Contents

Step 0: Introduction ............................................................................................................. 4
Step 0: Development environment requirements ............................................................... 4

Step 1: Installing Titanium And Your SDK .................................................................. 5
Step 1: Install Titanium and mobile SDK’s ...................................................................... 5
  Install the Sun Java 6 JDK .............................................................................................. 6
  Install Android SDK ...................................................................................................... 8
  Install Titanium Developer ............................................................................................ 13

Step 2: Running the Sample Applications ................................................................... 16
Step 2: Run the sample mobile apps ............................................................................... 16
  Create the HelloWorld project .................................................................................... 16
  Run the default project for Android ............................................................................. 18
  Edit the default project, creating “Hello World” on the Android ............................... 19
  Download the Kitchen Sink app .................................................................................. 21
  Import the Kitchen Sink project ................................................................................... 22
  Run the Kitchen Sink app on the Android emulator .................................................... 23
  Next steps ....................................................................................................................... 24
  Troubleshooting ............................................................................................................ 24
Step 0: Introduction

This section is part of the Titanium Getting Started series, covering the Steps 0 through 2 of getting Titanium installed and working on your development system.

You’ve heard that Appcelerator Titanium can help you build your next great app. Let’s get started with the process of installing and using Titanium. This document series covers “0 to 2” along your way to using Titanium for application development.

• Step 0 covers your existing computer environment, specifying the supported operating systems that you can use with Titanium.
• In Step 1, you’ll learn how to download and install Titanium and the Apple and/or Android SDK’s so that you’re ready to start building apps.
• And then in Step 2, you’ll use JavaScript to build your first “HelloWorld” app, and then run our Kitchen Sink app that demonstrates most of the API’s that you can use in your own applications.

With Titanium, you can build apps for iPhone, iPad, and Android mobile platforms, as well as desktop apps for Mac OS X, Windows, and Linux. This guide covers only the mobile platforms.

In Step 1 and Step 2, we’ll cover each of the mobile platforms, using each of the supported development environments. You don’t have to read each section – just skip over the sections that don’t match your development environment or your intended target platform.

Step 0: Development environment requirements

You’re ready to go – you’ve got your computer all set up, and ready to start working with Titanium. These are the development environments that are supported. You may find that slightly down-rev operating systems work fine as well, but these are the versions that we’ve tested.

If you’re developing for iPhone or iPad, you must use Mac OS X for your development. Note that the current iOS (iPhone and iPad) SDK will not run on Mac OS X systems earlier than Snow Leopard.

Mac OS X

Using Mac OS X, you can build mobile apps for iPhone, iPad and Android.

We’ve tested Mac OS X version 10.6.4, but earlier versions of 10.6 (Snow Leopard) should work as well. (Please let us know if you have any trouble.)

Windows

We have tested these procedures with Windows 7. You may be able to use earlier versions of Windows, but we haven’t fully tested them, and they aren’t supported.

Linux

We’ve tested Titanium mobile development with Ubuntu 9.10 (Karmic Koala).

You can try using other Linux distributions to develop with Titanium, but this Ubuntu distribution is the one that we test with and support. If you have trouble or success with another distribution, please let us know.

- Supported Environments
  - Mac OS X 10.6.4 (Snow Leopard)
  - Windows 7, XP and Vista
  - Ubuntu 9.10 (Karmic Koala)
Step 1: Installing Titanium And Your SDK

This section is part of the Titanium Getting Started series, covering the Steps 0 through 2 of getting Titanium installed and working on your development system.

This document is for development using Microsoft Windows. You can also develop using a Macintosh or Linux, which are covered in related sections.

The supported development environment is Windows 7, and this guide is written for users of Windows 7 Professional. The general procedures should work with other current versions of Windows as well.

For downloaded files, the filenames given here are current as of when we created this guide. In general, it should be ok to use a later update of the listed files.

Step 1: Install Titanium and mobile SDK’s

Appcelerator’s Developer Web site contains information you’ll need to configure your development system, and then begin development using the sample programs. Using Firefox, or another Web browser, navigate to http://developer.appcelerator.com/get_started

Here’s the Web page you’ll see:

At the bottom of the page, you’ll see several helpful getting started videos that you can watch. You can watch them now or later; when you’re ready, continue on with these instructions.
Step 1: Installing Titanium And Your SDK

You’ll be installing the components on the right that you’ll use to develop mobile applications with Titanium.

Note that Apple’s iOS (iPhone) SDK is not covered here, since iOS (iPhone) development is supported only on the Mac.

There are several API revision levels, and different versions of the SDK’s that we’ll be navigating through in this section. You’ll want to follow these installation steps in order.

Although only the Android SDK must be installed in a specific location, you can simplify overall development if you follow the pattern of suggested directories below.

We’ll be using a command window for some of these commands. To open a new command window, go to All Programs -> Accessories -> Command Prompt.

Install the Sun Java 6 JDK

To install the JDK, we’ll follow the following steps:
• Download the JDK
• Run the JDK installer
• Create/edit environment variables
• Verify the installation

Using your favorite Web browser, navigate to

On the download screen, select “Windows” or “Windows x64”, to choose the Java installer that matches your version of Windows 7.
• 32-bit Windows 7 jdk-6u21-windows-i586.exe
• 64-bit Windows 7 jdk-6u21-windows-x64.exe

Save the file to your hard disk. Once the download has finished, launch the installer. When you get to the Custom Setup page, you’ll see that the default path includes the Program Files folder.

Components to Install
• Sun Java Development Kit 6 (aka 1.6)
• Google’s Android SDK (for Android apps)
• Specific platform packages in the Android SDK
• Titanium Developer
Depending on the kind of development that you'll be doing, there could be an issue with spaces being included in the Java path. To fix this, click the Change button, and select the following path for the Java 6 SDK installation:

C:\Java\jdk1.6.0_21\

Your "Install to:" directory should look like this:

Work through the following screens to complete the installation. You don’t need to change the Java runtime path; that path should be the default:

C:\Program Files\Java\jre6\

Once the JDK is installed, you’ll need to make it available in the environment. We’ll make two changes to do this:

1. Set a JAVA_HOME environment variable to the JDK directory
2. Put %JAVA_HOME%bin in the PATH

Open the System Properties dialog (Control Panel -> System -> Advanced system settings). Then click the “Environment Variables…” button.
Click on “New…” in the System Variable section, and add a new System Variable with the name JAVA_HOME. The value should be “C:\Java\jdk1.6.0_21”, which is the path where you installed your JDK. This is what your New System Variable dialog should look like:

Click OK to dismiss this dialog. Now, select the “Path” system variable and click “Edit…” to enter a new value. The Path system variable is a list of directories separated by semicolons; we need to add the JDK’s bin directory to the beginning of this list. Being careful not to erase the existing text in the field, scroll and click at the beginning of the “Variable value:” field and enter “%JAVA_HOME%\bin;” exactly. Notice that you need to include the semicolon to separate your new entry from the existing text in the field. Your dialog should look more or less like this:

At this point, the Java JDK installation should be complete. Let’s verify that things are working as they should be. Open a new command window (don’t reuse a command window that had previously been open), and enter the following command:

> javac -version

If the JAVA_HOME and PATH environment variables are configured correctly, you’ll see the Java compiler report its version information like this:

Install Android SDK

Unlike iOS (iPhone) development on the Mac, there is no login or approval process for Android development. However, there are numerous steps that you have to follow to download and install the Android SDK, and then register it for use in Titanium. In preparation, it’s helpful to read about the Android SDK here:

http://developer.android.com/sdk/installing.html

When you’re ready, go to the Android SDK download page:

Step 1: Installing Titanium And Your SDK

In the following steps, we’ll show you how to:

- Download the Android SDK, and install to your C: drive
- Run Setup.exe to install it
- Start the Android SDK and AVD Manager GUI
- Modify the http fetch setting
- Download and install the proper Android APIs
- Add the Android SDK to the PATH variable
- Verify that the Android SDK is installed properly

Let’s get started with this. Here are the first steps to follow, to download and install the Android SDK to your C: drive. You can expect the installation to take up about 1 GB of disk space.

- Download android-sdk_r06-windows.zip
- Open this zip file; locate the android-sdk-windows folder
- Move android-sdk-windows to your C: drive, at the root level
- Rename that folder to C:android-sdk

Now, navigate to C:android-sdk and run “SDK Setup.exe”. (Note that you’ll have to run this from an administra-
tor account.)
As the installer opens windows, you'll see the "Choose Packages to Install" screen. The list in the left panel of this screen is populated with every Android SDK version.

![Screenshot of Choose Packages to Install](image)

Although you will use all these SDK versions in your later application development, this is more than we need at this point. Click "Cancel" to return to the "Android SDK and AVD Manager" screen where we'll choose the specific packages to install.

When you launch "SDK Setup.exe" for the first time, you may see an error about fetching the source files:

![Screenshot of error](image)

If you see this error, click "Close", and select the Settings option in the left column of the Android SDK and AVD Manager main page. On the Settings page, look for the "Misc" section near the bottom of the page. Check the box labeled "Force https:// ... sources to be fetched using http:// ..." – your display should look like this:

![Screenshot of Settings](image)

Click "Save & Apply" and you're good to go.
Now, back at the Android SDK and AVD Manager main page, click on “Available Packages” from the list on the left. Expand the list in the right panel to see the various versions of the Android tools. You’ll see a list of packages, something like this:

Select the following items from the list of available packages:
• SDK Platform Android 1.6, API 4, revision 3
• Google APIs by Google Inc., Android API 4, revision 2

Now, click on “Install Selected”, click “Accept All” and “Install” on the confirmation screen, and the selected components will be downloaded and installed.
Once this has completed, click on “Installed Packages” in the left panel to see what got installed. Your display should look something like this:

![Android SDK and AVD Manager](image)

After the installation has finished, close the installer.

The Android tools need to be in your directory search path. We need to prepend the Android SDK directories to the PATH environment variable in Windows, just as we did when installing the Java JDK.

Again, open the System Properties dialog and select the Advanced Settings tab. Then click the “Environment Variables…” button.

![System Properties](image)

Select the PATH system variable and click “Edit…” to enter a new value. Scroll and click at the beginning of the “Variable value:” field and enter

```
C:\android-sdk\tools;C:\android-sdk\platforms\android-4\tools;
```
Step 1: Installing Titanium And Your SDK

1. **Getting Started with Appcelerator Titanium (Windows)**

   **Release GSW, 8/17/10**

   Exactly. Be sure that semicolons separate each directory in the field, and again be sure not to delete the existing contents of the field.

   Finally, let’s verify that the Android SDK installation works. Open a new command window (don’t reuse a command window that had previously been open), and enter the following commands:

   ```
   > aapt v
   > android list
   ```

   If all is well, you’ll see a report of the installed Android tools, something like this:

   ![Command Prompt](Image)

   With these steps complete, all the tools needed by Titanium Developer are in place. When you create a Titanium Mobile application, Developer will be able to find the tools it needs to compile your application, run the emulator, and install your app.

   **Install Titanium Developer**

   In this section, you’ll install Titanium, and the Titanium Developer user interface. We’ll cover the following steps:

   - Download and install Titanium
   - Create an Appcelerator account for Titanium development
   - Connect your Android SDK with Titanium Developer
   - Use the “New Project” button to verify proper SDK installation

   Using your Web browser, again navigate to

   [http://developer.appcelerator.com/get_started](http://developer.appcelerator.com/get_started)
You’ll be returning to the Getting Started page you saw earlier:

Let’s install Titanium. First, click on the “Download and Install” link under Step 2 on the getting started page. This will download “Titanium Developer.msi” to your Downloads folder. When the download has completed, locate “Titanium Developer.msi” and double-click it to start the installation.

We’ll cover Step 3 on the getting started page a bit later, after we’ve finished installing Titanium, and after we build a “Hello World” app from Titanium’s default sample application.

Now, start Titanium Developer. The first time you do this, you’ll need to sign up to create an account with Appcelerator. This is free – just follow the instructions that appear on the screen in Developer. Be sure, of course, to remember the login credentials you create since you’ll use them to log in each time you use Titanium Developer.
After you’re logged in, click the Profile mini-icon in the Perspectives area, in the upper left corner of Titanium Developer’s main window. In Developer’s profile configuration screen, you need to specify where to find your Android SDK. Your display will look like this:

At the bottom of that window, you’ll see a field called “Android SDK” – click on the icon at the right of that field, and locate your Android SDK. If you followed the above directions to install the Android SDK, you would enter the following pathname in the Android SDK field:

C:\android-sdk

This is your ADK root – it’s the folder that contains Android’s tools and platforms folders.

Now select “Projects” from the Perspectives buttons at the top right corner of the Titanium Developer screen to begin development using Titanium. Let’s start with a New Project – click on the “New Project” button at the top of the screen. In the New Project panel, click on the “Project type” field and choose “Mobile” from the pop-up menu. In the lower part of the screen, you’ll see a line called “Installed Mobile Platforms”. After a few seconds Titanium Developer should locate your Android SDK, and a green checkmark should appear for it. (And the red X for the iPhone SDK is expected, since iPhone development needs to be done on a Mac, not on Windows.)
Now you’ve successfully installed Titanium Developer, the Java JDK and the Android SDK. Next, in Step 2, you’ll start working with a “Hello World” app, and the Titanium sample applications.

Your Titanium account is available in several types, at different price points, with different support and training resources. To get started, we’ll continue to use a Community account, which is free. When you get ready to develop your own app using Titanium, you’ll want to consider upgrading your account to Titanium Professional or Titanium Enterprise to help speed up your work.

**Step 2: Running the Sample Applications**

This section is part of the Titanium Getting Started series, covering the Steps 0 through 2 of getting Titanium installed and working on your development system.

This document is for development using Windows. You can also develop using Linux or Macintosh, which are covered in related sections.

**Step 2: Run the sample mobile apps**

In this section, we’ll build a simple “HelloWorld” application in Titanium, and then we’ll run one of the sample apps available on the Appcelerator Web site.

Step-by-step, here are the topics we’ll cover.

- Create a default project using Titanium — See [Create the HelloWorld project on page 16](#)
- Run the default project on the Android emulator — See [Run the default project for Android on page 18](#)
- Edit the default project’s source file (“Hello World”) — See [Edit the default project, creating “Hello World” on the Android on page 19](#)
- Download Titanium’s Kitchen Sink app — See [Download the Kitchen Sink app on page 21](#)
- Import the Kitchen Sink source code into Developer — See [Import the Kitchen Sink project on page 22](#)
- Run the Kitchen Sink app on the Android emulator — See [Run the default project for Android on page 18](#)

**Create the HelloWorld project**

Let’s start where we left off at the end of Step 1. You should have Titanium’s New Project window open, ready to enter the details of your first app.
Before you enter things into Titanium Developer’s New Project window, decide on a directory somewhere where the app’s source files will live. (Expect this to take about 300 MB.) Perhaps you created an Appcelerator directory inside your home folder, C:\Users\Devon.

We’ll call our app “HelloWorld”, so for our example, we’ll let Titanium Developer create the HelloWorld directory inside your existing C:\Users\Devon\Appcelerator folder. Let’s assume that your company name is devoncompanyinc, so your company’s URL would be www.devoncompanyinc.com.

Now make entries like these in the New Project window:

<table>
<thead>
<tr>
<th>Project type:</th>
<th>Mobile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>HelloWorld</td>
</tr>
<tr>
<td>App Id:</td>
<td>com.devoncompanyinc.helloworld</td>
</tr>
<tr>
<td>Directory:</td>
<td>C:\Users\Devon\Appcelerator</td>
</tr>
<tr>
<td>Company/Personal URL:</td>
<td><a href="http://www.devoncompanyinc.com">www.devoncompanyinc.com</a></td>
</tr>
<tr>
<td>Titanium SDK version:</td>
<td>1.4.0</td>
</tr>
</tbody>
</table>

When you’ve filled in the fields, click “Create Project” to have Titanium Developer create all the source files for your HelloWorld project. Titanium Developer will then open the HelloWorld project screen, which looks like this one:

At this point, you can explore the contents of the new HelloWorld directory. You’ll see various files that are used by Titanium, plus a Resources directory – that’s where your JavaScript code goes. With only a few exceptions which you’ll learn later, you should make all your changes in the Resources directory; the other files and directories are used by Titanium, and you should not change them.

In the Resources directory, open app.js using your favorite text editor. You’ll see the JavaScript code for a simple default project. In the next section we’ll run this application so you can see what it does. Later you’ll learn how to deploy an app to a physical device; for now, we’ll run the HelloWorld app in an emulator on your Windows system.
Run the default project for Android

To run the HelloWorld project we just created, click on the “Test & Package” tab at the top of the Titanium Developer window. That will bring up this screen:

Click the Android tab if it’s not already selected to indicate that you’ll be using the Android emulator. If Titanium Developer has properly located the Android SDK, you’ll see the SDK field at the bottom of the screen populate with a pop-up list of available SDK versions. Select “APIs 1.6” from that list if it’s not already selected.

Now, click on the “Launch” button to compile the JavaScript code and start the application in the Android emulator. You'll see messages in the Titanium Developer window like these:

It will take several minutes for the application to build and for the emulator to launch and start up the application. There’s no user intervention needed here; you just have to wait for Titanium Developer and the Android
emulator to eventually start the app. If the Android emulator shows the “Locked” screen, press the emulator’s “Menu” button to unlock it.

When you finally see the emulated Android screen appear, you’ll see that this window looks just like an Android display. As the app starts executing in the emulated Android, it will automatically put up the Appcelerator logo:

Then the image will automatically flip to the “I am Window 1” display, with two tabs named (unsurprisingly) “Tab 1” and “Tab 2”.

This sample application doesn’t do much, but you’ll see that if you click on “Tab 2” at the bottom of the screen, the text changes to “I am Window 2”. When you’re finished experimenting with the two tabs (which probably won’t be very long!), return to the Titanium Developer window and click the “Stop” button at the bottom of it. This will terminate the Android emulator.

If you don’t get the results you’re expecting, take a look at the Troubleshooting section below, and see if the suggestions there might help.

**Edit the default project, creating “Hello World” on the Android**

Let’s change the message in the window to “Hello World”. This requires only very simple edits to the app.js file in the Resources directory, in the application’s source directory. In our example, that’s the C:\Users\Devon\Appcelerator\HelloWorld source directory.

Using your favorite text editor, open app.js, in the Resources directory. Search for “I am Window 1” – you’ll find this in the first half of the app.js source file. Search also for “I am Window 2” which you’ll find in the second half of the app.js file.

Make the following changes in the app.js file:

• Change “I am Window 1” to “Hello World!”
• Change “I am Window 2” to “Hello again”
Look at the following code extracts from app.js to see exactly how to make these changes. Here’s the first section of code, before the change of “I am Window 1” to “Hello World!”:

```
var label1 = Titanium.UI.createLabel(
    color:'#999',
    text:'I am Window 1',
    font:{fontSize:20,fontFamily:'Helvetica Neue'},
    textAlign:'center',
    width:'auto'
);
```

And here’s what it will look like after your change:

```
var label1 = Titanium.UI.createLabel(
    color:'#999',
    text:'Hello World!',
    font:{fontSize:20,fontFamily:'Helvetica Neue'},
    textAlign:'center',
    width:'auto'
);
```

Now, here’s the second section of code, before the change of “I am Window 2” to “Hello again”:

```
var label2 = Titanium.UI.createLabel(
    color:'#999',
    text:'I am Window 2',
    font:{fontSize:20,fontFamily:'Helvetica Neue'},
    textAlign:'center',
    width:'auto'
);
```

And here’s what the second section will look like, after your change:

```
var label2 = Titanium.UI.createLabel(
    color:'#999',
    text:'Hello again',
    font:{fontSize:20,fontFamily:'Helvetica Neue'},
    textAlign:'center',
    width:'auto'
);
```
Save your changes to app.js, go back to the Titanium Developer window, and again click “Launch” to start the changed application. Again Titanium will compile the JavaScript code (because you changed it), and eventually it will relaunch the Android emulator. If you experience timing issues between Titanium Developer and the Android emulator, just “Launch” again as described in the Troubleshooting section below.

You’ll see your HelloWorld app automatically run on the emulated Android, and the message on the Android’s screen will be “Hello World!”. If you click “Tab 2”, the message should change to “Hello again”. When you’re finished playing with the tabs, return again to Titanium Developer’s window and click the “Stop” button to terminate the Android emulator.

**Download the Kitchen Sink app**

Now that you’ve successfully built Titanium’s default project, let’s move on to build the much larger KitchenSink sample app. Appcelerator’s Kitchen Sink application demonstrates many of the APIs that are available in Titanium. You can adapt the code in the Kitchen Sink app to your own applications.

The first step is to get the source code for the KitchenSink app, which you can download from Appcelerator’s Web site:

http://developer.appcelerator.com/doc/kitchensink

The source code repository lives on Github, and you can get the source code from there. Click on the link “Kitchen Sink on Github”, and from there, click on the “Download Source” button near the top of the Github page. The downloaded file will be a zip file called something like “appcelerator-KitchenSink-1.3.3-1-g4d1c4f8.zip”. Save this file and it should end up in your Downloads folder.

Open the archive and drill down one level where you’ll see the KitchenSink and KitchenSink-iPad folders. Copy the KitchenSink folder to your home directory. The Kitchen Sink source files will then live in C:\Users\Devon\KitchenSink.
Import the Kitchen Sink project

Because the Kitchen Sink app is already a fully structured project, we’ll be importing this into Titanium Developer. This contrasts with what we did earlier with the default project, where we created a new project from scratch. In that case, Titanium Developer created the default project, ready to edit.

To import the Kitchen Sink app’s source code, click the “Import Project” button at the top of Titanium Developer’s window. Navigate to the Kitchen Sink directory, and click OK to import the project.

After a few moments, Titanium Developer will present a project screen similar to this:

Now the KitchenSink project is ready to go – the next step is to execute it on the Android emulator.
Run the Kitchen Sink app on the Android emulator

As you did above, go to the Test & Package tab, note that the Android tab is selected, select “APIs 1.6” for the SDK field, and then click the “Launch” button.

Titanium Developer will build the project. When the build is finished, Developer should install the app on your Android emulator. You should see the Kitchen Sink logo automatically appear on the Android emulator, and then automatically flip to reveal Kitchen Sink’s tabs and buttons.
**Next steps**

Now that you’ve successfully built the default app and imported the Kitchen Sink app, you’re ready to start looking at the JavaScript code for these and other sample apps.

**Troubleshooting**

You may encounter timing issues with Titanium Developer and the emulator. The last message you should see in the Titanium Developer window is "[INFO] Deployed Hello World … Application should be running." When using the Android emulator, Titanium Developer sometimes times out before the emulator is ready to accept the new application as input. If that happens, try again by clicking “Launch” again. Generally, this works after two or three tries. But be sure to wait a minute or two after the "… Application should be running." message appears in Titanium Developer before deciding that the sample app didn’t start properly on the emulator. When this works properly, the sample app should automatically start on the emulator – there’s nothing you have to do on the emulator to start up the sample app. If your app doesn’t start, there are several additional things you can try. When using the Android emulator, some of these issues are more common after you’ve started the emulator several times.

1. Read the messages in the Titanium Developer window to see if there are any clues to the problem. Also try changing the Filter to “Trace”, to view additional messages.
2. Make sure you have enough free disk space and available RAM. The RAM requirements are very modest, so that’s unlikely to be a problem, and you should have at least 1 GB of free disk space at all times.
3. When using the Android emulator, perhaps the adb server is stuck. In this case, you would restart the adb server like this:

   ```
   $ adb kill-server
   $ adb devices
   ```

4. When using the Android emulator, check to see that there is exactly one Android emulator running. If there are two, then kill one of them.
5. When using the Android emulator, in the Android SDK and AVD Manager, try killing any existing AVD’s, which will force Titanium to recreate them. This is not normally necessary, but can be needed if the AVD’s are out of order.
6. Perhaps you have added too many apps to your emulated mobile device. If that happens, just remove some of the apps.
7. Sometimes when importing Titanium projects, the build assets don’t import correctly, especially if you’re importing into a platform different from where the project was originally created. If this happens, you may need to copy the assets into a new project. To do this, create a new project and then replace its Resources directory with the Resources directory of the project that didn’t import successfully. Be aware that the TiApp.xml file includes the project id and other information, so it may have to be updated as well to reflect changes.
8. Check how long the path to your project directory is. This is the full name of your project directory, starting from the root. Some Titanium errors are caused by a path that’s too long. Try using a shorter path to your source directory by moving your sources higher in the directory hierarchy.
9. Titanium Developer has advanced proxy-detection capabilities, and will usually find a way to connect during authentication when you log in. However, if you’re in an unusual corporate network environment, you may require custom configuration of some environment variables.
10. Check out the help resources on Appcelerator’s Developer Web. Point your browser to: [http://developer.appcelerator.com/](http://developer.appcelerator.com/) and click the Q&A tab. Many developers find answers to their questions here.
11. Submit a support request. Be sure to include the following information with your support request:
   - What problem you’re experiencing
   - The version of Titanium you’re using
   - Whether you’re building a mobile or desktop app
   - Trace messages from Titanium Developer
   - The version(s) of the mobile tools (emulator) you have
   - The JDK version you’re using
• Your operating system and version (e.g., Windows 7)
• Anything else you think might be helpful to understanding your issue
Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/5/2010</td>
<td>Initial release</td>
</tr>
<tr>
<td>8/17/2010</td>
<td>Minor formatting changes</td>
</tr>
</tbody>
</table>