Managing Projects from Eclipse with ADT

Eclipse and the ADT plugin provide GUIs and wizards to create all three types of projects (Android project, Library project, and Test project):

- An Android project contains all of the files and resources that are needed to build a project into an .apk file for installation. You need to create an Android project for any application that you want to eventually install on a device.
- You can also designate an Android project as a library project, which allows it to be shared with other projects that depend on it. Once an Android project is designated as a library project, it cannot be installed onto a device.
- Test projects extend JUnit test functionality to include Android specific functionality. For more information on creating a test project, see Testing from Eclipse with ADT.

Creating an Android Project

The ADT plugin provides a *New Project Wizard* that you can use to quickly create a new Android project (or a project from existing code). To create a new project:

1. Select **File** > **New** > **Project**.
2. Select **Android** > **Android Project**, and click **Next**.
3. Select the contents for the project:
   - Enter a *Project Name*. This will be the name of the folder where your project is created.
   - Under Contents, select **Create new project in workspace**. Select your project workspace location.
   - Under Target, select an Android target to be used as the project's Build Target. The Build Target specifies which Android platform you'd like your application built against.

     Select the lowest platform with which your application is compatible.

     > **Note:** You can change your the Build Target for your project at any time: Right-click the project in the Package Explorer, select **Properties**, select **Android** and then check the desired Project Target.
   - Under Properties, fill in all necessary fields.
     - Enter an *Application name*. This is the human-readable title for your application — the name that will appear on the Android device.
     - Enter a *Package name*. This is the package namespace (following the same rules as for packages in the Java programming language) where all your source code will reside.
     - Select *Create Activity* (optional, of course, but common) and enter a name for your main Activity class.
     - Enter a *Min SDK Version*. This is an integer that indicates the minimum API Level required to properly run your application. Entering this here automatically sets the `minSdkVersion` attribute in the <uses-sdk> of your Android Manifest file. If you're unsure of the appropriate API Level to use, copy the API Level listed for the Build Target you selected in the Target tab.
4. Click **Finish**.

> **Tip:** You can also start the New Project Wizard from the *New* icon in the toolbar.

Setting up a Library Project

A library project is a standard Android project, so you can create a new one in the same way as you would a new application project.

When you are creating the library project, you can select any application name, package, and set other fields as needed, as shown in figure 1.

Next, set the project's properties to indicate that it is a library project:

1. In the **Package Explorer**, right-click the library project and select **Properties**.
2. In the **Properties** window, select the "Android" properties group at left and locate the **Library** properties at right.
3. Select the "is Library" checkbox and click **Apply**.
4. Click **OK** to close the *Properties* window.

The new project is now marked as a library project. You can begin moving source code and resources into it, as described in the sections below.

You can also convert an existing application project into a library. To do so, simply open the Properties for the project and select the "is Library" checkbox. Other application projects can now reference the existing project as a library project.

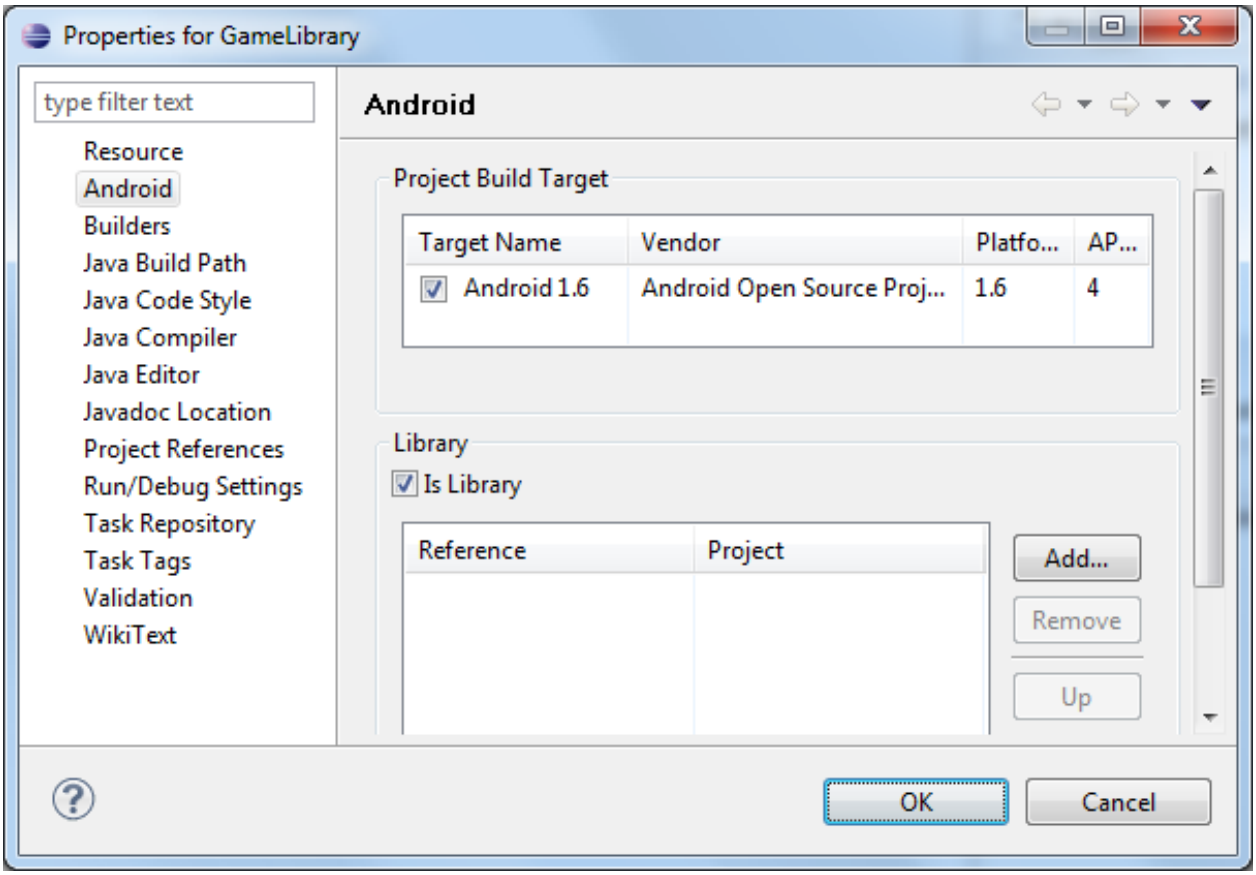

Figure 1. Marking a project as an Android library project.

Creating the manifest file

A library project's manifest file must declare all of the shared components that it includes, just as would a standard Android application. For more information, see the documentation for AndroidManifest.xml.

For example, the TicTacToeLib example library project declares the Activity `GameActivity`:

```
<manifest>
  ...
  <application>
```

```
    ...
    <activity android:name="GameActivity" />
    ...
  </application>
</manifest>
```

Referencing a library project

If you are developing an application and want to include the shared code or resources from a library project, you can do so easily by adding a reference to the library project in the application project's Properties.

To add a reference to a library project, follow these steps:

1. In the **Package Explorer**, right-click the dependent project and select **Properties**.
2. In the **Properties** window, select the "Android" properties group at left and locate the **Library** properties at right.
3. Click **Add** to open the **Project Selection** dialog.
4. From the list of available library projects, select a project and click **OK**.
5. When the dialog closes, click **Apply** in the **Properties** window.
6. Click **OK** to close the **Properties** window.

As soon as the Properties dialog closes, Eclipse rebuilds the project, including the contents of the library project.

Figure 2 shows the Properties dialog that lets you add library references and move them up and down in priority.

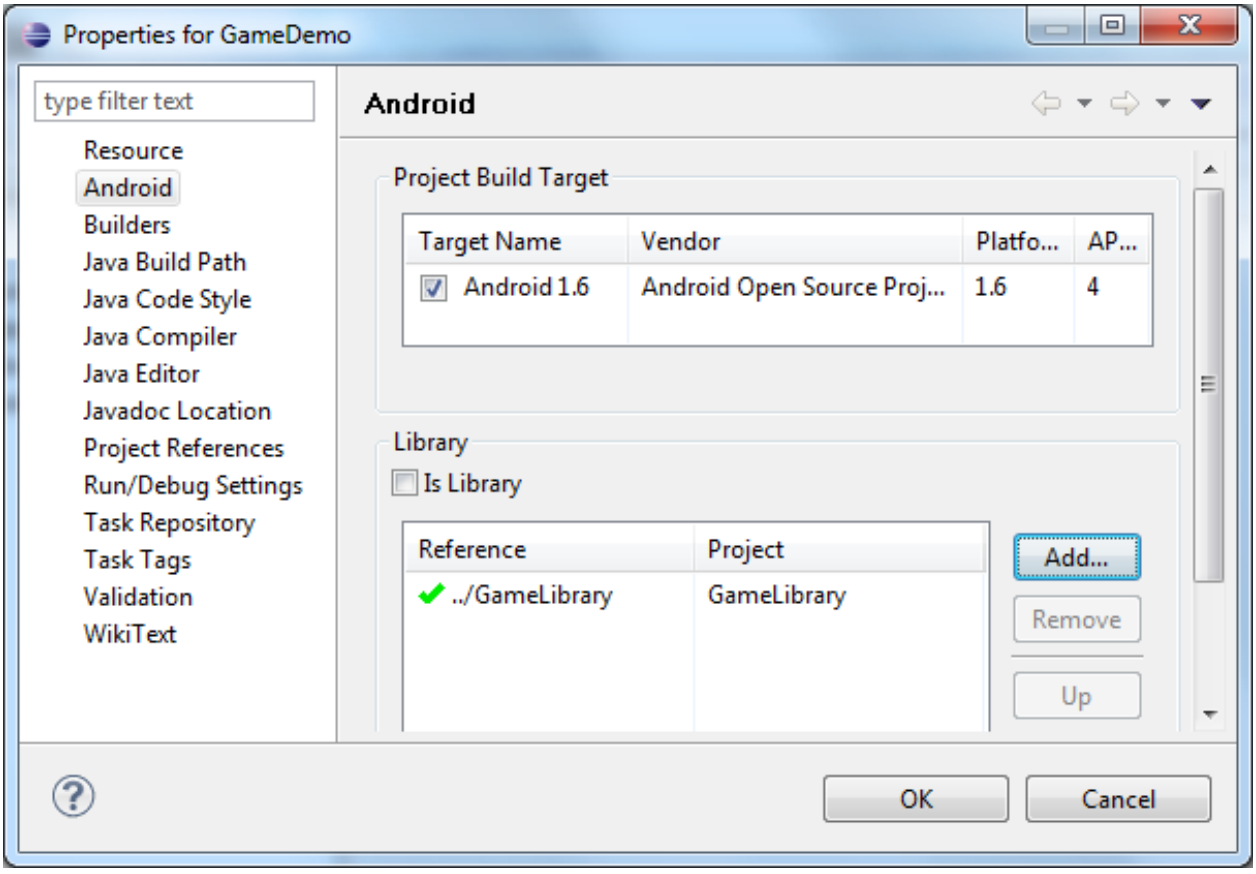

Figure 2. Adding a reference to a library project in the properties of an application project.

If you are adding references to multiple libraries, note that you can set their relative priority (and merge order) by selecting a library and using the **Up** and **Down** controls. The tools merge the referenced libraries with your application starting from lowest priority (bottom of the list) to highest (top of the list). If more than one library defines the same resource ID, the tools select the resource from the library with higher priority. The application itself has highest priority and its resources are always used in preference to identical resource IDs defined in libraries.

Declaring library components in the the manifest file

In the manifest file of the application project, you must add declarations of all components that the application will use that are imported from a library project. For example, you must declare any `<activity>`, `<service>`, `<receiver>`, `<provider>`, and so on, as well as `<permission>`, `<uses-library>`, and similar elements.

Declarations should reference the library components by their fully-qualified package names, where appropriate.

For example, the TicTacToeMain example application declares the library Activity `GameActivity` like this:

```
<manifest>
  ...
  <application>
    ...
    <activity android:name="com.example.android.tictactoe.library.GameActivity" />
    ...
  </application>
</manifest>
```

For more information about the manifest file, see the documentation for AndroidManifest.xml.

← Back to Managing Projects ↑ Go to top