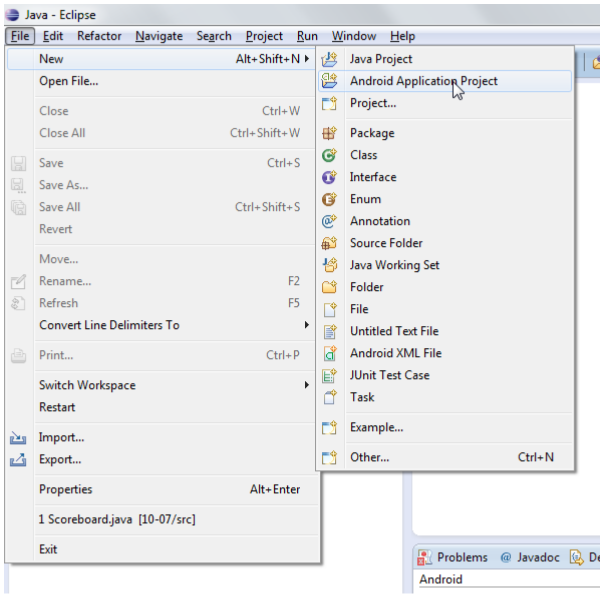
Eclipse for Android Development

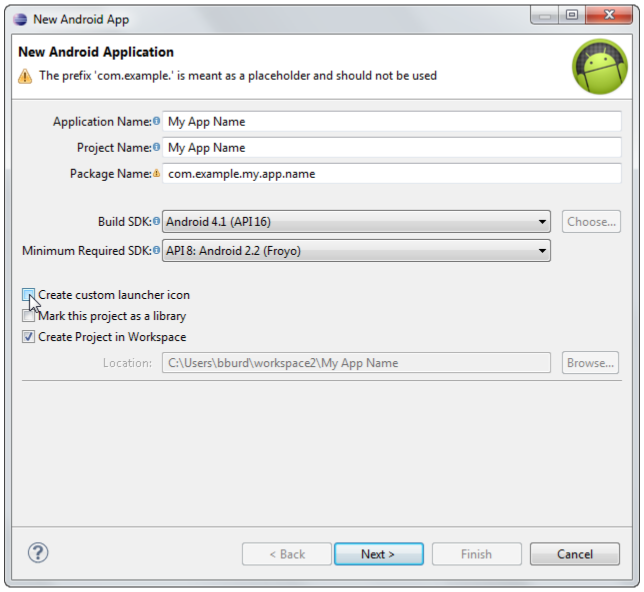
(Current as of September 2012)

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.



New Android Application

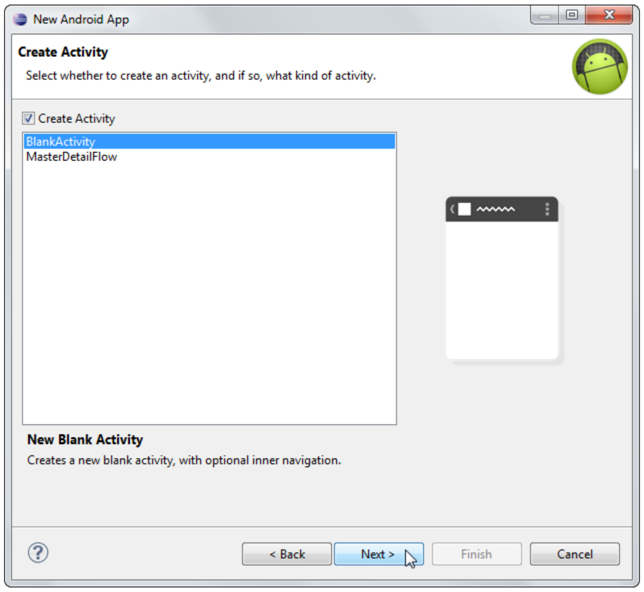
The prefix 'com.example.' is meant as a placeholder and should not be used

Application Name: My App Name
Project Name: My App Name
Package Name: com.example.my.app.name
Build SDK: Android 4.1 (API 16)
Minimum Required SDK: API 8: Android 2.2 (Froyo)

☐ Create custom launcher icon
☐ Mark this project as a library
☑ Create Project in Workspace

Location: C:\Users\bburd\workspace2\My App Name

Accept the defaults to have Eclipse create a blank activity for your project.


New Android App

Create Activity

Select whether to create an activity, and if so, what kind of activity.

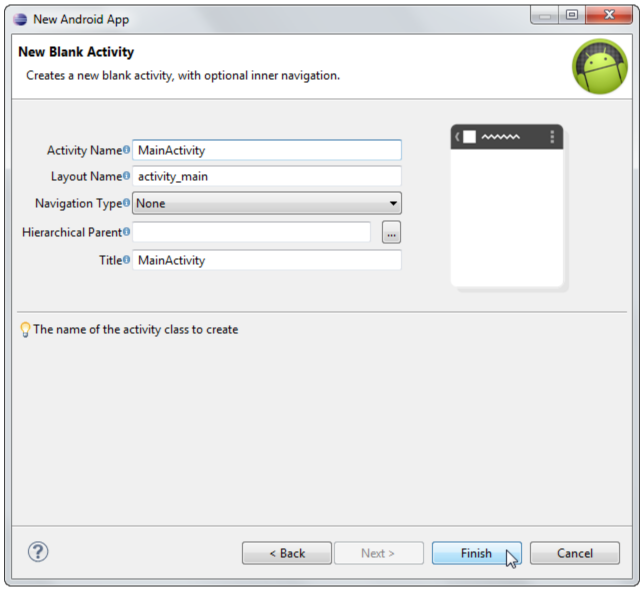
☑ Create Activity

BlankActivity
MasterDetailFlow

New Blank Activity

Creates a new blank activity, with optional inner navigation.

Accept the defaults to name your activity MainActivity.java, and to name your layout file activity_main.xml.


New Android App

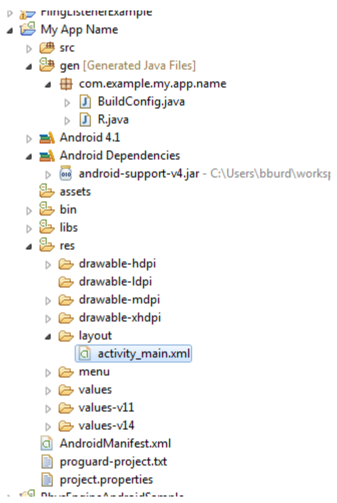
New Blank Activity

Creates a new blank activity, with optional inner navigation.

Activity Name	MainActivity
Layout Name	activity_main
Navigation Type	None
Hierarchical Parent	
Title	MainActivity

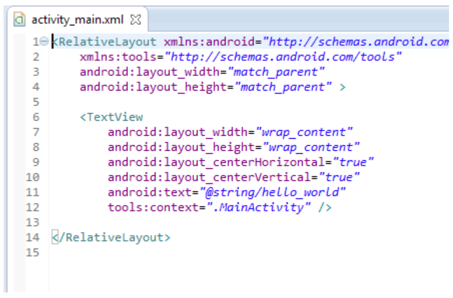
The name of the activity class to create

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.



```
FlingListenerExample
My App Name
    src
    gen [Generated Java Files]
        com.example.my.app.name
            BuildConfig.java
            R.java
    Android 4.1
    Android Dependencies
        android-support-v4.jar - C:\Users\bburd\works
    assets
    bin
    libs
    res
        drawable-hdpi
        drawable-ldpi
        drawable-mdpi
        drawable-xhdpi
        layout
            activity_main.xml
        menu
        values
        values-v11
        values-v14
    AndroidManifest.xml
    proguard-project.txt
    project.properties
```

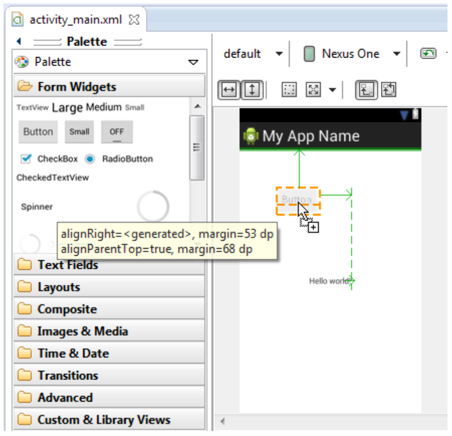
The new project's layout is no longer a LinearLayout. Instead, it's a RelativeLayout.



activity_main.xml

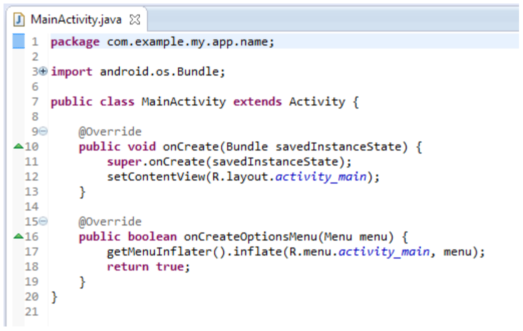
```xml
<RelativeLayout xmlns:android="http://schemas.android.com
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="@string/hello_world"
        tools:context=".MainActivity" />

</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.



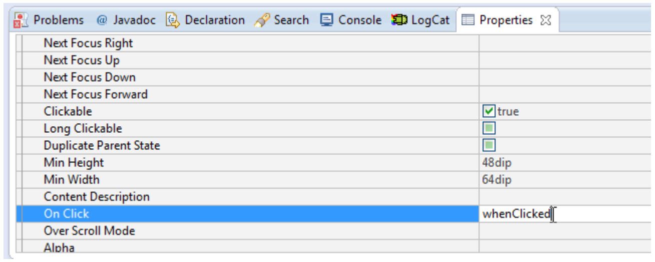
alignRight=<generated>, margin=53 dp
alignParentTop=true, margin=68 dp

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.


MainActivity.java

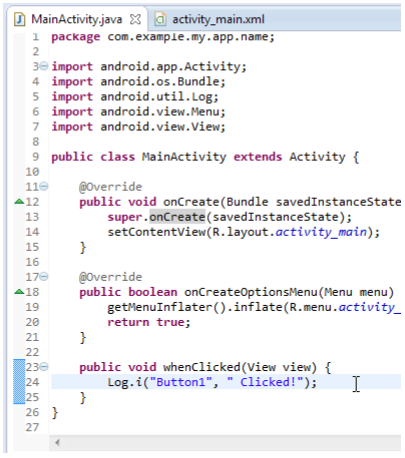
```
package com.example.my.app.name;

import android.os.Bundle;

public class MainActivity extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }
}
```

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



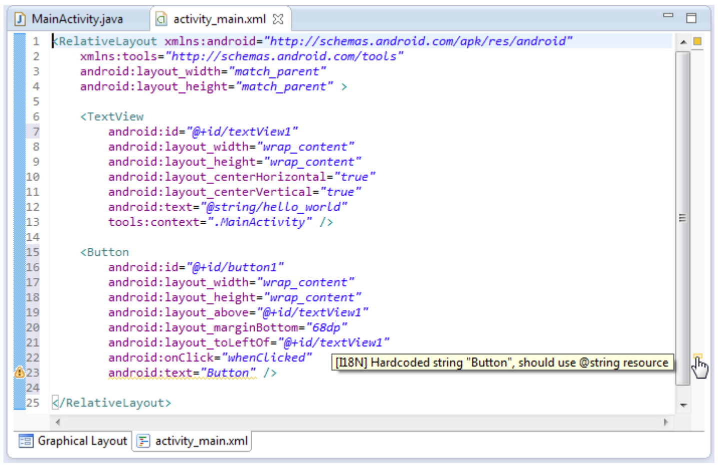
Next Focus Right	
Next Focus Up	
Next Focus Down	
Next Focus Forward	
Clickable	✓ true
Long Clickable	☐
Duplicate Parent State	☐
Min Height	48dip
Min Width	64dip
Content Description	
On Click	whenClicked
Over Scroll Mode	
Alpha	

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


MainActivity.java   activity_main.xml

```
package com.example.my.app.name;

import android.app.Activity;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.View;

public class MainActivity extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu)
        getMenuInflater().inflate(R.menu.activity_
        return true;
    }

    public void whenClicked(View view) {
        Log.i("Button1", " Clicked!");
    }
}
```

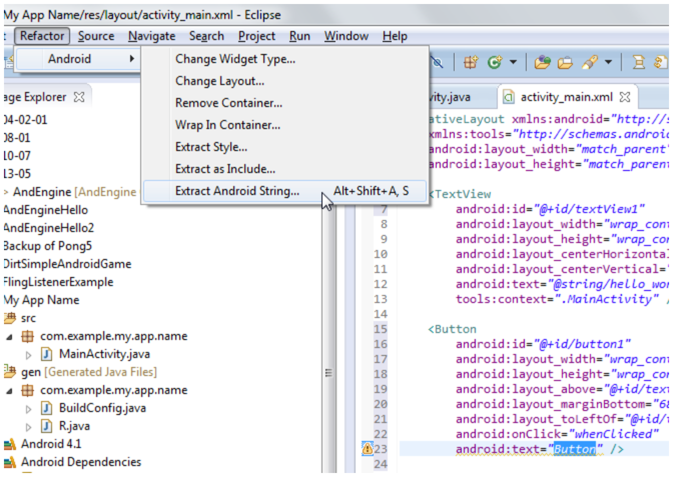
Eclipse gives you more warnings than it used to give you. For example, Eclipse warns you when you use a string (instead of a resource from the strings.xml file).



```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="@string/hello_world"
        tools:context=".MainActivity" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/textView1"
        android:layout_marginBottom="68dp"
        android:layout_toLeftOf="@+id/textView1"
        android:onClick="whenClicked"
        android:text="Button" />

</RelativeLayout>
```

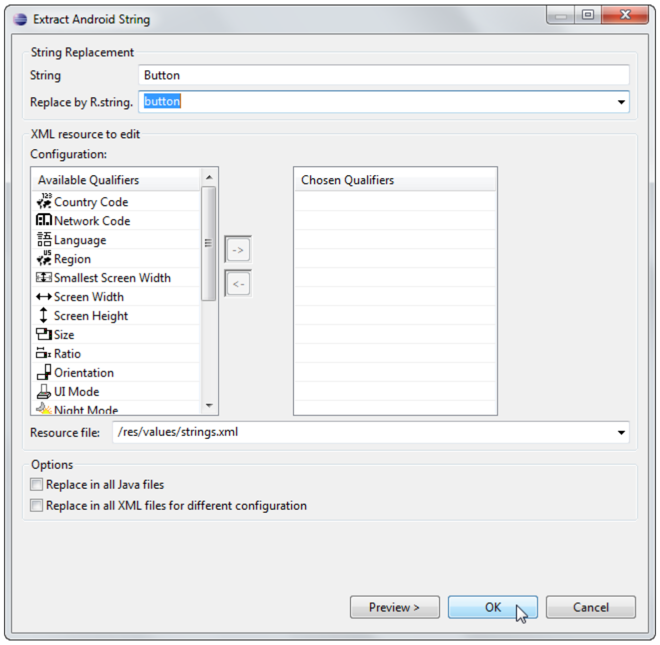
[I18N] Hardcoded string "Button", should use @string resource

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.



```
ativeLayout xmlns:android="http://:
xmlns:tools="http://schemas.androic
android:layout_width="match_parent'
android:layout_height="match_paren'

<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_con'
    android:layout_height="wrap_cor
    android:layout_centerHorizonta:
    android:layout_centerVertical='
    android:text="@string/hello_wor
    tools:context=".MainActivity" ,

<Button
    android:id="@+id/button1"
    android:layout_width="wrap_con'
    android:layout_height="wrap_cor
    android:layout_above="@+id/tex'
    android:layout_marginBottom="6l
    android:layout_toLeftOf="@+id/'
    android:onClick="whenClicked"
    android:text="Button" />
```

In this example, I accept the defaults in the Extract Android String dialog box.



String Replacement

String: Button

Replace by R.string.: button

XML resource to edit

Configuration:

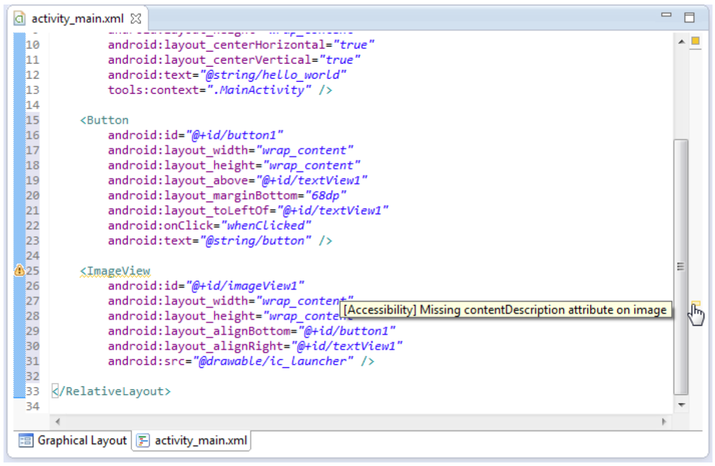
Available Qualifiers	Chosen Qualifiers
Country Code	
Network Code	
Language	
Region	
Smallest Screen Width	
Screen Width	
Screen Height	
Size	
Ratio	
Orientation	
UI Mode	
Night Mode	

Resource file: /res/values/strings.xml

Options

- [ ] Replace in all Java files
- [ ] Replace in all XML files for different configuration

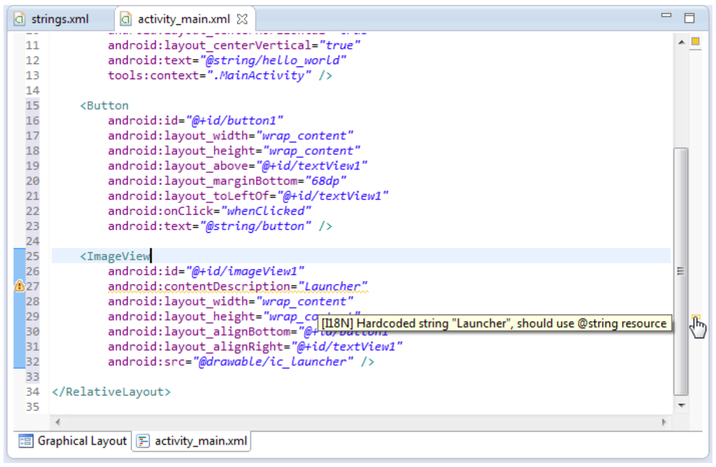
Eclipse wants me to add a text hint to my image…



```
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="@string/hello_world"
        tools:context=".MainActivity" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/textView1"
        android:layout_marginBottom="68dp"
        android:layout_toLeftOf="@+id/textView1"
        android:onClick="whenClicked"
        android:text="@string/button" />

    <ImageView
        android:id="@+id/imageView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_conte
        android:layout_alignBottom="@+id/button1"
        android:layout_alignRight="@+id/textView1"
        android:src="@drawable/ic_launcher" />

</RelativeLayout>
```

[Accessibility] Missing contentDescription attribute on image

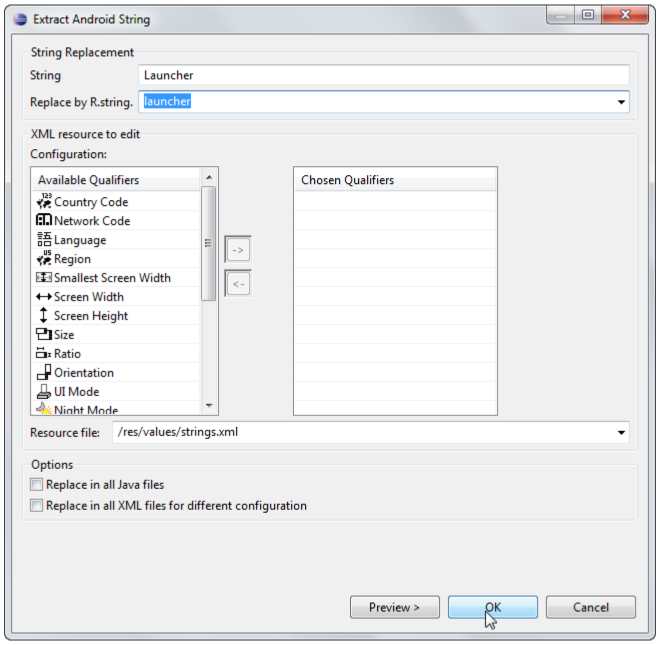
To add a text hint, I type an android:contentDescription attribute. Again, Eclipse wants me to use a resource from the strings.xml file.



```
        android:layout_centerVertical="true"
        android:text="@string/hello_world"
        tools:context=".MainActivity" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/textView1"
        android:layout_marginBottom="68dp"
        android:layout_toLeftOf="@+id/textView1"
        android:onClick="whenClicked"
        android:text="@string/button" />

    <ImageView
        android:id="@+id/imageView1"
        android:contentDescription="Launcher"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/button1"
        android:layout_alignRight="@+id/textView1"
        android:src="@drawable/ic_launcher" />

</RelativeLayout>
```

[I18N] Hardcoded string "Launcher", should use @string resource

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



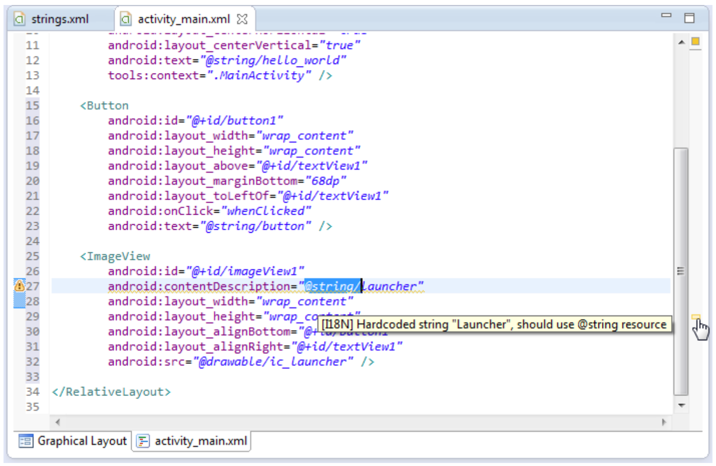
String Replacement
String: Launcher
Replace by R.string.: launcher

XML resource to edit
Configuration:

Resource file: /res/values/strings.xml

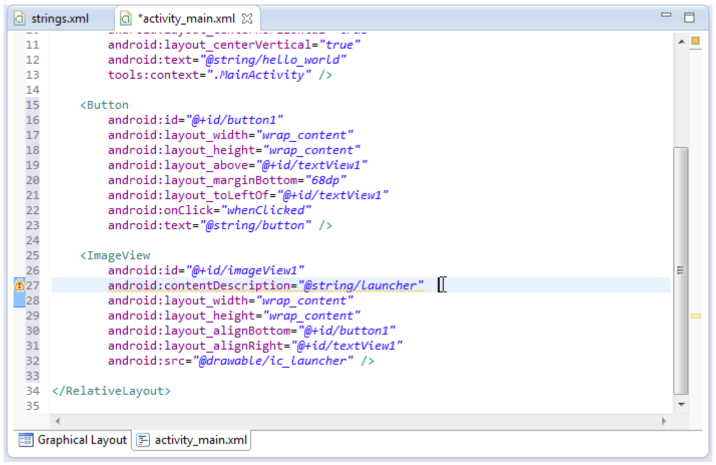
Options
☐ Replace in all Java files
☐ Replace in all XML files for different configuration

I've replaced a hardcoded string with a resource from the strings.xml file. But, for some reason, Eclipse doesn't recognize the change!



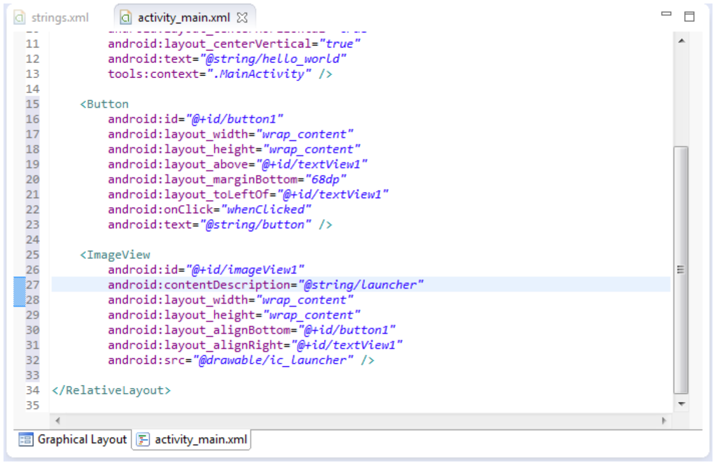
```
        android:layout_centerVertical="true"
        android:text="@string/hello_world"
        tools:context=".MainActivity" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/textView1"
        android:layout_marginBottom="68dp"
        android:layout_toLeftOf="@+id/textView1"
        android:onClick="whenClicked"
        android:text="@string/button" />

    <ImageView
        android:id="@+id/imageView1"
        android:contentDescription="@string/Launcher"
        android:layout_width="wrap_content"
        android:layout_height="wrap_co
        android:layout_alignBottom="@  [I18N] Hardcoded string "Launcher", should use @string resource
        android:layout_alignRight="@+id/textView1"
        android:src="@drawable/ic_launcher" />

</RelativeLayout>
```

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.



```
        android:layout_centerVertical="true"
        android:text="@string/hello_world"
        tools:context=".MainActivity" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/textView1"
        android:layout_marginBottom="68dp"
        android:layout_toLeftOf="@+id/textView1"
        android:onClick="whenClicked"
        android:text="@string/button" />

    <ImageView
        android:id="@+id/imageView1"
        android:contentDescription="@string/Launcher"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/button1"
        android:layout_alignRight="@+id/textView1"
        android:src="@drawable/ic_launcher" />

</RelativeLayout>
```

Resaving the layout file makes Eclipse happy.



```
        android:layout_centerVertical="true"
        android:text="@string/hello_world"
        tools:context=".MainActivity" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/textView1"
        android:layout_marginBottom="68dp"
        android:layout_toLeftOf="@+id/textView1"
        android:onClick="whenClicked"
        android:text="@string/button" />

    <ImageView
        android:id="@+id/imageView1"
        android:contentDescription="@string/Launcher"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/button1"
        android:layout_alignRight="@+id/textView1"
        android:src="@drawable/ic_launcher" />

</RelativeLayout>
```