Assignment  
PhoneGap on Android

# Creating the project in Eclipse

Follow these steps to create a new Android project in Eclipse:

1. Choose New > Android Project (see Figure 1).



Figure 1. Creating a new Android project.

After you create a new, standard Android project you will update that project to use PhoneGap.

1. In the New Android Project dialog box, type a project name and select Create New Project In Workspace (see Figure 2).
2. Click Next.

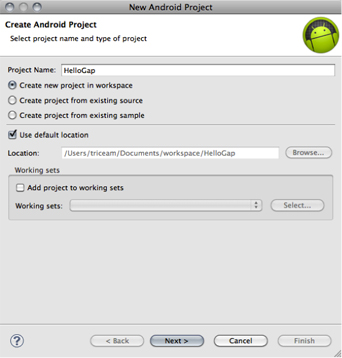


Figure 2. The New Android Project dialog box.

1. Select the Android 2.2 build target, and click Next (see Figure 3).

**Note:** Choosing the Android 2.2 build target will configure the compiler to target the Android 2.2 SDK, and will ensure that your PhoneGap application will work on devices running Android 2.2 and newer versions of the operating system.

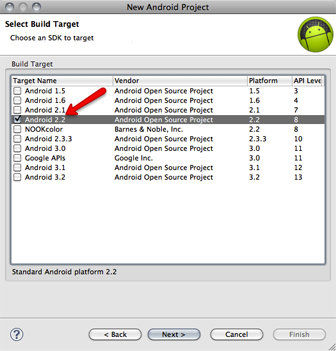


Figure 3. Selecting a build target

1. On the Application Info screen, type a package name for your main Android application (see Figure 4). This should be a namespace that logically represents your package structure; for example, **com.yourcompany.yourproject**.
2. Click Finish.

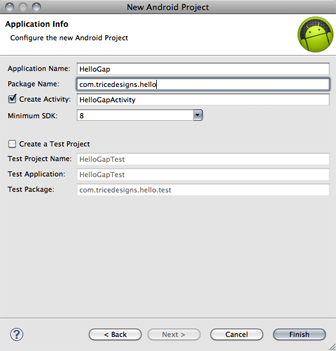


Figure 4. Specifying a package name.

# Configure the project to use PhoneGap

1. Create an assets/www directory and a libs directory inside of the new Android project. All of the HTML and JavaScript for your PhoneGap application interface will reside within the assets/www folder (see Figure 5).

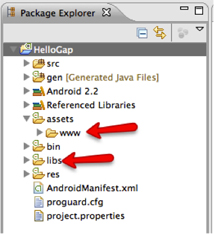


Figure 5. New project directories.

1. To copy the required files for PhoneGap into the project, first locate the directory where you downloaded PhoneGap, and navigate to the lib/android subdirectory (see Figure 6).

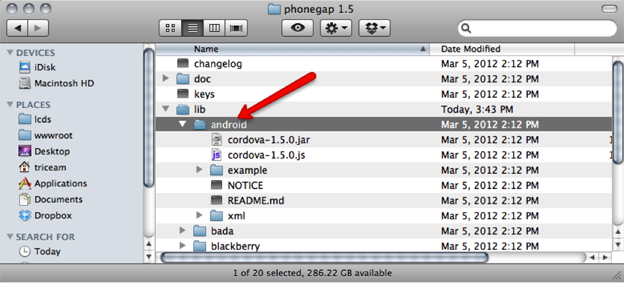


Figure 6. The PhoneGap lib/android directory.

1. Copy **cordova.js** to the assets/www directory within your Android project.
2. Copy **cordova.jar** to the libs directory within your Android project.
3. Copy the xml directory into the res directory within your Android project (see Figure 7).

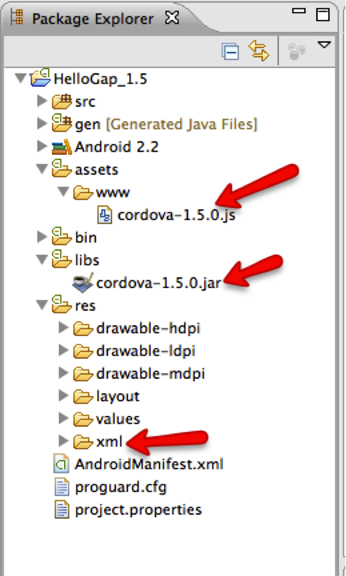


Figure 7. Copied resources.

1. Next, create a file named index.html in the assets/www folder. This file will be used as the main entry point for your PhoneGap application's interface.
2. In index.html, add the following HTML code to act as a starting point for your user interface development:

<!DOCTYPE HTML> <html> <head> <title>PhoneGap</title> <script type="text/javascript" charset="utf-8" src="cordova.js"></script> </head> <body> <h1>Hello PhoneGap</h1> </body> </html>

1. Add the cordova.jar library to the build path for the Android project. Right-click cordova.jar and select Build Path > Add To Build Path (see Figure 8).

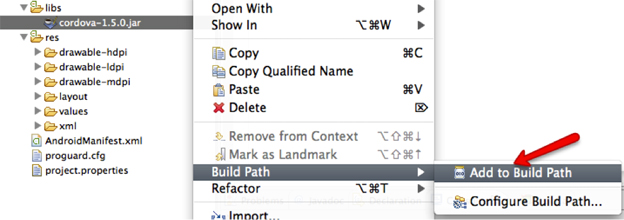


Figure 8. Adding cordova.jar to the build path.

# Update the Activity class

Now you are ready to update the Android project to start using PhoneGap.

1. Open your main application Activity file. This file will have the same name as your project, followed by the word "Activity". It will be located under the src folder in the project package that you specified earlier in this process.

For my project, which I named HelloGap, the main Android Activity file is named HelloGapActivity.java, and is located in the package com.tricedesigns.hello, which I specified in the New Android Project dialog box.

1. In the main Activity class, add an import statement for org.apache.cordova.DroidGap:

**import** org.apache.cordova.DroidGap;

1. Change the base class from Activity to DroidGap ; this is in the class definition following the word extends :

**public** **class** MainActivity **extends** DroidGap {

1. Replace the call to setContentView() with a reference to load the PhoneGap interface from the local assets/www/index.html file, which you created earlier (see Figure 9).

**super**.loadUrl("file:///android\_asset/www/index.html");

**Note:** In PhoneGap projects, you can reference files located in the assets directory with a URL reference file:///android\_asset, followed by the path name to the file. The file:///android\_asset URI maps to the assets directory.



Figure 9. Updates to the main Activity class.

# Configure the project metadata

You have now configured the files within your Android project to use PhoneGap. The last step is to configure the project metadata to enable PhoneGap to run.

1. Begin by opening the AndroidManifest.xml file in your project root. Use the Eclipse text editor by right-clicking the AndroidManifest.xml file and selecting Open With > Text Editor (see Figure 10).

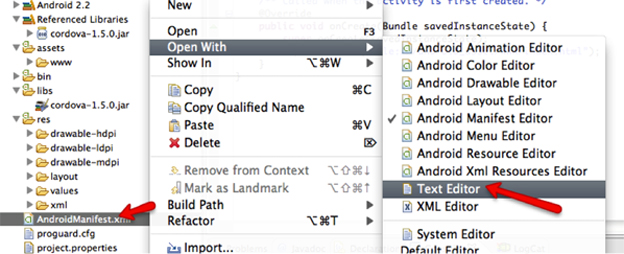


Figure 10. Opening AndroidManifest.xml.

1. In AndroidManifest.xml, add the following supports-screen XML node as a child of the root manifest node:

<supports-screens

android:anyDensity=*"true"*

android:largeScreens=*"true"*

android:normalScreens=*"true"*

android:resizeable=*"true"*

android:smallScreens=*"true"* />

The supports-screen XML node identifies the screen sizes that are supported by your application. You can change screen and form factor support by altering the contents of this entry. To read more about <supports-screens>, visit the [Android developer topic on the supports-screen element](http://developer.android.com/guide/topics/manifest/supports-screens-element.html).

Next, you need to configure permissions for the PhoneGap application.

1. Copy the following <uses-permission> XML nodes and paste them as children of the root <manifest> node in the AndroidManifest.xml file:

<uses-permission android:name=*"android.permission.CAMERA"* />

<uses-permission android:name=*"android.permission.VIBRATE"* />

<uses-permission android:name=*"android.permission.ACCESS\_COARSE\_LOCATION"* />

<uses-permission android:name=*"android.permission.ACCESS\_FINE\_LOCATION"* />

<uses-permission android:name=*"android.permission.ACCESS\_LOCATION\_EXTRA\_COMMANDS"* />

<uses-permission android:name=*"android.permission.READ\_PHONE\_STATE"* />

<uses-permission android:name=*"android.permission.INTERNET"* />

<uses-permission android:name=*"android.permission.RECEIVE\_SMS"* />

<uses-permission android:name=*"android.permission.RECORD\_AUDIO"* />

<uses-permission android:name=*"android.permission.MODIFY\_AUDIO\_SETTINGS"* />

<uses-permission android:name=*"android.permission.READ\_CONTACTS"* />

<uses-permission android:name=*"android.permission.WRITE\_CONTACTS"* />

<uses-permission android:name=*"android.permission.WRITE\_EXTERNAL\_STORAGE"* />

<uses-permission android:name=*"android.permission.ACCESS\_NETWORK\_STATE"* />

<uses-permission android:name=*"android.permission.GET\_ACCOUNTS"* />

<uses-permission android:name=*"android.permission.BROADCAST\_STICKY"* />

The <uses-permission> XML values identify the features that you want to be enabled for your application. The lines above enable all permissions required for all features of PhoneGap to function. After you have built your application, you may want to remove any permissions that you are not actually using; this will remove security warnings during application installation. To read more about Android permissions and the <uses-permission> element, visit the [Android developer topic on the uses-permission element.](http://developer.android.com/guide/topics/manifest/uses-permission-element.html).

After you have configured application permissions, you need to modify the existing <activity> node.

1. Locate the <activity> node, which is a child of the <application> XML node. Add the following attribute to the <activity> node:

android:configChanges="orientation|keyboardHidden"

1. Next, you need to create a second <activity> node for the org.apache.cordova.DroidGap class. Add the following <activity> node as a sibling of the existing <activity> XML node:

<activity android:name="org.apache.cordova.DroidGap" android:label="@string/app\_name" android:configChanges="orientation|keyboardHidden"> <intent-filter></intent-filter> </activity>

# Running the application

To launch your PhoneGap application in the Android emulator, right-click the project root, and select Run As > Android Application (see Figure 11).

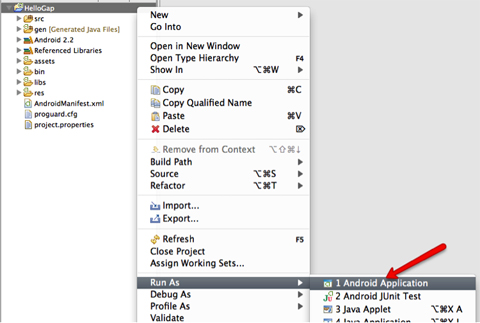


Figure 11. Launching the Android application.

If you don't have any Android virtual devices set up, you will be prompted to configure one. To learn more about configuring Android emulator virtual devices, visit the [Android developer guide for devices](http://developer.android.com/guide/developing/devices/index.html).

Eclipse will automatically start an Android emulator instance (if one is not already running), deploy your application to the emulator, and launch the application (see Figure 12).

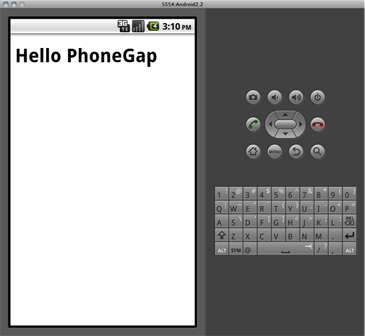


Figure 12. The application in the Android emulator.

After you get your application running in the Android emulator, you'll want to test it out on a physical device. I strongly recommend that you always test your applications on a physical device before deploying the application into production environments. Physical devices always have different computing abilities and form factors than emulators, and device testing can uncover issues that may not have been detected in the emulator environment.

Follow these steps to launch your application on a physical Android device:

1. Make sure the device is connected to your computer via USB.
2. Choose Run > Run Configurations (see Figure 13).

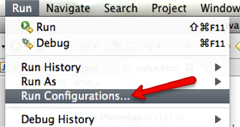


Figure 13. Updating run configurations.

1. Select your application under Android Application on the left side of the Run Configurations dialog box.
2. Click the Target tab, and then select Manual as the Deployment Target Selection Mode.
3. When you are ready to launch your application, click Run (see Figure 14).

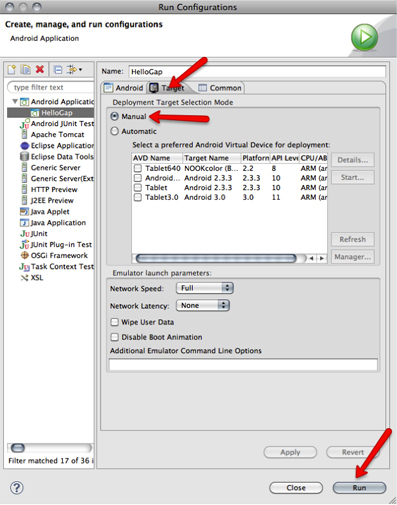


Figure 14. Preparing to run the application on a device.

In the Android Device Chooser dialog box, you can select either an emulator or a connected Android device. All connected Android devices will be displayed in this list.

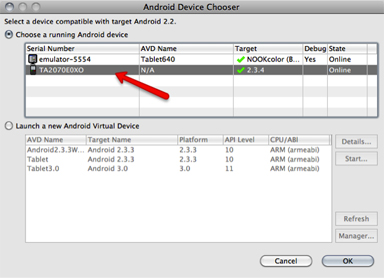


Figure 15. Choosing an Android device.

1. Select the device that you want to use (see Figure 15), and click OK.

Your PhoneGap application will be installed and launched on the device.