Preparing Rapise for Apple iOS Mobile Testing

Rapise lets you record and play automated tests against native applications on a variety of mobile devices using Apple iOS. Rapise gives you the flexibility to test your applications on either real or simulated devices.

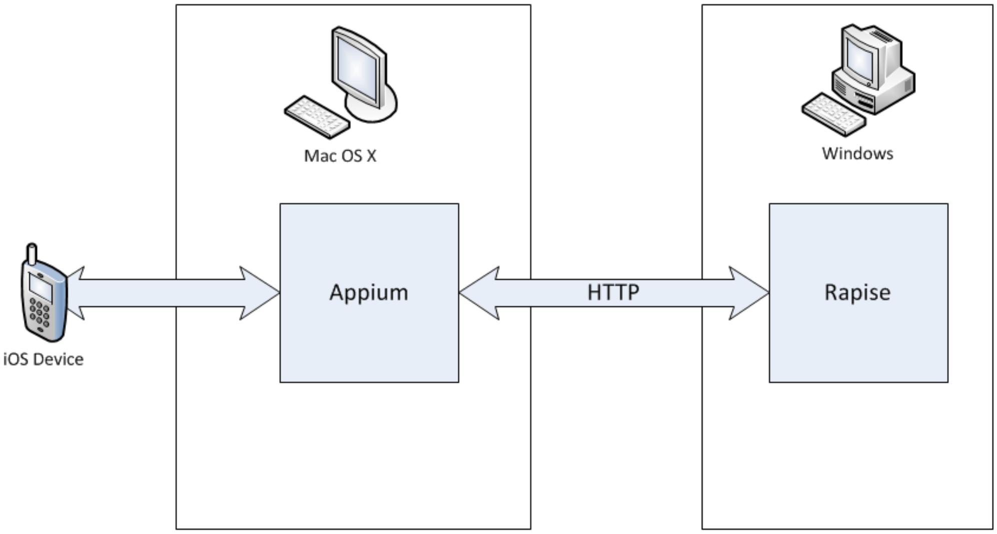
This section explains **how to setup your environment for mobile testing**, once that is done, you can the go to the section that explains the process for using Rapise to actually perform mobile testing.

Rapise uses a third-party open-source tool called **Appium** (http://appium.io) that is used to actually host the mobile devices and Rapise essentially communicates to the device via. Appium:

Testing Architecture

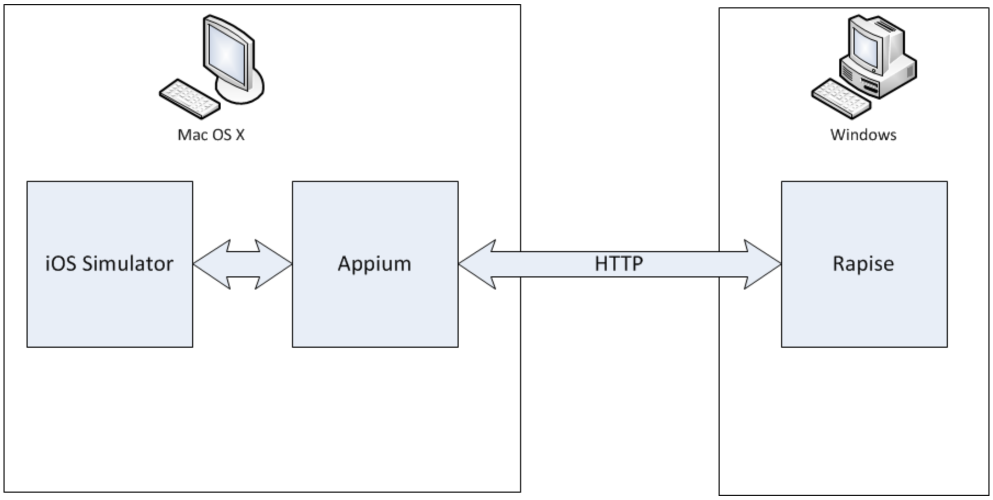
Since Rapise runs on Windows computers (PC) and iOS devices (both real and simulated) can only be tested on an Apple Macintosh (Mac) computer, it is necessary that you install **Appium** and **Apple Xcode** onto the Mac and connect to Appium over the network from Rapise running on your PC.

For Physical iOS devices the architecture looks like:



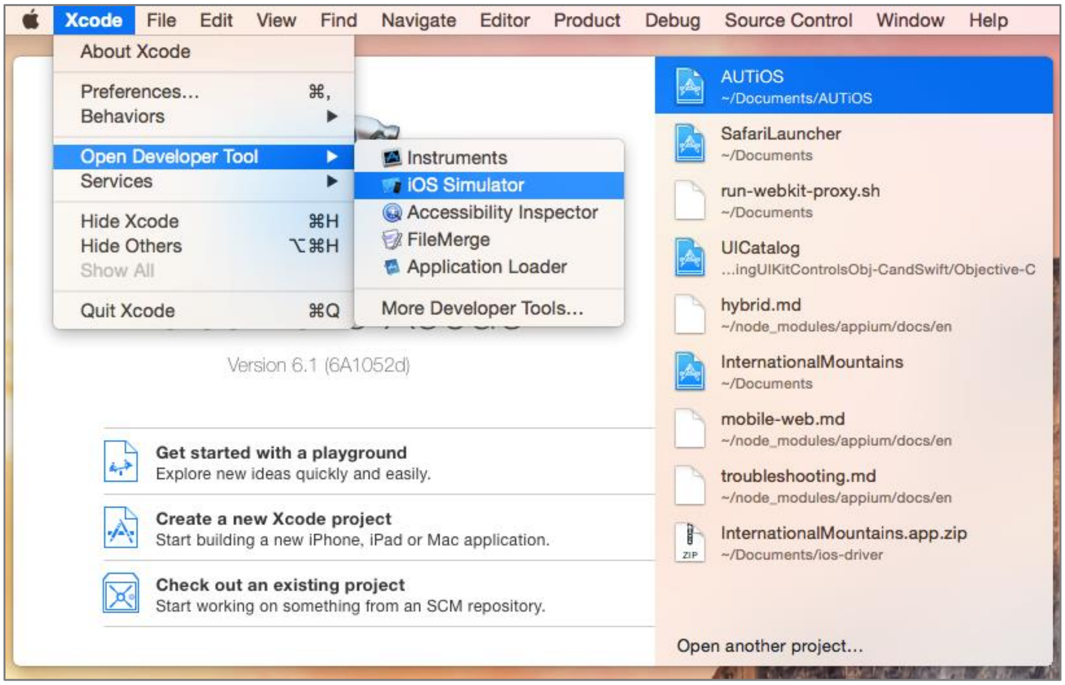
For simulated iOS devices (using the Xcode iOS Simulator) the architecture looks like:

Rapise®



Using a Mac to Host iOS Devices

The first thing you need to do is install Xcode from the Apple Mac app store. Make sure you include the **iOS SDK**, and also the **iOS Simulator** if you intend to test simulated iOS devices.


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
About Xcode
Preferences… ⌘,
Behaviors ▶
Open Developer Tool ▶
Services ▶
Hide Xcode ⌘H
Hide Others ⌥⌘H
Show All
Quit Xcode ⌘Q
Instruments
iOS Simulator
Accessibility Inspector
FileMerge
Application Loader
More Developer Tools…
Version 6.1 (6A1052d)
Get started with a playground
Explore new ideas quickly and easily.
Create a new Xcode project
Start building a new iPhone, iPad or Mac application.
Check out an existing project
Start working on something from an SCM repository.
AUTiOS
~/Documents/AUTiOS
SafariLauncher
~/Documents
run-webkit-proxy.sh
~/Documents
UICatalog
…ingUIKitControlsObj-CandSwift/Objective-C
hybrid.md
~/node_modules/appium/docs/en
InternationalMountains
~/Documents
mobile-web.md
~/node_modules/appium/docs/en
troubleshooting.md
~/node_modules/appium/docs/en
InternationalMountains.app.zip
~/Documents/ios-driver
Open another project…

(Please refer to the Apple tutorial https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/ if you are writing your first iOS application and need an introduction into how to develop for iOS).

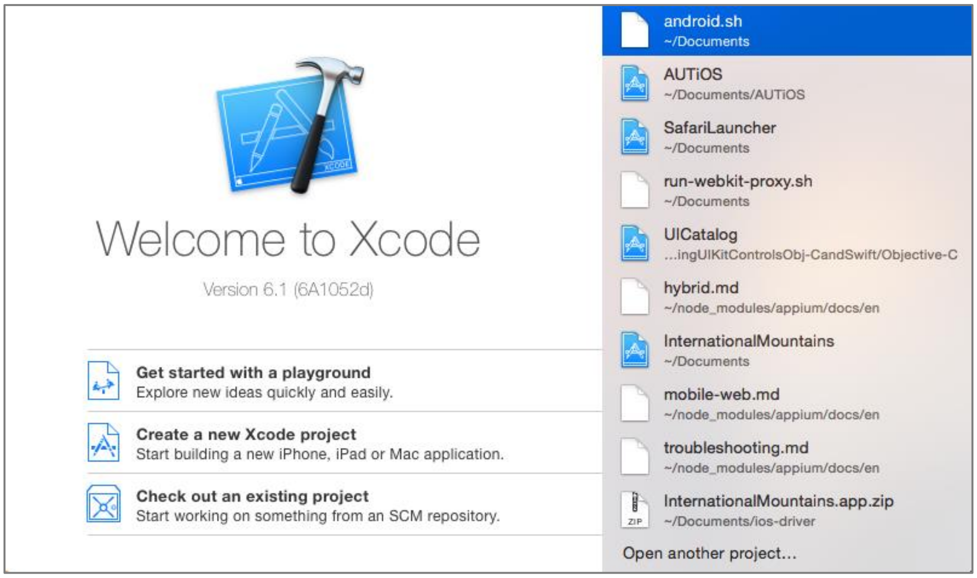
Get the AUTiOS Source Code

When you install Rapise, the sample AUT for iOS (AUTiOS) is placed in the following folder on your PC:

```
C:\Users\Public\Documents\Rapise\Samples\UsingMobile\AUT\AUTiOS
```

You will need to **copy this folder across onto your Mac** so that you can open it in Xcode.

Once you have done that, launch Xcode on the Mac:



Welcome to Xcode

Version 6.1 (6A1052d)

Get started with a playground
Explore new ideas quickly and easily.

Create a new Xcode project
Start building a new iPhone, iPad or Mac application.

Check out an existing project
Start working on something from an SCM repository.

android.sh
~/Documents

AUTiOS
~/Documents/AUTiOS

SafariLauncher
~/Documents

run-webkit-proxy.sh
~/Documents

UICatalog
...ingUIKitControlsObj-CandSwift/Objective-C

hybrid.md
~/node_modules/appium/docs/en

InternationalMountains
~/Documents

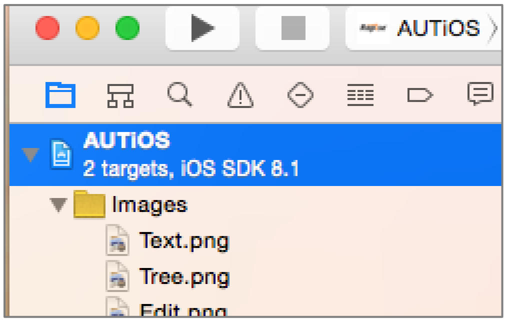
mobile-web.md
~/node_modules/appium/docs/en

troubleshooting.md
~/node_modules/appium/docs/en

InternationalMountains.app.zip
~/Documents/ios-driver

Open another project...

Open the AUTiOS project and select the root node:



Before you can actually build and deploy this project, you will need to register for an **Apple ID** and setup an Apple Developer account. You should check with your company to see if they have already joined the **Apple iOS Developer Program**, if not, you will need to join yourself and become a member. You can learn more about this at the Apple developer website: https://developer.apple.com.

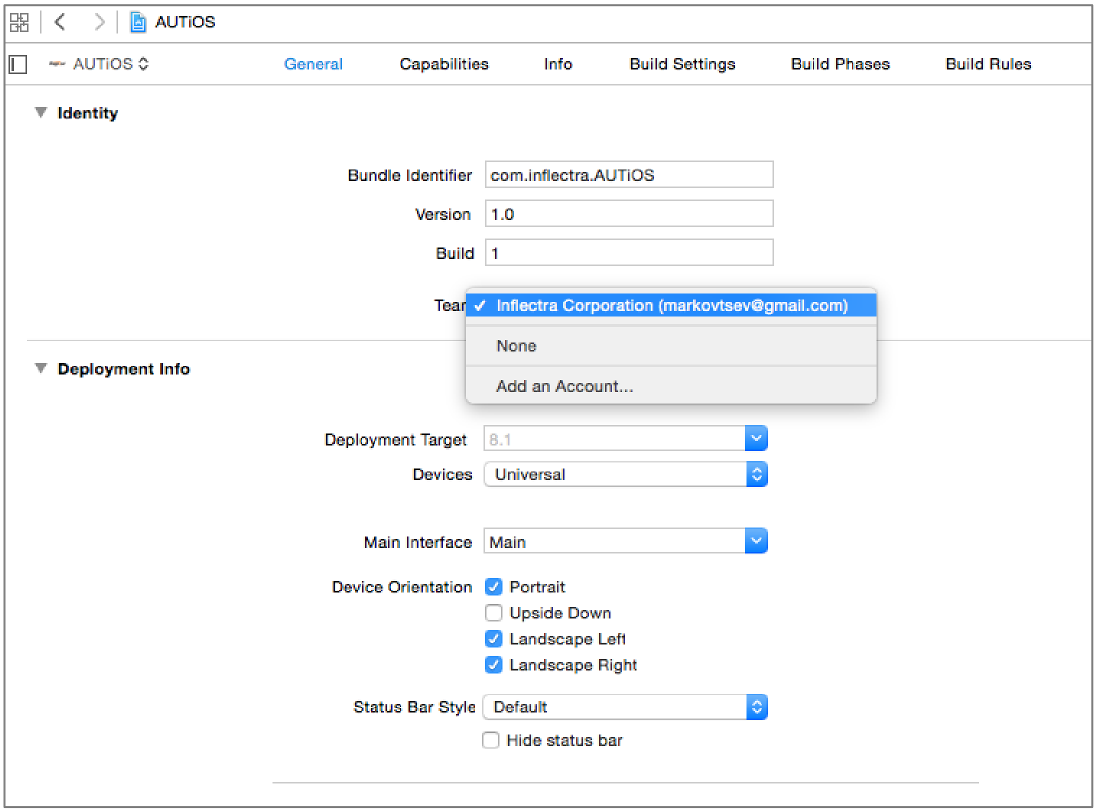
Join Your iOS Development Team

Assuming that either you or your company already has signed up for the iOS Developer Program, you will need to ask the administrator of your account (it might be you) to send an invitation to you if you are not already a member. The link for accepting such an invitation is typically:

https://developer.apple.com/programs/start/jointeam/index.php?success=%2Fios%2Finvitation%2Faccept.action

Click on this link and accept the invitation.

Meanwhile, back in XCode Use the **'Add an Account…'** to login with your **Apple ID**:

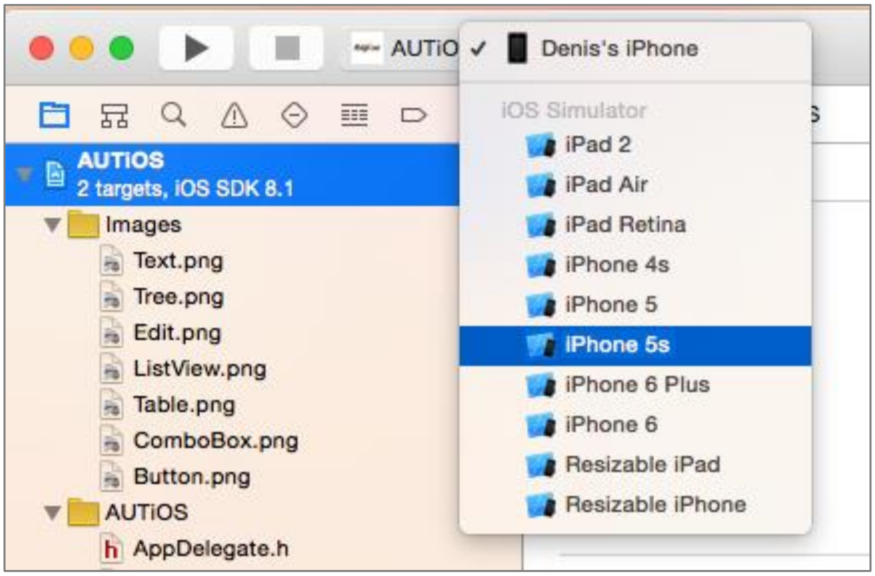


Bundle Identifier: com.inflectra.AUTiOS
Version: 1.0
Build: 1
Team: ✓ Inflectra Corporation (markovtsev@gmail.com)

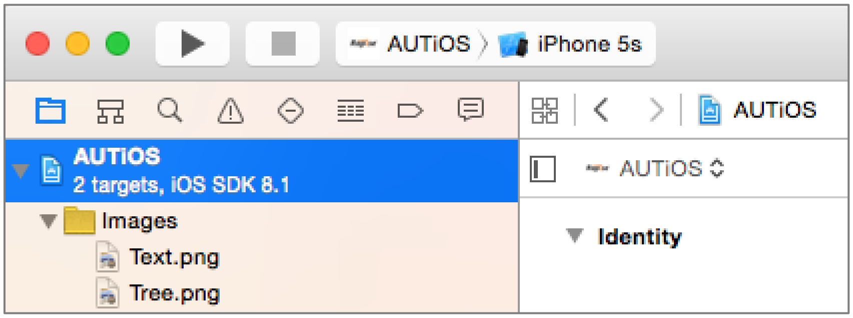
Deployment Target: 8.1
Devices: Universal
Main Interface: Main
Device Orientation: ☑ Portrait
□ Upside Down
☑ Landscape Left
☑ Landscape Right
Status Bar Style: Default
□ Hide status bar

Building and Deploying on a Simulated Device

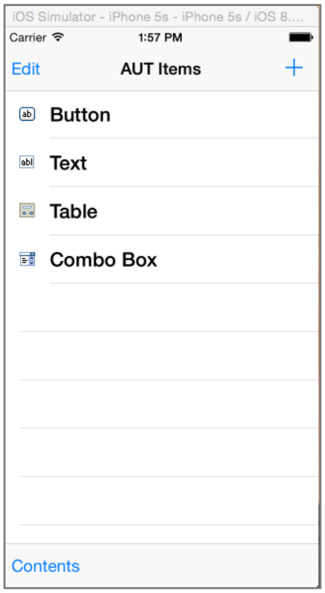
Now that you have signed into Xcode using your developer account, you can select a simulated device and run the project on it:



Once you have selected the simulated iOS device you want to use, click the **Product > Build** option to build the app for the targeted device. You can use the **Run** option to make sure that the app actually launches on this device before testing it with Rapise.



Assuming that this is successful, you will see the AUTiOS running in the iOS Simulator:

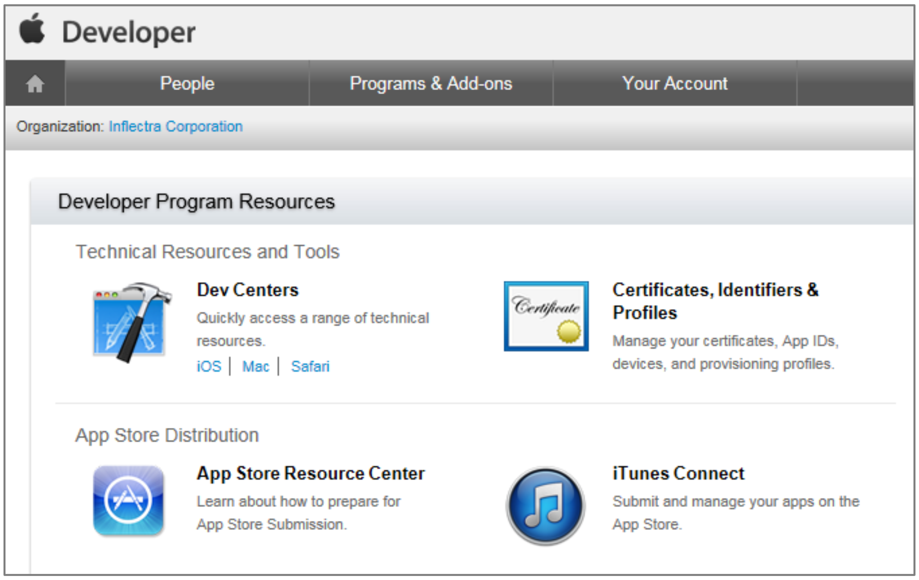


If you are only going to use Simulated devices (not recommended) then you can skip the next section and just continue with setting up **Appium**, as described in the main Mobile Testing topic.

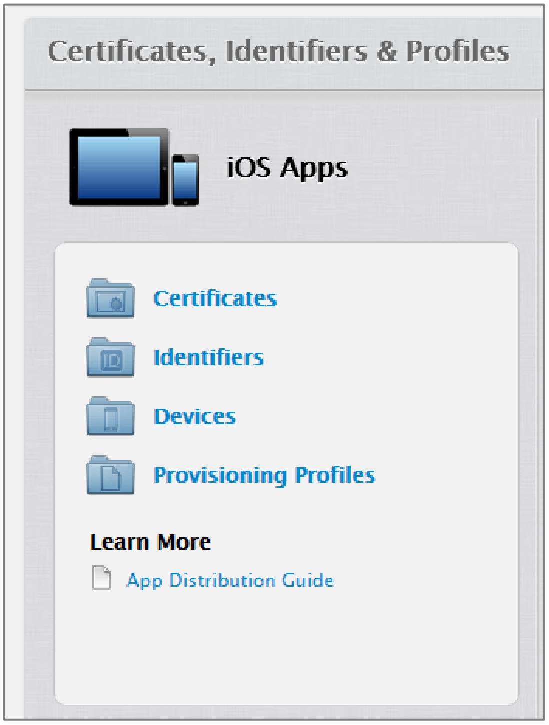
Building and Deploying on a Physical Device

Login with your **Apple ID** to http://developer.apple.com

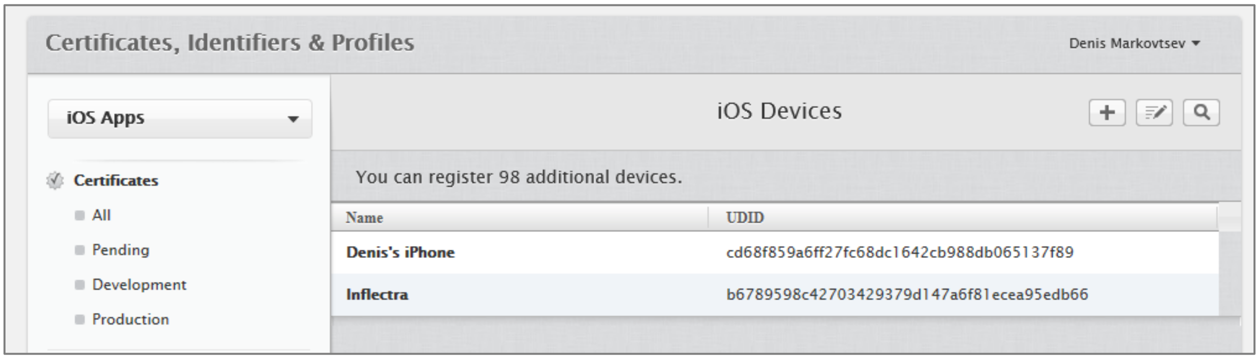
Choose Certificates, Identifiers & Profiles:


Developer
People
Programs & Add-ons
Your Account
Organization: Inflectra Corporation
Developer Program Resources
Technical Resources and Tools
Dev Centers
Quickly access a range of technical resources.
iOS | Mac | Safari
Certificates, Identifiers & Profiles
Manage your certificates, App IDs, devices, and provisioning profiles.
App Store Distribution
App Store Resource Center
Learn about how to prepare for App Store Submission.
iTunes Connect
Submit and manage your apps on the App Store.

Select Devices:



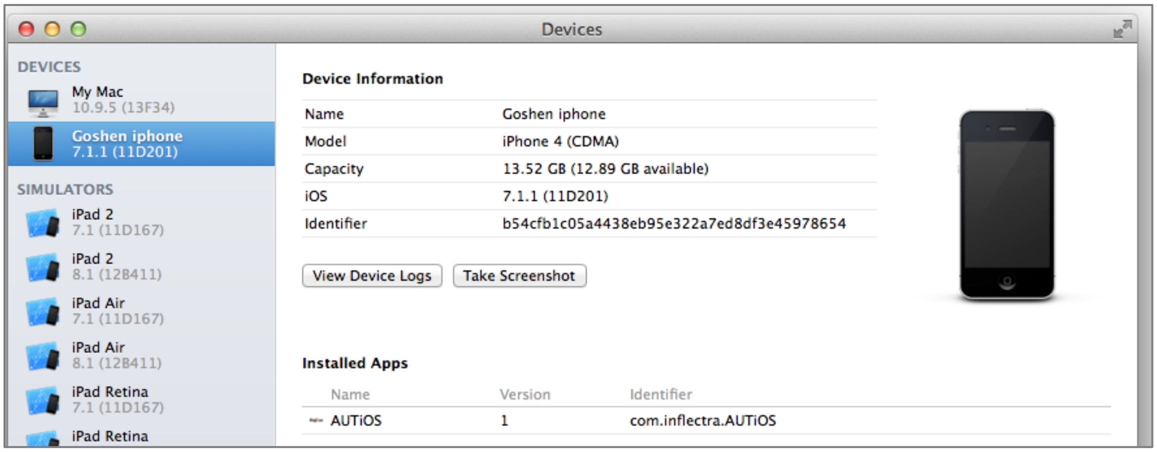
Add your device's UDID to the list of registered iOS devices in the developer account:



You can register 98 additional devices.

Name	UDID
Denis's iPhone	cd68f859a6ff27fc68dc1642cb988db065137f89
Inflectra	b6789598c42703429379d147a6f81ecea95edb66

You can find out the UDID by connecting it to the Mac and viewing the device inside Xcode.



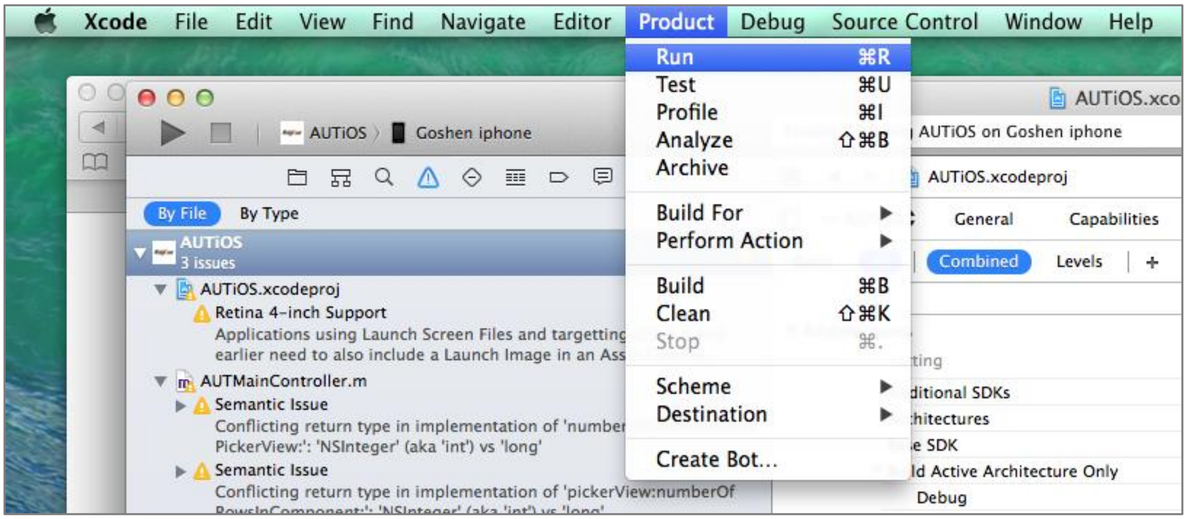
Device Information

Name	Goshen iphone
Model	iPhone 4 (CDMA)
Capacity	13.52 GB (12.89 GB available)
iOS	7.1.1 (11D201)
Identifier	b54cfb1c05a4438eb95e322a7ed8df3e45978654

Installed Apps

Name	Version	Identifier
AUTiOS	1	com.inflectra.AUTiOS

Then, back in Xcode choose your physical device, and use the **Product > Build and Run** option to test that the app launches on the device:

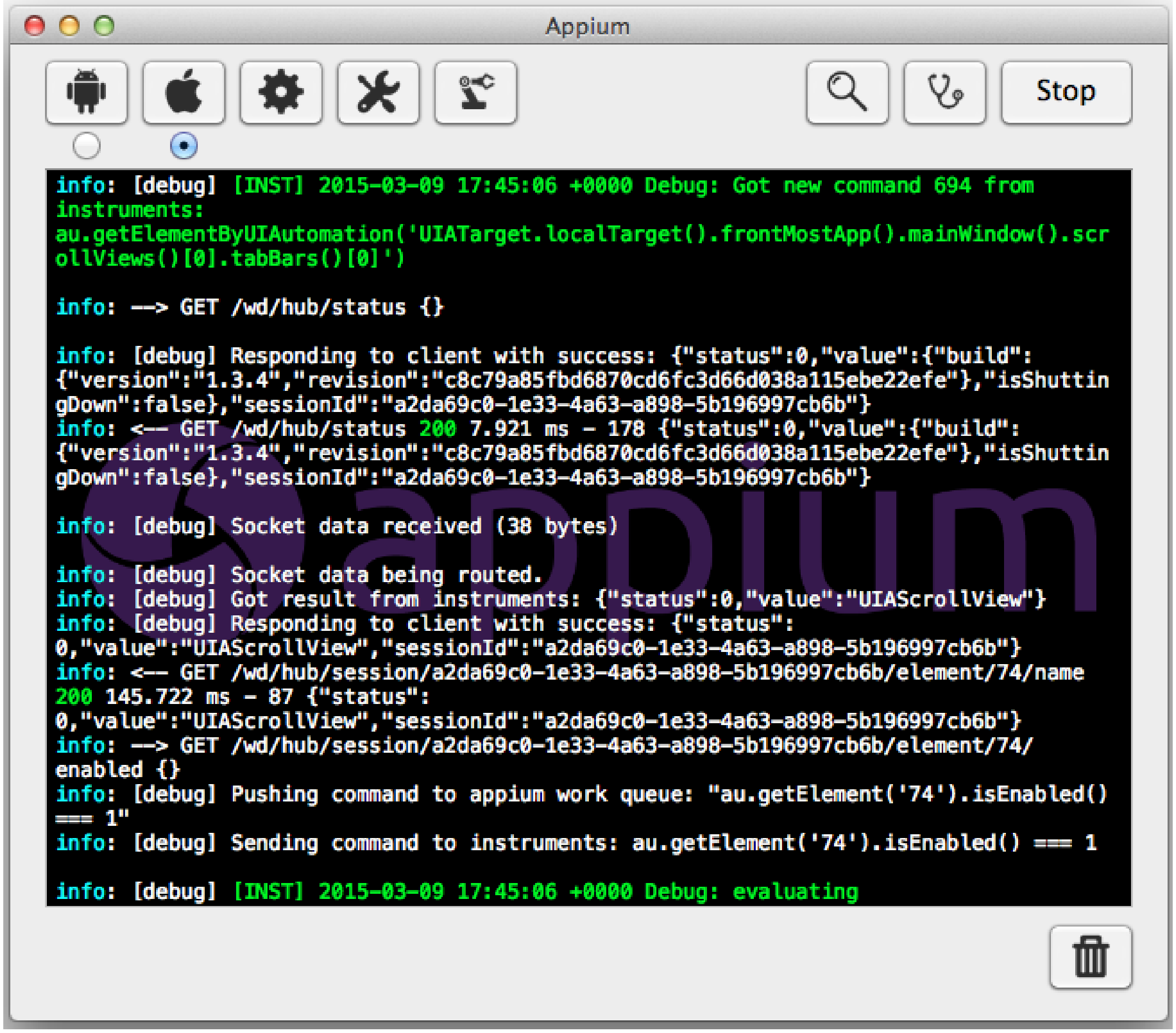


AUTiOS.xcodeproj
Retina 4-inch Support
Applications using Launch Screen Files and targetting
earlier need to also include a Launch Image in an Ass

AUTMainController.m
Semantic Issue
Conflicting return type in implementation of 'number
PickerView:': 'NSInteger' (aka 'int') vs 'long'
Semantic Issue
Conflicting return type in implementation of 'pickerView:numberOf
RowsInComponent:': 'NSInteger' (aka 'int') vs 'long'

Installing Appium on the Mac

Once you have the iOS environment configured, you need to do is go to the **Appium** website (http://appium.io) and install the latest version of Appium. Once it is installed, you need to select the option for **iOS** and click the Play button to start the Appium server:



```
info: [debug] [INST] 2015-03-09 17:45:06 +0000 Debug: Got new command 694 from
instruments:
au.getElementByUIAutomation('UIATarget.localTarget().frontMostApp().mainWindow().scr
ollViews()[0].tabBars()[0]')

info: --> GET /wd/hub/status {}

info: [debug] Responding to client with success: {"status":0,"value":{"build":
{"version":"1.3.4","revision":"c8c79a85fbd6870cd6fc3d66d038a115ebe22efe"},"isShuttin
gDown":false},"sessionId":"a2da69c0-1e33-4a63-a898-5b196997cb6b"}
info: <-- GET /wd/hub/status 200 7.921 ms - 178 {"status":0,"value":{"build":
{"version":"1.3.4","revision":"c8c79a85fbd6870cd6fc3d66d038a115ebe22efe"},"isShuttin
gDown":false},"sessionId":"a2da69c0-1e33-4a63-a898-5b196997cb6b"}

info: [debug] Socket data received (38 bytes)

info: [debug] Socket data being routed.
info: [debug] Got result from instruments: {"status":0,"value":"UIAScrollView"}
info: [debug] Responding to client with success: {"status":
0,"value":"UIAScrollView","sessionId":"a2da69c0-1e33-4a63-a898-5b196997cb6b"}
info: <-- GET /wd/hub/session/a2da69c0-1e33-4a63-a898-5b196997cb6b/element/74/name
200 145.722 ms - 87 {"status":
0,"value":"UIAScrollView","sessionId":"a2da69c0-1e33-4a63-a898-5b196997cb6b"}
info: --> GET /wd/hub/session/a2da69c0-1e33-4a63-a898-5b196997cb6b/element/74/
enabled {}
info: [debug] Pushing command to appium work queue: "au.getElement('74').isEnabled()
=== 1"
info: [debug] Sending command to instruments: au.getElement('74').isEnabled() === 1

info: [debug] [INST] 2015-03-09 17:45:06 +0000 Debug: evaluating
```

You are now ready to start mobile testing of your iOS device.