iOS Simulator User Guide

Developer

Contents

Figures and Tables

About iOS Simulator

The iOS Simulator allows you to rapidly prototype and test builds of your app during the development process. Installed as part of the Xcode tools along with the iOS SDK, iOS Simulator runs on your Mac and behaves like a standard Mac app while simulating an iPhone or iPad environment. Think of the simulator as a preliminary testing tool to use before testing your app on an actual device.

iOS Simulator enables you to simulate several iOS devices and several versions of the iOS operating system. Each simulated software version is considered its own simulation environment, independent of the others, with its own settings and files. These settings and files exist on every device you test within a simulation environment.

At a Glance

By simulating the operation of your app in iOS Simulator, you can:

- Find major problems in your app during design and early testing
- Test your app using developer tools that are available only for iOS Simulator
- Learn about the Xcode development experience and the iOS development environment before becoming a member of the the iOS Developer Program

This guide walks you through iOS Simulator, starting with the basics on how to use it and moving on to the tools found within iOS Simulator that can assist you in testing and debugging your apps.

Organization of This Document

Read the following chapters to learn how to use iOS Simulator:

- "Getting Started in iOS Simulator" (page 7), to understand the functionality of iOS Simulator, and gain a working knowledge of the various ways to launch it
- "Interacting with iOS Simulator" (page 21), to learn about the various ways of interacting with iOS Simulator, including gestures and hardware manipulation
- "Testing and Debugging in iOS Simulator" (page 30), to understand the tools available within iOS Simulator to assist you with testing and debugging your apps
- "Customizing Your iOS Simulator Experience with Xcode Schemes" (page 37), to learn about additional ways to customize your iOS Simulator experience through Xcode schemes

See Also

Apple provides these related documents that you may find helpful:

- To learn the basics of developing iOS apps, see *Start Developing iOS Apps Today*.
- To learn more about how you can customize your development experience within Xcode, see *Xcode Overview*.
- To learn about the process of testing your app on a device, submitting it to the App Store, and distributing it, see *App Distribution Guide*.

Getting Started in iOS Simulator

The iOS Simulator app, available within Xcode, presents the iPhone or iPad user interface in a window on your Mac computer. You interact with iOS Simulator by using the keyboard and the mouse to emulate taps, device rotation, and other user actions.

This chapter explains the basic capabilities of iOS Simulator. You will use an iPhone app to gain some hands-on experience with iOS Simulator, allowing you to become familiar with the tool. If you do not have an iPhone app to use, you can use the *HelloWorld* app. For more detailed information on interacting iOS Simulator and using it to test and debug your apps, refer to the later chapters in this book.

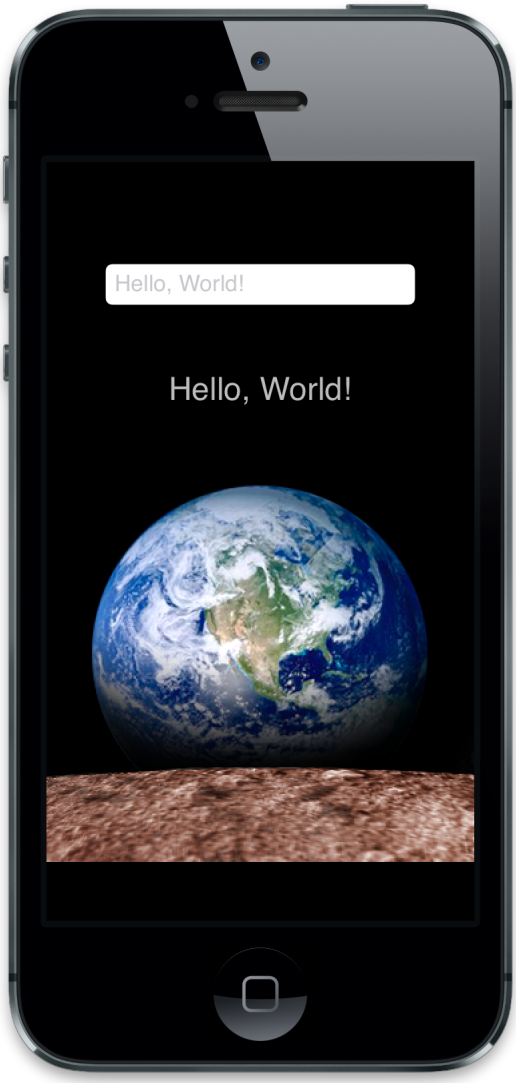
Access iOS Simulator from Xcode

There are two different ways to access iOS Simulator through Xcode. The first way is to run your app in iOS Simulator, and the second way is to launch iOS Simulator without running an app.

Running Your App in iOS Simulator

When testing an app in iOS Simulator, it is easiest to launch and run your app in iOS Simulator directly from your Xcode project. To run your app in iOS Simulator, choose iPhone Retina (4-inch) from the Xcode scheme pop-up menu, and click Run. Xcode builds your project and then launches the most recent version of your app running in iOS Simulator on your Mac's screen, as shown in Figure 1-1.

Figure 1-1 Simulated iPhone running the HelloWorld app



Launching iOS Simulator Without Running an App

At times, you may want to launch iOS Simulator without running an app. This approach is helpful if you want to test how your app launches from the Home screen of a device or if you want to test a web app in Safari in iOS Simulator.

To launch iOS Simulator without running an app

1. Launch Xcode.
2. Do one of the following:
   - Choose Xcode > Open Developer Tool > iOS Simulator.
   - Control-click the Xcode icon in the Dock, and choose Open Developer Tool > iOS Simulator from the shortcut menu.

iOS Simulator opens and displays the Home screen of whichever simulated device was last used.

View the Installed Apps

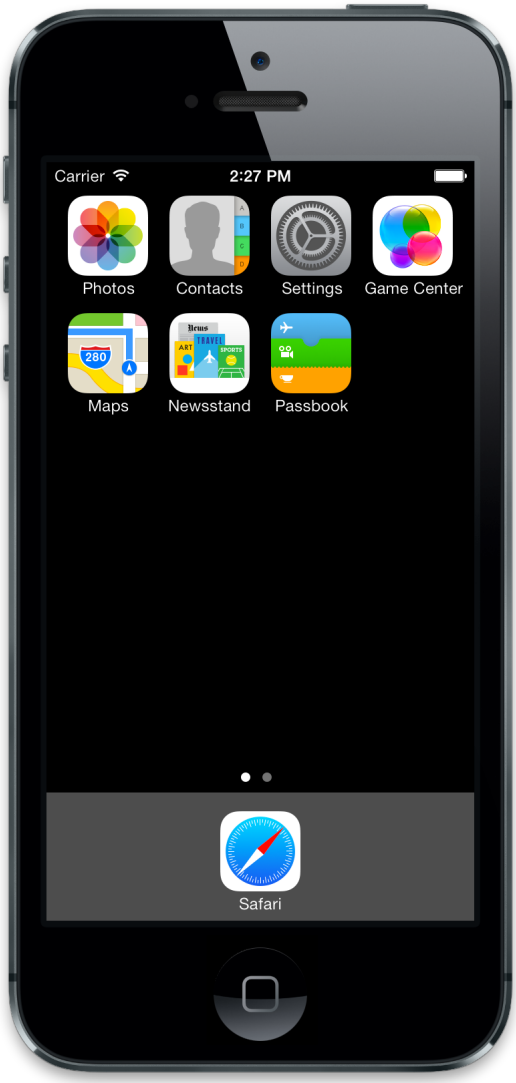
From the Home screen, you have access to all of the apps that are installed in the iOS 7.0 simulation environment. There are two ways to access the Home screen in iOS Simulator from your app:

- Click the Home button.
- Choose Hardware > Home.

Much like the Home screen on an iOS device, the simulator's Home screen has multiple pages. After clicking the Home button (or accessing the Home screen through the Hardware menu), you arrive at the second page of the Home screen. To get to the first page, where all of the preinstalled apps are found, swipe to the first Home screen by dragging to the right on the simulator screen.

On the Home screen, you see all of the apps that have been preloaded into iOS Simulator, as shown in Figure 1-2.

Figure 1-2 The iOS Simulator Home screen for a simulated iPhone in the iOS 7.0 simulation environment



The apps you see on the Home screen are specific to the simulated iPhone in the iOS 7.0 simulation environment. Because the Calendar app is available only in iOS simulator on iOS 7.0 and because Passbook is available only for the iPhone, these apps won't appear if you switch to a legacy simulator or to simulating an unsupported device type.

Use the installed apps to test your app's interaction with them. For example, if you are testing a game, you can use iOS Simulator to ensure that the game is using Game Center correctly.

Use Safari to Test Web Apps

From the Home screen you can access Safari within iOS Simulator. Use Safari to test your iOS web apps directly on your Mac.

1. From the Home screen, click Safari.
2. In the address field in Safari, type `apple.com` and press the Return key.

If your Mac is connected to the Internet, Safari displays the Apple website. See Figure 1-3.

Figure 1-3 The Apple website running in Safari in iOS Simulator

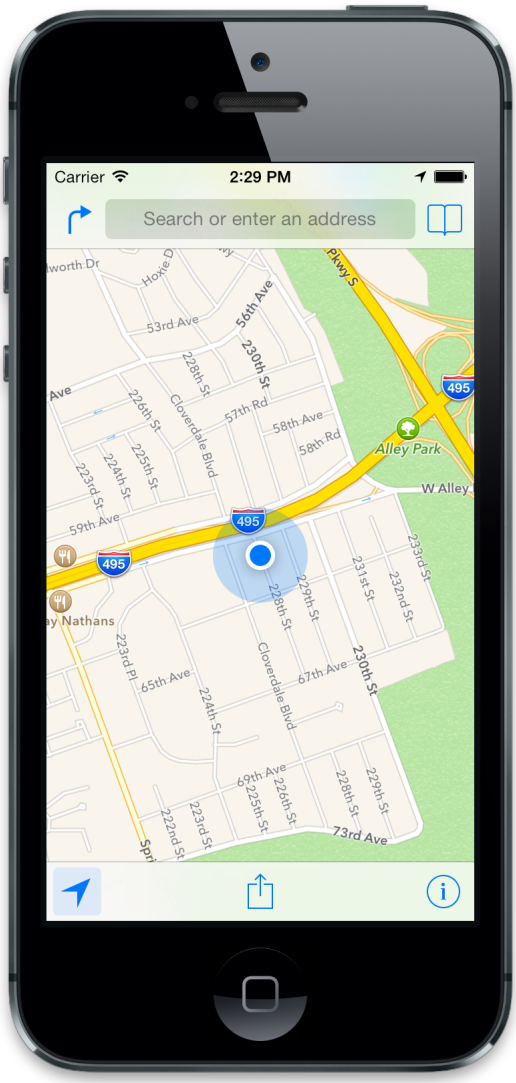
Use Maps to Simulate Location Awareness

iOS Simulator provides tools to assist you in debugging your iOS app. One of the many features you can debug in iOS Simulator is location awareness within your app. Here's an example of how to simulate a different location, which can be seen in the Maps app:

1. From the Home screen, click Maps.
2. Choose Debug > Location > Custom Location.
3. In the window that appears, type the number `40.75` in the latitude field, and the number `–73.75` in the longitude field.
4. Click OK.
5. Click the Current Location button in the bottom-left corner of the simulated device screen.

After completing this task, notice that the blue dot representing your location is in New York, NY, near the Long Island Expressway, as shown in Figure 1-4.

Figure 1-4 Running Maps and simulating a latitude and longitude in iOS Simulator



Change the Simulated Device and iOS Version

iOS Simulator provides the ability to simulate six devices:

- iPhone
- iPhone Retina (3.5-inch)
- iPhone Retina (4-inch)

- iPhone Retina (4-inch 64-bit)
- iPad
- iPad Retina

> **Note:** Use the simulated iPad to test apps for the iPad mini.

In addition to simulating various hardware devices, you can also change the simulated software version. Each simulated software version is considered its own simulation environment, and each simulation environment has its own settings and applications. There are two software versions available:

- iOS 6.1
- iOS 7.0

> **Important:** Not all hardware software combinations are available. Any limitations in hardware are mirrored in iOS Simulator.

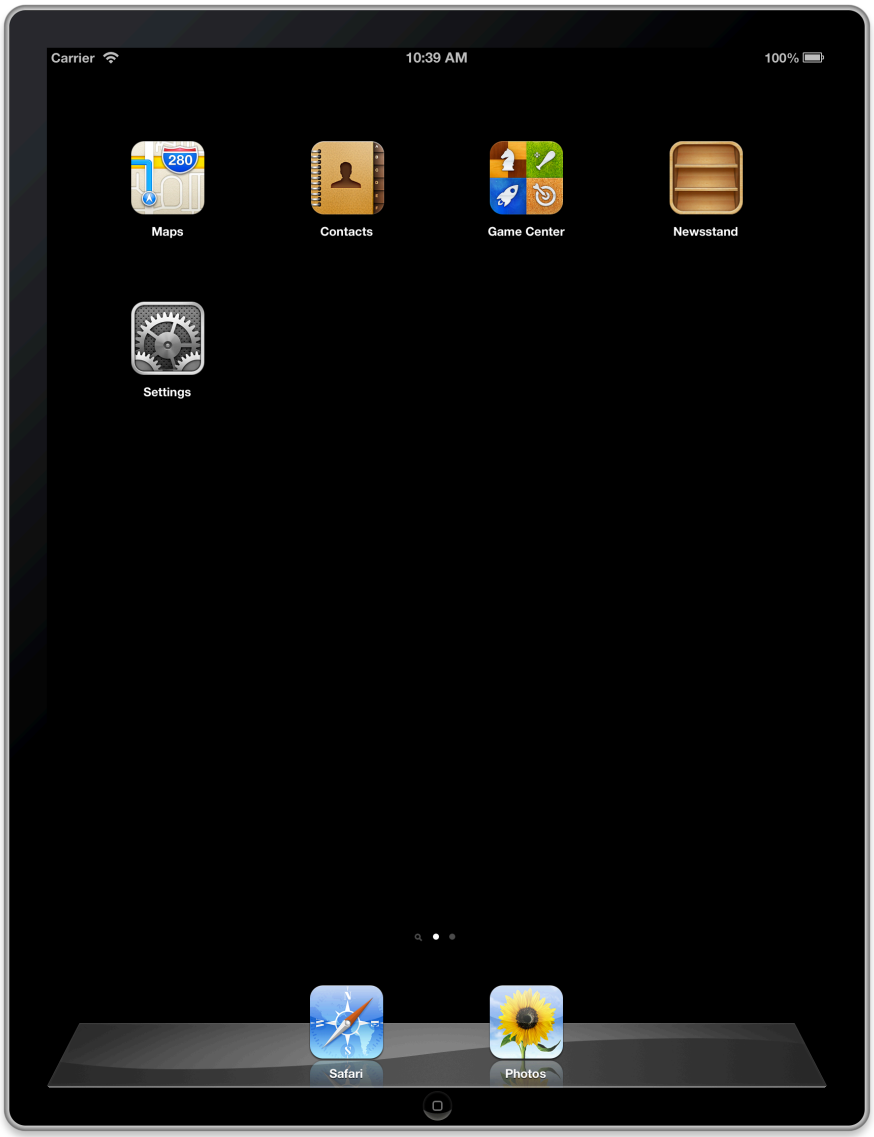
To change the device you're simulating to an iPad running iOS 6.1, choose Hardware > Device > iPad > iOS 6.1. If iOS 6.1 is not listed as an option in the menu, you need to download the legacy simulator in Xcode and then relaunch iOS Simulator.

To download a legacy simulator

1. In Xcode, choose Xcode > Preferences.
2. In the Preferences window, click Downloads.

3. In Components, find the legacy simulator version you want to add, and click the Install button.

Figure 1-5 iOS Simulator displays a simulated iPad running iOS 6.1



The iOS Simulator display you saw in Figure 1-2 (page 10) changes from an iPhone running iOS 7.0 to an iPad running iOS 6.1, as shown in Figure 1-5.

Because the Passbook app is available only on iPhone, its icon, which was visible in Figure 1-2 (page 10) for iPhone, is not visible in Figure 1-5 for iPad. Additionally, since the new user interface (UI) was first introduced in iOS 7.0, notice that the simulated iPad running iOS 6.1 is now displaying the old UI and icon designs.

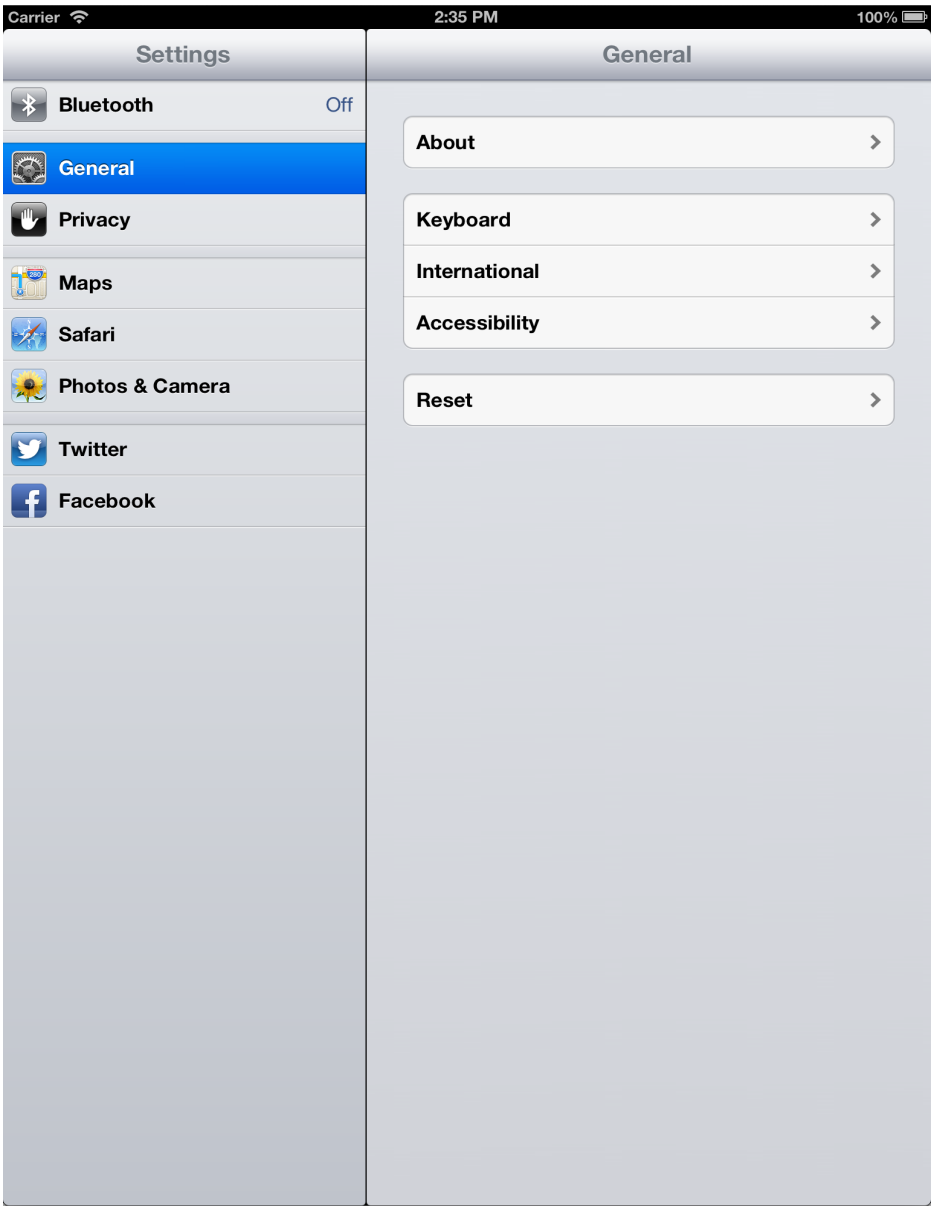
If you are using a computer whose screen isn't large enough to display an entire simulated iPad, only the screen of the iPad is displayed, not the device border around it. If this is true for you, you must access the Home screen though the Hardware menu.

If you are testing an iPad app, you can test only on a simulated iPad. If you are testing an iPhone or a universal app, you can test on either a simulated iPhone or a simulated iPad.

Alter the Settings in iOS Simulator

You can alter the settings within iOS Simulator to help test your app. To open the Settings app in iOS Simulator, go to the Home screen in iOS Simulator and click Settings. In Figure 1-6 you see the Settings app as it appears when launched in the iOS 6.1 simulation environment.

Figure 1-6 Example of the Settings app in a simulated iPad device



The iOS Simulator settings differ from the settings found on a hardware device. iOS Simulator is designed for testing your apps, whereas a hardware device is designed for use. Because iOS Simulator is designed for testing apps, its settings are naturally focused on testing too. For example, in iOS Simulator the Accessibility menu provides the ability to turn on the Accessibility Inspector, and the Accessibility menu on a device allows you to turn on and off different accessibility features.

Through the settings, you can test both accessibility and localization of your app. See "Testing and Debugging in iOS Simulator" (page 30) for information on how to manipulate your settings for the various types of testing you are interested in.

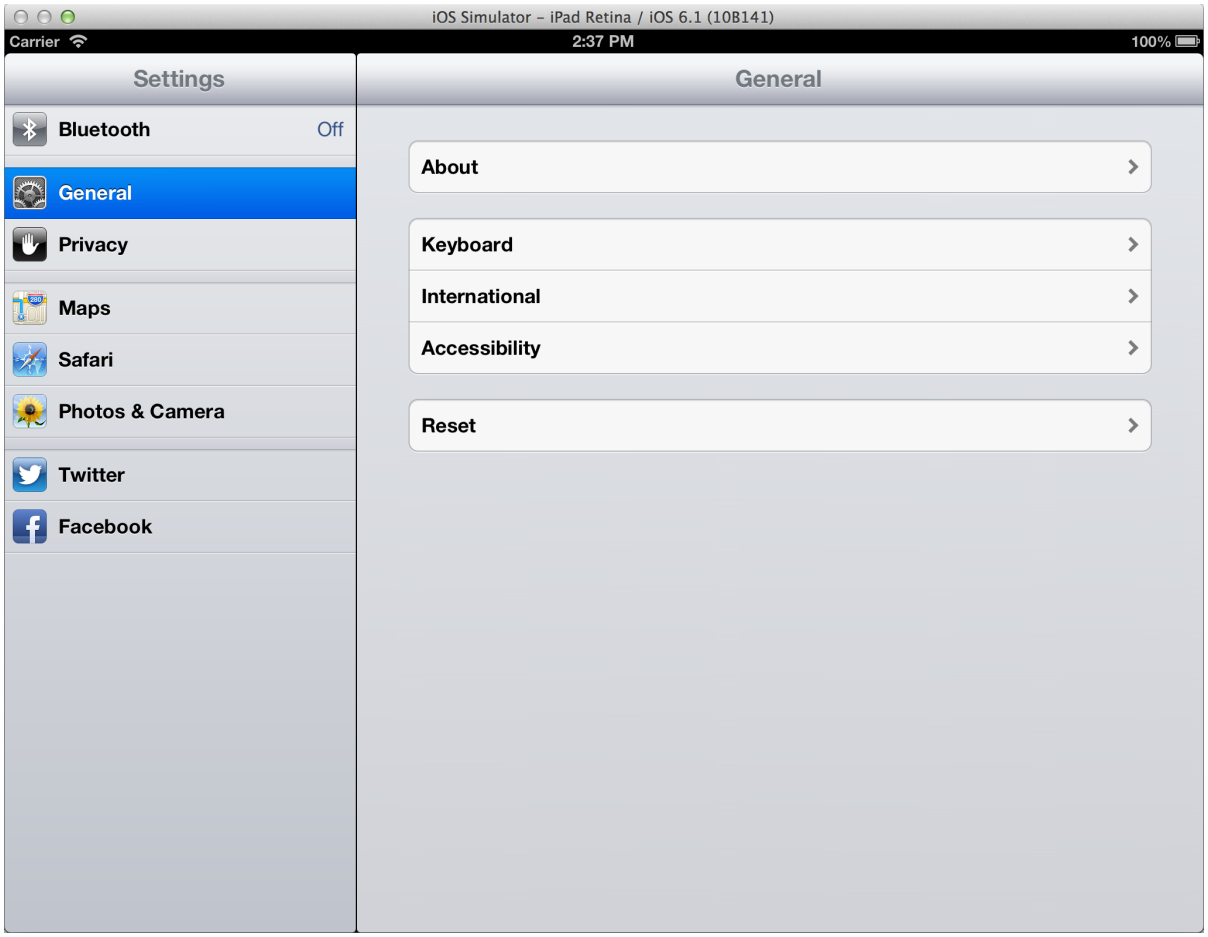
> **Remember:** Changes made in the Settings app of iOS Simulator affect only the simulation environment that is currently running.

Rotate the Device

You can use iOS Simulator to manipulate the simulated device much as you do a physical device.

To rotate your simulated device, choose Hardware > Rotate Left. When you rotate your simulated device, you see that Settings rotates (see Figure 1-7), just as it would on a hardware device.

Figure 1-7 A rotated simulated iPad running in the iOS 6.1 simulation environment



Test in iOS Simulator and on a Device

iOS Simulator is designed to assist you in designing, rapidly prototyping, and testing your app, but it should never serve as your sole platform for testing. One reason is that not all apps are available in the simulator. For example, the Camera app is available only on hardware devices and cannot be replicated in the simulator.

In addition, not all bugs and performance problems can be caught through testing in iOS Simulator alone. You'll learn more about performance limitations in "Testing and Debugging in iOS Simulator" (page 30). You can also find more information on testing your app on a device in "Launching Your App on Devices" in *App Distribution Guide*.

Quit iOS Simulator

iOS Simulator continues running until you quit it. Even if you quit Xcode, iOS Simulator continues to run because it is a separate app. To quit iOS Simulator, choose iOS Simulator > Quit iOS Simulator. If Xcode is running, Xcode remains open.

Interacting with iOS Simulator

Interacting with iOS Simulator differs from interacting with an actual device. In this chapter you learn how to:

- Simulate hardware actions such as rotate and shake
- Simulate Multi-Touch gestures using a mouse and keyboard
- Uninstall an app you previously installed in a simulation environment
- Copy and paste text and images between the simulator and your Mac

Simulating Hardware Interactions

With iOS Simulator, you can simulate most of the actions a user performs on a device. Table 2-1 lists hardware manipulations you can perform in iOS Simulator by using the Hardware menu.

Table 2-1 Manipulating iOS Simulator from the Hardware menu

Menu option	Hardware action
Rotate Left	Rotates the simulator to the left.
Rotate Right	Rotates the simulator to the right.
Shake Gesture	Simulates shaking the device.
Home	Displays the Home screen of the simulated device.
Lock	Displays the Lock screen.
Simulate Memory Warning	Sends the frontmost app a simulated low-memory warning. For information on how to handle low-memory situations, see "Observing Low-Memory Warnings" in *iOS App Programming Guide*.
Toggle In-Call Status Bar	Toggles the status bar between its normal state and its in-call state. This command shows how your app's user interface looks when a user launches your app during a call or while navigation is running. The in-call state bar is used when a phone call is in progress, a FaceTime call is in progress, or Maps in iOS 6 is navigating. The status bar is taller in its in-call state than in its normal state.

Menu option	Hardware action
Simulate Hardware Keyboard	Toggles the software keyboard on an iPad simulator. Turn off the software keyboard to simulate using a keyboard dock or wireless keyboard with an iPad device. **Note:** The Mac's keyboard can be used as input into the simulator at all times.
iOS Uses Same Keyboard Layout As OS X	Automatically selects the iOS keyboard that most closely matches the keyboard layout of your Mac.
TV Out	Opens a window simulating a device's TV Out signal. **Note:** Several TV Out resolutions are available by choosing Hardware > TV Out > *desired resolution*.

Simulating User Gestures

With iOS Simulator, you can perform traditional Multi-Touch gestures using the mouse and keyboard. Table 2-2 lists gestures you can perform in iOS Simulator. See *iOS Human Interface Guidelines* for more about gestures.

> **Note:** All gestures can be performed using a mouse or a trackpad.

Table 2-2 Performing gestures in iOS Simulator

Gesture	Desktop action
Tap	Click.
Touch and hold	Press and hold down the mouse button or trackpad.
Double-tap	Double-click.
Drag	Drag.
Swipe	Drag.
Flick	Drag quickly.

Gesture	Desktop action
Two-finger Drag	1. Place the pointer where you want the two-finger drag to occur. 2. Hold down the Option key. 3. Move the circles that represent finger touches to the start position. 4. Move the center of the pinch target by holding down the Shift key, moving the circles to the desired center position, and releasing the Shift key. 5. Hold down the Shift key and the mouse button, move the circles in the direction you want to drag, and release both the Shift key and the mouse button.
Pinch	1. Place the pointer where you want the pinch to occur. 2. Hold down the Option key. 3. Move the circles that represent finger touches to the start position. 4. Move the center of the pinch target by holding down the Shift key, moving the circles to the desired center position, and releasing the Shift key. 5. Hold down the mouse button, move the circles in and out to the end position, and release the Option key.
Rotate	1. Place the pointer where you want the rotation to occur. 2. Hold down the Option key. 3. Move the circles that represent finger touches to the start position. 4. Move the center of the pinch target by holding down the Shift key, moving the circles to the desired center position, and releasing the Shift key. 5. Hold down the mouse button, rotate the circles to the end position, and release the Option key.

Simulating Keyboards in iOS Simulator

In order for you to most accurately simulate a device on iOS Simulator, the simulator uses iOS keyboard layouts, as opposed to OS X keyboard layouts. If you have the option, iOS Uses Same Keyboard Layout As OS X, selected, iOS simulator selects the keyboard that most closely matches the keyboard layout of your Mac. For most cases you want to leave this checked, but if you do feel a need to disable it—allowing you to select completely different keyboard layouts for your Mac and iOS Simulator—simply choose Hardware > iOS Uses Same Keyboard Layout As OS X.

In addition to using the keyboard that most closely matches your Mac keyboard layout, you can also manually select a keyboard layout through the iOS Simulator settings. This approach can be helpful if you're using a keyboard layout that iOS Simulator cannot automatically associate with a keyboard.

To change a hardware keyboard layout in iOS Simulator

1. Open Settings from the Home screen, and select General.
2. Select Keyboard.
3. Select Keyboards.
4. Select the keyboard language you want to change the hardware layout for.
5. Under Hardware Keyboard Layout, select the desired keyboard layout.

Installing and Uninstalling Apps

When you build your app for iOS Simulator, Xcode automatically installs it in the selected simulation environment. In the same way that the Maps app disappears when you switch from the 6.0 simulation environment to the 5.1 simulation environment, your own apps exist only within the specific simulation environments where you installed them.

Note: You cannot install apps from the App Store in simulation environments.

To uninstall apps that you have installed in a simulation environment

1. Select the simulation environment from which to remove the app by choosing Hardware > Version > iOS version.
2. Place the pointer on the icon of the app you want to uninstall, and then press and hold down the mouse button or trackpad until the icons start to jiggle and a close button appears.
3. Click the close button on the app you want to uninstall from the simulation environment.
4. Click the Home button, or if the Home button is not visible, choose Hardware > Home to stop the icons from jiggling.

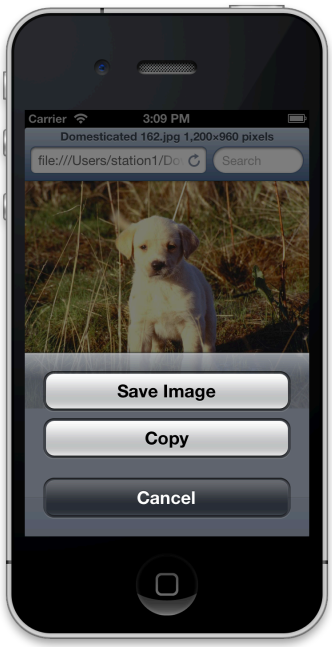
Copying and Pasting in iOS Simulator

iOS Simulator provides a variety of copy and paste operations, both within the simulator and between the simulator and your Mac. The actual copy and paste operations in iOS Simulator are performed in the same way they are on an iOS device, but if you are trying to copy and paste between the simulator and your Mac, additional steps must be taken. Copy and paste operations can be used on strings and images.

If you are copying an image from a webpage in iOS Simulator, you need to save it to the Photos app first.

To save an image from a webpage to the Photos app

1. Press and hold down the mouse button or trackpad on the image you want to save.
2. When the menu appears, click Save Image to save the image to the Photos app in iOS Simulator.

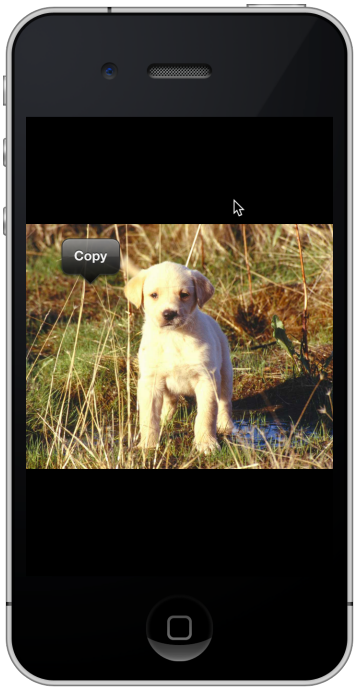


The image is saved to the Saved Photos album in the Photos app.

To copy an image in iOS Simulator

1. Open the photo you want to copy in the Photos app.

2. Press and hold down the mouse button or trackpad on the image you want to copy.

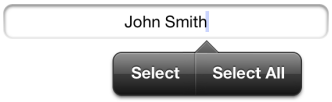


3. Click Copy.
4. If you intend to paste the image on your Mac outside of iOS Simulator, choose Edit > Copy.

   This copies the image to the Mac's pasteboard. To paste the image in another app on the Mac, use that app's paste command.

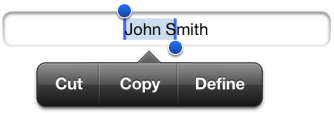
To copy text in iOS Simulator

1. Click the insertion point to display the selection buttons.



2. Click the Select button to select the adjacent word, or click Select All to select all text.
3. Drag the grab points to select more or less text.

4. Click Copy.



5. If you are pasting the text on your Mac outside of iOS Simulator, choose Edit > Copy.

   This copies the text to the Mac's pasteboard. To paste the text in another app on the Mac, use that app's paste command.

To paste an image into iOS Simulator

1. If the image you are pasting was copied from your Mac, choose Edit > Paste.

   This action copies the image from the Mac's pasteboard to the simulator's pasteboard.

2. Navigate to the location where you want to paste the image you just copied.
3. Press and hold down the mouse button or trackpad in the location where you want to paste your copied image, and then click Paste.

To paste text into iOS Simulator

1. If the text you are pasting was copied from your Mac, choose Edit > Paste.

   This action copies the text from the Mac's pasteboard to the simulator's pasteboard.

2. Navigate to the location where you want to paste the text you just copied.
3. Double-click in the location where you want to paste the text, and then click Paste.

Taking a Screenshot of the Simulator

In iOS Simulator you can copy a screenshot of the simulator to your Mac's pasteboard, or you can save a copy of the screenshot as a file.

- To take a screenshot and save it to your Mac's pasteboard, choose Edit > Copy Screen.
- To save a screenshot as a file, choose File > Save Screen Shot. The screenshot is saved to the desktop of your Mac.

Viewing the Simulated Device's Screen

Even though iOS Simulator runs on all Macs, how it appears may differ between models. If you are using a computer whose screen is not large enough to display the entire simulator, only the screen of the simulated iOS device is shown, not the device border around it.

You can scale iOS Simulator by choosing Window > Scale > *percentage of choice*, but scaling shows only the device screen for any size other than 100 percent. For other scaling sizes, only the screen of the device is simulated, as shown in Figure 2-1.

Figure 2-1 iOS Simulator scaled to 100 percent (left) and 75 percent (right)

Testing Retina and Non-Retina Display Devices

With iOS Simulator, you can simulate iOS devices both with and without Retina displays, regardless of whether you have a Mac with Retina display.

> **Note:** The following equivalencies pertain when iOS Simulator is scaled to 100 percent; when scaled differently, the equivalencies scale in the same manner.

When working on a Mac without a Retina display, the simulator is mapped from pixel to pixel instead of from point to point. When simulating an app for an iOS device with Retina display on a Mac without a Retina display, the simulator appears twice as large as it would for a non-Retina display app to account for the extra pixels in a Retina display.

When working on a Mac with Retina display, your computer maps each point in the iOS app to a point on the Mac's screen. If the simulated app is for an iOS device with a Retina display, each point is composed of 1 pixel. If the app being simulated is for an iOS device without a Retina display, each point is composed of 2 pixels.

To learn more about mapping points to pixels, see "Points Versus Pixels" in *Drawing and Printing Guide for iOS*.

Testing and Debugging in iOS Simulator

iOS Simulator is a great tool for rapid prototyping and development before testing your app on a device. iOS Simulator also has features that can assist you in testing and debugging both iOS apps and web apps. By understanding the tools that iOS Simulator offers, you can more efficiently develop your app.

Limitations of Testing in iOS Simulator

Even though iOS Simulator is a useful tool, never make it the only way you test an app. Because iOS Simulator is an app running on a Mac, it has access to the computer's memory, which is much greater than the memory found on a device. As a result of the increased memory, iOS Simulator is not an accurate test of an app's memory usage. For this same reason, you should always test the performance of your app's user interface (UI) on a device. In iOS Simulator, your app's UI may appear to run both faster and smoother than on a device.

Also keep in mind that some UI elements can be easier to interact with in iOS Simulator using a mouse than when trying to interact with the app via touch on a device.

Finally, there are some hardware and API limitations of iOS Simulator. These limitations may affect your app when testing in iOS Simulator.

Hardware Limitations

While most of the functionality of iOS devices can be simulated in iOS Simulator, there are some hardware features that must be tested directly on a device. The hardware features that cannot be simulated are:

- Accelerometer
- Gyroscope
- Camera
- Proximity Sensor
- Microphone Input

To test your app on a device, you must be a member of the iOS Developer Program. To learn more about enrolling in the iOS Developer Program, see "Managing Accounts" in *App Distribution Guide*.

OpenGL ES Limitations

iOS Simulator includes complete implementations of OpenGL ES 1.1, 2.0, and 3.0 that you can use to start developing your app. The capabilities of iOS Simulator are similar to those of the A7 GPU; for more information on the iOS hardware, see *iOS Device Compatibility Reference* . iOS Simulator differs from the hardware processor in a few ways:

- iOS Simulator does not use a tile-based deferred renderer.
- iOS Simulator does not provide a pixel-accurate match to the graphics hardware.
- Rendering performance of OpenGL ES in iOS Simulator has no relation to the performance of OpenGL ES on an actual device.

Important: The OpenGL ES support in iOS Simulator should be used to help you get started writing an OpenGL ES app. Never assume that iOS Simulator reflects the real-world performance or the precise capabilities of the graphics processors used in iOS devices. Always profile and optimize your drawing code on a real device.

API Limitations

Within iOS Simulator, there are some limitations to the APIs and features, including:

- Apple Push Services
- Privacy alerts for access to Photos, Contacts, Calendar, and Reminders
- The `UIBackgroundModes` key
- iCloud document syncing and key-value storage support

Unsupported frameworks include:

- External Accessory
- Media Player
- Message UI
- Event Kit
- In UIKit, the `UIVideoEditorController` class

iOS Version Limitations

iOS Simulator does not have complete backward compatibility. In addition to supporting iOS 7.0, iOS Simulator supports iOS 6.1.

To use iOS Simulator for iOS 7.0, you must be running Xcode 5.

Testing for the iPad mini

Although an iPad mini isn't an option for a simulated device in iOS Simulator, you can still test apps for the iPad mini in the simulator. To do this, run your app on the simulated iPad without Retina display.

Testing for 64-bit Compatibility

iOS Simulator includes the ability to simulate 64-bit hardware. Choose one of the simulator targets that includes *64-bit* in the name; your app is automatically compiled using the 64-bit data types described in *64-Bit Transition Guide for Cocoa Touch*. Use Simulator to test any changes in your code that were necessary to support the new data sizes and alignments.

Some changes in the 64-bit ABI are not testable in Simulator. For example, any changes related to function or method calls can only be discovered when running on actual iOS hardware. So, as with developing any other iOS app, Simulator is best used only for the initial work to develop and test your app.

Testing App Accessibility

Use the Accessibility Inspector to test the accessibility of your app. The Accessibility Inspector displays accessibility information about each accessible element in an app. In Figure 3-1 you can see what the Accessibility Inspector looks like as it runs in iOS Simulator.

Figure 3-1 The Accessibility Inspector running on a simulated iPhone

To start the Accessibility Inspector

1. When iOS Simulator is running, click the Home button to reveal the Home screen.
2. Click Settings.
3. Go to General > Accessibility.
4. Slide the Accessibility Inspector switch to On.

> **Note:** The Accessibility Inspector remains active until you turn it off, even if you quit and restart iOS Simulator.

Turning on the Accessibility Inspector in iOS Simulator alters the behavior of the simulator. After the Accessibility Inspector is on, clicking an element moves the focus of the inspector to that element instead of activating it. To activate an element, you must double-click it. Additionally, swiping and dragging gestures are unsupported

while the Accessibility Inspector is running. To perform these gestures, you must first disable the Accessibility Inspector. To disable and re-enable the Accessibility Inspector, click the close control in the upper-left corner of the inspector panel (the close control looks like a circle with an “X” in it).

For more information on using the Accessibility Inspector and testing the accessibility of your app, see *Verifying App Accessibility on iOS*.

Testing App Localization

If you have created an app with multiple localizations, you can test them in iOS Simulator by changing the Internationalization settings.

To change the language of a simulation environment

1. Build and run your app in iOS Simulator.
2. Select the simulation environment whose settings you want to change by choosing Hardware > Version > iOS version.
3. Click the Home button to reveal the Home screen, and click Settings.
4. Go to General > International > Language.
5. Select the language, and click Done.

For more information on localizing your app, see *Internationalization Programming Topics*.

Testing Web Apps

If you are building a web app and want to test its usability on an iOS device, let iOS Simulator assist you.

To test a web app in iOS Simulator

1. Select the simulator environment you would like to test in by choosing Hardware > Version > iOS version.
2. Open Safari from the Home screen of iOS Simulator.
3. Navigate to the location of your web app in the browser.

For more information on creating web apps for iOS, see *Getting Started with iOS Web Apps*.

Testing iCloud

Important: iCloud simulator only works when simulating iOS 7.0.

If you are building an app the uses iCloud, you can test iCloud syncing from within iOS Simulator before testing on physical devices. This can also assist you in testing iCloud syncing across many devices if you haven a limited number of devices to test on.

To simulate iCloud syncing, you must first sign in to the iOS Simulator using an Apple ID. It is strongly encouraged that you create and use separate Apple ID specifically for testing iCloud in iOS Simulator.

To sign in to iOS Simulator with your Apple ID

1. Launch iOS Simulator with a simulated device running iOS 7.0.
2. Open Settings from the Home screen, and select iCloud.
3. Enter your Apple ID and Password, and click Sign In.

After signing in with your Apple ID, you can then test your iCloud syncing. To test to see whether your app is syncing properly with iCloud choose Debug > Trigger iCloud sync.

Using the Debugging Tools in iOS Simulator

Access the debugging tools in iOS Simulator through the Debug menu, as shown in Table 3-1.

Table 3-1 Performing debugging through the iOS Simulator Debug menu

Menu item	Debug result
Toggle Slow Animations	Slows down the animation taking place within the app. Use to identify any problems in the animation. Toggle Slow Animations can also be activated by pressing the Shift key three times.
Color Blended Layers	Shows blended view layers. Multiple view layers that are drawn on top of each other with blending enabled are highlighted in red. Reduce the amount of red in your app when this option is selected to dramatically improve your app's performance. Blended view layers are often the cause of slow table scrolling.
Color Copied Images	Shows images that are copied by Core Animation in blue.
Color Misaligned Images	Places a yellow overlay over images whose source pixels are not aligned to the destination pixels.

Menu item	Debug result
Color Off Screen Rendered	Places a magenta overlay on content that is rendered offscreen.
Location	Allows you to set the Core Location to be used by your app. Choose from the different location settings: • **None.** Does not return a location. Use for testing how an app responds when no location data is available. • **Custom Location.** Allows use of a custom latitude and longitude. • **Apple Stores.** Uses the coordinates for an Apple Retail Store. • **Apple.** Uses the coordinates of the Apple Headquarters. • **City Bicycle Ride.** Simulates a bike ride in Cupertino, CA. This option simulates the device moving on a predefined route. • **City Run.** Simulates a run in Cupertino, CA. This option simulates the device moving on a predefined route. • **Freeway Drive.** Simulates a drive through Cupertino, CA. This option simulates the device moving on a predefined route.

Viewing Crash Logs

If your app experiences a problem that causes it to crash, a crash log can help you determine what problem occurred. You open the crash log using Console.

To view a crash log

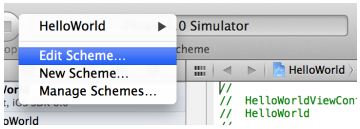
1. Open Console by going to `Applications/Utilities/Console` in the Finder.
2. Look for the line in Console that reads “Saved Crash Report for.”
3. Expand this item using the arrow at the left.
4. Click Open Report.

Customizing Your iOS Simulator Experience with Xcode Schemes

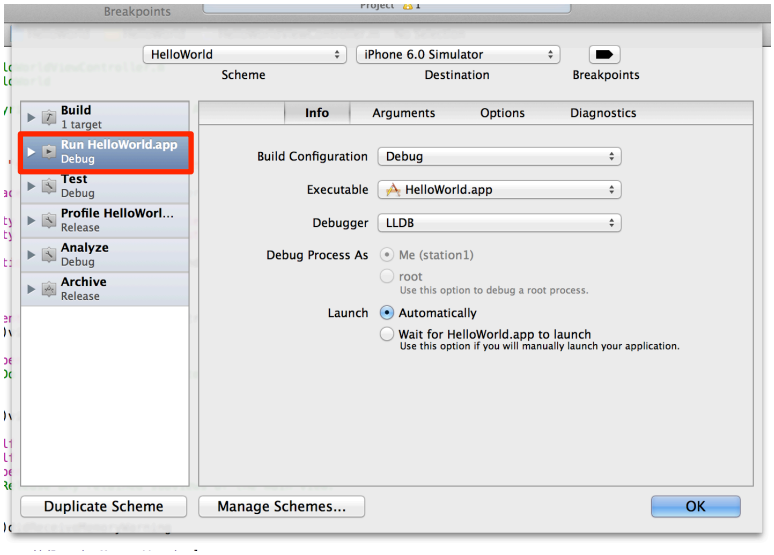
You can customize your iOS Simulator experience using the Xcode scheme editor. In fact, some iOS Simulator features are accessible only within this editor. The biggest advantage of using an Xcode Scheme is the ability to load application data files and routing app coverage files.

To access scheme settings

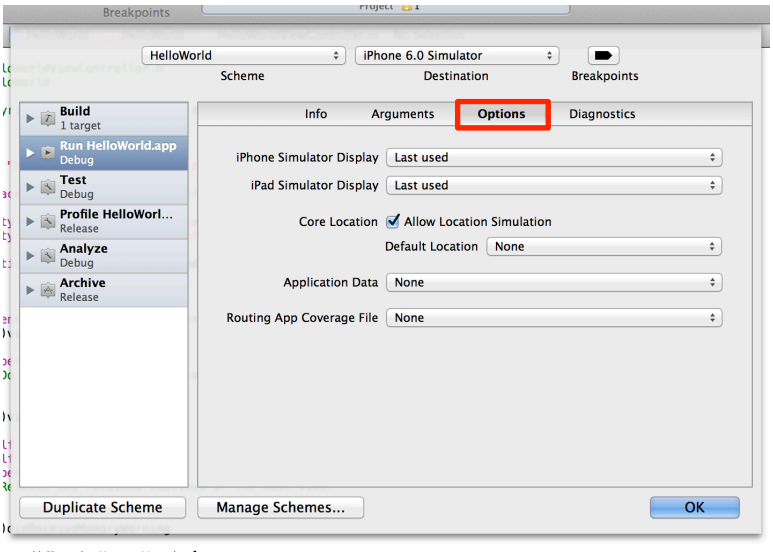
1. Click the scheme name in the scheme pop-up menu, and choose Edit Scheme.



2. When the Edit Scheme window opens, click the Run option in the scheme editor's left pane.



3. In the main pane, click Options.



4. Specify the options you want.
    - **iPhone Simulator Display.** Specify the type of display your simulated iPhone will have (Retina or non-Retina).
    - **iPad Simulator Display.** Specify the type of display your simulated iPad will have (Retina or non-Retina).
    - **Core Location.** If you want to define the default Core Location setting, select Allow Location Simulation and choose a default location from the pop-up menu.
    - **Application Data.** If you want to load app data into the simulator, select the application data file from the pop-up menu. In this way, you can replicate the settings that were present when a problem occurred.
    - **Routing App Coverage File.** If your app uses routing, use this file to define the location boundaries in which your app will provide routes. For more information on routing app coverage files, see *Location and Maps Programming Guide*.

For more information on using schemes in Xcode, see "Create, Edit, and Manage Schemes" in *Xcode Overview*.

Document Revision History

This table describes the changes to *iOS Simulator User Guide*.

Date	Notes
2013-10-22	Updated links to App Distribution Guide.
2013-09-18	Updated to include iOS Simulator changes in Xcode 5.
2013-04-23	Minor updates and the addition of information on testing Bluetooth using iOS Simulator.
2013-01-28	New document that explains how to test iOS apps on a Mac computer during development.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010