Building "HelloWorld" Android Application Step by Step.

Sang Shin
Michèle Garoche
www.javapassion.com
"Learn with Passion!"

Disclaimer

- Portions of this presentation are modifications based on work created and shared by the Android Open Source Project
  - http://code.google.com/policies.html
- They are used according to terms described in the Creative Commons 2.5 Attribution License
  - http://creativecommons.org/licenses/by/2.5/

Goals of This Presentation

- Get you started with Android application development with minimum knowledge
- Get you know the development environment
  - Will focus on Eclipse-based Android development environment (instead of command-line-based one)
- Get you be exposed to the basic building blocks of an Android application in a cursory manner
  - The details will be covered in the rest of the course

Topics

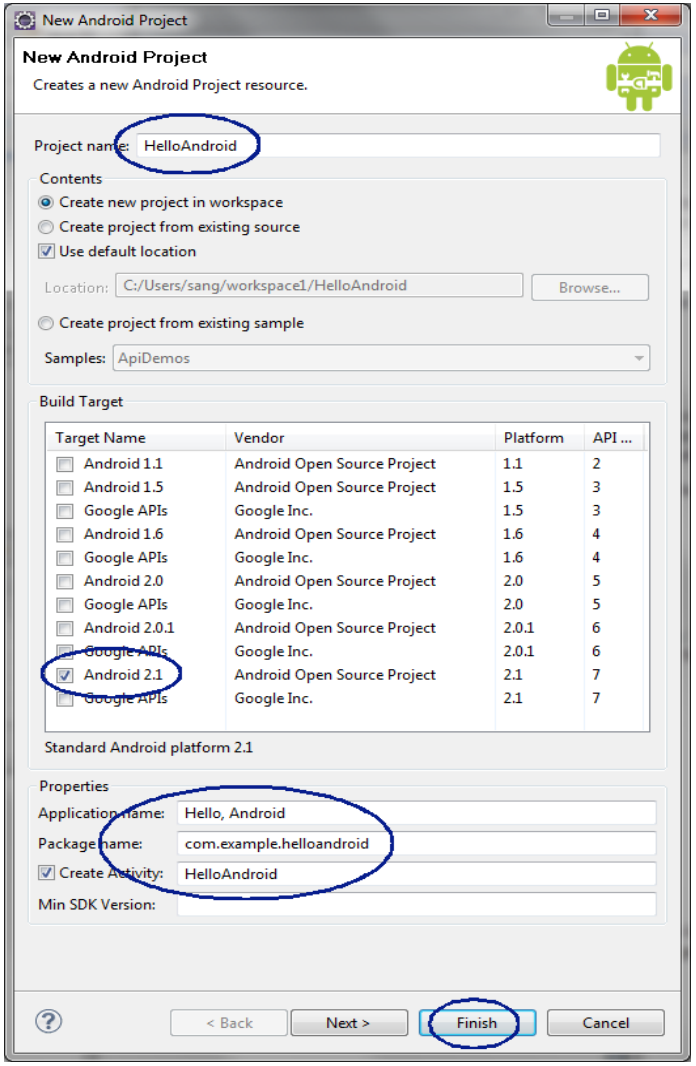
- Installation and configuration
- Building blocks of Android application
  - Activities
  - Layout resource files
  - AndroidManifest.xml
  - Resource files - strings.xml
  - R.java (created for you)

Installation & Configuration

Software Needed

- JDK (Java Development Kit)
  - Java programming language is used to build Android application
- Eclipse IDE
  - Editor, Debugger, profiler, deployment
- Android SDK
  - Libraries, samples, documentation, handset emulators, debugger, command line tools
- Eclipse ADT (Android Development Tools) plug-in
  - Give you a powerful, integrated Android development environment

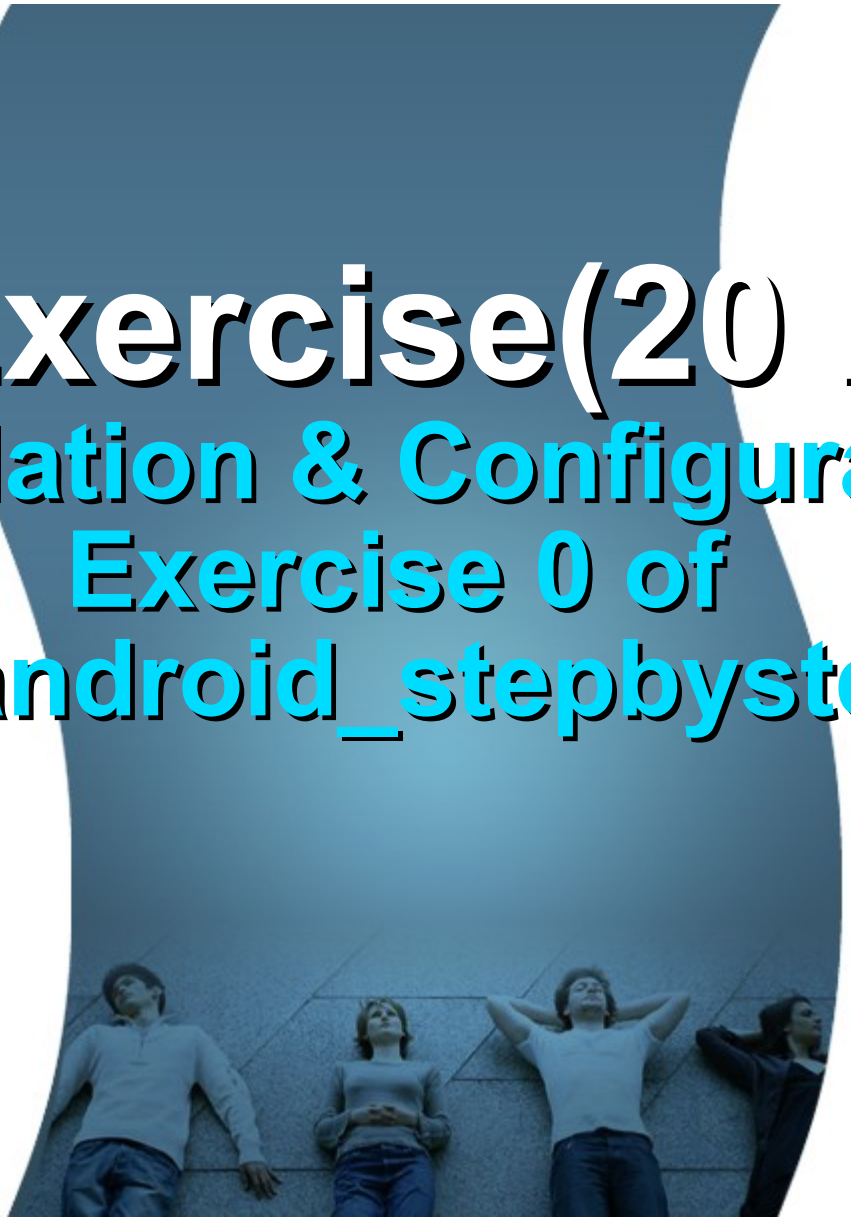
Creating Android Project using Eclipse



- Automatic creation of building blocks of an Android application
  - Starting Activity class
  - Layout resource file
  - AndroidManifest.xml
  - strings.xml
  - R.java
  - Android library

Lab Exercise(20 min):

Installation & Configuration Exercise 0 of 6110_android_stepbystep.zip

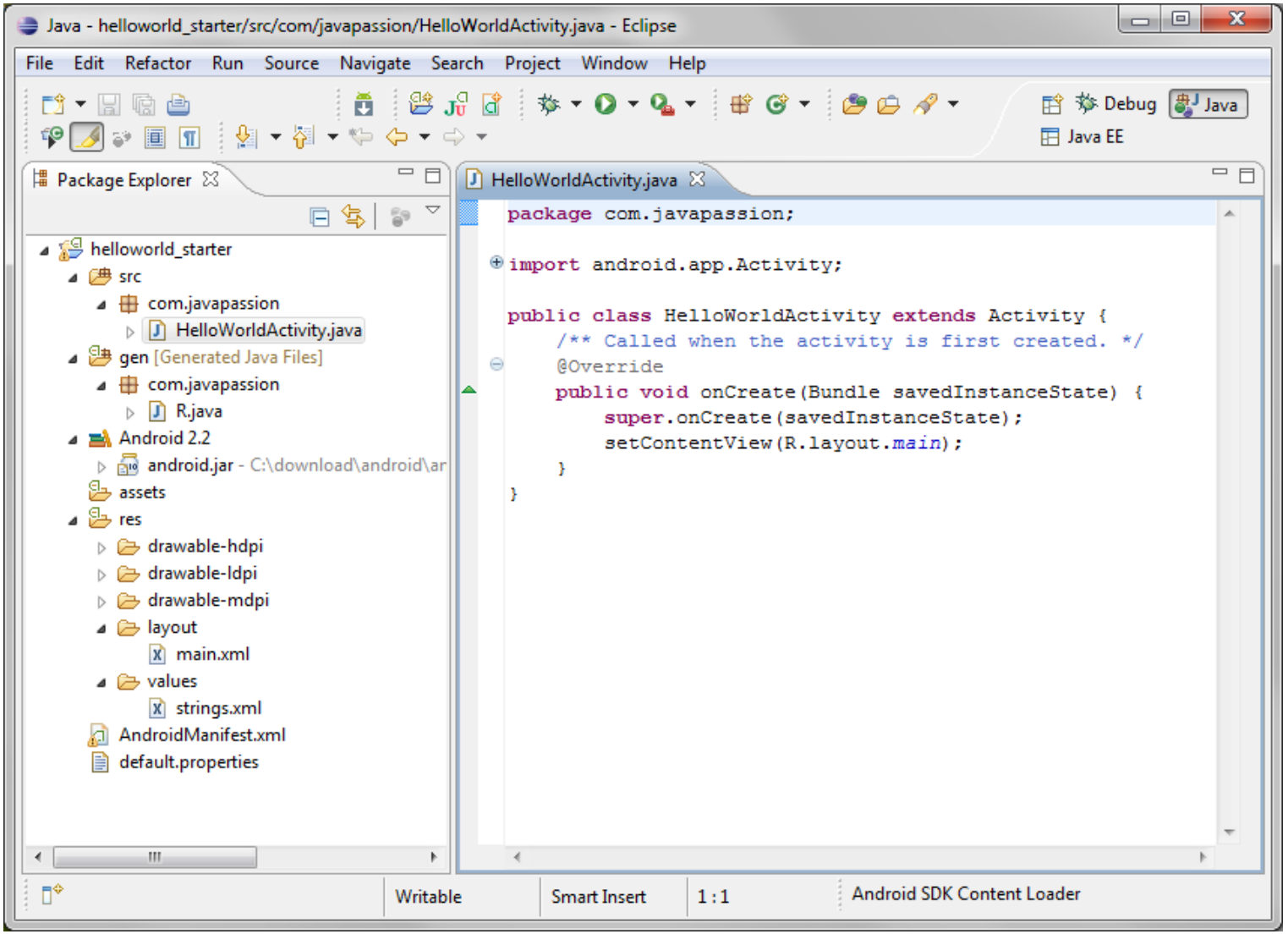


Building Blocks of Android Application

Building Blocks of Android Application

- Activity classes
- Layout resource files
- Resource files
  - strings.xml, etc..
- AndroidManifest.xml
- R.java (automatically created from resource files)
- Android library (automatically configured)

Building Blocks of Android Application



Activity Class - Quick Overview

- Each Activity class typically represents a screen
  - Like a JSP page in a Web application
- The *onCreate()* method of Activity class gets called by the Android system when your Activity starts
  - You create your screen UI inside *onCreate()* method
- Every Activity has to be described in the *AndroidManifest.xml* file
- An Activity is typically chosen as a starting one of your application - like a class that has a main() method in Java
  - Through special configuration in *AndroidManifest.xml*

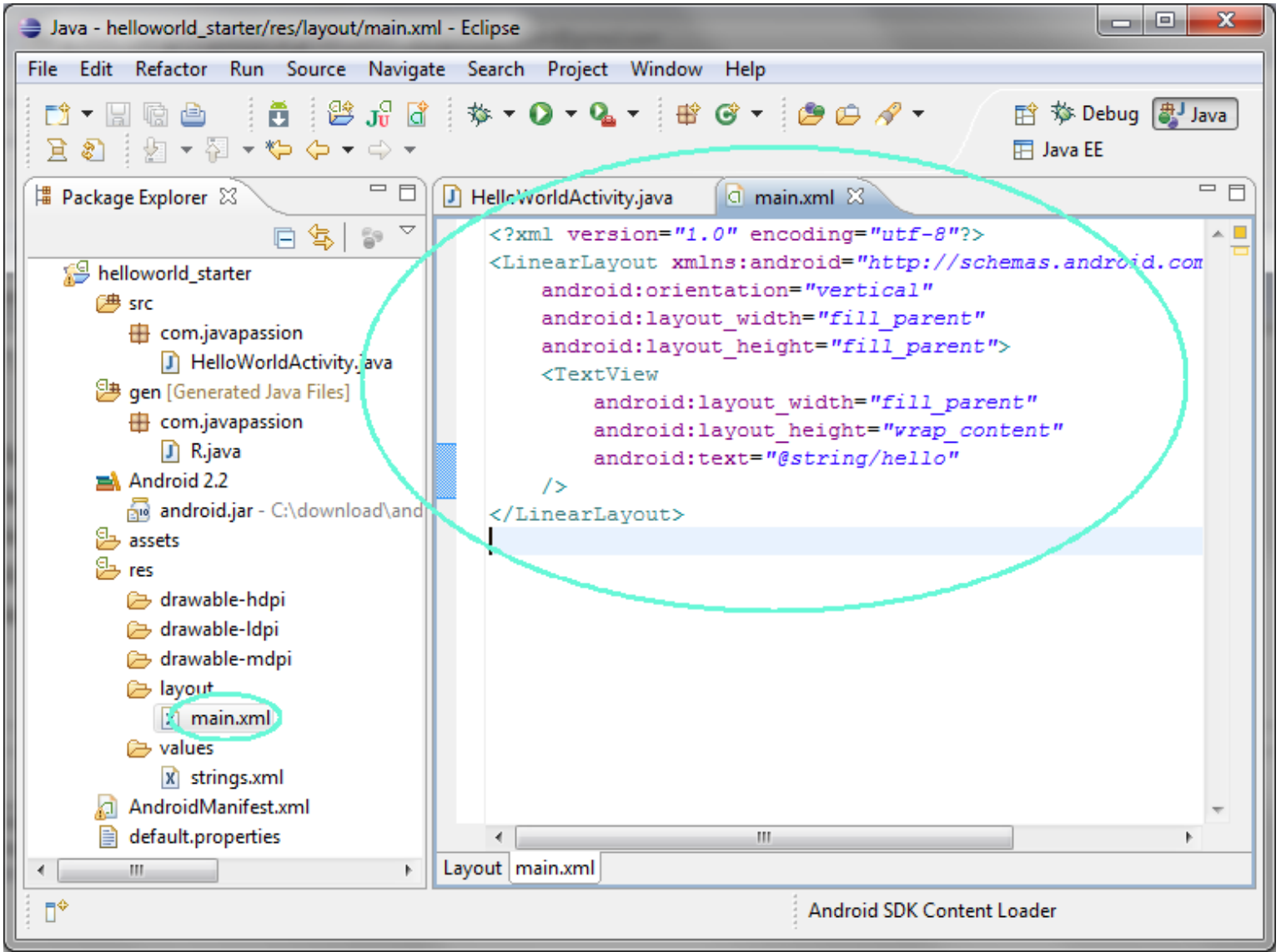
Activity Class Example

```
package com.javapassion;

import android.app.Activity;
import android.os.Bundle;

public class HelloWorldActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Create UI using Layout resource file
        setContentView(R.layout.main);
    }
}
```

Layout Resource Files

- Every screen has a corresponding layout resource file
  - Unless you create screen UI programmatically
- Each activity class specifies which layout resource file to use for each screen it represents
  - Using **`setContentView(R.layout.main);`**
- Located under */res/layout* directory
  - /res/layout/main.xml layout resource file is referred to as **`R.layout.main`**

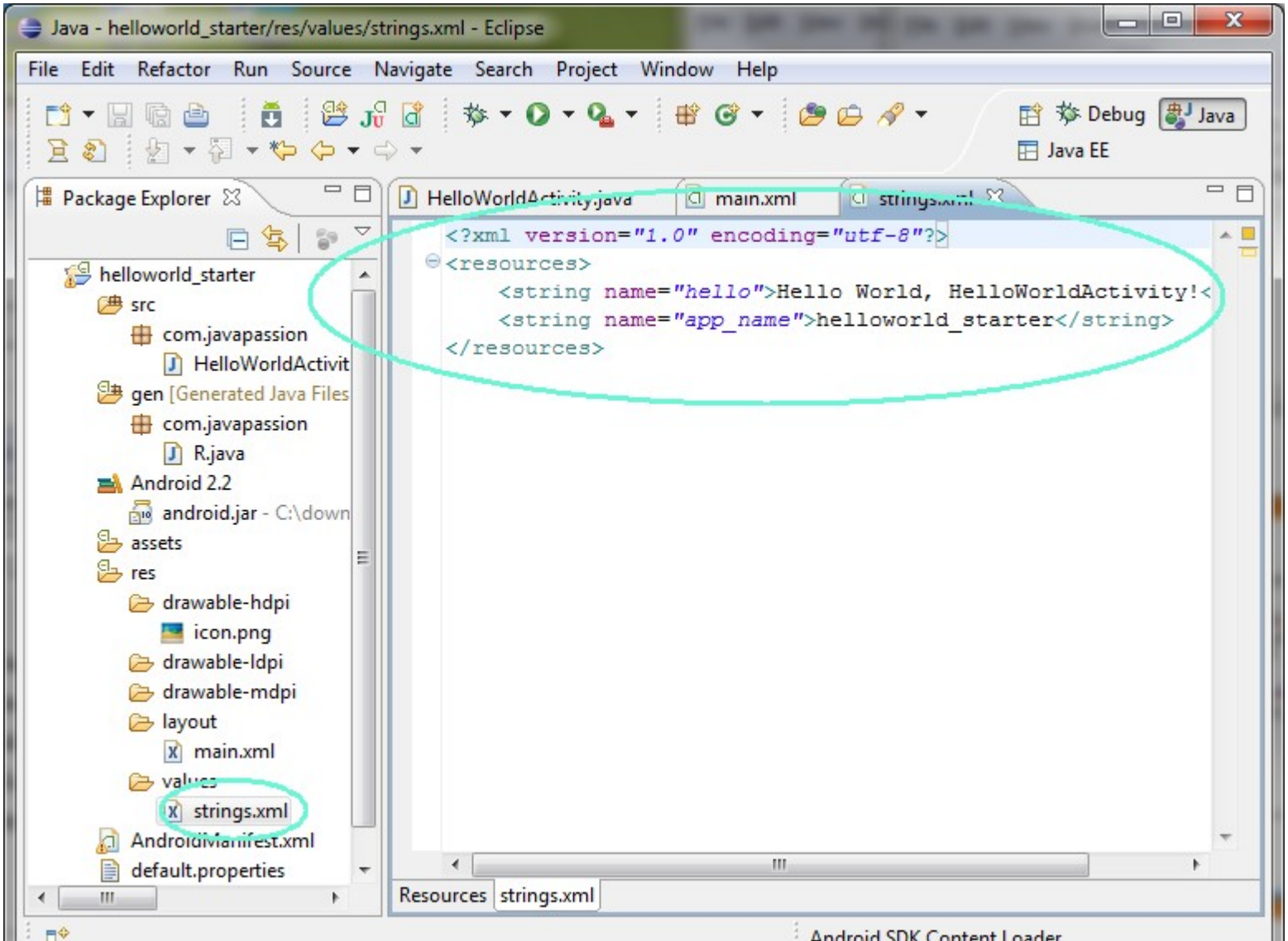
Layout Resource File Example



Resource Files - strings.xml

- Let you define the text strings of your applications in a well-known file
  - Rather than in Java code
  - Rather than in Layout resource files
- The strings are then referred to through the names assigned to them
  - The mapping is done through R.java
  - *@string/hello* (in the layout resource file)
  - *R.string.hello* (in the Java code)
- Located under /res/values directory

strings.xml Resource File



AndroidManifest.xml file

- Every application must have a manifest file called *AndroidManifest.xml* file (with precisely that name) in its root directory.
- The manifest presents essential information about the application to the Android system, information the system must have before it can run any of the application's code.

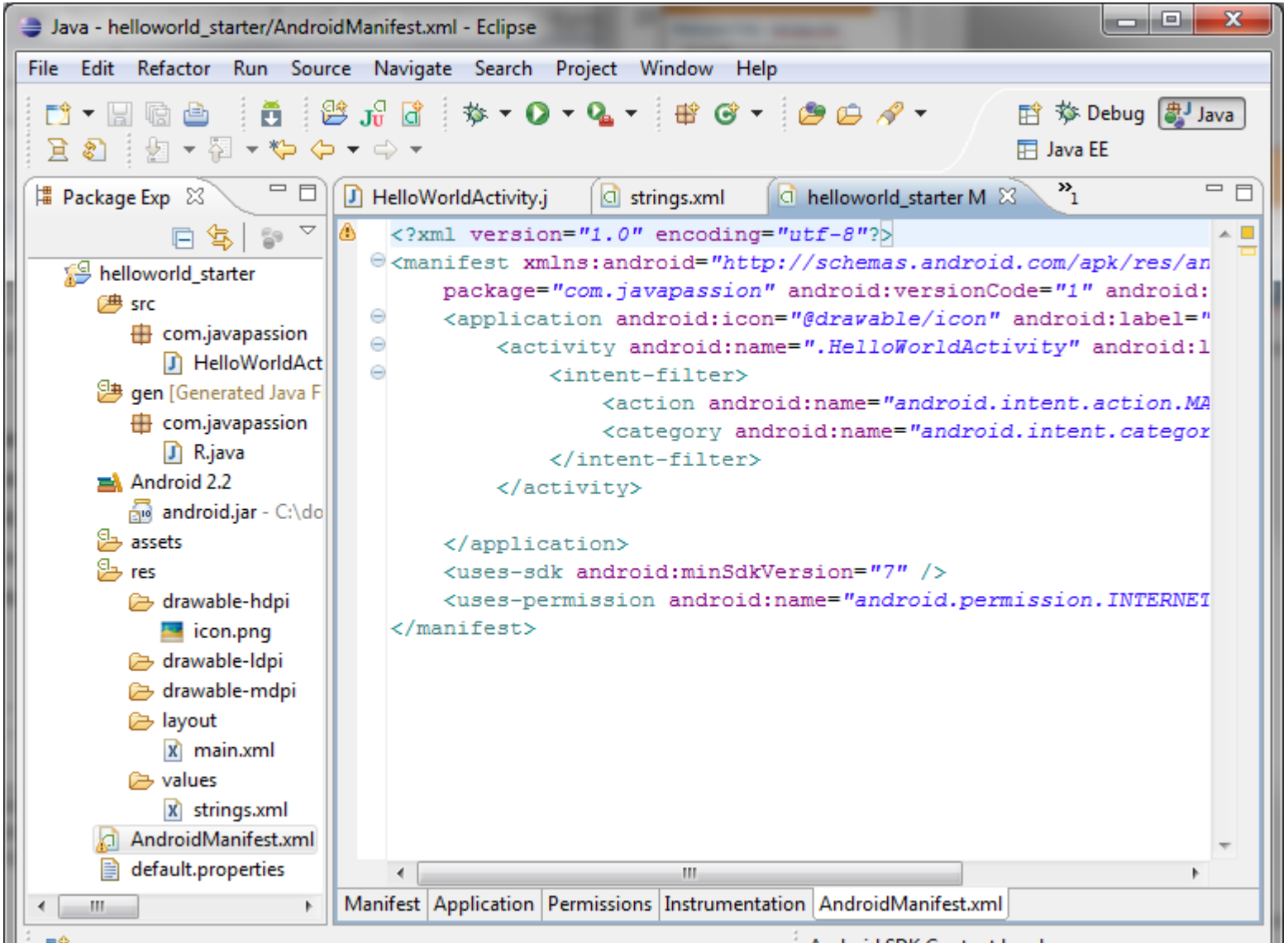
Info. in AndroidManifest.xml file

- Java package for the application.
  - The package name serves as a unique identifier for the application.
- Application name and icon
- Application version information
- Activities
  - One activity is designated as a starting Activity

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.example.helloandroid"
      android:versionCode="1"
      android:versionName="1.0">
  <application android:icon="@drawable/icon"
                  android:label="@string/app_name">
     <activity android:name=".HelloAndroid"
                   android:label="@string/app_name2">
        <intent-filter>
          <action android:name="android.intent.action.MAIN" />
          <category
              android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
     </activity>
  </application>
  <uses-sdk android:minSdkVersion="7" />
</manifest>
```

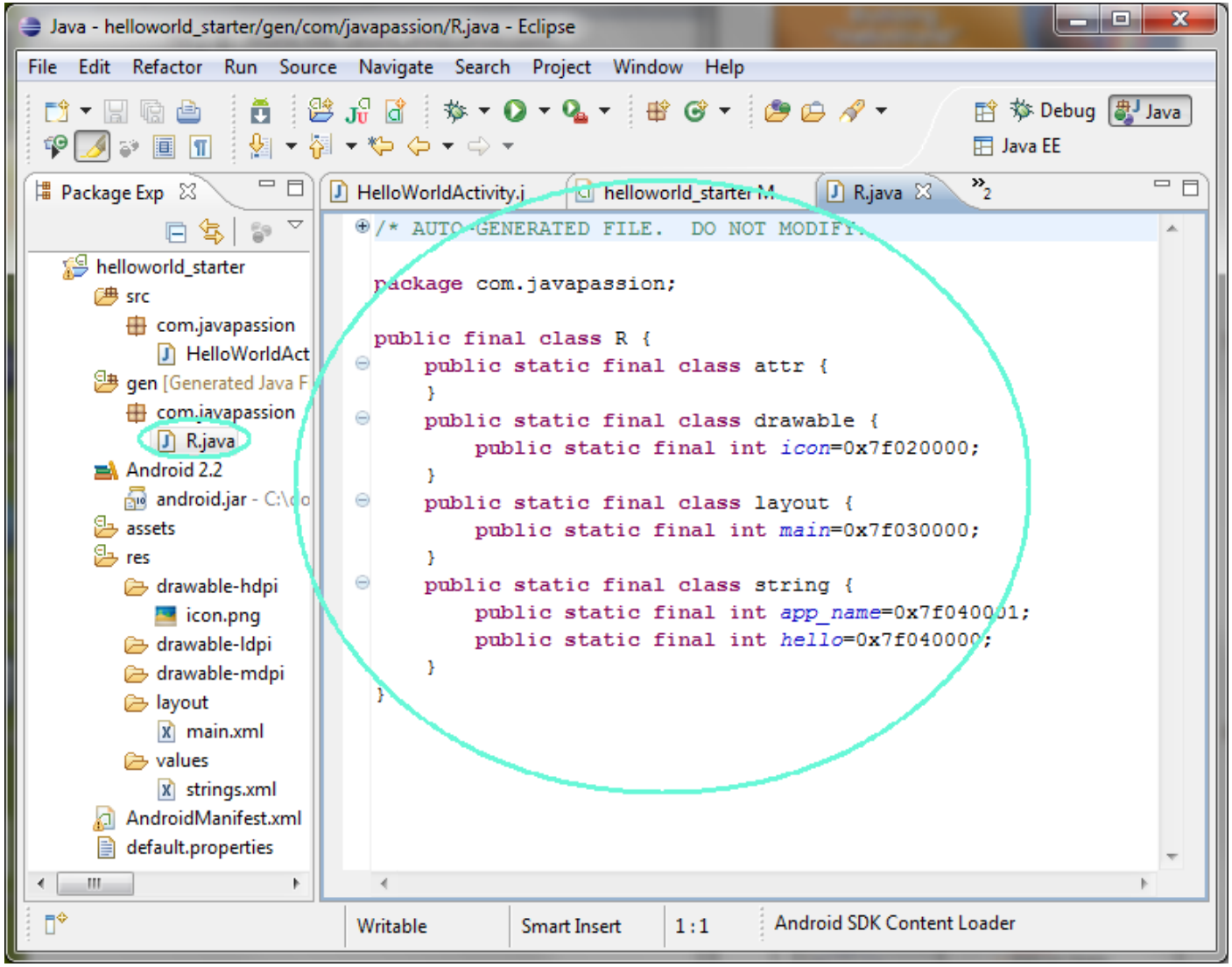
AndroidManifest.xml



R.java

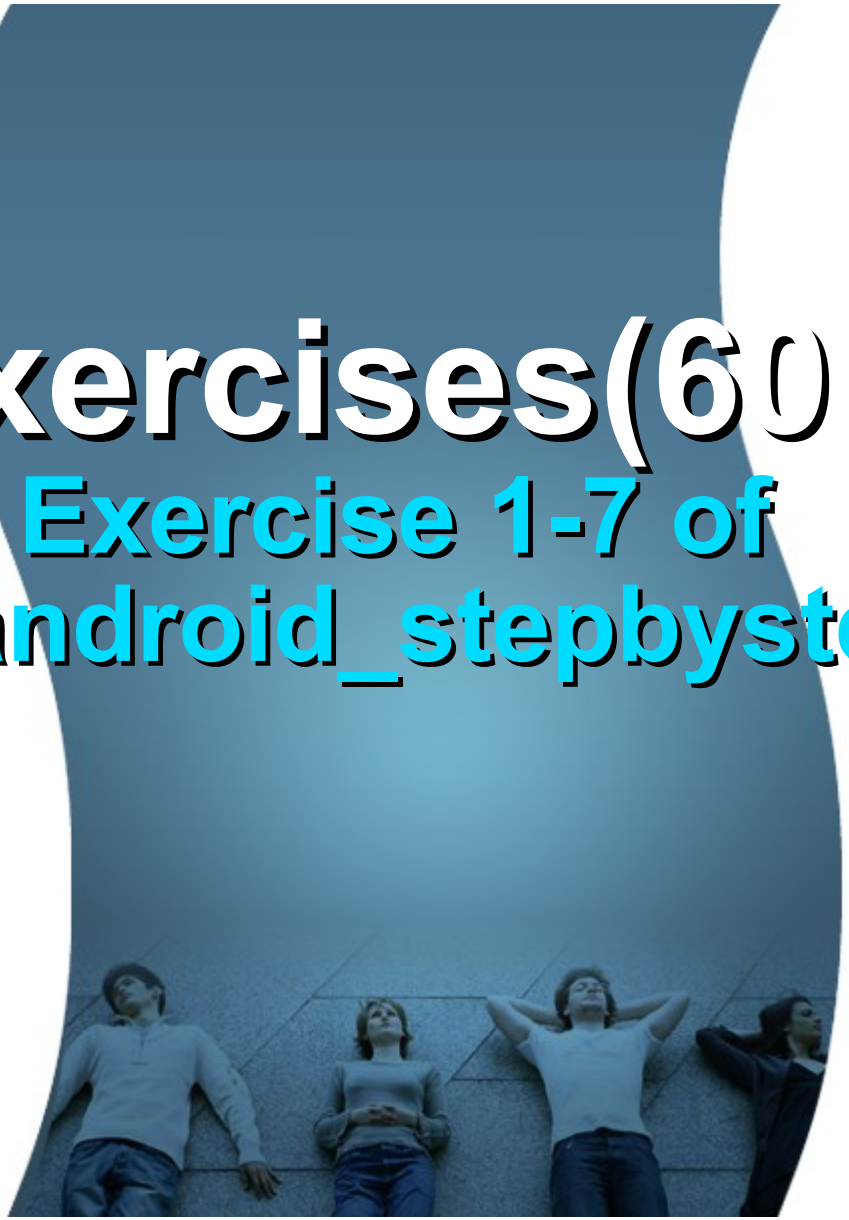
- Automatically created by Android system for all resources defined

R.java


Java - helloworld_starter/gen/com/javapassion/R.java - Eclipse
File Edit Refactor Run Source Navigate Search Project Window Help
Debug
Java
Java EE
Package Exp
HelloWorldActivity.j
helloworld_starter M
R.java
helloworld_starter
src
com.javapassion
HelloWorldAct
gen [Generated Java F
com.javapassion
R.java
Android 2.2
android.jar - C:\
assets
res
drawable-hdpi
icon.png
drawable-ldpi
drawable-mdpi
layout
main.xml
values
strings.xml
AndroidManifest.xml
default.properties
/* AUTO-GENERATED FILE. DO NOT MODIFY.
package com.javapassion;
public final class R {
public static final class attr {
}
public static final class drawable {
public static final int icon=0x7f020000;
}
public static final class layout {
public static final int main=0x7f030000;
}
public static final class string {
public static final int app_name=0x7f040001;
public static final int hello=0x7f040000;
}
}
Writable
Smart Insert
1 : 1
Android SDK Content Loader

Lab Exercises(60 min):

Exercise 1-7 of
6110_android_stepbystep.zip



Thank you!

Check JavaPassion.com Codecamps!
http://www.javapassion.com/codecamps
"Learn with Passion!"