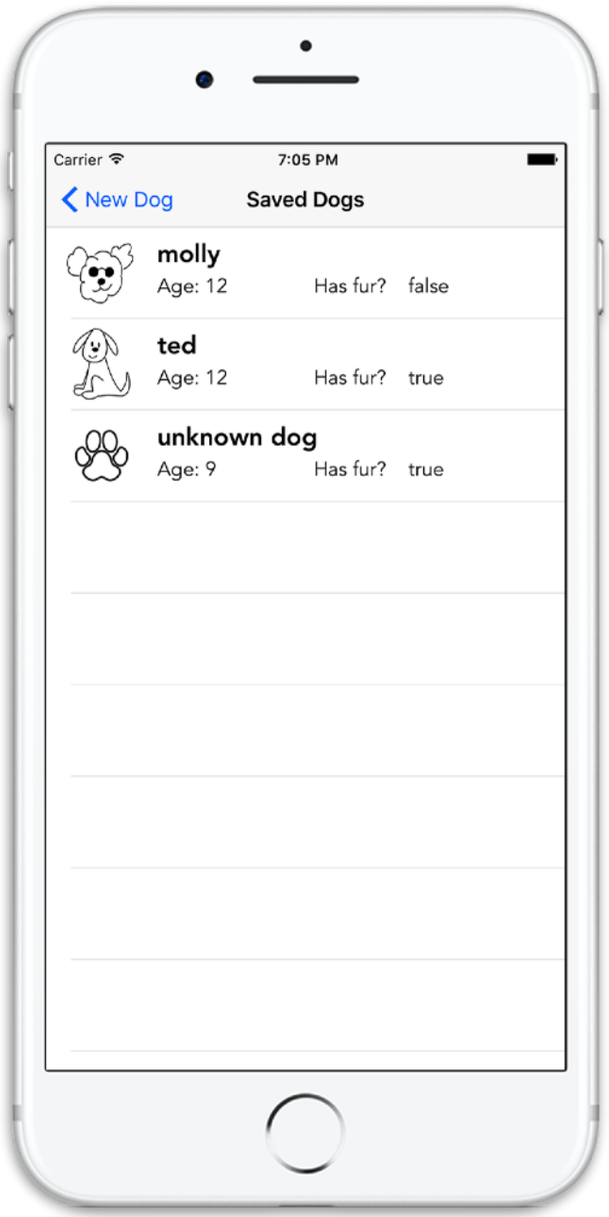
Managed Object - NSManagedObject

Fetch Request - NSFetchRequest

```
func fetch(_ request: NSFetchRequest)
throws -> [Any]
```

Core Data : Today's Example





Allow user to add a name, age, and set whether or not dog has fur.

Using Core Data, save this user data to the user's device, so they can store a list of dogs

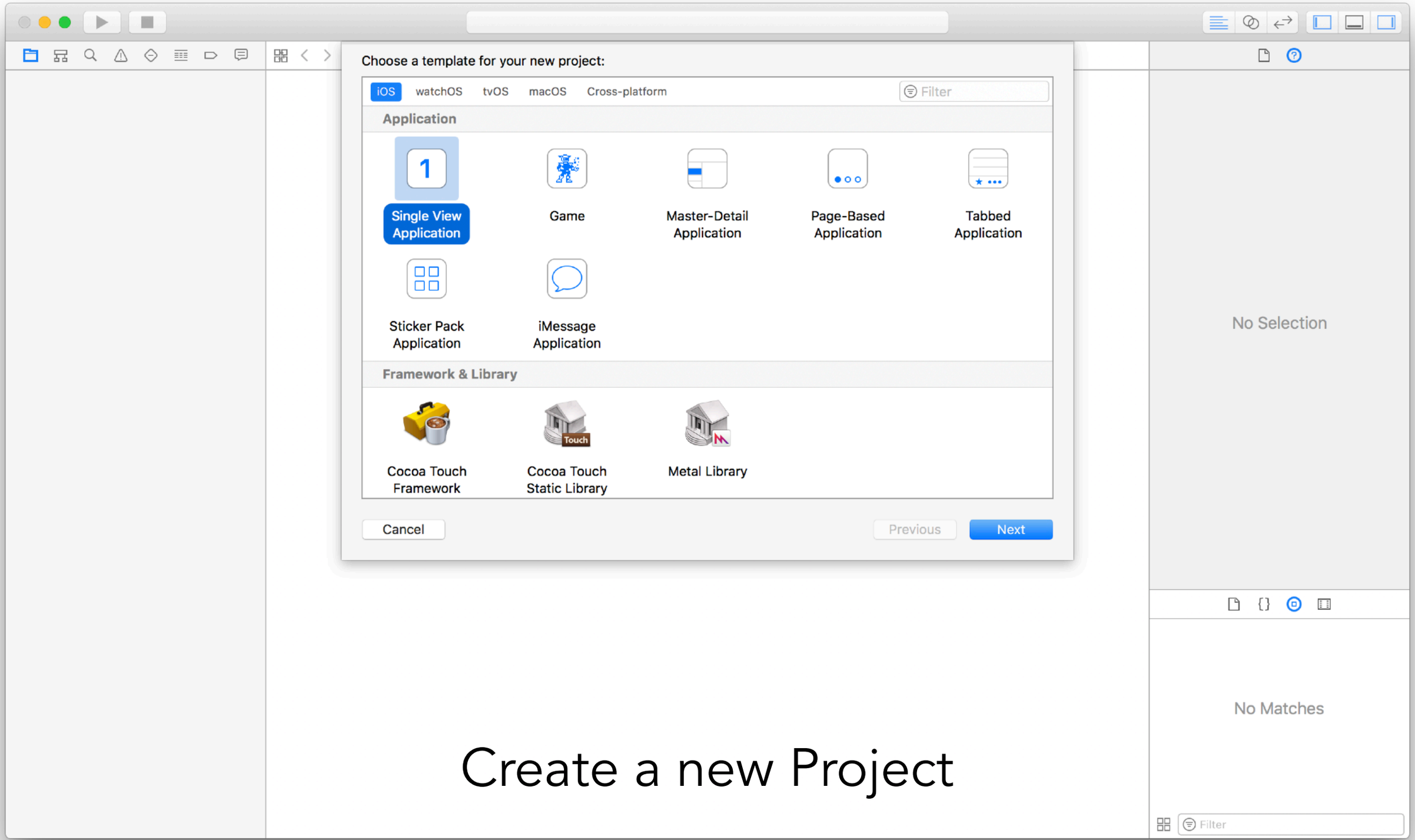
Code available at **github.com/paigeplan/Core-Data-Demo**

Core Data Checklist

1. **Enable Core Data:** Check **“Use Core Data”** when creating a new application
2. **Create an Entity:** for whatever you want to save the state for (i.e. Dog, Person, Profile)
3. **Store Data to Core Data:** Save user input data in an Entity

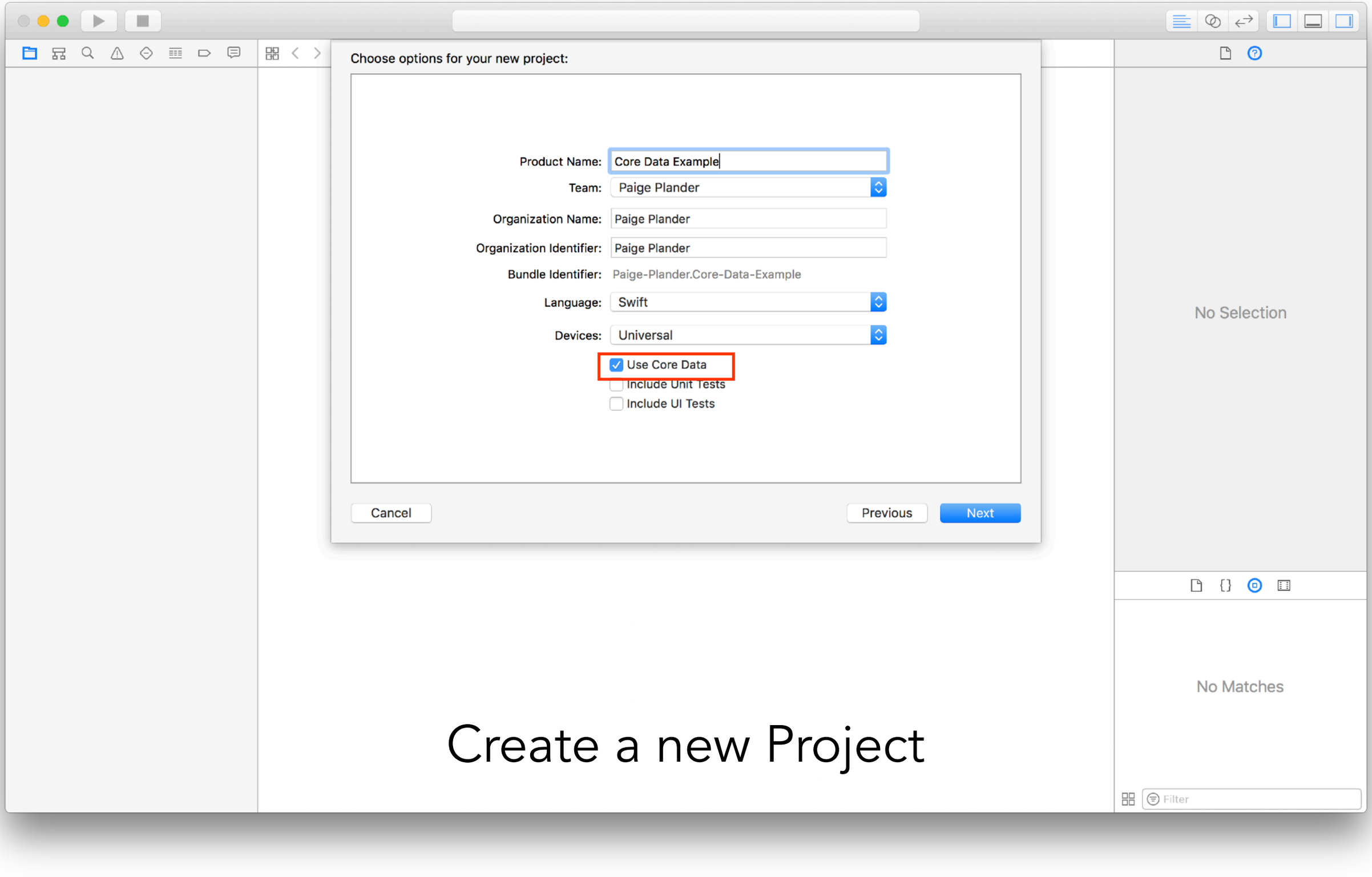
   You’ll need your “scratchpad” `NSManagedObjectContext`, which is found in you app’s `NSPersistentContainer`
4. **Fetch Data from Core Data:** Access stored data using your context via the method `fetchRequest()`

Core Data : Creating an Entity



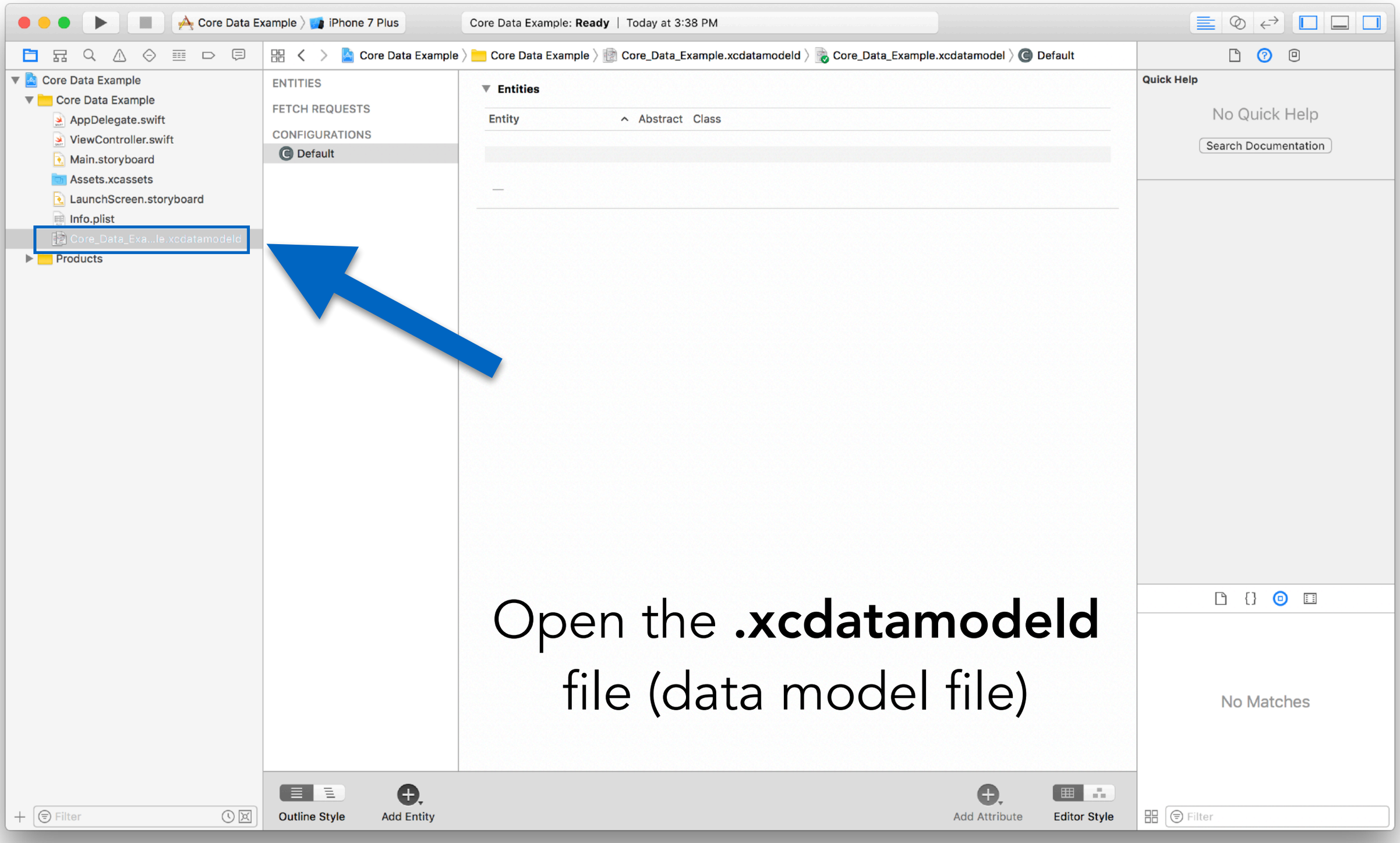
Create a new Project

Core Data : Creating an Entity

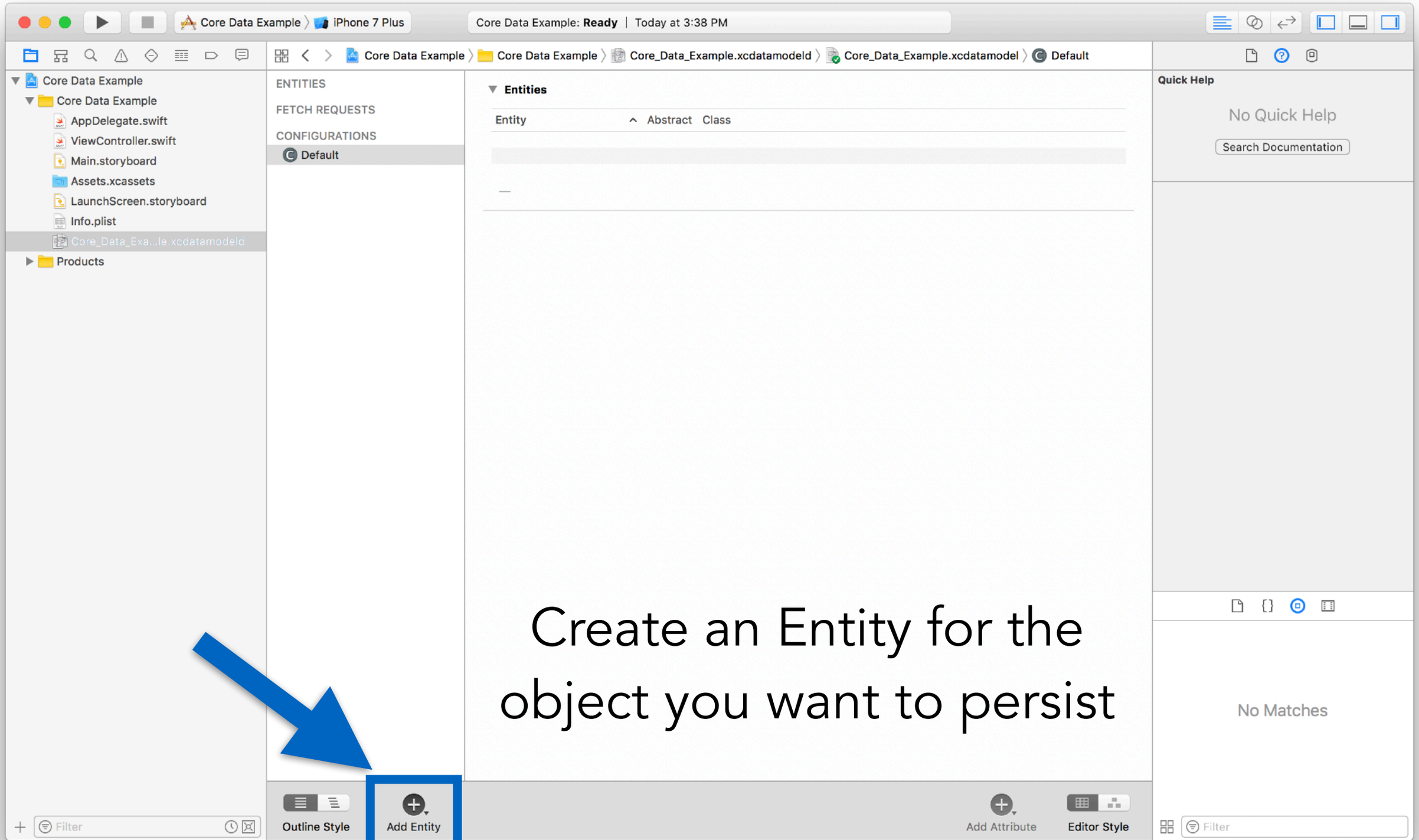


Create a new Project

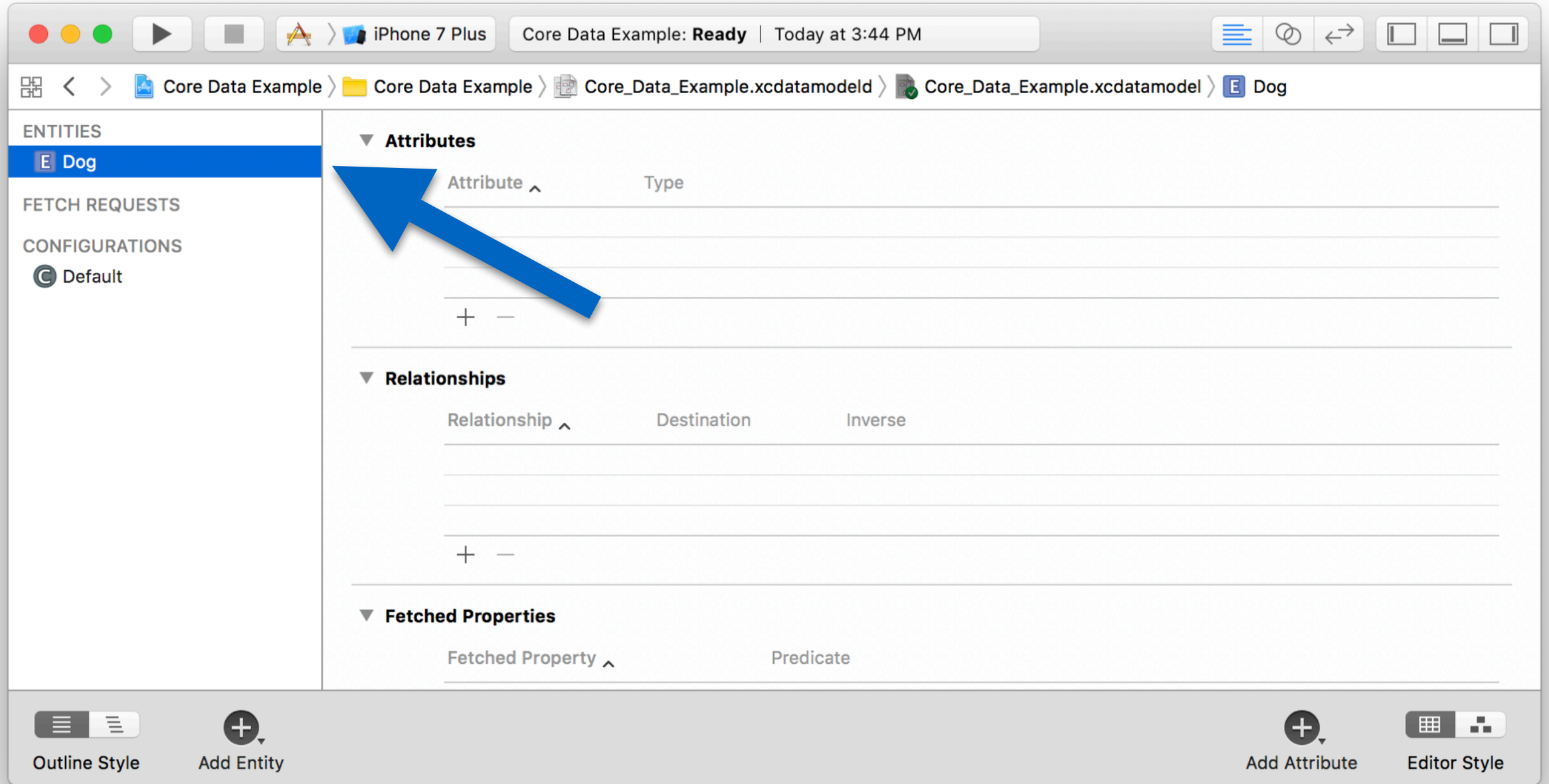
Core Data : Creating an Entity



Core Data : Creating an Entity

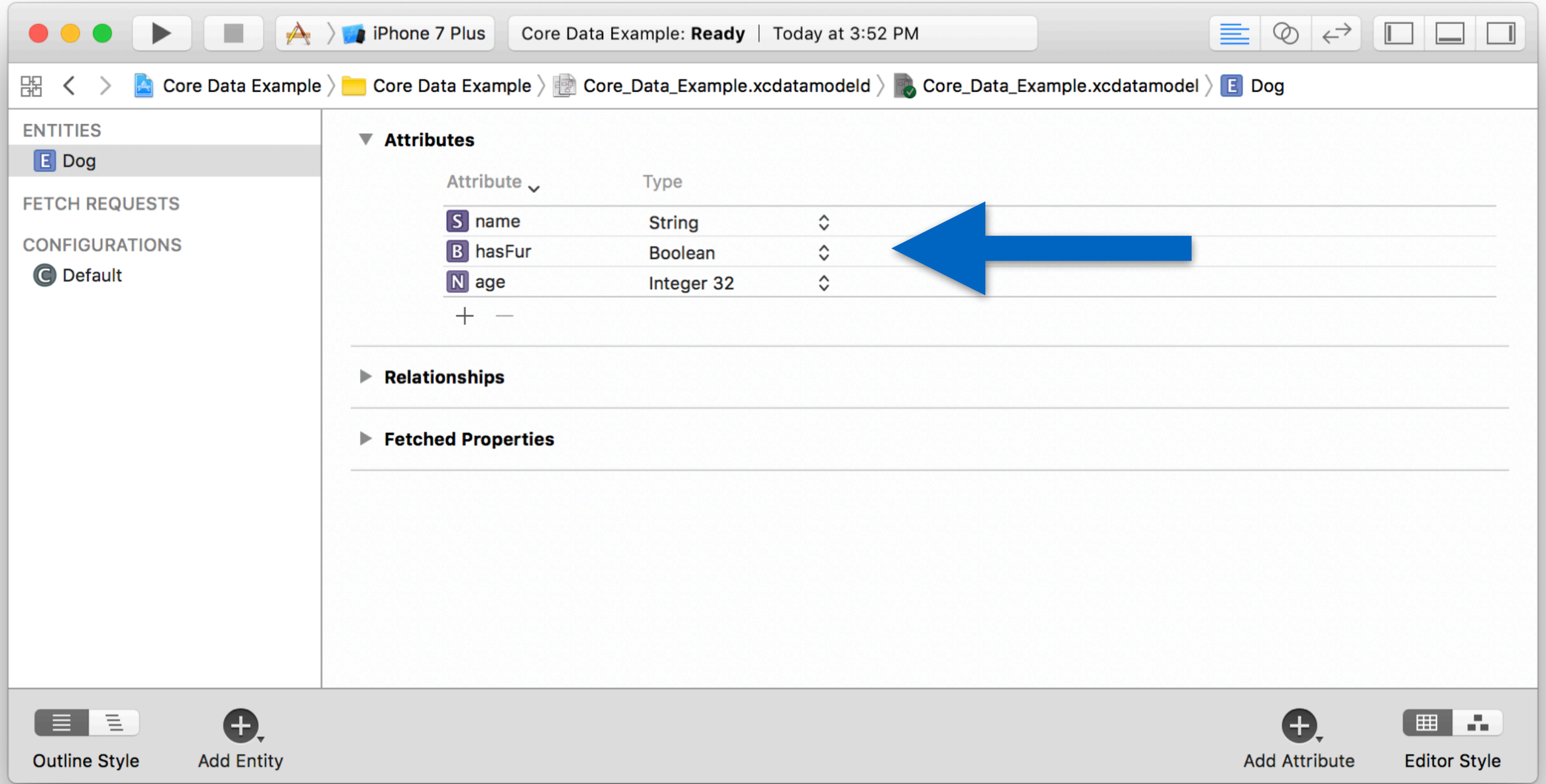


Core Data : Creating an Entity



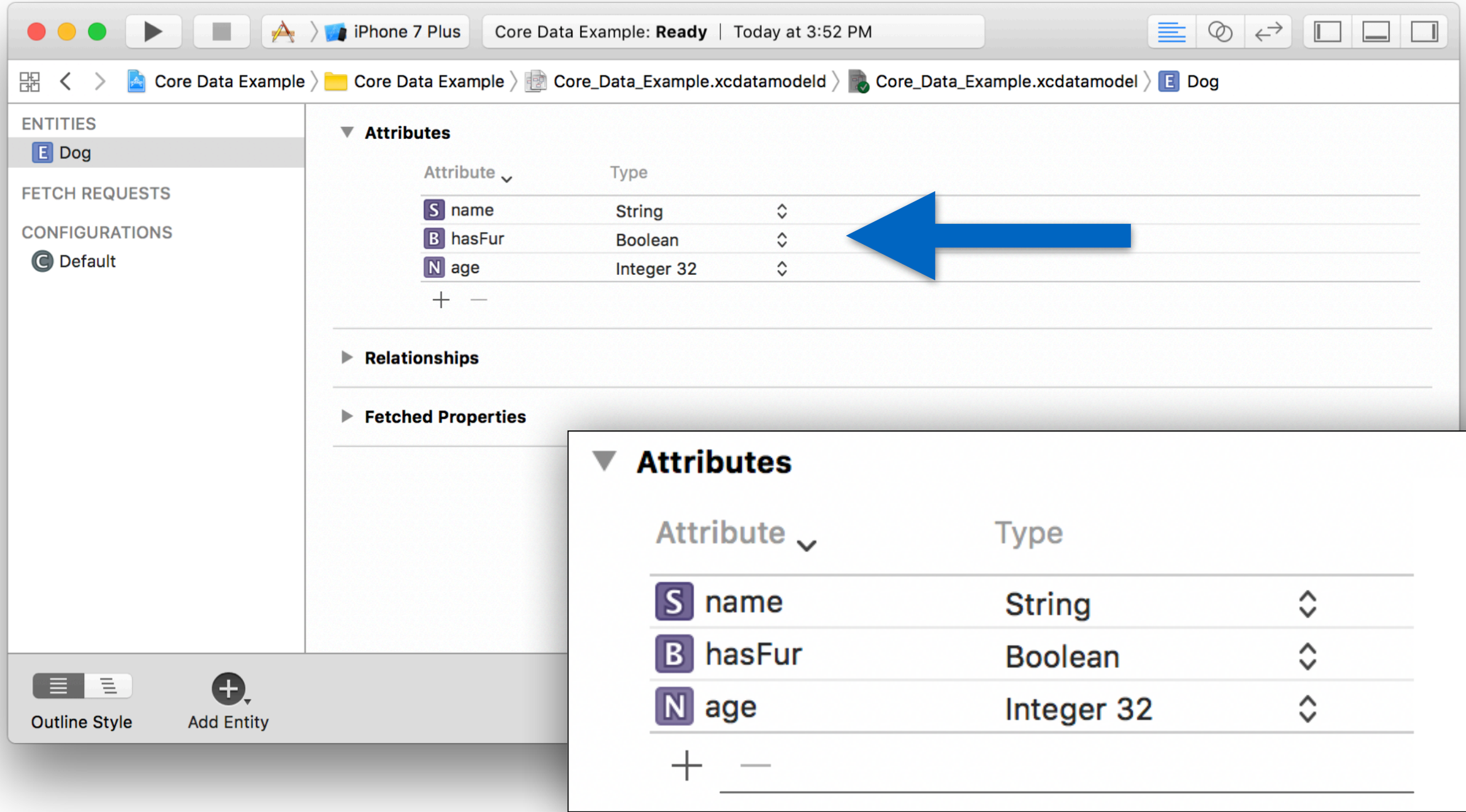
Name your entity (object)

Core Data : Creating an Entity



Add attributes (Model object instance variables)

Core Data : Creating an Entity



Add attributes (Model object instance variables)

Core Data : Storing Data

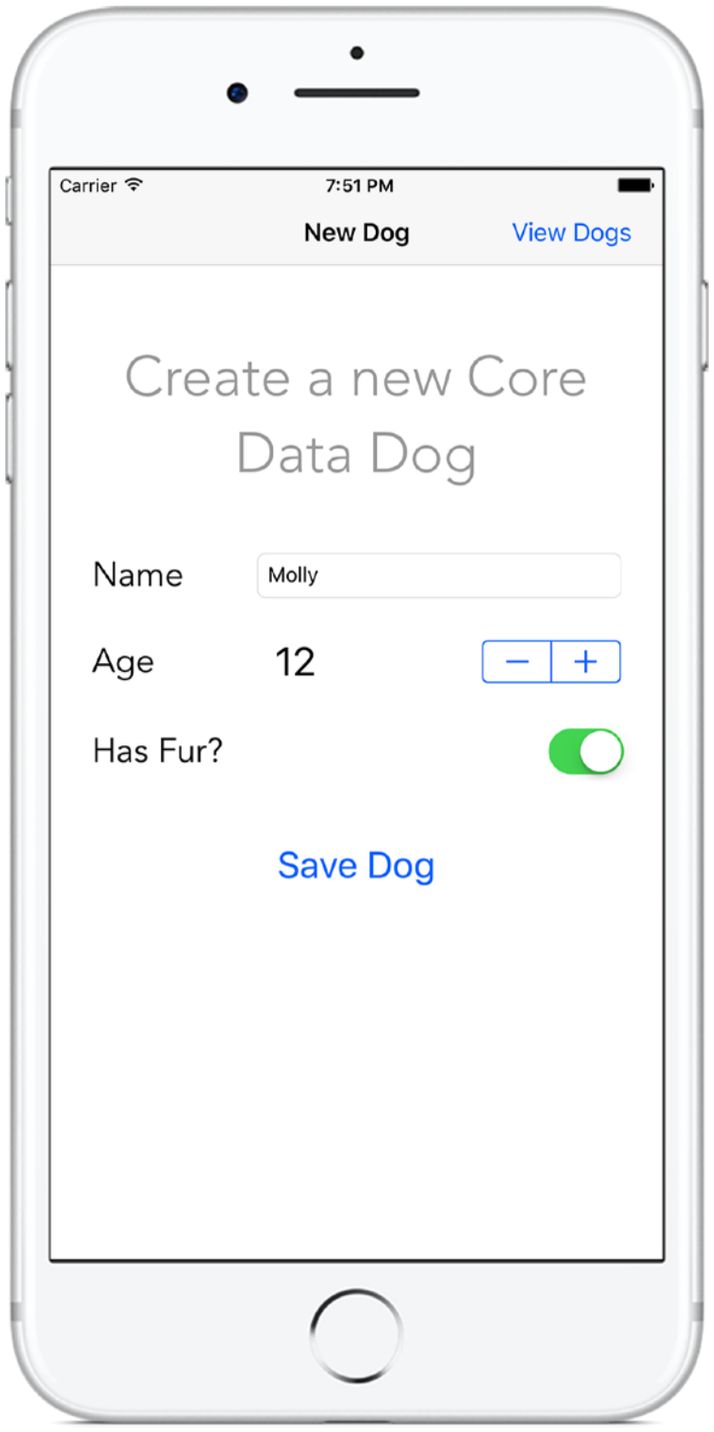
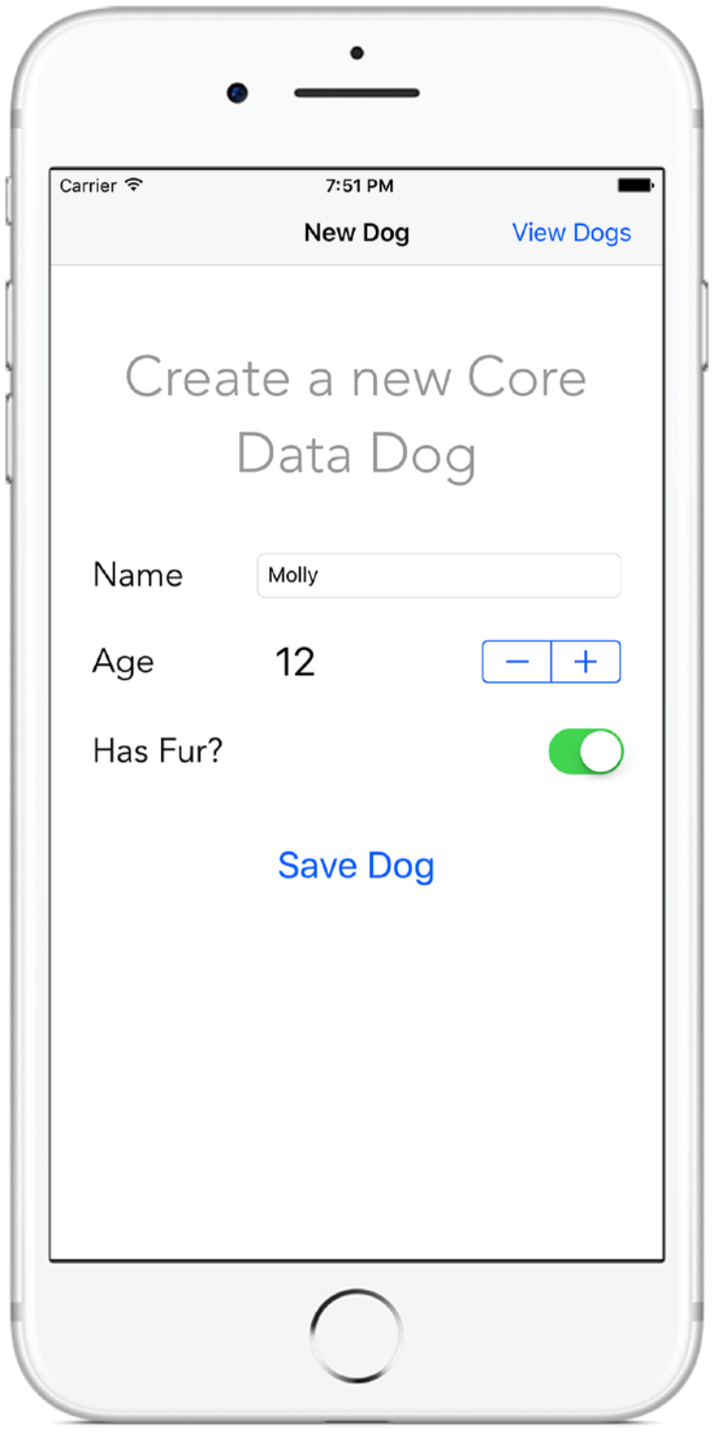


Figure out what user input that you want to save.

In this example, this involves getting the Name TextView's text, Age label text, and switch value

In the next slide, we'll store these values in Core Data

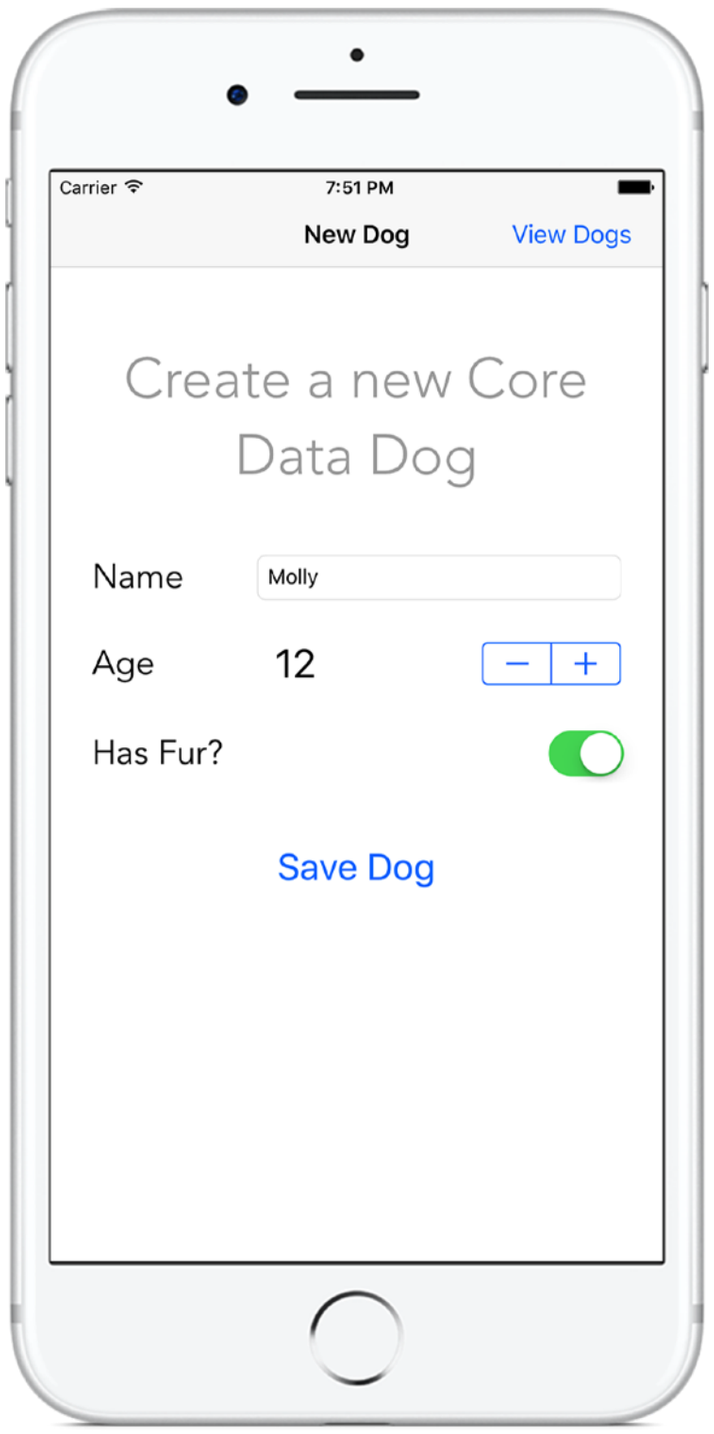
Core Data : Storing Data



To store an object, we will need to:

1. add the new managed object to the object context
2. configure the object within the context
3. commit the changes within the context to save it to disk

Core Data : Storing Data



You do not need to initialize a context - this is already created for you if you have checked the “Use Core Data” box when you created your application

Remember: you can find this context within your `NSPersistentContainer`

Adding the new managed object to the object context

This can be done easily using a handy NSManagedObject initializer

```
init(context: NSManagedObjectContext)
```

Initializes a managed object and inserts it into the specified managed object context.

configure the object within the context and commit changes

To configure: set the properties (attributes) of your NSManagedObject you just initialized

To commit changes: saveContext() in your App Delegate file

Core Data : Storing Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
if let dogName = dogNameTextField.text {
    let dog = Dog(context: context)
    dog.name = dogNameTextField.text
    dog.hasFur = furSwitch.isOn
    dog.age = Int16(ageLabel.text)!
    appDelegate.saveContext()
}
```

Core Data : Storing Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
if let dogName = dogNameTextField.text {
    let dog = Dog(context: context)
    dog.name = dogNameTextField.text
    dog.hasFur = furSwitch.isOn
    dog.age = Int16(ageLabel.text)!
    appDelegate.saveContext()
}
```

First, get a reference to your App Delegate

Core Data : Storing Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
if let dogName = dogNameTextField.text {
    let dog = Dog(context: context)
    dog.name = dogNameTextField.text
    dog.hasFur = furSwitch.isOn
    dog.age = Int16(ageLabel.text)!
    appDelegate.saveContext()
}
```

Get the context from the App Delegate. We need it to save our new Dog Object

Core Data : Storing Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
if let dogName = dogNameTextField.text {
    let dog = Dog(context: context)
    dog.name = dogNameTextField.text
    dog.hasFur = furSwitch.isOn
    dog.age = Int16(ageLabel.text)!
    appDelegate.saveContext()
}
```

Initialize NSManagedObject **Dog** and insert it into **context** from our app delegate.

Core Data : Storing Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
if let dogName = dogNameTextField.text {
    let dog = Dog(context: context)
    dog.name = dogNameTextField.text
    dog.hasFur = furSwitch.isOn
    dog.age = Int16(ageLabel.text)!
    appDelegate.saveContext()
}
```

Set the dog's **attributes**

Core Data : Storing Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
if let dogName = dogNameTextField.text {
    let dog = Dog(context: context)
    dog.name = dogNameTextField.text
    dog.hasFur = furSwitch.isOn
    dog.age = Int16(ageLabel.text)!
    appDelegate.saveContext()
}
```

Save **dog** to Core Data using `saveContext()`

Core Data : Retrieving Data

Now that we can store user data to Core Data, we need a way to retrieve this data so we can display / use it.

To do this we'll need to use our App Delegate and context again

Core Data : Retrieving Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
var dogs: [Dog] = []

func fetchDogsFromCoreData() {
    do {
        dogs = try context.fetch(Dog.fetchRequest())
    }
    catch {
        print(“Fetch failed :( ")
    }
}
```

Core Data : Retrieving Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
var dogs: [Dog] = []

func fetchDogsFromCoreData() {
    do {
        dogs = try context.fetch(Dog.fetchRequest())
    }
    catch {
        print(“Fetch failed :( ")
    }
}
```

Again, get App Delegate and context

Core Data : Retrieving Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
var dogs: [Dog] = []

func fetchDogsFromCoreData() {
    do {
        dogs = try context.fetch(Dog.fetchRequest())
    }
    catch {
        print(“Fetch failed :( ")
    }
}
```

Initialize an array to store your fetched Objects

Core Data : Retrieving Data

```
let appDel = UIApplication.shared.delegate
                                    as! AppDelegate
let context = appDel.persistentContainer.viewContext
var dogs: [Dog] = []

func fetchDogsFromCoreData() {
    do {
        dogs = try context.fetch(Dog.fetchRequest())
    }
    catch {
        print(“Fetch failed :( ")
    }
}
```

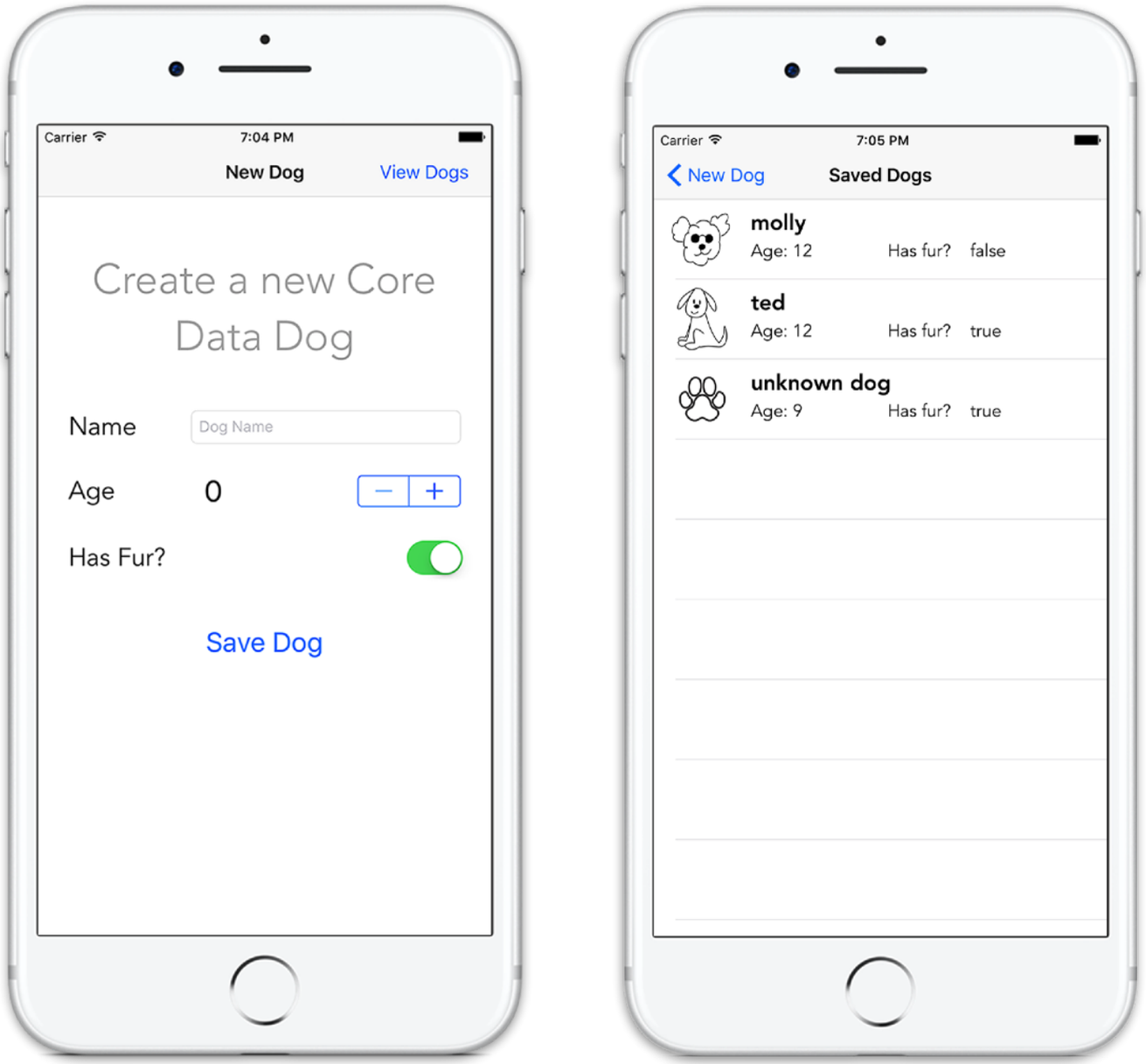
Populate this array with the Objects the user saved

Core Data : Retrieving Data

```
/// Uses the App Delegate's Context to get the dogs
saved to Core Data
func fetchDogsFromCoreData() {
  do {
     let request = NSFetchRequest<NSManagedObject>
                                        (entityName: "Dog")
     // only get 20 objects at a time
     myRequest.fetchBatchSize = 20
     // only give the first 100
     myRequest.fetchLimit = 100
     savedDogs = try context.fetch(myRequest) as! [Dog]
  } catch {
     print("Fetching Dogs from Core Data failed :( ")
  }
}
```

Can also set request limits / batch sizes if dealing with a lot of data

Core Data : Result!



Now the dogs the user has added will now be saved to disc.

We now don’t have to worry about data disappearing when the user force closes app or turns off phone