Contents

1 **Introduction to iOS Development** 7

Introduction 8
Opening Xcode 11
Creating a New Project 12
Xcode Project Files 13
Xcode User Interface 14
Running Your First Project 19
Testing on a Device 21

2 **Swift Playgrounds** 23

Playgrounds Setup 24
Variables 26
Conditional Statements 30
Arrays and Dictionaries 32
Loops 33
Functions 36

3 **User Interaction** 37

Basic User Input 38
Switches 43
Web Views 45
Auto Layout 52
Basic Debugging 55

4 Camera & Photo Library 57

User Interface Setup 58
Writing the Camera Code 61
Writing the Library Code 62
Image Placement 63
saveImage() Method 64
Running the Project 65
Constraints 66

5 Location & Table Views 67

Setting Up a Map View 68
Import & Permission 71
CLLocation Manager 72
Location Update 73
Geocode 74
Table Views 76

6 Firebase: Login & Database 83

Firebase Setup 84
Building the Login UI 88
AppDelegate Setup 92
Register Method 93
Login & Logout 96
Profile Data 100
Login State on Load 106

7 Game Development 107

Game Setup 108
BlockTile Class 112
Model Class 113
Random Number & Populate 115
Set Up Sprites 117
Responsive Tiles 118
Tile Interaction 121
Finding Matches 124
Game Sounds 130
Particle Effects 131
Scoreboard & Finishing Up 133

8 Advanced Swift 137

ARKit 138
Machine Learning 143
API Calls 153
Core Data 158
Local Notifications 166
Google AdMob 171

9 Submitting Your Apps 175

Apple Developer Account 176
Code Signing 177
App ID 179
App Store Connect 180
Adding an App 181
Upload App Using Xcode 182
Prepare & Submit 184

Index 187

Preface

This book has allowed me to share my knowledge of the Swift programming language with you. I really hope the examples in this book will help you on your way to publishing your own apps on the Apple App Store. It really is an exciting time to be an app developer, as there is a lot of opportunity out there to get your own product into the hands of so many people. This is also a great chance to enhance your skillset in a language that Apple will be using for development across all their devices for the foreseeable future.

Color coding

All code written inside this book has been color-coded to help you understand each example better.

Swift components are colored in blue; numerical and string values are colored in black; and programmer-specified names are colored in red. This includes all chosen function and method names. Below is an example:

```
//Here is a comment
var message = "Good luck with Swift!"
print(message)
```

Source code

All examples featured in this book are available to download in a single ZIP file, which you can get by following these simple steps:

1. Go to **http://ineasysteps.com/resource-centre/downloads/**
2. Find **Swift Programming in easy steps** and click on the hyperlink entitled **All code examples** to start downloading the file
3. Extract the archive contents to the location of your choosing on your computer

If you don't achieve the result illustrated in any example, simply compare your code to that in the original example files you've downloaded to discover where you went wrong.

I hope you enjoy learning how simple yet powerful Swift is as a programming language. I look forward to seeing what apps you come up with. Good luck!

1 Introduction to iOS Development

This first chapter will introduce you to iOS development, and getting to grips with Xcode. You will also get the chance to set up your own iOS device for development.

8 Introduction

11 Opening Xcode

12 Creating a New Project

13 Xcode Project Files

14 Xcode User Interface

19 Running Your First Project

21 Testing on a Device

Introduction

You can get all the source code for this book at https://ineasysteps.com/resource-centre/downloads/

Apps have taken over the world. Most of us use them every day for things like Facebook, Instagram, WhatsApp, and even games like Angry Birds. We have them on our phones, tablets and even our TVs, and they are not going away anytime soon.

This book will teach you how to build iOS apps from scratch using Swift 5. You won't need any prior programming knowledge at all, and I will walk you through the process of UI (user interface) design, coding, all the way to publishing apps to the Apple App Store.

What is Swift?

Swift allows you to build applications for iPhone, iPad, Apple Watch, Apple TV, and Mac. The first version of Swift was first introduced in 2014, and the current version is Swift 5, which was released in March 2019.

Why should I learn Swift and build for iOS?

Swift is the future of iOS development on all platforms. It's very easy to learn; it's more readable than most languages; it's very safe to use due to the excellent error handling model; and the coding environment is more engaging compared with other languages. Apple announced in early 2018 that there are over 1.3 billion active iOS devices worldwide, which means there is a huge customer base, plus it's very simple to distribute your app through the App Store, which all iOS users use to download their apps.

What will I need?

Xcode is free from the Mac App Store.

- **Mac computer**: This can be a MacBook Air, MacBook Pro, iMac or even a Mac Mini.
- **Xcode**: This is the software you will need to write apps in Swift. Xcode version 10.2 or higher is required for Swift 5.
- **Apple ID**: You will need this to download Xcode.
- **iPhone/iPad**: For some of the chapters, you will need an iPhone or iPad to test your apps.

If you plan to publish your app(s) you will need an Apple Developer account and a subscription to the program, which currently costs $99 annually (see page 176 for more details).

Apple ID

First, you will need to create an Apple ID at **https://appleid.apple.com/account**
It is free to create an account. If you have downloaded apps before, you will already have one of these.

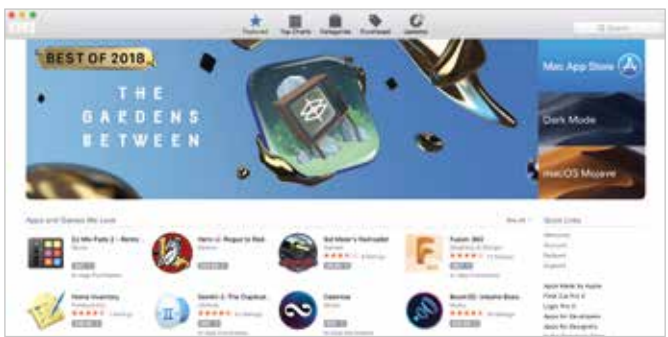
The App Store is a great place to find approved software for your Mac.

Downloading Xcode

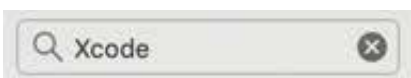
Once you have set up an Apple ID, you will need to install Xcode on your machine.

Xcode will only run on a Mac computer.

1 Open the App Store on your Mac



2 Use the Search bar located in the top-right corner and search for "Xcode"



3 Select the Xcode logo

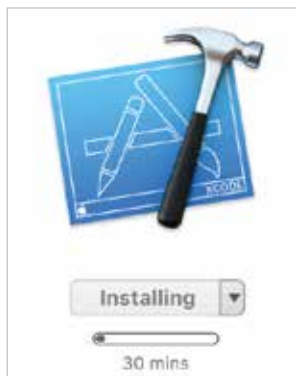
...cont'd

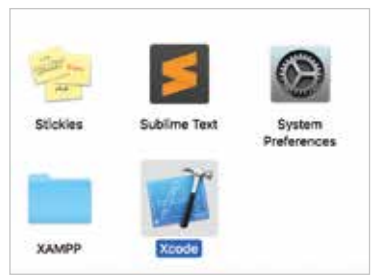
If you have an Apple Developer account (see page 176), you can also download older and beta versions of Xcode at https://developer.apple.com

4 Tap on the **Get** button underneath the icon. This will then change the text of the button to **Install App**. Press the button for a second time

5 After that, it will prompt you to enter your Apple ID and password

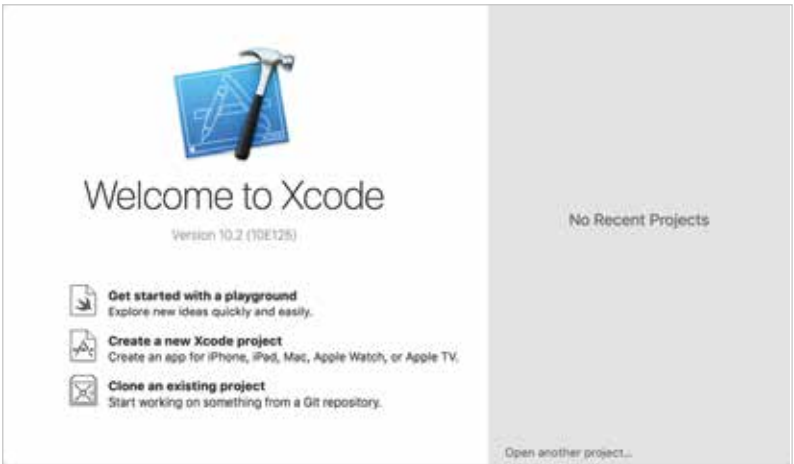
6 Once you enter your credentials, Xcode will begin downloading. You should be able to find it inside your Applications folder once it has finished downloading





Opening Xcode

1 Double-click the Xcode icon. You will be greeted with the following welcome window:



This is the Xcode welcome window. It has two panes. The left-hand pane displays three options:

Getting started with a playground:
This option allows you to create a playground file. A playground will allow the user to write a few lines of code, execute it and see the results in the adjacent window. Playgrounds is an interactive development and testing environment. This will be covered in Chapter 2.

Create a new Xcode project:
This option allows you to create a new Xcode project from a list of available templates. You will be selecting this option to start off, as it will allow you to explore Xcode's development environment.

Clone an existing project:
This option allows you to download and open an existing project from an SCM repository like SVN, Git; etc. A repository is a place where you can store project files.

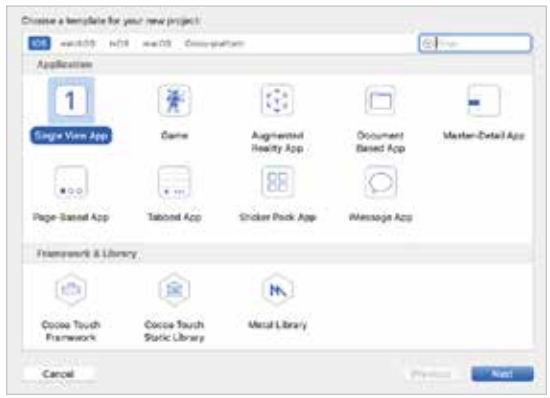
The right-hand pane displays the list of recently opened projects. This should be empty at the moment. You also have the option to open another project, if there is a particular project that isn't listed under the most recent ones.

Version control like Git is a great way to record changes to files over time. This way you can refer back to an older version if something goes wrong.

Creating a New Project

1. Click the **Create a new Xcode project** option on the "Welcome to Xcode" window

2. You will now be asked what type of template you want to create. There are many different choices, but for now just select **Single View App** and then click **Next**



At the moment, it doesn't matter what name you use as a product name, as you are just walking through Xcode. Your organization name is usually your full name or company name.

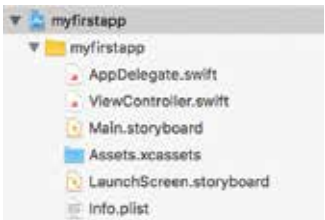
3. Give your app a product name, and choose an organization name. Make sure Swift is selected as the language and leave the bottom three check/tick boxes deselected. The Bundle Identifier will be your Organization Identifier (your domain backwards), and product name to make up **com.yourdomain.productname**; however, you can change this if you wish. Click **Next** to continue

Make sure that Swift is always selected as the language.

4. Click **Create** to save your project

Xcode Project Files

When the project has been created, Xcode will open the project inside a new workspace window. You will learn more about the user interface (UI) on pages 14-18. On the left-hand side, you will see a list of project files, which are explained below:



Don't worry too much about understanding each file right now, as they will make more sense as you cover various projects throughout the book.

- **AppDelegate.swift**

  This file is the starting point of your application. It takes care of initializing the app and displaying the first view of your app. It also helps deal with events that happen when the app goes into the background, or becomes active again.

- **ViewController.swift**

  The ViewController file is responsible for setting up the main view inside the **Main.storyboard** file and displaying it on the screen to the user. There can be multiple ViewController files where each file will have a corresponding UI in the **Main.storyboard**. The ViewController file is where the user writes code, and it acts as the Controller in the MVC architecture.

- **Main.storyboard**

  The Main.storyboard file contains the UI of your application. It corresponds to View in the MVC architecture. You will develop the app's UI in the Main.storyboard file.

- **Assets.xcassets**

  The Assets.xcassets is a folder that contains all the images that will be used inside your app.

- **LaunchScreen.storyboard**

  The LaunchScreen.storyboard file contains the Launch (Splash) screen of the app. The Launch screen is shown on the launch of the app before showing the first view of the app.

- **Info.plist**

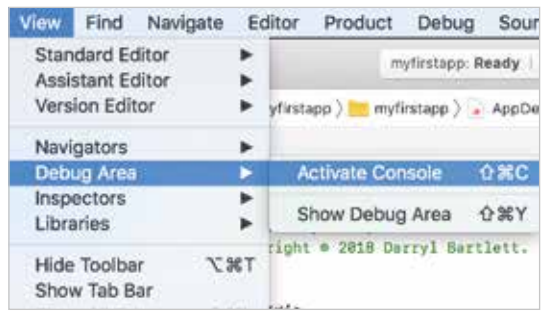
  The Info.plist file is a key-value pair-based XML file that contains the run-time configuration of your app.

MVC stands for Model View Controller. The Model is responsible for any data-related logic. The View is used for anything related to the user interface. The Controller acts as the middle man for the UI and the Model.

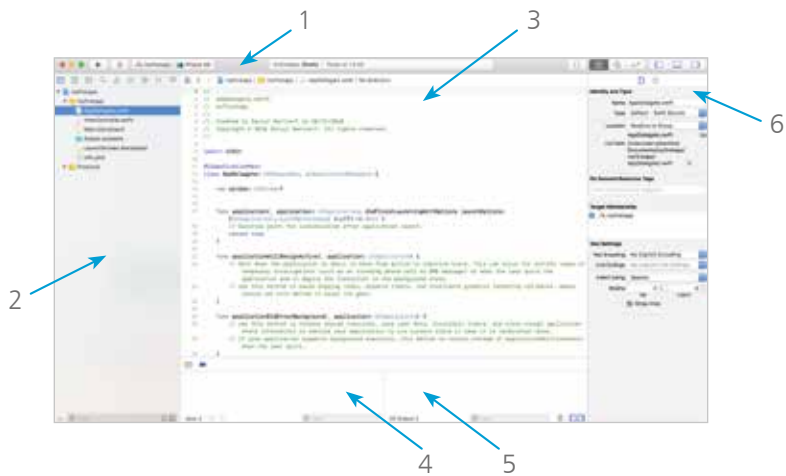
Xcode User Interface

Try not to get too overwhelmed by the user interface. You will get used it to more as you work through projects.

You are now going to have a look at the various sections of the Xcode development environment. Start by clicking the **AppDelegate.swift** file in the left-hand pane. In the Xcode menu go to **View** > **Debug Area** > **Activate Console**.



Your main window should look like the screenshot below:



The Xcode user interface is broken down into six main sections, which are numerically labeled above.

- **Toolbar (1)**
- **Navigator (2)**
- **Editor (3)**
- **Debug Area (4 & 5)**
- **Utility Area (6)**

Toolbar (1)

These are the Run and Stop buttons. The **Run** button will build and run your app in the iOS simulator. The **Stop** button will stop the execution of your application and return you to Xcode.

The left-hand button (**myfirstapp**) indicates which build target you want to run. The build target is basically a specific configuration of your app. Your app can have multiple build targets so that you can run the same code with multiple configurations.

The right-hand button (**iPhone XR**) allows you to choose if you want to run the app under the iPhone or iPad simulator or an actual iOS device, if you have it connected to your Mac.

Because you currently have a Swift file open (**AppDelegate.swift**), you will see the **Snippet** button, which allows you to add a code snippet or template to the Editor.

If a Storyboard file was open you would see the **Object Library** button that brings up a list of all the iOS objects. You would need this if you wanted to drag objects (Buttons, Labels; etc.) onto the Storyboard. You will see this in more detail on page 19.

The **Standard Editor** button (left) is selected by default. It will keep your setup the way it is now. The **Assistant Editor** button (center) will split your **Editor** into two. Half of the Editor will become the Storyboard, and the other half will be the **Code Editor**. You may also have two Code Editors side by side.

It's always important to test your app on a device as well as a simulator, as you can never truly tell how an app will react until it's running on an actual device.

...cont'd

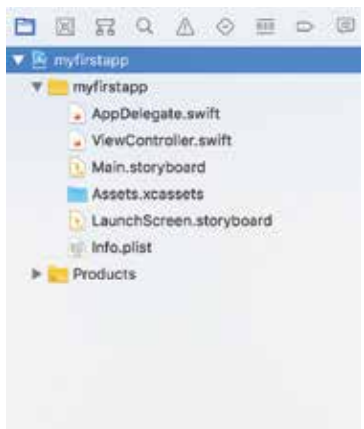
It can be difficult to fit everything on screen at once. Only keep open the areas that you need.

The **Version Editor** button (on the right in the last image on page 15) makes it easier to compare two files. It splits the pane to show two different versions of the same file.

The three buttons at the end of the toolbar will hide and show various sections inside of Xcode. The left-hand button will show or hide the **Navigator** – this is automatically shown by default. The center button will show or hide the Debug Area – by default this is hidden when opening for the first time. The right-hand button will show or hide the Utility Area.

Navigator (2)

There are various sections to the Navigator; by default, the **Project Navigator** will be the one selected when you first start a project with Xcode. The Project Navigator is an area where you'll see all the files associated with your project. As a beginner, this is really the only part of the Navigator you will need at this point.



Within the Project Navigator, you can also create groups to organize your files in. The yellow folder (myfirstapp) shown in the screenshot (left) is a group. You can right-click inside the Project Navigator to create new files or add existing files to your project.

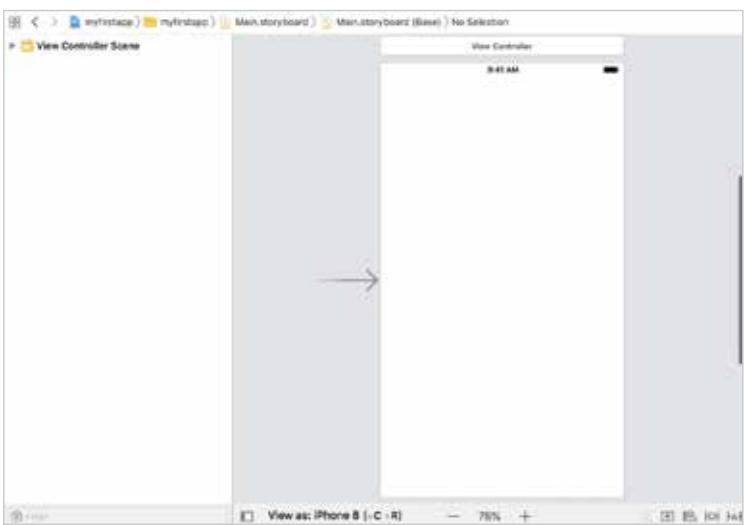
Alternatively, you can also drag folders or files from your computer directly onto the Project Navigator.

If you end up with a rather large project, then you can click the Search icon on the Navigator's toolbar and type in keywords to help find a particular file or section of code.

Editor (3)

The **Editor** is the most important area, as the majority of your development work will be done here. When you click a **.swift** file, it will open in the **Code Editor**. It will look similar to the screenshot below:

If you open up a **.storyboard** file, it will display the app's Storyboard. This is where you will create the views for your app. It will look very similar to the screenshot below:



The Storyboard will only have one view when you first open it

...cont'd

The Debug Area also displays any error messages if something inside your code fails.

Debug Area (4 & 5)

As you can see in the screenshot below, the **Debug Area** is made up of two different sections.

The left-hand pane is a great way of reading particular variable values at certain times using breakpoints.

The right-hand pane is the **Console Area**. You might use this if there are certain messages or values you want to print to the console.

You will learn more about breakpoints and debugging on pages 55-56.

Utility Area (6)

The Utility Area will look different depending on the type of object, or file, you have selected.

This will mainly be used when working with the Storyboard.

When dragging UI objects onto the Storyboard, you can view and change attributes for the particular object you are working with.

For example, for a Label, this section will allow you to change the text, font size; etc.

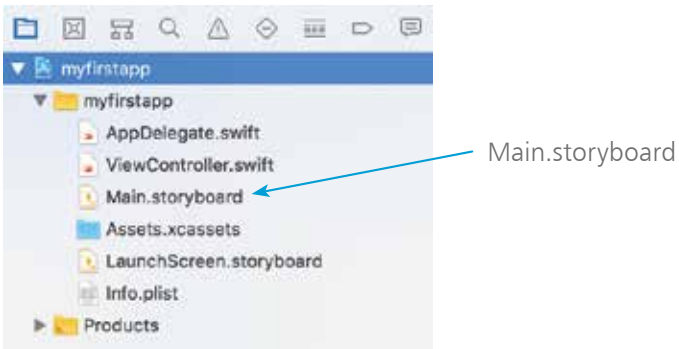
Essential Xcode shortcuts

Cmd + ,	Open Xcode Preferences
Cmd + B	Build the project
Cmd + S	Save project
Cmd + C	Copy
Cmd + P	Paste
Cmd + K	Clear console
**Cmd + **	Add breakpoint
Cmd + R	Run the project on the iOS simulator or device
Shift + Cmd + 0	Open documentation
Shift + Cmd + F	This will bring up the Search box in the Project Navigator

Running Your First Project

Now that you have a better knowledge of Xcode, it's time to build and run your first project! You are going to work with the project that you just created, and add a simple Label to the Storyboard with a message that says "My first app!".

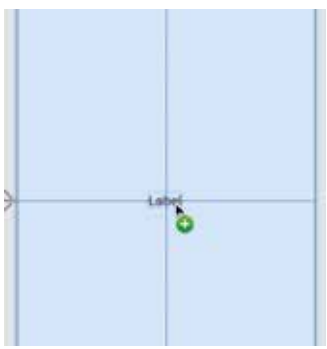
1 Open **Main.storyboard** from the Project Navigator



2 Click the **Object Library** button in the toolbar

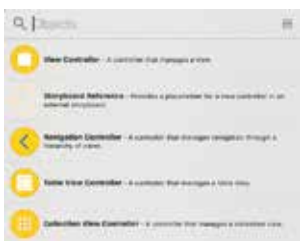
3 This will bring up the Object Library. Type "Label" into the Search box at the top

4 Drag the Label onto the view. Make sure it's centered, as shown in the screenshot (right)



5 Double-click on the Label

You can either search or scroll through the list inside the Object Library.

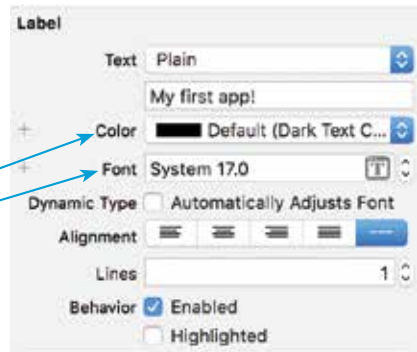


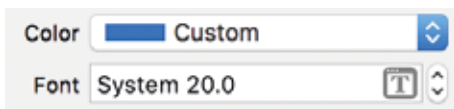
...cont'd

6 Type "My first app!"

Inside the Utility Area, you will see a list of options for your Label:

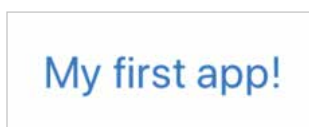
7 Change the color to dark blue and the font size to 20.0

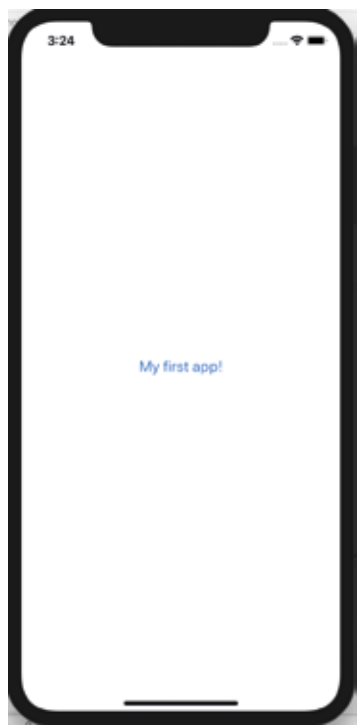




You can also change the appearance of the view by clicking on it, and then changing the attributes inside the Utility Area – just like you did with the Label.

8 Press the **Run** button to run the app in the simulator





There might be a slight delay when starting the simulator.

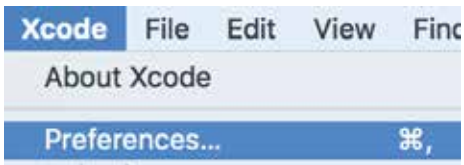
9 To stop running the app, just press the **Stop** button

Testing on a Device

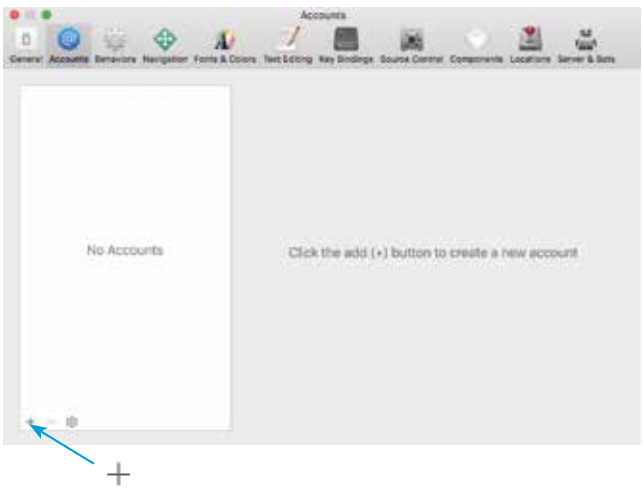
You will need to test out your apps on a physical device for a several projects throughout the book, so this is a good time to walk through the setup process.

1. Connect your iPhone or iPad to your Mac using USB

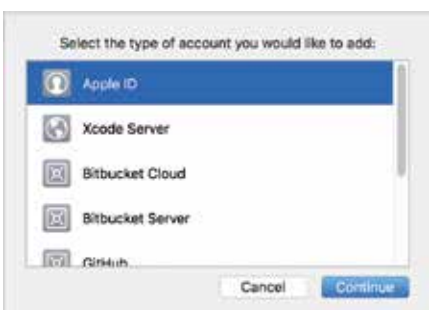
2. From the top menu inside Xcode, select **Preferences...**



3. Inside the Accounts tab, click the + button



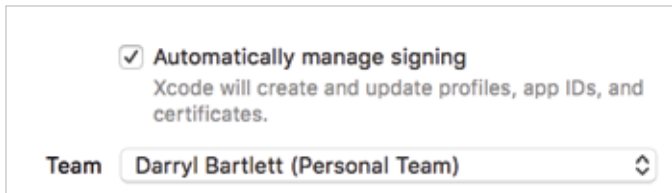
4. Select **Apple ID** from the drop-down and click **Continue**



5. Sign in with your Apple ID and password

6. Select the top file in the Project Navigator

7. Select your name from the **Team** drop-down under **Signing**. Make sure **Automatically manage signing** is also ticked



Running the app

1. Select your iOS device in the toolbar

If you try to run the app now, you will get an error message saying that you need to trust the developer app certificate on your device.

2. Go to **General** > **Device Management** on your iOS device and select your developer profile

3. Click the **Trust** button

4. Click the **Run** button to run the app on your iOS device

Once you trust the developer profile on your device you will be able to run all other apps that you build without having to run through Steps 2 and 3 again.