OS X Setup for Embarcadero Development Tools

Graeme Chandler

OS X Setup
for
Embarcadero
Development Tools

Graeme Chandler

From the WYN (What You Need) Series

OS X Setup for Embarcadero Development Tools

Copyright © 2013 by Graeme Chandler

All Rights Reserved

This material is subject to copyright. All rights are reserved by the author, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software or by similar or dissimilar methodology not known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for the exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the publisher's location, in its current version, and permission for use must always be obtained from the author.

Some of this material is drawn from sources on the Internet, Embarcadero documentation and newsgroups. Appropriate credit will be given where possible. Please accept the apologies of the author if appropriate credit has not been provided.

Trademark names, logos and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo or image, we use the names, logos and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use, in this publication, of trade names, trademarks, service marks and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are the subject of proprietary rights.

The information is provided as-is and is, to the best of the author's knowledge and experience, correct at the time of writing.

This material should not be regarded as exhaustive or be relied upon to guarantee brilliant, bug free and dependable software. That is what your experience and testing are for. In other words, Coogara Consulting cannot be held responsible for your use of this material or the accompanying source code.

Acknowledgements

Special thanks to Jan, Ernie and Craig for their encouragement and assistance with proofing. Their efforts make this a better book. Thanks to Hugo and Kasper. As cats, they are a calming influence when things aren't going well.

OS X Setup
for
Embarcadero
Development Tools

Table of Contents

About the Author **1**
Also by Graeme Chandler 1
Introduction **3**
A note about spelling 3
Who is this book for? 3
The structure of this book 3
Book format 4
Training and Consulting 4
Third Party Products 4
Notes 4
1. Preparing for iOS Development **5**
Connecting to a Mac Remotely 5
Obtaining the Right Version of Xcode 8
Xcode 4.6.3 Installation 10
PhoneGap Support for HTML5 Builder 12
Obtaining the PhoneGap Installer 12
Installing PhoneGap 13
Connecting an iOS Device to your Mac 15
Signing up for the iOS Developer Program 16
Provisioning 17
Using Xcode 26
The Organiser 26
Navigators 29
Project Summary 31
Project Info/Settings 32
Accessing your Windows projects from a Mac 33
Simplifying Access to a Windows Share 37
2. Apple App Store Submission **39**
App Store Review Guidelines 39
iTunes Connect 39
Contracts 40
Distribution Provisioning 42
Preparing for Application Submission 45
Company Name 45
Application Information 46
Availability and Pricing Tier 46
Product Version 47
Product Rating 47
Metadata 48
EULA 48
Uploads 48
Large App Icon 49
iPhone and iPhone Touch Screenshots 49
iPad Screenshots 49
Review Time 51
Index **53**

About the Author

Graeme Chandler has been a software developer since 1980 with experience in over a dozen languages including Fortran, Assembler, Pick, C#, Java and Delphi.

As a trainer, he is respected for his technical knowledge and practical experience. From 2001 to 2003 he provided Delphi training for Borland Australia. In 2003 he began developing his own training material as Coogara Consulting, training in Delphi, C# and Delphi Prism. People have come from as far away as the Philippines to be trained.

When not working, he makes pens and other wood products (www.strikingwood.com) and enjoys the country life in regional Victoria, Australia, with his wife and cats.

You can contact Graeme via the Contact page at http://www.coogara.com.au

Also by Graeme Chandler

FireMonkey Development for iOS and OS X with Delphi XE2

Application Development with HTML5 Builder

Introduction

I have written books for two of Embarcadero's mobile development tools: Delphi XE2 FireMonkey and HTML5 Builder. Since both of these require the use of a Mac, and much of the setup is common to each, I have decided to provide this small book to cover the steps involved in using a Mac and the necessary steps on the Apple Developer site and iTunes Connect. The method of submission of apps is different for each tool and is covered separately in the books concerning those tools.

Even in terms of setup, there are some aspects that are peculiar to a specific development tool. Where possible I will cover those differences here. Where necessary, those differences will be covered in more detail in the individual books.

Note that, over time, the interfaces on Apple's sites change. Some of the steps on the Apple Developer site and iTunes Connect may now appear different. The required steps, however, are still basically the same.

A note about spelling

I am Australian and prefer to spell in the British style. However, for consistency with the spelling of reserved words and property names in HTML5 Builder and Delphi, "colour" will be spelled without a "u" and "centre" will be spelled "center". No doubt there are other examples of this.

Who is this book for?

This book applies to Delphi FireMonkey XE4 or later and HTML5 Builder XE3 or later.

If you are new to mobile development and have not set up your Mac then this book is for you. It will guide you through the Apple sign-up process, installation of Xcode, device provisioning and key generation. Finally it will help you install the necessary tools for Delphi FireMonkey development and HTML5 Builder Development.

The structure of this book

This book is divided into two main parts:

Chapter 1, Preparing for iOS Development, provides an introduction to what you need to get started. It introduces you to the resources required from Apple and takes you through the steps to set up your Mac and Windows environments for iOS development.

Chapter 2, Apple App Store Submission, covers the steps involved in submitting an app to the Apple Store for review.

Book format

The book may be purchased in either PDF or Mobi formats or in printed format directly from Coogara Consulting. Mobi is suitable for Kindle and other devices. The book may will also become available from Amazon.com and Lulu.com.

Training and Consulting

Training and Consulting are available in Australia. Contact the author for more information or visit www.coogara.com.au.

Third Party Products

Some third-party products may be mentioned. I have no personal affiliation with the companies producing them and, unless stated otherwise, have not undertaken an assessment of their usefulness. They are mentioned only as relevant products that may be useful to developers.

Notes

Note

General notes are shown in this format

Note

Notes specific to HTML5 Builder are shown in this format.

Note

Notes specific to Delphi FireMonkey are shown in this format.

1. Preparing for iOS Development

To create standalone iOS (iPhone and iPad) applications, Xcode must be installed on an Intel-based Mac running OS X Lion version 10.7.4 or higher. Xcode provides the build environment for Delphi FireMonkey and HTML5 Builder applications targeting iOS devices.

Note

To build a Client Mobile application, as opposed to a web-based solution, one additional piece of software is required on the Mac: PhoneGap.

If you have used FireMonkey for Delphi XE2 and built iOS applications, you will be familiar with the need for the Free Pascal Compiler to be installed on the Mac so that Xcode can compile a FireMonkey for iOS application. PhoneGap is to HTML5 Builder what FPC is to FireMonkey.

After setting up Xcode we will set up PhoneGap on the Mac.

For my installation I am using OS X Lion 10.7.4 and Xcode 4.6.3. Xcode 4.5 is required to support devices with iOS 6 installed. Xcode 4.6 is required for iOS 6.1.

Connecting to a Mac Remotely

There are a number of ways to connect to a Mac, including SSH, VNC (OS X has a VNC server built right in), Windows and OS X folder sharing, TeamViewer (free for personal use) and other solutions.

If using VNC to connect, a VNC client program is needed. There are plenty of free ones such as TightVNC Viewer, RealVNC and UltraVNC.

I have been a TightVNC user for many years and am happy with it. A suggestion I found for a better VNC experience is to set Preferred encoding to ZRLE (under Connection Options). I generally use a combination of TightVNC and TeamViewer.

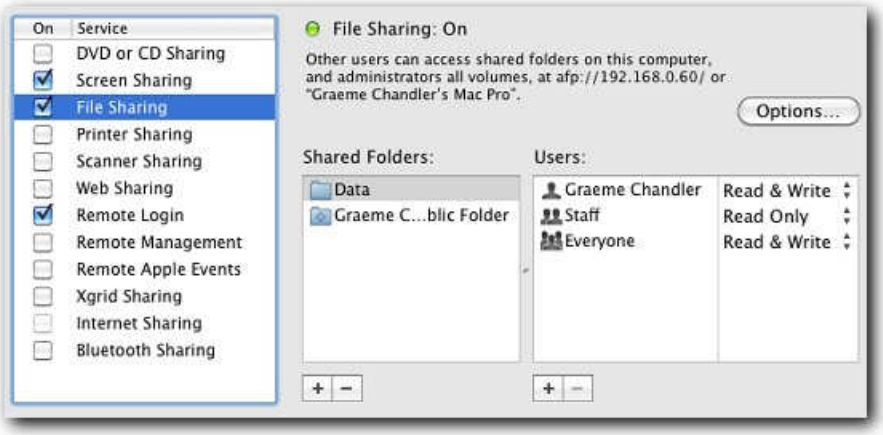
To access remote folders from Windows and to connect via SSH or VNC, you will need to know the IP address of your Mac. Once you have logged on to your Mac, open System Preferences. This is easily done by clicking the cogwheel icon in the dock or clicking the Apple symbol on the menu bar:

From the System Preferences dialog select Network (under Internet & Wireless options). Your Mac's IP address should be shown.

To allow a VNC client to connect to your Mac you need to permit Screen Sharing. Again, open System Preferences on the Mac. Under Internet & Wireless, select Sharing. In the list on the left, check Screen Sharing, then click Computer Settings. In Computer Settings, check both options, as shown below, and set a password. Setting the password seems to be critical for successfully establishing a connection from your VNC client:

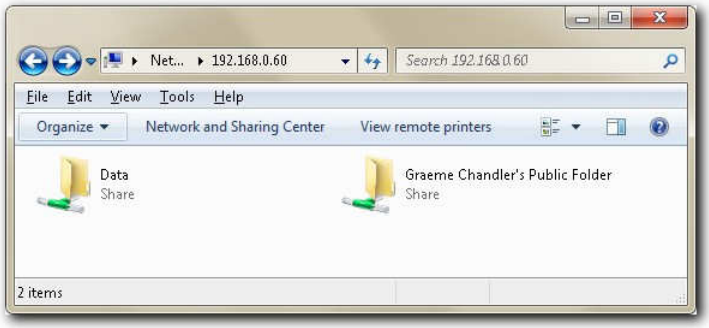
To share folders on your Mac (I'm assuming you already know how to do this under Windows), again open System Preferences on the Mac. Under Internet & Wireless, select Sharing.

In the list on the left, check File Sharing. In the fields to the right, add the folder(s) you want to share and set permissions to enable read/write access:



Click the Options button. Ensure that "Share files and folders using SMB" is checked. Also put a check beside the account that is to be used when sharing files:

From Windows, check that you can see these folders when accessing the Mac via its IP address and try to create a folder or file inside one of the shared folders to confirm that you have full access:



It almost goes without saying that to support deploying, running and debugging an OS X or iOS application you require a Windows computer and an OS X computer to able to actively see each other on a LAN, even if your Windows installation is a guest in an OS X virtual machine.

Obtaining the Right Version of Xcode

Identifying the right version of Xcode depends, at least in part, on the highest version of iOS on the mobile devices to be connected to your Apple computer and on which testing is to be performed. This is because the Xcode environment must be able to identify your iOS device when connected to the host computer. An older version of Xcode, with the associated version of the iOS SDK, is unlikely to successfully identify an iPhone with iOS 6.x installed.

At the time of writing it is mid 2013. At this time many iPads and iPhones are on the most recent version of iOS, which is 6.1, and you will need at least Xcode 4.6, which provides support for iOS SDK version 6.1. iOS 7 has only just been released and is starting to appear on devices around the world. For iOS 7 support you will require a later version of Xcode. Note that all we are talking about here is the version of Xcode required to support devices you plan to connect to your Mac. It has nothing to do with the versions of iOS on which you will be deploying your applications. An application designed with your 6.x device will function perfectly well on a device with a later version of iOS.

There are at least two ways to obtain and install Xcode:

OS X Lion and above include the App Store icon on the dock:

The latest version of Xcode can be found in the App Store and installed from there.

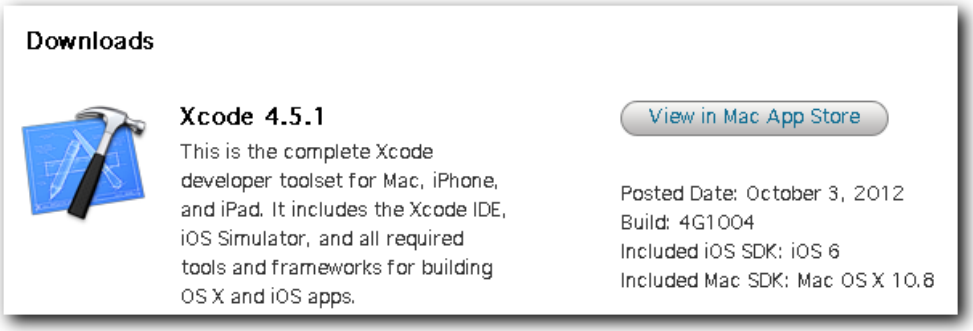
Sometimes you don't simply want the latest version that can be installed via the App Store. To obtain Xcode outside of the App Store, you must be signed up as an Apple Developer. You can do this at:

https://developer.apple.com

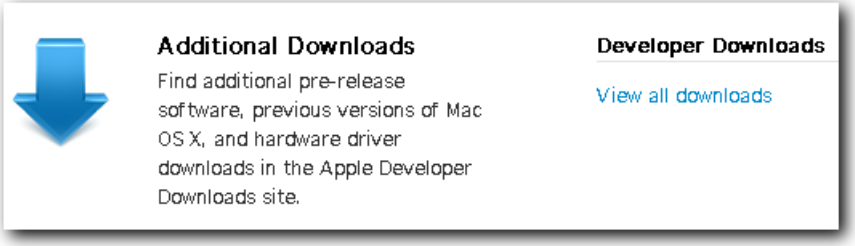
Log in with the Apple ID and password used to register your devices - it is not necessary to create a new ID. The site will run you through the usual email verification process. In the end, you will be signed up as an Apple developer. Once you are signed in as an Apple Developer you will have access to MSDN-like resources: downloads, code samples, documentation and so on.

Go to the Member Center. From there access the Mac Dev Center by selecting it from the Dev Centers | Mac link:

Scroll down to the Downloads section. Here you are given the option to view/download in the App Store. This is not what we want to do as it is no better than downloading via the App Store icon on the OS X Launchpad:



Scroll down a little further to view Additional Downloads and choose "View all downloads":



These are some of the versions that can be downloaded from the above link (filter the list by Developer Tools to see these and/or search for Xcode):

Xcode version	iOS SDK version
3.2.3	4.0.1
3.2.3	4.0.2
3.2.4	4.1
3.2.5	4.2
3.2.6 for Snow Leopard	4.3
4.2 for Snow Leopard	4.3?
4.1 for Lion	?
4.2 for Lion	?
4.2.1 for Lion	?
4.3 for Lion	5
4.3.1 for Lion	5.1
4.3.2 for Lion 10.7.3	5.1
4.3.3 for Lion 10.7.3	5.1
4.4.1 for Lion 10.7.3	5.1
4.5.1 for Lion 10.7.4	6.0
4.6.3	6.1
5.0 for OS X 10.8.4	7.0

Note

> The base Lion OS version is 10.7. Xcode 4.6.3 requires that you upgrade Lion to at least 10.7.4. Xcode 5.0 requires Lion 10.7.4 or later (ie. OS X Lion or Mountain Lion).

The iOS SDK is installed with the installation of Xcode. Each successive version of Xcode provides additional support for iOS features. Through project settings, an Xcode project can target an earlier iOS SDK, providing support for devices running older versions of iOS but reducing the SDK features available to you as a developer.

Xcode 4.6.3 is approximately 1.61 GB. Xcode 5 is 1.91 GB.

Xcode 4.6.3 Installation

If you have downloaded Xcode 4.6.3 for Lion from the developer site, the installation is not really intuitive.

Double-click the DMG file you downloaded. This will display the Xcode icon inside the disk image.

DON'T RUN THIS!!!

In reality it is not an installer. It is an .app package and that's why you can't double-click to install it.

In OS X Finder locate the /Applications folder and drag the above Xcode icon (package) into it. This will take some time to complete as it is copying the hundreds of megabytes from the disk image. When this is done, launch Xcode from within the /Applications folder.

Depending on other Xcode versions installed on your Mac, you will either need to install or update additional components:

Installation will confirm your password to ensure you have permissions for the installation. The duration of this process could be five to ten minutes, depending on the updates required.

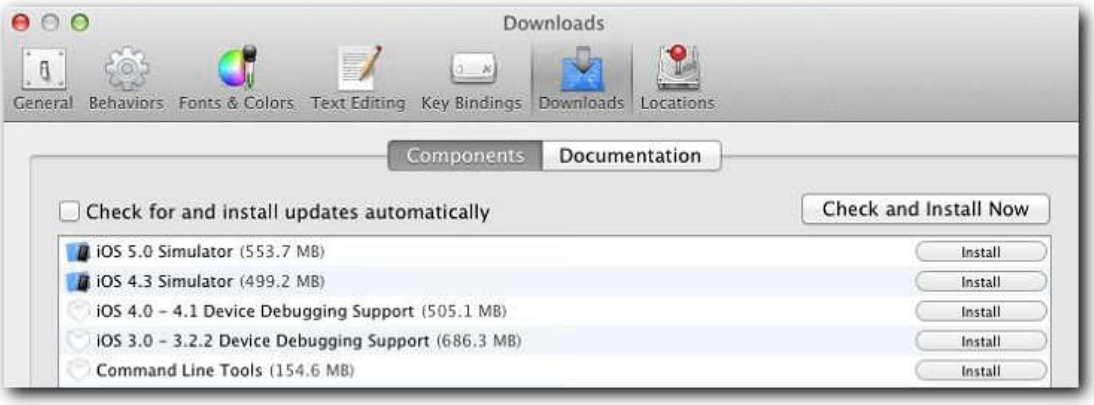
You may be offered the opportunity to remove older Xcode Tools, if installed:

If you are using Delphi XE2 for iOS development on the same machine, don't remove the older Xcode versions.

Once installed (which at this point is straightforward), you can subsequently launch Xcode from the Applications folder using the same icon with which you installed it.

Once Xcode is installed there is a little more to do. With Xcode running, select Xcode | Preferences from the menu:

Check the Xcode Documentation tab. A number of downloads may have started automatically. If you don't want them, or are shy of download capacity, stop the documentation downloads:



You will require the command line tools so ensure that this is downloaded.

PhoneGap Support for HTML5 Builder

PhoneGap is required for building HTML5 Builder applications on the Mac. It is used by Xcode in much the same way that the Free Pascal Compiler is used to build Delphi XE2 FireMonkey applications on the Mac for iOS.

Note

There is another option for building an HTML5 Builder iOS application: it can be built in the cloud. This requires signing up for an Adobe ID. As part of the build process, your application is uploaded to the build service, compiled and returned to you. This is the approach you will often take for other mobile devices types. For iOS we can install PhoneGap into Xcode. This offers a more convenient and flexible solution.

To sign up for an Adobe ID, go to:

https://build.phonegap.com/people/adobeid

Obtaining the PhoneGap Installer

Download the PhoneGap 1.8.1 installer from:

https://github.com/phonegap/phonegap/zipball/1.8.1

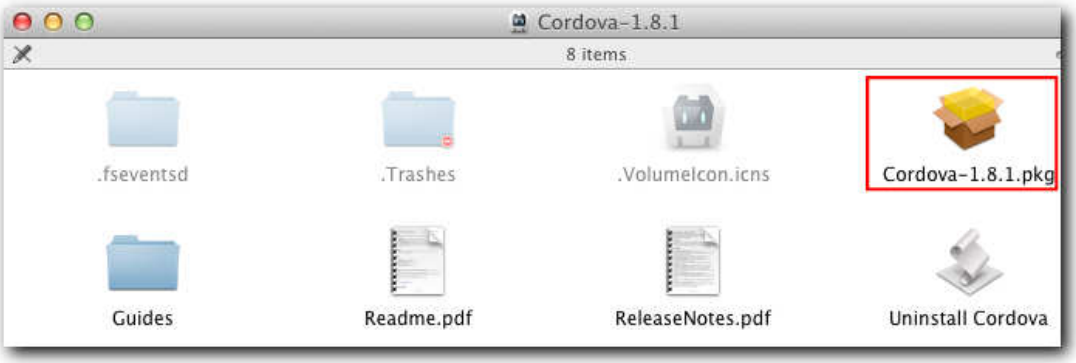
The download is approximately 24 MB.

If using a download manager, the name may end up as simply 1.8.1. The name should be: phonegap-phonegap-1.8.1-0-gce32a3f.zip. At least name it as a ZIP file so the contents can be extracted.

Installing PhoneGap

The download includes PhoneGap for several platforms, each of which is found in the ZIP files lib folder. Inside lib/ios is a disk image: Cordova-1.8.1.dmg.

Extract the .dmg file on the Mac and double-click to open the disk image:

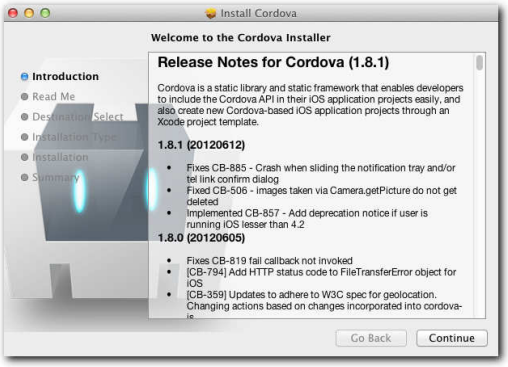


Inside the disk image is the Cordova installation package.

> **Note**
>
> PhoneGap was a product of Adobe/Nitobi - hence the need to sign up for an Adobe ID to build in the cloud. In 2011, Adobe/Nitobi donated the PhoneGap codebase to the Apache Software Foundation (ASF) for incubation. This contribution required a new name to ensure that intellectual property was protected. They didn't want any conflict or confusion with existing trademarks. To cut a long story short, PhoneGap was a product of Nitobi when they were on Cordova Street in Vancouver and the name Cordova came into use.

> **Note**
>
> Throughout this document "PhoneGap" will be used to refer to the Cordova product. This is the name referred to in the Embarcadero documentation.

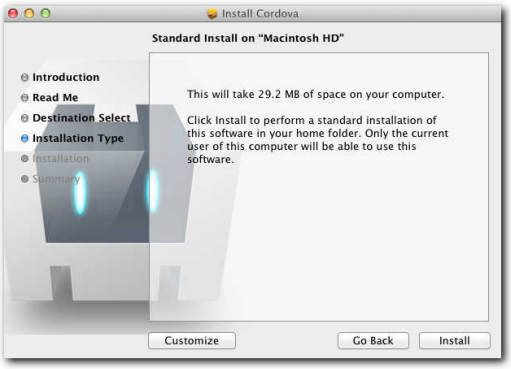
Double-click the package to launch the installer and click Continue:



Review the release notes

Review the pre-requisites

Note that the reference to building and installing the installer package only applies if you downloaded the source package from the Apache Cordova source distribution. Click Continue.



Review the installation type and click Install

Upon completion, review the installation summary

This completes the PhoneGap installation. We now need to ensure that your devices are correctly seen by Xcode and that they can be provisioned.

Connecting an iOS Device to your Mac

When I plugged in my iOS 5.1 iPhone for the first time, the Xcode Organiser automatically launched and displayed my connected device:

If the organiser is not displayed, click the Organiser button at the top right of the Xcode main window or select Window | Organiser from the Xcode menu:

Above, you can see my iPhone 3GS. At the bottom of the screen is the option to "Use for Development".

Until you have set up provisioning on your Mac, **DO NOT PRESS THIS BUTTON**. Actually it isn't as critical as that. You can always tell the Organiser to ignore the device and start again. This is done in the Devices pane by right-clicking the device node and choosing "Ignore Device".

If you receive a message concerning the version of iOS on the device, and the lack of support from within the installed Xcode, you don't have the right version of Xcode installed. Go back and read the Xcode installation instructions again. Alternatively you may need to consider upgrading the OS on your device.

As an Apple developer, you can register up to 100 devices for testing. Each year your devices must be re-registered. At the start of each new year you will be able to restore the list of devices already registered. You won't have to register them individually all over again.

The device must be provisioned (identified and prepared) in the iOS Provisioning Portal. The Provisioning Portal is where you manage certificates and create profiles for development, testing and application distribution.

Signing up for the iOS Developer Program

You may have already joined the Apple Developer center, but if you haven't yet joined the iOS Developer Program, now is the time to do it. Otherwise, you can't test your program on a device, let alone submit it to the App Store.

Log into developer.apple.com. Go to the Member Center and click on Your Account. Alternatively, go straight to: developer.apple.com/programs/ios.

Listed under Developer Programs is iOS Developer Program. Click "Join Today" or "Enroll Now" and follow the enrollment process. Once you have signed up, it may take up to 24 hours for the enrollment process to complete and to receive your activation email:

You have the choice of signing up as an individual developer or for a company/organisation. As an individual developer, you are limited to submitting applications to the App Store as a means to distribute them. This will be the only way for others to obtain your completed applications. You can, of course, register devices for testing for the duration of your membership.

Registered as a company/organisation, applications can be installed on company devices, bypassing the App Store. Additionally, you can assign team members to your developer account. Team members can participate in the development process. To sign up as a company, you will need a D-U-N-S number. This is a Dun & Bradstreet assigned number that identifies business entities on a location specific basis. Don't

worry. The process is straightforward. Just follow the prompts and read the information Apple provides.

The individual program is $99 per year. The company program is $299 per year.

When your first application is added to the App Store, you have the chance to name your company. This can either be your own name or a business name. **Once this has been specified it cannot be changed.**

Within a reasonable time of signing up for the iOS Developer Program, usually no more than 24 hours, two emails arrived - one to say Welcome! and the other to say that you have access to iTunes Connect. iTunes Connect is where, as a developer, "you can enter into a Paid Applications agreement, setup and edit information about your application, view sales and trend information, retrieve financial reports (if applicable), and create new user accounts for other people at your company":

https://itunesconnect.apple.com/WebObjects/iTunesConnect.woa/wa/jumpTo?page=contracts

Provisioning

There is still more to do before an application can be tested on an iOS device. Ultimately, you need to have your device "provisioned", which is to say that it has been prepared, within the Apple developer environment, to participate in the testing of applications. This is done, in part, in the iOS Provisioning Portal as well as in Xcode on your Mac.

The Device ID

Each device has a Unique Device Identifier (UDID) - a 40 character string uniquely identifying it. The UDID is added to a provisioning profile. This allows (or restricts, depending on your perspective) applications built with the provisioning profile to run on devices that are included in the profile.

> **Note**
>
> Each device you add counts towards the annual limit of 100 devices. Even if you remove a device, that doesn't make room for another. It only means the device is no longer provisioned. Each time you add a device it uses up one of the 100 provisioning slots.

The UDID is found in the Xcode Organiser under the details for your device, next to the "Identifier" label.

The Keychain

You have probably already noticed that the Mac Keychain is used to store your credentials when connecting to servers. Keychain Access is used to reduce the number of passwords you have to keep track of. Keychain Access can store passwords for applications, servers and website, cryptographic keys and even non-computer related information such as credit card numbers and PINs for bank accounts. An App

ID (Application ID) is used during iOS development to share Keychain data among applications.

The App ID

The App ID is a combination of a Bundle Seed ID and a Bundle Identifier. The Bundle Seed ID is like a GUID in that it is a globally unique identifier, 10 characters long.

Bundle Identifiers look a bit like reversed domain names. My sole trader name is Coogara Consulting. Taking Coogara, along with an application named Container Tracking, the Bundle Identifier would be com.coogara.containerTracking. You eventually add the Bundle Identifier to your application.

The Bundle Identifier can contain a wildcard for the application name part, leaving com.coogara.*. Using a wildcard allows the same App ID (combination of Bundle Seed ID and Bundle Identifier) to be used with multiple applications. It is also possible to use a wildcard for the entire Bundle Identifier. Ultimately you could have these combinations, assuming a Bundle Seed ID of "A1B2C3D4E5":

A1B2C3D4E5.com.coogara.containerTracking	This App ID applies only to Coogara's Container Tracking app
A1B2C3D4E5.com.coogara.*	This App ID applies to all of Coogara's apps
A1B2C3D4E5.*	This App ID applies to all applications

The first App ID, above, is entirely restrictive and limited to the containerTracking app.

The Development Provisioning Profile

A development provisioning profile is used to associate developers, devices and App IDs with a development team. This is what allows you to install and run applications on a device for testing. The development provisioning profile needs to be installed on the device to make this possible. A device is not limited to a single profile. We will deal with the development provisioning profile shortly.

Setting up Your Mac Development Machine

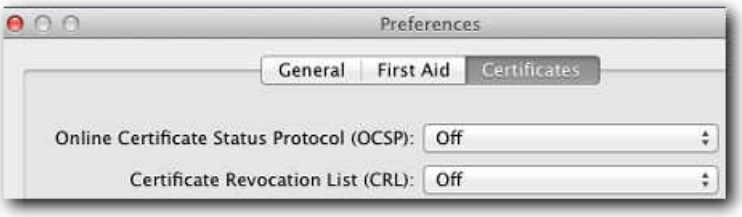
Your Mac must be set up for code signing. This is not so much for your Mac as it is for your application. Code signing securely identifies the application's author and, because of the way it is applied to the application, ensures that the application has not been altered once it leaves your hands and gets onto a device, whether for testing or for deployment via the App Store. The application must be digitally signed before it lands on any device.

Code signing involves having a private and public key pair and a digital certificate. During development, a development certificate is used; on release, a distribution certificate is used to sign the app.

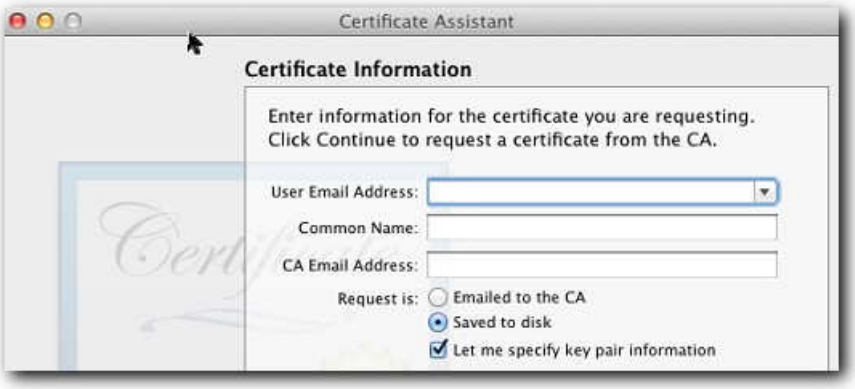
First you must request a development certificate for your development environment on your Mac.

From Launchpad | Utilities choose Keychain Access. Keychain Access is used to generate a Certificate Signing Request (CSR):

Before doing anything else, open the Keychain Access | Preferences menu. Go to Certificates and change Online Certificate Status Protocol and Certificate Revocation List to "off":

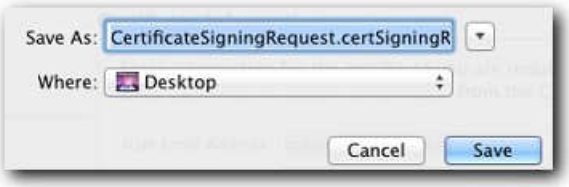


Close the Preferences window and select Keychain Access | Certificate Assistant | Request a Certificate from a Certificate Authority. The values for Email Address and Common Name should be the same as submitted to Apple for iOS Developer Program membership:



Leave CA Email Address blank, select "Saved to disk" and check the check box to specify key pair information then click Continue.

You will be prompted to save the CSR - put it somewhere convenient like the desktop. I ended up saving a number of certificates from this process to the same folder on the desktop then put the folder away for safe keeping:

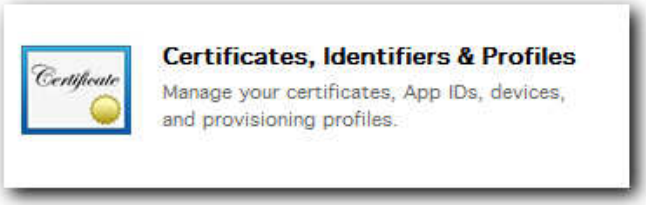


Because you chose "Let me specify key pair information" you are prompted for Key Pair Information. Ensure that 2048 bits and RSA are selected (which should be the defaults) and then continue:

The CSR is generated and saved to the specified location. Click Done. A private and public key pair has been generated and added to the login Keychain. You can find it under the Keys category in Keychain Access.

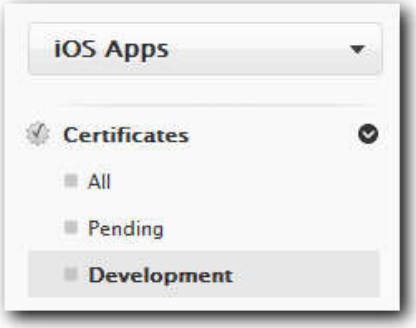
The CSR for this computer now needs to be submitted to your Developer Center account for approval. Who approves it? A team agent or admin, that is, someone on your iOS Developer Program team. In all likelihood, that means you.

To submit the CSR, log in to the Developer Center and, if necessary, click the Home button on the toolbar to display the Developer Program Resources interface. Choose Certificates, Identifiers and Profiles:



You are now presented with each of the Apple Developer programs. Each program for which you are signed up will provide options for managing aspects of that program. For iOS we have the following options:

Choose Certificates. From the tree on the left choose Certificates | Development:

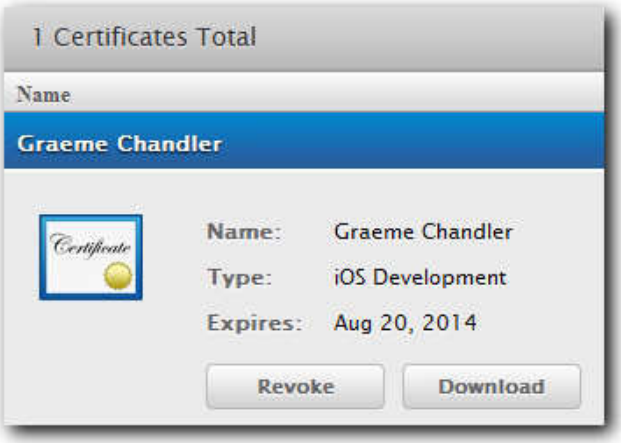


To add the CSR you generated on the Mac, click the "plus" button at the top right. You are asked what kind of certificate you are adding: yours is an iOS App Development certificate so choose this option:

Scroll to the bottom of the web page. **Before clicking Continue**, note the reference to the Intermediate Certificates and the Worldwide Developer Relations Certificate Authority. As specified here, normally Xcode will install this certificate automatically. It is required for the development process. If you ever need to reinstall it, or Xcode complains that there is no WWDR certificate this is where you will find it. I find it is just as well to download it now and store it with your other certificates for safe keeping, but you can always get it here again if necessary.

Click Continue. You will be prompted to take the Keychain Access steps we have already taken to generate a CSR. Click Continue once more and you will be able to browse for your CSR file, upload it and generate a certificate.

Having submitted the CSR, a provisioning profile certificate is now in the list of your current developer certificates. Click an individual certificate to expand its details:



You must download and install the certificate on your development machine, **but complete reading this section before doing so**. This may seem very circular, but consider that it is intended to support multiple developers working on multiple machines and potentially multiple developer profiles.

Before you download your first certificate to a development machine ensure you have downloaded and installed the WWDR intermediate certificate as described above.

Once downloaded, click the Downloads icon in the Dock or open Finder to locate the file (AppleWWDRCA.cer). Double-click the file to install it. You will be prompted to add it to your Keychain:

You may now download the provisioning profile certificate. It will be saved as ios_development.cer. Locate the file and double-click to install to the login Keychain. It is visible in Keychain Access under Login Certificates as iPhone Developer: *your name*.

Your development machine is now set up. Make sure you safely store the certificates you downloaded or you will need to request new ones should you need to reinstall.

Preparing Your Device for Testing

With the necessary certificates installed, we must now ensure that Xcode recognises your device as one suitably prepared for use in development. You can use Xcode to prepare your device, but there will be some restrictions. The Xcode Organiser will add your device and generate a wildcard App ID. A wildcard App ID cannot be used with Apple services like Game Center, In-App Purchase, iCloud and Push Notifications. However, Xcode does a nice job when you plan to develop general applications that run on the device without needing these special services.

Make sure your device is connected to your computer and open the Xcode Organiser (from the menu choose Window | Organiser to do this). If necessary, launch Xcode to ensure that it is running and is the current application on the OS X menu bar. Your devices are listed with a colored orb beside them. A white orb means the device is not ready for development. A green orb means you are set to go. A yellow orb means the device is busy.

Click Use for Development and sign in with your Apple Developer ID:

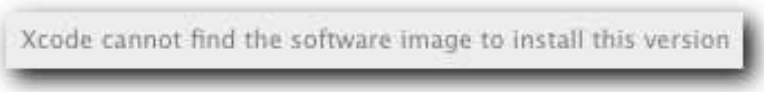
Xcode automatically sets up your device for development, registering your device and allocating a wildcard App ID. If no iOS Distribution Certificate is found, you will be prompted to obtain one by submitting a request.

This process can take some time. During the provisioning process the device's status icon will be yellow. The device should not be removed until the process is completed

and the icon changes to green. Along the way you will be asked if you want to export a Developer Profile for installation on other Macs. Once this is completed, a provisioning profile has been assigned: "iOSTeam Provisioning Profile:*".

With everything installed correctly, the Organiser Devices screen showed:

Don't be concerned about the message: "Xcode cannot find the software image to install this version":



According to the Apple Technical Q&A QA1569:

http://developer.apple.com/library/ios/ - qa/qa1569/_index.html

this is a notification and not an error:

> "This message does not prevent you from using the device for development purposes."

On my iPhone, in Settings | General | Profile, it records the provisioning profile:

The same information is presented on my provisioned iPad. The name of the profile depends on how the profile was generated. The iOS Developer Centre allows you to name the profile however you wish.

You are now ready to build and run iOS applications on your device. When you run the app to a physical device you will be asked to sign the app:

I chose Always Allow. Ultimately it is more convenient.

Terminating a Program Running under Xcode

To end a program running from Xcode on the device, don't terminate it on the device. This can cause Xcode to crash. Instead, click the Xcode Stop button. This will terminate it on both Xcode and the device. In fact I have found fewer problems with Xcode 4.5.1 than with earlier versions. Stopping an application on the device while debugging is not nearly as nasty as with earlier versions of Xcode.

Taking Screen Shots from your Device

It is always nicer to have a screen shot from a program running on a device rather than a simulator.

You probably already know that you can take a screen shot on any iOS device by holding down the Home and Sleep buttons at the same time. But then you have to transfer the image from the device's camera roll to your machine.

There is an easier way when you have Xcode installed. With your device connected, bring up the Organiser window and click the Screenshots node under your device:

On the bottom right of the Organiser window is a New Screenshot button:

When clicked, a screenshot is saved under the Screenshots view in Organiser. These can be exported or saved as launch images (splash screens to you and me).

Here is part of an image snapped from my device:

Using Xcode

Several aspects of Xcode have already been mentioned. This section will summarise those and bring together other Xcode features that will be useful during development.

> **Note**
>
> This section is useful to HTML5 Builder developers and Delphi XE2 FireMonkey developers as they will interact with Xcode directly. Users of Delphi XE4 and later do not interact directly with Xcode.

The Organiser

The Organiser is broken down into 5 sections: Devices, Repositories, Projects, Archives and Documentation.

To open the Organiser, choose the Organiser button from the top right side of the Xcode window:

or choose Window | Organiser from the Xcode menu.

The Devices section is where your devices are listed. Previously connected devices that are currently disconnected are also shown. Provisioned and connected devices become available in the main Xcode window as deployment targets, along with available simulators:

In addition to features helping you manage devices, there are options to help you manage device-related information such as provisioning profiles, software images, device logs and screenshots:

Provisioning Profiles will usually provide the same list of profiles as you would see from any of your device's provisioning profiles unless a profile has been removed from a device.

Software Images are used to restore the firmware of an iOS device to a specific version. Files are of type .ipsw. They can be downloaded from:

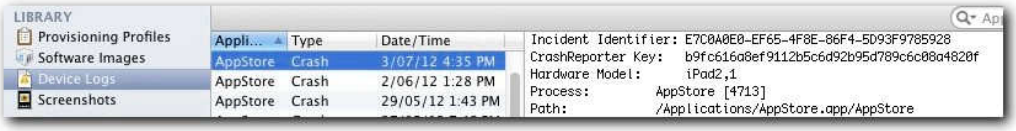
http://www.appleipsw.com/

http://ios.e-lite.org/

The downloaded .ipsw files can be added by dragging and dropping from Finder or by using the Add button. To restore a device to its original state without changing firmware version, select your device in iTunes and choose the Restore button. Further information about updating and restoring your device can be found here:

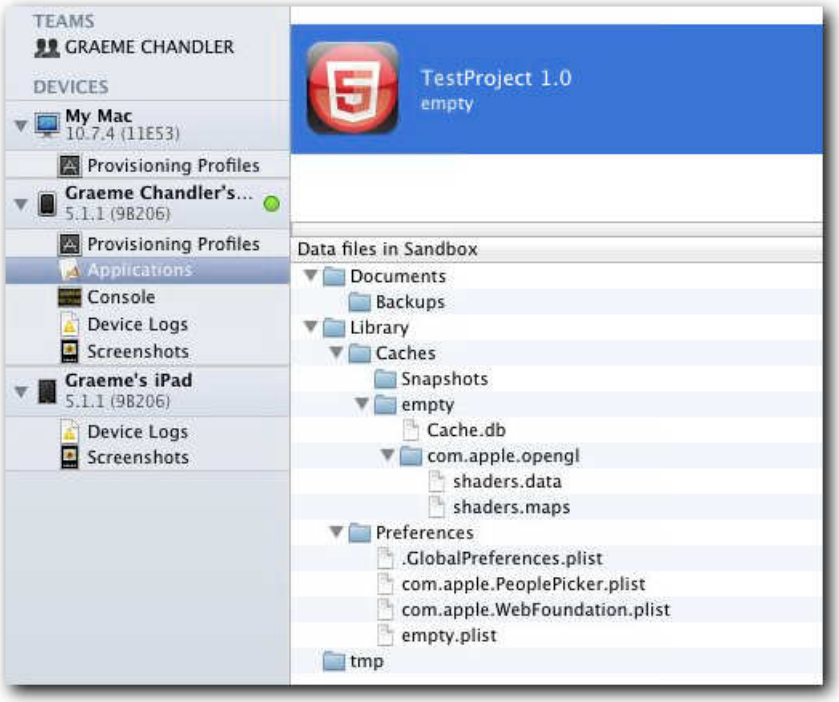
http://support.apple.com/kb/HT1414

Device Logs list the logs of all devices that have been connected to the Organiser. It provides details that may prove helpful for diagnosing application crashes:



Applications

Particularly useful is the Applications section of a device:

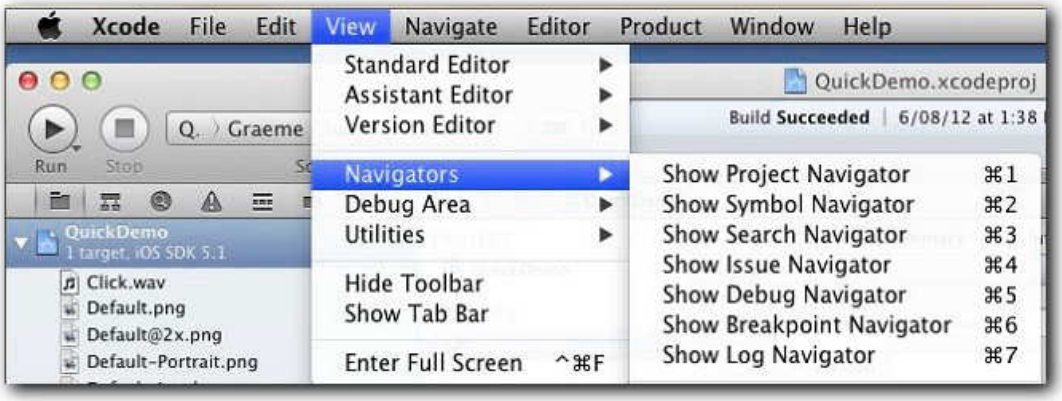


Each application you have built and installed to the device is listed at the top. Selecting an application will show the folders and files that have been placed on the device for the selected application. The only folder not shown is the application bundle.

Beneath the list of folders and files is a tool bar of four icons. The first two permit adding and deleting applications. The second two permit downloading an application's files and uploading files to an existing application. The last two can be especially helpful for diagnosing the data output of an application or for loading particular files for testing specific scenarios.

Navigators

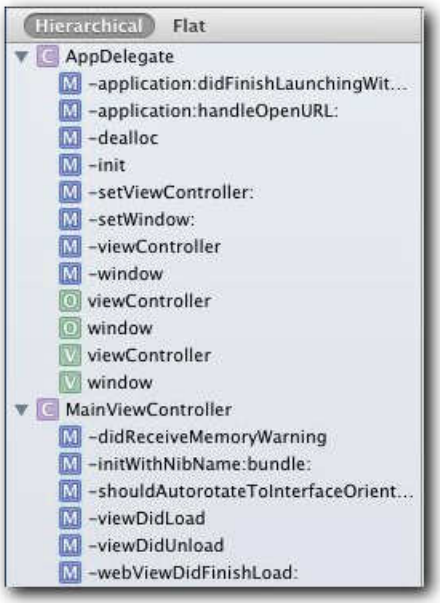
The view provided in the left Navigator column can be changed by selecting an option from the View menu:



Each menu option corresponds to a button above the Navigator view:

The **Project Navigator** lets you access project members. Selecting a graphic node will show a preview in the main window. Selecting a source node allows source code to be edited.

The **Symbol Navigator** shows class symbols/elements and allows a specific symbol to be navigated/edited by selecting the symbol. For the rest of us, symbols refer to method names, property declarations and so forth. This example shows symbols related to the classes involved in a simple HTML5 Builder project:



By choosing a symbol such as ~application:handleOpenURL:, the handleOpenURL method in the application class is displayed for editing:

```
// this happens while we are running ( in the background, or from within our own app )
// only valid if Info.plist specifies a protocol to handle
- (BOOL) application:(UIApplication*)application handleOpenURL:(NSURL*)url
{
    if (!url) {
        return NO;
    }

    // calls into javascript global function 'handleOpenURL'
    NSString* jsString = [NSString stringWithFormat:@"handleOpenURL(\"%@\");", url];
    [self.viewController.webView stringByEvaluatingJavaScriptFromString:jsString];

    // all plugins will get the notification, and their handlers will be called
    [[NSNotificationCenter defaultCenter] postNotification:[NSNotification notificationWithName:
        CDVPluginHandleOpenURLNotification object:url]];

    return YES;
}
```

The **Search Navigator** provides a project-wide search mechanism. While it could be very useful for finding all references to a string, the HTML5 Builder and Delphi IDEs are probably still your IDE of choice for searching and editing tasks.

The **Issue Navigator** may be the most important tool in your Xcode arsenal. Compile errors (issues) are listed in this navigator. Single-click an issue to display it in the main window:

```
ld: file is universal (3 slices) but does not contain a(n) armv7s slice: /Users/
  Shared/Cordova/Frameworks/Cordova.framework/Cordova for architecture armv7s
clang: error: linker command failed with exit code 1 (use -v to see invocation)
File is universal (3 slices) but does not contain a(n) armv7s slice: /Users/Shared/Cordova/Frameworks/... more
Linker command failed with exit code 1 (use -v to see invocation)
```

Double-click an issue to open the error detail in a separate window. If the issue relates to source code, as opposed to project settings, the location of an issue is usually provided. Very often the first issue triggers a number of others. Fixing the first can clean up the remaining ones.

The **Debug Navigator** shows thread and call stack information for a running program.

The **Breakpoint Navigator** provides a breakpoint list.

The **Log Navigator** displays a log of project-related activities including builds and debugging. The status of builds is shown as well as various debug-related logs:

```
Debug TestProject
11/10/12 11:21 AM
Build TestProject
11/10/12 11:21 AM

2012-10-11 11:22:01.380 TestProject[1136:15b03]
2012-10-11 11:22:01.383 TestProject[1136:15b03]
2012-10-11 11:22:01.443 TestProject[1136:15b03]
root view controller at the end of application l
(lldb)
```

Project Summary

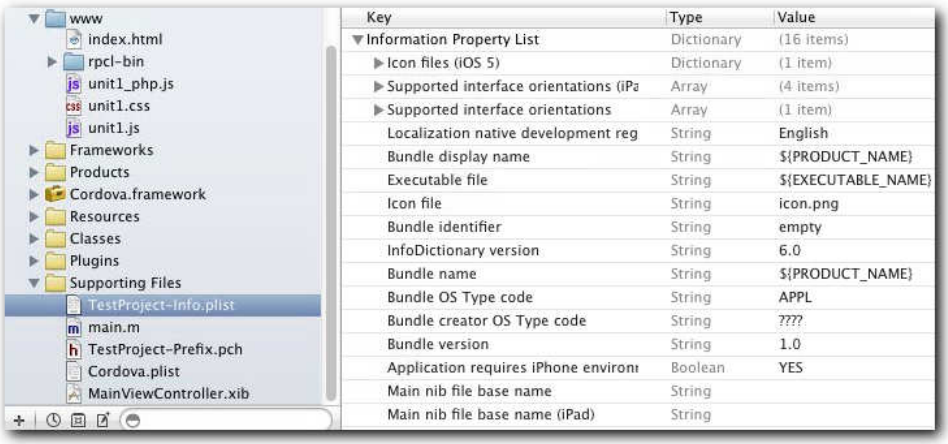
Having chosen the Project Navigator, project settings can be made in the central Summary screen. Permitted orientations for your application need to be specified by device type. There is a section for iPhone/iPod and a section below this for iPhone:

Note

For Delphi FireMonkey XE4 and later, settings such as orientation are made in the Delphi project settings. You will usually have no interaction with Xcode directly for FireMonkey projects in Delphi XE4 and later.

The settings are generated by the HTML5 Builder and FireMonkey XE2 export process and written to your <project name>-Info.plist file in the xcode folder. Choosing the .plist file in the Project Navigator lets you view and change its settings.

Here is a screen shot of an HTML5 Builder TestProject-Info.plist file:



Further up the tree you can see the www folder into which web page files were written.

Project Info/Settings

Instead of changing settings by selecting the .plist file, it is often easier to change settings in the Info tab of the Project navigator:

Summary | Info | Build Settings | Build Phases | Build Rules

Custom iOS Target Properties

Key	Type	Value
Bundle identifier	String	${PRODUCT_NAME:rfc1034identifier}
InfoDictionary version	String	6.0
Bundle version	String	1.0
Executable file	String	${EXECUTABLE_NAME}
Application requires iPhone environmer	Boolean	YES
▶ Icon files	Array	(3 items)
▶ Supported interface orientations	Array	(3 items)
Icon already includes gloss effects	Boolean	YES
Bundle display name	String	${PRODUCT_NAME}
▶ CFBundleSupportedPlatforms	Array	(1 item)
Bundle OS Type code	String	APPL
Bundle creator OS Type code	String	????
Status bar is initially hidden	Boolean	NO
Localization native development region	String	en
▶ Supported interface orientations (iPad)	Array	(4 items)
Bundle name	String	${PRODUCT_NAME}

Many different settings and properties are available. Rather than list them all here, they are best read in their entirety at:

http://developer.apple.com/library/ios/#DOCUMENTATION/iPhone/Conceptual/iPhoneOSProgrammingGuide/App-RelatedResources/App-RelatedResources.html

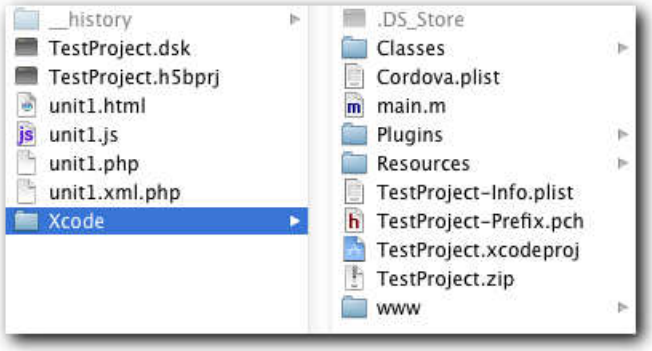
A lot of the properties are presented, in one way or another, on the Xcode interface. For example, device orientations are available visually on the Summary screen and also by using the `UISupportedInterfaceOrientations` key.

Accessing your Windows projects from a Mac

HTML5 Builder and Delphi XE2 FireMonkey projects need to reside on a Windows share so that they can be accessed from your Mac. Switch to your Mac. You will need to navigate to the Windows share on which your project is saved. Open Finder. From the menu, select Go | Network. If your computer is not shown in the list, select Go | Connect to Server. To connect to a Windows share, the server address needs to be in the form: smb://MachineName:

Click the Connect button. Supply your Windows user name and password when prompted (not your Mac credentials), and save the credentials to your Mac Keychain so you don't have to enter them each time:

You will be prompted to choose the volumes to mount. Choose the share on which your project is saved and navigate to the folder to which you exported your project. From OS X it looks something like this:

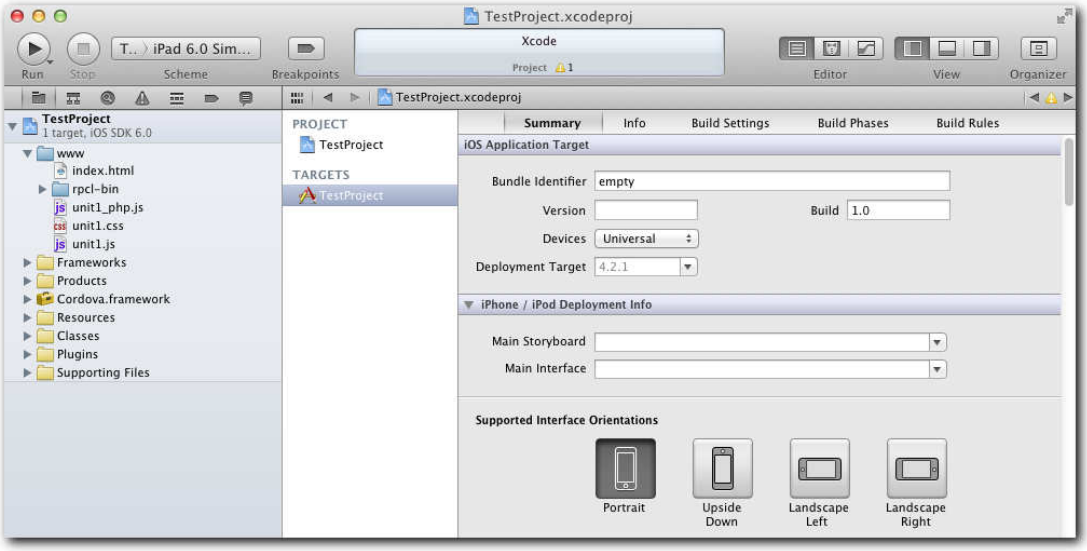


My HTML5 Builder project was called TestProject and in the export process I did not change it. Consequently, the exported files are given names based on the base name of my project: TestProject.

On the Mac, TestProject.xcodeproj is seen as a package, not a folder. You can still see the contents of the package with a right-click, but the contents are not meant to be a concern on the Mac, other than from within Xcode.

> **Note**
>
> Because I saved the project on Windows and access it from the Mac via a share, I find it easiest on the Mac to navigate to the share, locate the .xcodeproj item, and double-click it to launch Xcode. Trying to open the project from the Xcode IDE before the share is accessible is problematic.

On the Mac, double-click TestProject.xcodeproj to launch the IDE. This will launch the Xcode IDE:



For now we want to make certain that you can run the test project in a simulator.

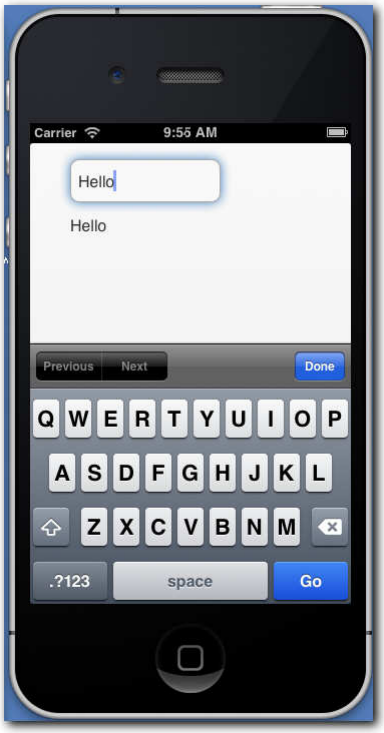
Before running your application, look at the top left of the Xcode window. When a project is opened for the first time, it will target the iPad simulator:

The iPad simulator is quite large, the Retina display version even larger. For a smaller interface choose the iPhone simulator. The Retina version is still very large. When the simulator is running, the target simulator can be changed from the simulator's Hardware menu. The standard iPhone makes a good choice for this test:

Click the Run button (top left). There will be a brief Build Succeeded message:

This is followed by an iPhone launch screen. This launch screen comes from an image generated by the export process, in this case from HTML5 Builder:

Finally the application will run:

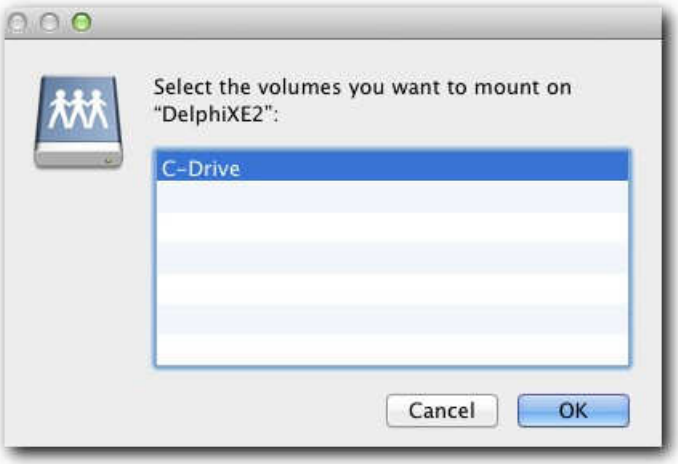


To end the simulator, right-click the simulator icon in the dock and select Quit or click the Stop button in Xcode:

Simplifying Access to a Windows Share

Re-establishing a connection to a Windows share each time you start OS X is annoying. Here is a way to simplify it.

If you haven't already done so, follow the steps in the previous section to connect to a server (Windows share) and choose the volume to mount. The shared volume on my DelphiXE2 machine is called C-Drive:



In Finder, find the mounted volume and ensure it is connected:

On Lion:

Drag the item that represents your share (in my case C-Drive) from within Finder up to Finder's tool bar. In the case of Lion, you can drag it from the title bar onto the tool bar. What you end up with is a link to your share on the toolbar - the last icon shown below:

Next time you start OS X, the icon will be a question mark. This indicates that the media to which the link refers is not currently available or no attempt has yet been made to access the shared media:

Clicking the question mark will re-establish the connection to the share, if available.

2. Apple App Store Submission

App Store Review Guidelines

By reviewing the App Store Review Guidelines before you submit your application, your app has a better chance of being accepted sooner.

Many of the guidelines are obvious: Apps that crash will be rejected. But some of the guidelines are amusing: Apps that duplicate apps already in the App Store may be rejected, particularly if there are many of them, such as fart, burp, flashlight, and Kama Sutra apps.

The review guidelines for iOS applications can be found here:

https://developer.apple.com/appstore/resources/approval/guidelines.html

iTunes Connect

In the previous chapter you signed up for the iOS Developer Program and were given access to iTunes connect. Now that you have finished your application and want it listed in the App Store, you must submit it to iTunes Connect to obtain Apple's approval. Log in to:

https://itunesconnect.apple.com/

There are two main areas on the screen. The top area provides news and information relevant to application publishers. The lower area is where you:

- Manage applications
- Enter and update banking details
- Manage contracts, tax and banking details
- View sales information
- View financial reports
- Manage users
- Access developer forums

Manage Users is where you can manage iTunes Connect and In-App Purchase test user accounts.

Contracts

If you want to be paid for your applications, you must enter into a contract with Apple and provide tax and banking information. Click "Contracts, Tax and Banking" or skip this section if you are happy with free apps for now:

You will have to sign two contracts. One is for iOS paid applications and the other is for the iAd Network:

Contract Region	Contract Type	Legal Entity	
All	iOS Paid Applications		Request
World	iAd Network		Request

Contracts In Effect

Contract Region	Contract Type	Contract Number	Contact Info	Bank Info	Tax Info	Effective Date	Expiration Date	Download
Worldwide	iOS Free Applications		N/A	N/A	N/A	Jul 01, 2012	Jul 01, 2013	N/A

There is one contract automatically in effect. This is for free iOS applications.

Click the Request button for each contract, read thoroughly and indicate that you accept the terms of the agreement. Note that Apple collects tax in various countries. Read the agreement carefully so that you understand how the agreement applies to your country.

Having clicked the submit button for each contract, the contracts will be listed as "Contracts in Process". A copy of the contracts will be emailed to you.

Contract Region	Contract Type	Contract Number	Contact Info	Bank Info	Tax Info	Download	Status
All (See Contract)	iOS Paid Applications		Set Up	Set Up	Set Up		Pending Tax, Bank, Contact
World	iAd Network		Set Up	Set Up	Set Up		Pending Tax, Bank, Contact

For each contract, you are required to set up contact, bank and tax information. If you are an individual developer, this is straightforward. In the Company Contacts form, click Add New Contact to add yourself as a contact:

Once your information is added, choose a contact for each role: senior management, financial, technical, legal and marketing. As an individual developer, it will be you in each case. Click Done when you are finished.

You are back at the main screen and can now set up banking information. Click "Set Up":

Click Add Bank Account to proceed. It will guide you through selecting your country and entering the details of your bank account, including account name and currency. On the final screen, check the box to accept the agreement and click Save. Once an account has been added it can be chosen from the drop-down list.

Back on the main screen, you will see that the banking information has been applied to the second contract and it can be viewed:

Contract Region	Contract Type	Contract Number	Contact Info	Bank Info	Tax Info
All (See Contract)	iOS Paid Applications		Edit	View	Set Up
World	iAd Network		Set Up	View	Set Up

Proceed to set up your tax information. Depending on your location, the set up will vary. In Australia:

An Australian seller must have an Australian Business Number (ABN) and be registered for GST. Proof of registration must be uploaded to Apple.

Everyone must set up the U.S. Tax forms. The steps are straightforward for non U.S. citizens. The information you enter cannot be changed. For this reason, they go to some trouble to make sure you are happy with the information.

Once your information is submitted, you are back at the main screen. Setting up the iAd Network is all that remains. The steps are the same as for the first contract. When you are done, the Contracts, Tax and Banking screen should be complete:

Contracts In Process

Once you complete setup and the effective date has been reached, the contract will be moved to the Contracts In Effect section.

Contract Region	Contract Type	Contract Number	Contact Info	Bank Info	Tax Info	Download	Status
All (See Contract)	iOS Paid Applications		Edit	View	View		Processing
World	iAd Network		Edit	View	View		Processing

Contracts In Effect

Contract Region	Contract Type	Contract Number	Contact Info	Bank Info	Tax Info	Effective Date	Expiration Date	Download
Worldwide	iOS Free Applications		N/A	N/A	N/A	Jul 01, 2012	Jul 01, 2013	N/A

All set up? Written your killer app? Now you can proceed with preparing the application for submission to the App Store.

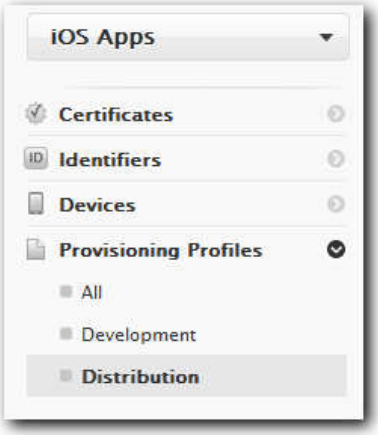
Distribution Provisioning

The provisioning profile that was created previously was a development provisioning profile. A distribution provisioning profile is now required.

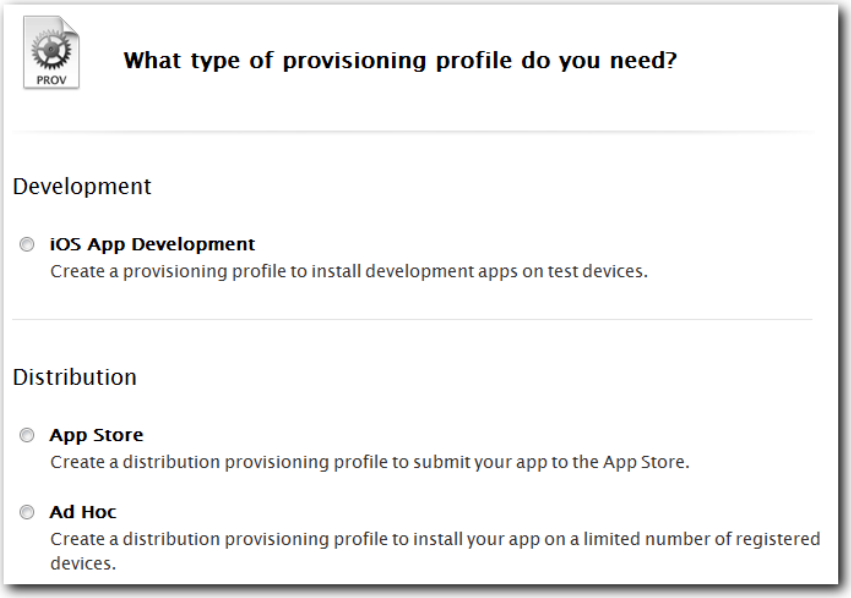
Log into:

https://developer.apple.com

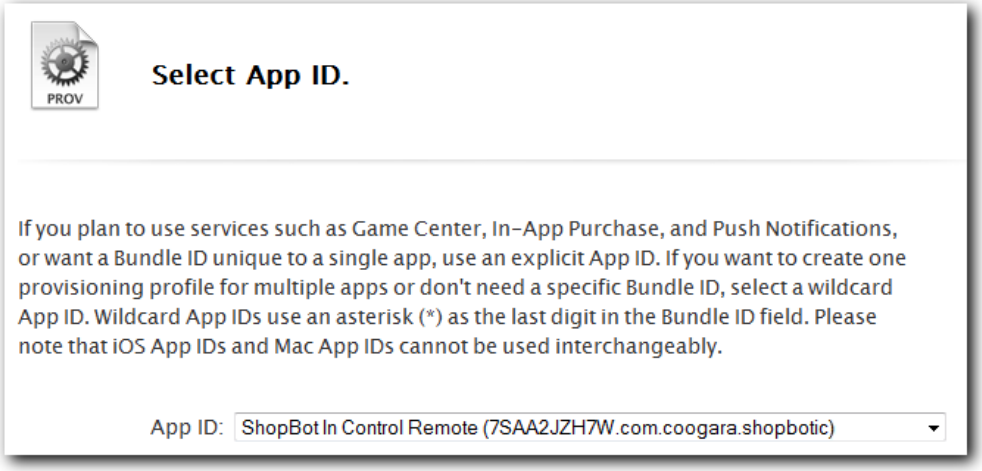
Navigate to the Member Center and go to the Certificates, Identifiers and Profiles section. From the iOS Apps group choose the Provisioning Profiles option. From the menu on the left choose Distribution:



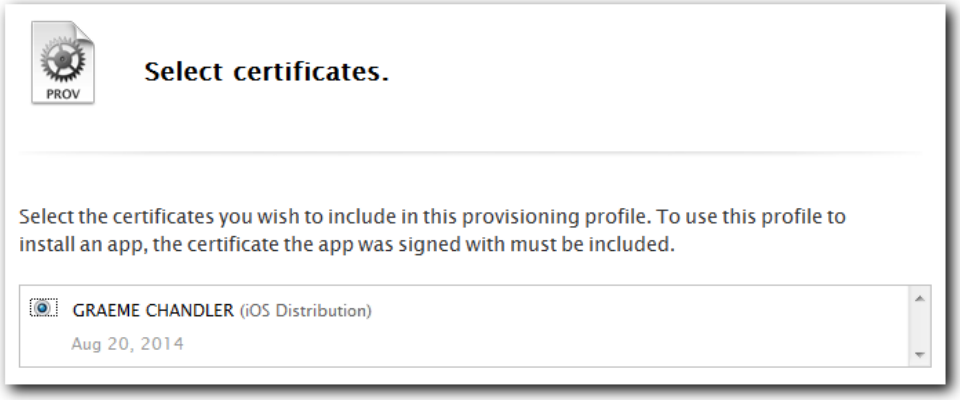
There are probably no profiles listed. All of your profiles, up to this point, are development profiles. Click the "+" button at top right to add a new profile. This is a generic process and is used to create both development and distribution profiles. From the Distribution group choose App Store:



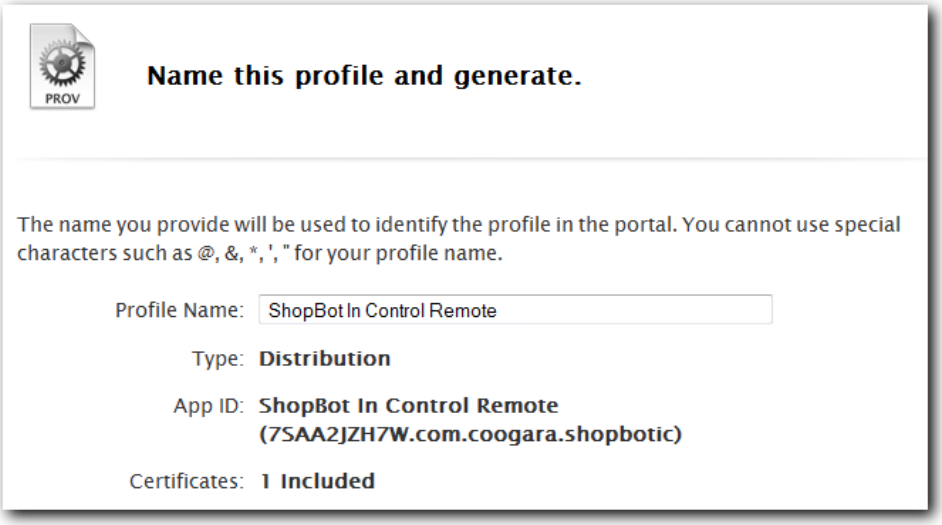
Click Continue. The next screen wants to know the AppID to which this provisioning profile applies. This should already have been created during the development process so choose your App ID and click Continue:



Choose the certificates to be included in the provisioning profile. This identifies the developer signed up for iOS distribution:



Finally you can enter the name for the profile to identify the profile in the portal:



You can now click Generate to produce the Distribution Provisioning Profile.

The final step allows you to download the generated profile so it can be included in the Mac Keychain. Double click the downloaded file to install to the Keychain and make it available to Xcode. Downloading can also be done later from the main interface.

At this point your application must be prepared for submission to the App Store. As the steps are different depending on the IDE you are using, these steps are covered in my books on those topics.

Preparing for Application Submission

Assuming that your application is prepared for submission, we can now take the remaining steps in iTunes Connect to prepare for the submission of your application.

Log on iTunes Connect:

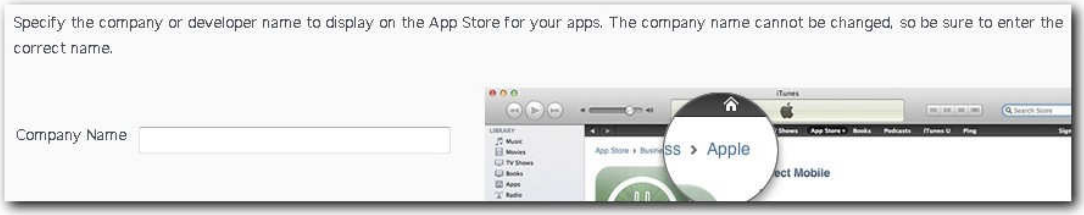
https://itunesconnect.apple.com/

> **Note**
>
> These steps can be completed almost any time before your application is finished. It does not have to be finished to prepare the application's details in iTunes Connect.

Manage Your Applications is where you should go to add or manage an application. Click the Add New App button:

Company Name

If this is your first application, you will be asked to enter a company name. This is the name, for all time, that will be used to identify the producer of your applications. It is included in your application details on the App Store, so make certain you make the right choice and spell it correctly:



Application Information

Once this is set up, you are ready to enter information about your application, including default language, application name, SKU and Bundle ID:

There is plenty of help in the "?" buttons on the right side of the controls.

Availability and Pricing Tier

The next screen asks for availability and pricing information:

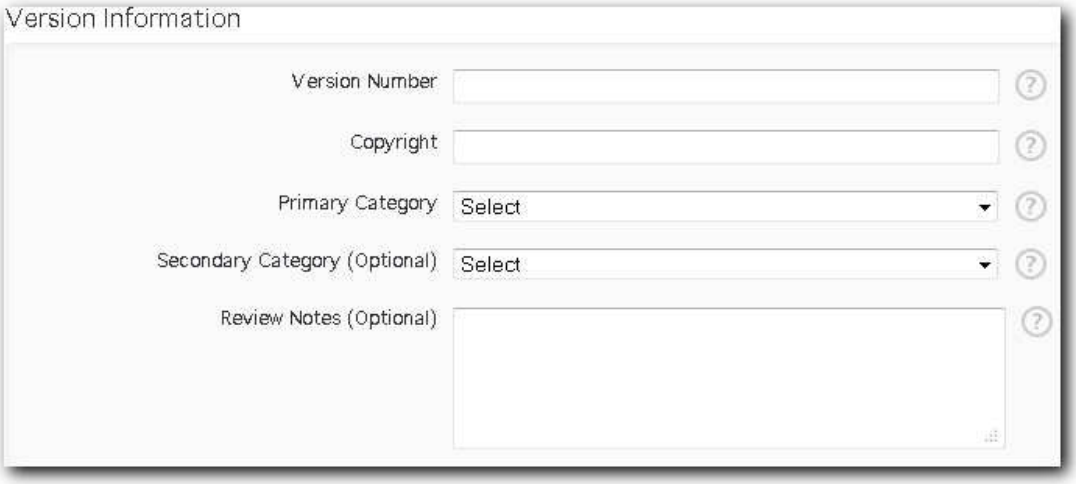
The Availability Date gives you the flexibility to delay the release of an application until some future date. This might be useful if, for example, additional work is required on a supporting web site, server processes or other infrastructure, prior to the application's release.

The pricing tier identifies the price, Apple's take and your return after taxes. Price structures are available in the Pricing Matrix by choosing this option under the Price Tier combo. The price may be changed at any time and may take anything from a few minutes to a few hours to be updated.

You can choose to give a discount to educational institutions. It is checked by default.

Product Version

You are next asked for information about this version of your application. It must be entered in English:



For Copyright information, prefix the name of the copyright holder with the year. For example, "2013 Coogara Consulting". Review Notes are notes for the Apple reviewers, such as details of test accounts for In-App Purchase, user names, passwords, access codes and so on.

What if it is impossible for your app to be tested by Apple's reviewers? This can occur, for example, when your app is designed to access your company servers or a piece of hardware to which Apple either does not have access or is not entitled to have access. In these circumstances you should provide a video of the app in action. The URL of the video should be provided in the Review Notes section.

Product Rating

Further down on this page, you are asked for Rating information. This assists Apple in assigning an age rating. Note the content warning:

For example, Angry Birds is rated 4+ because it contains no objectionable material. Twisted Lands: Insomniac Collector's Edition is rated 9+ for:

- Infrequent/Mild Cartoon or Fantasy Violence
- Infrequent/Mild Horror/Fear Themes

Your responses to the following questions establishes the rating for your application:

Apple Content Descriptions	None	Infrequent/Mild	Frequent/Intense
Cartoon or Fantasy Violence			
Realistic Violence			
Sexual Content or Nudity			
Profanity or Crude Humor			
Alcohol, Tobacco, or Drug Use or References			
Mature/Suggestive Themes			
Simulated Gambling			
Horror/Fear Themes			
Prolonged Graphic or Sadistic Realistic Violence			
Graphic Sexual Content and Nudity			

Metadata

Metadata includes:

- Description
- Keywords
- Support Email Address
- Support URL
- Marketing URL *
- Privacy Policy URL *

* - Optional

When users search the app store, their search terms are matched against your keywords. Separate keywords by commas and limit the size to 100 bytes. Keywords cannot be edited once your application is under review by Apple.

EULA

Your own EULA (End User License Agreement) can be used or you can accept the one provided by Apple. Note that there are some minimum terms that apply when supplying your own EULA. Read the supplied information carefully.

Uploads

At the foot of the page you can supply icons and other graphics. These are not necessarily the icons used in your application. These are used in the App Store.

> **Note**
>
> All graphics must be at least 72 DPI and in the RGB color space. The file type must be .jpeg, .jpg, .tif, .tiff, or .png.

Large App Icon

This is a large version of your app icon that will be used on the App Store. It must be at least 1024 x 1024 pixels (it cannot be scaled up). It must be flat artwork without rounded corners. This rather large icon is for Retina displays.

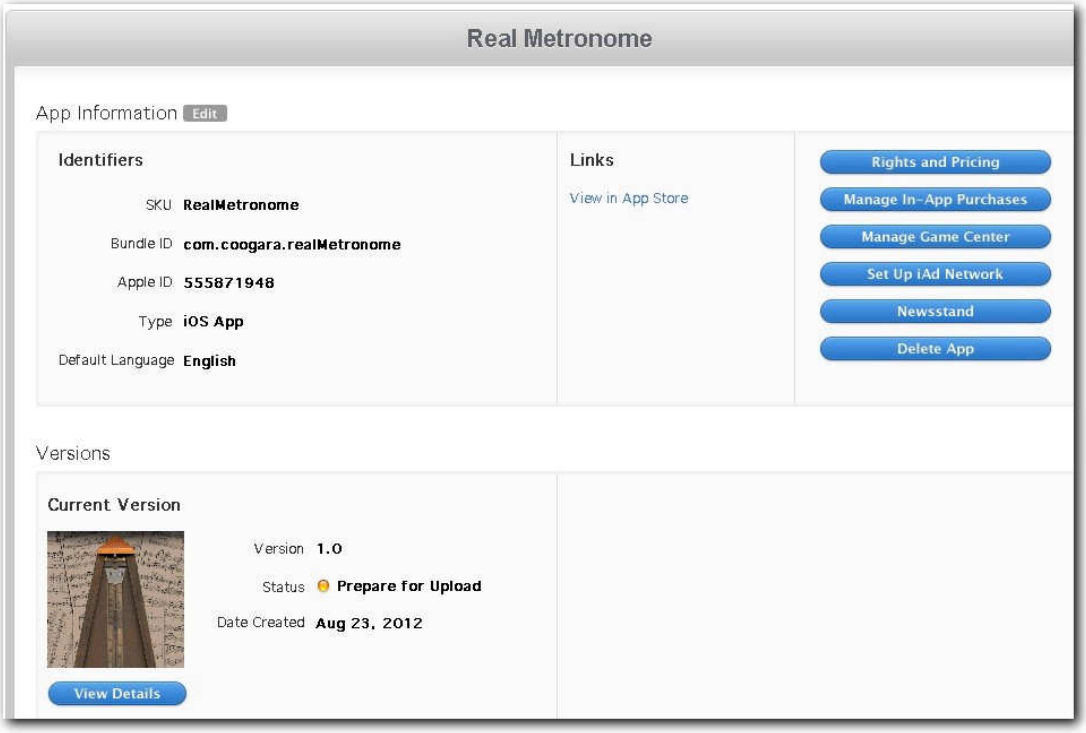
iPhone and iPhone Touch Screenshots

iPhone and iPod touch Screenshots must be 960x640, 960x600, 640x960 or 640x920 pixels.

iPad Screenshots

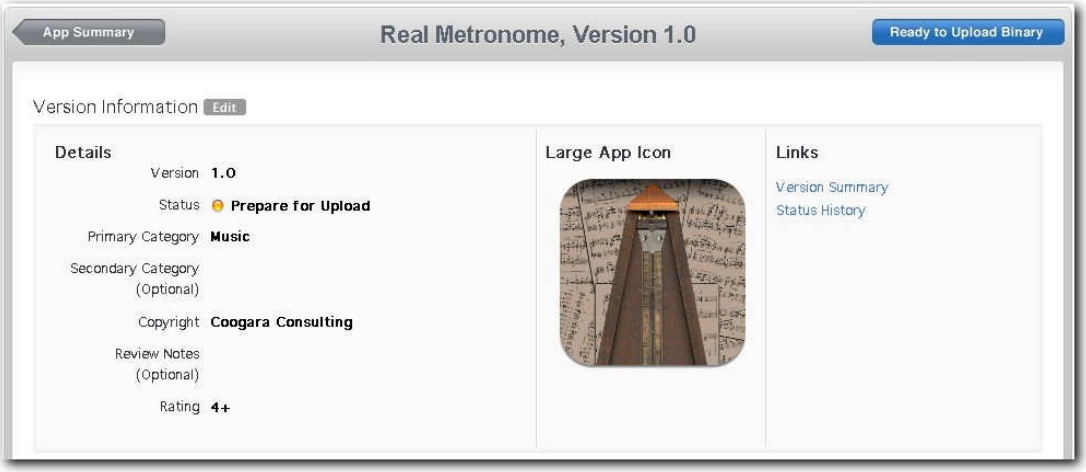
iPad Screenshots must be 1024x768, 1024x748, 768x1024, 768x1004, 2048x1536, 2048x1496, 1536x2048 or 1536x2008 pixels.

Click Save to finish. A summary screen is shown:

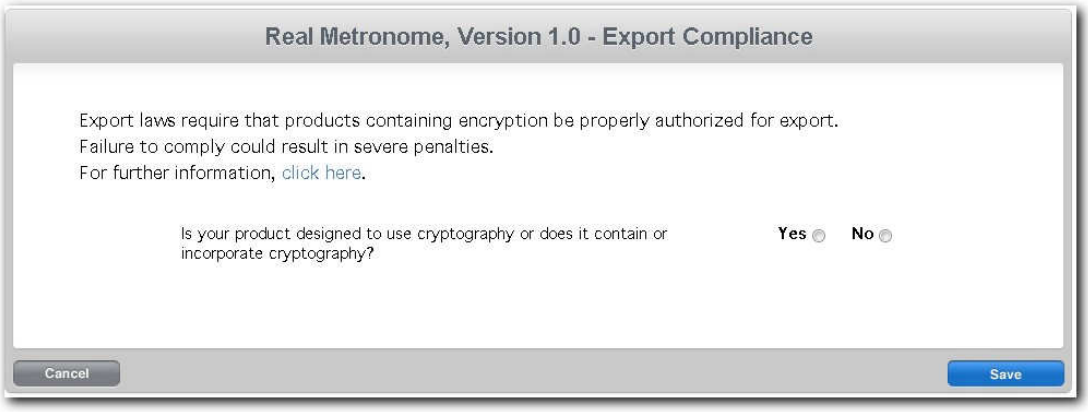


On the summary page, click View Details. The button is beneath the graphic in the lower left corner.

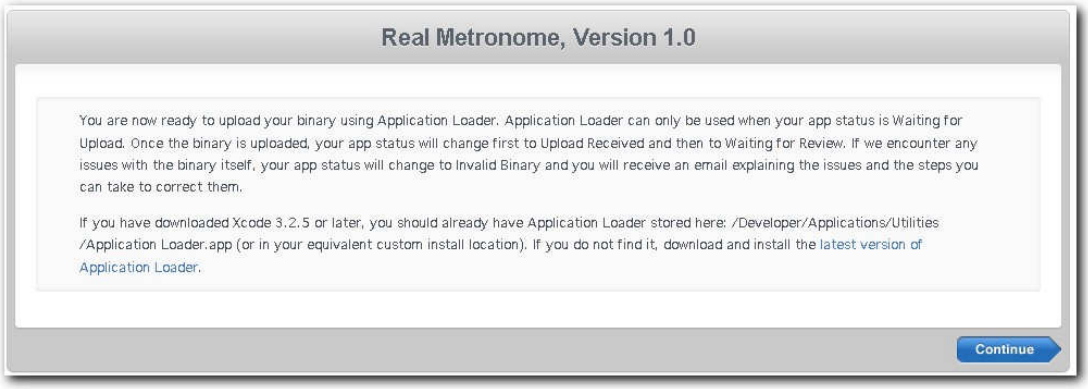
This screen is the final step before you can upload your binary:



Click Ready to Upload Binary. Your final question concerns cryptography:



You are now ready to upload. This screen explains the upload mechanism. The software for performing the upload is part of Xcode 3.2.5 or later:



At some point you will receive an email confirming that you have finished preparing for upload and your app status is "Waiting for Upload". You should now take the appropriate steps for your IDE to upload the prepared app.

Once your application is uploaded its status becomes "Waiting for Review". An email will be sent to confirm this status.

All you can do now is wait.

Review Time

Reviews may take a couple of weeks and applications may be rejected without reason. The folk at Mobile Orchard provide an excellent list of things to consider when your application is rejected. It is worth perusing before submitting your application:

http://mobileorchard.com/avoiding-iphone-app-rejection-from-apple/

http://mobileorchard.com/avoiding-iphone-app-rejection-part-2/

It is interesting how many times the issue of HIG (Human Interface Guidelines) arises, even when a specific human interface issue is not mentioned in the guidelines. One issue mentioned is that the 512x512 iTunes Store icon must match the 57x57 icon displayed on the iPhone. Failure to do so has resulted in application rejections.

Good luck with your application.

Index

A

activation email, 16
Apache Software Foundation, 13
App ID, 17
App Store, 45
App Store Review Guidelines, 39
AppleWWDRCA.cer, 22
application metadata, 48
application submission time, 51

B

Breakpoint Navigator, 31
Bundle ID, 46
Bundle Identifier, 17
Bundle Seed ID, 17

C

certificate, 18, 21
Certificate Assistant, 19
Certificate Signing Request, 19
Code signing, 18
Connecting to a Mac, 5
Cordova, 13
CSR, 19, 20

D

Debug Navigator, 31
Developer Center, 9, 20
Developer Profile, 24
Developer Program, 16
  activation email, 16
Developer Program Resources, 20
development certificate, 18
development provisioning profile, 18
Device ID, 17
digital signing, 18
distribution certificate, 18, 23
distribution provisioning, 42
Documentation
  Xcode, 12
downloads
  Dev Center, 9

E

EULA, 48

F

file sharing, 6
file type
  cer, 22
folder sharing, 6

G

Game Center, 23

I

iCloud, 23
icons, 49
In-App Purchase, 23
Intermediate Certificates, 21
Internet & Wireless, 6
iOS Developer Program, 16
iOS distribution certificate, 23
iOS Provisioning Portal, 15
iOSTeam Provisioning Profile, 24
Issue Navigator, 31
iTunes Connect, 16, 39

K

Keychain, 17, 19, 20, 33

L

LAN, 7
launch screen, 36
Log Navigator, 31

M

Mac
  folder sharing, 6
Mac setup
  PhoneGap, 12
Member Center, 8
Metadata, 48

N

Navigators
  Breakpoint, 31
  Debug, 31
  Issue, 31
  Log, 31
  Project, 29

Search, 30
Symbol, 30
Xcode, 29
Nitobi, 13

O

Organiser, 15, 24, 26
OS X Lion, 5

P

package, 34
PhoneGap, 12
pricing matrix, 46
Pricing Tier, 46
private/public key pair, 18, 20
Product Rating, 47
Product Version, 47
Project Navigator, 29, 31
provisioning, 42
Provisioning, 17
Provisioning Portal, 15
provisioning profile, 18, 25
provisioning profile certificate, 22
public/private key pair, 18, 20
Push Notifications, 23

R

review guidelines, 39

S

screen sharing, 6
screen shots, app submission, 49
screen shots, Xcode, 25, 26
Search Navigator, 30
share
Windows, 37
simulator, 35, 37
SKU, 46
SMB, 7, 33
software images, 27
splash screen, 36
SSH, 5
Symbol Navigator, 30
System Preferences, 6

T

Terminating a Program Running under Xcode, 25

U

UDID, 17

V

VNC, 5, 6

W

wildcard App ID, 23
wildcard identifier, 18
Worldwide Developer Relations Certificate Authority, 21
WWDR, 22

X

Xcode, 5, 8, 34, 35
documentation, 12
Navigators, 29
Project info/settings, 32
Project Summary, 31
Xcode 4.6.3 Installation, 10
Xcode tools, 11

Z

ZRLE encoding, 5