Eclipse for Android Development

(Current as of September 2012)

The Android Development Toolkit for Eclipse has changed since the publication of Android Application Development All-in-One For Dummies. This document describes some of the differences.

Start by selecting File→New→Android Application Project

When you type the name of your application, Eclipse fills in the Project Name and Package Name fields for you. (You can change these fields if you want.) If you’re new to Android, you can uncheck the Create Custom Launcher Icon box.
Accept the defaults to have Eclipse create a blank activity for your project.

Accept the defaults to name your activity MainActivity.java, and to name your layout file activity_main.xml.
Eclipse creates more folders and files than it did when I wrote the book. As a new Android developer, you don't have to worry about the new files and folders.

The new project's layout is no longer a LinearLayout. Instead, it's a RelativeLayout.

It's not easy to compose (from scratch) the XML code for a RelativeLayout. So I recommend using the Graphical Layout tab in order to add widgets to your layout. Afterward, you can use the activity_main.xml tab to tweak the XML code.
The "blank activity" now has an `onCreateOptionsMenu` method. You can ignore this method. (Nothing will happen when you click Android's menu button, but when you're just starting out, that's okay.) For more information on the `onCreateOptionsMenu` method, see `onCreateOptionsMenu.pdf`.

![Snippet of MainActivity.java]({})

The Properties view looks fancier, but you can use it the same way you did with older versions of the Android Development Toolkit.

![Properties view](image)

If you create an "On Click" property, you still have to type your own handler method (in this example, the `whenClicked` method).

![Snippet of MainActivity.java]({})
Eclipse gives you more warnings than it used to give you. For example, Eclipse warns you when you use a string (instead of a resource from the strings.xml file).

For a beginning developer I often say “Warning; Schmawr-ning!” But if the warning bothers you, highlight the offending hardcoded string. Then select Refactor→Android→Extract Android String.
In this example, I accept the defaults in the Extract Android String dialog box.

Eclipse wants me to add a text hint to my image...
To add a text hint, I type an android:contentDescription attribute. Again, Eclipse wants me to use a resource from the strings.xml file.

![Image showing strings.xml and activity_main.xml with some hardcoded string highlighted](image)

Again, I highlight the offending hardcoded string, and then select Refactor→Android→Extract Android String.

![Image showing the Extract Android String dialog box](image)
I’ve replaced a hardcoded string with a resource from the strings.xml file. But, for some reason, Eclipse doesn’t recognize the change!

To make Eclipse happy, I make a small change to the layout file (… in this example I add a few blank spaces…) and then I click Save.

Resaving the layout file makes Eclipse happy.