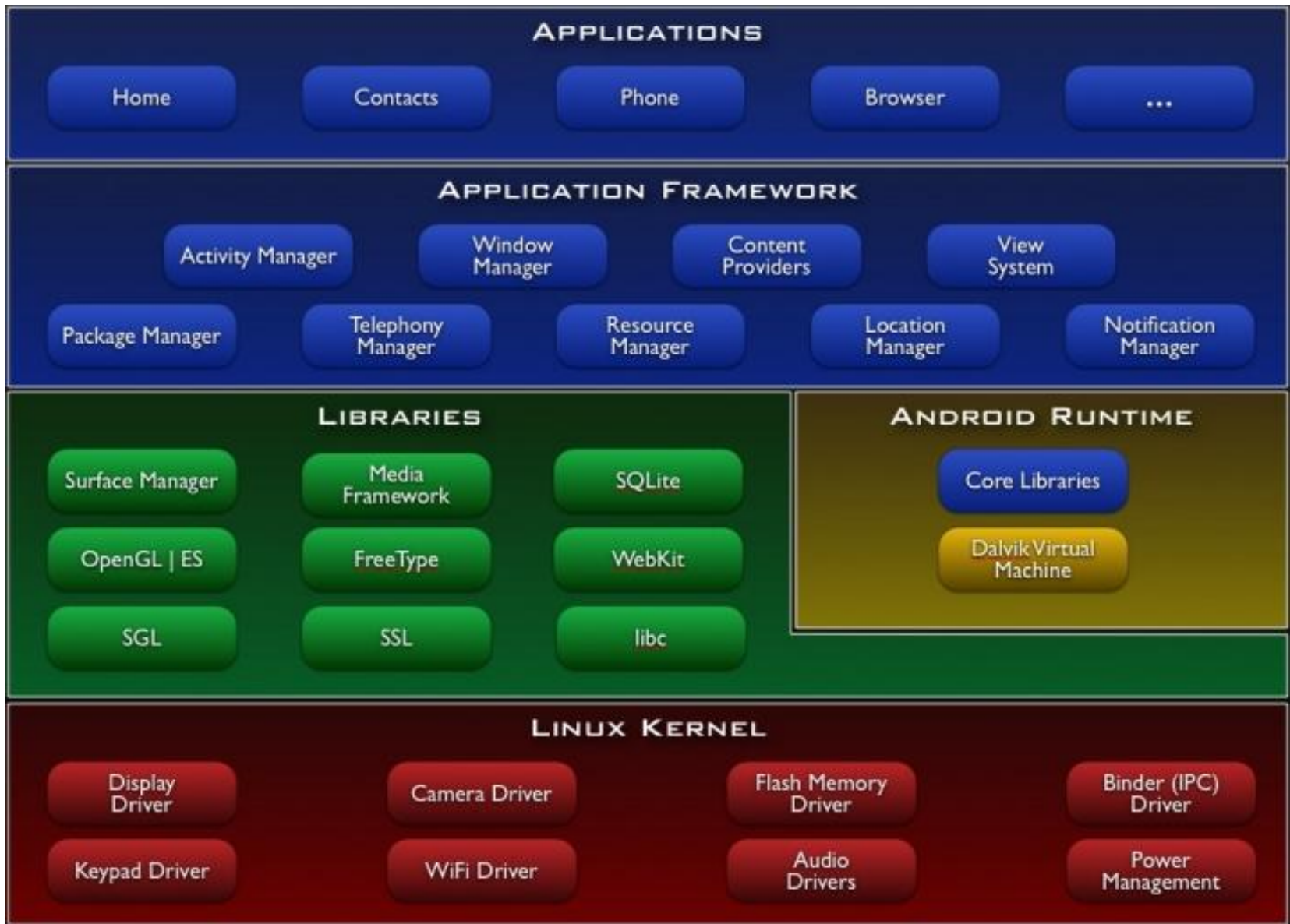


# Introduction to Android Development Environment

ECOD - Aug 2015

# What is Android?

- A software stack for mobile devices that includes
  - An operating system
  - Middleware
  - Key Applications
- Uses Linux to provide core system services
  - Security
  - Memory management
  - Process management
  - Power management
  - Hardware drivers



<http://developer.android.com/guide/basics/what-is-android.html>

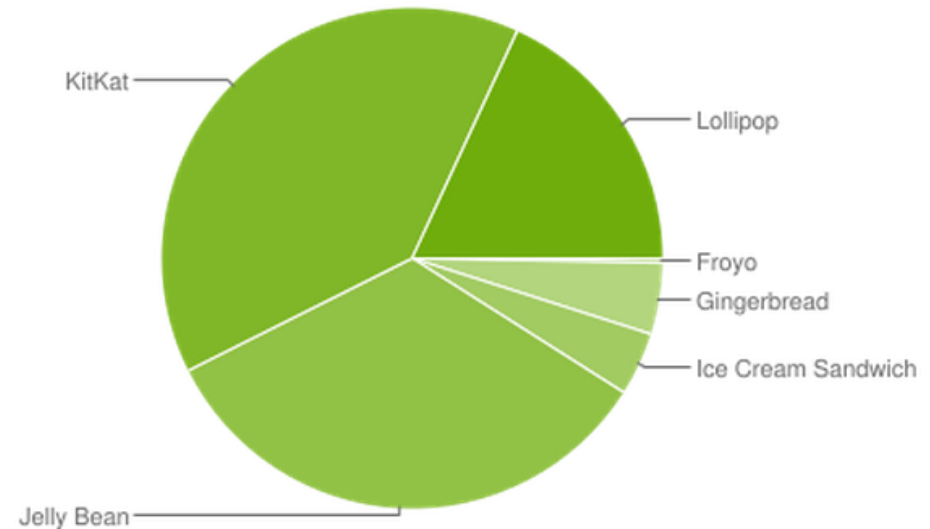
# Android Features

- **Application framework** enabling reuse and replacement of components
- **Dalvik virtual machine** optimized for mobile devices
- **Integrated browser** based on the open source [WebKit](#) engine
- **Optimized graphics** powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- **SQLite** for structured data storage
- **Media support** for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- **GSM Telephony** (hardware dependent)
- **Bluetooth, EDGE, 3G, and WiFi** (hardware dependent)
- **Camera, GPS, compass, and accelerometer** (hardware dependent)
- **Rich development environment** including a device emulator, tools for debugging, memory and performance profiling, and a plugin for the Eclipse IDE

<http://developer.android.com/guide/basics/what-is-android.html>

# Android Distribution Aug 2015

Version	Codename	API	Distribution
2.2	Froyo	8	0.3%
2.3.3 - 2.3.7	Gingerbread	10	4.6%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	4.1%
4.1.x	Jelly Bean	16	13.0%
4.2.x		17	15.9%
4.3		18	4.7%
4.4	KitKat	19	39.3%
5.0	Lollipop	21	15.5%
5.1		22	2.6%



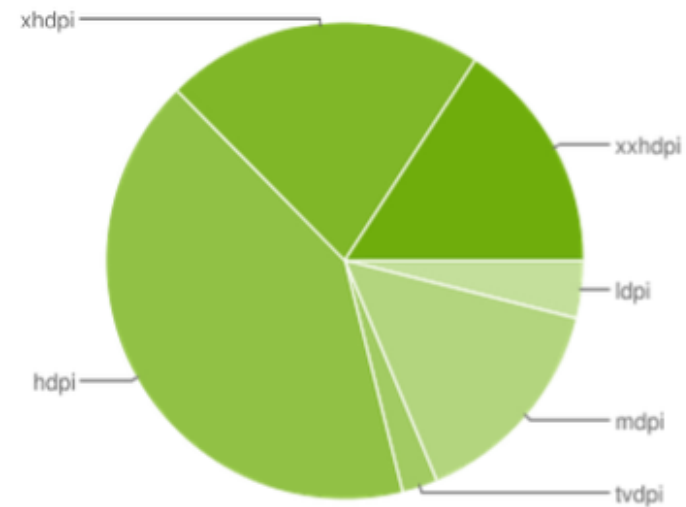
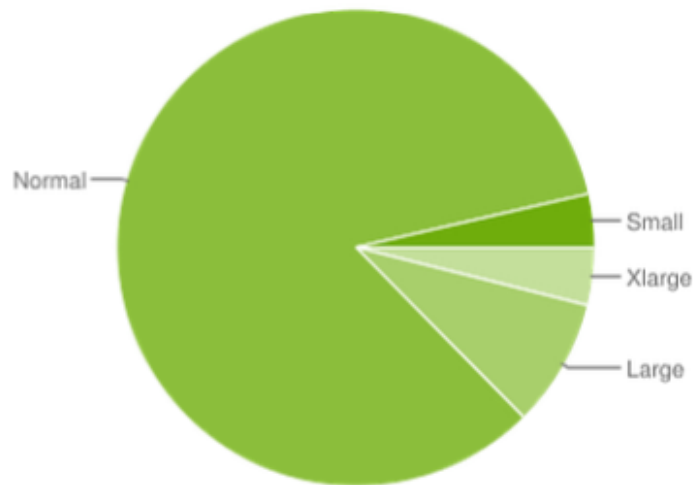
<https://developer.android.com/about/dashboards/index.html>

*Data collected during a 7-day period ending on August 3, 2015.*

*Any versions with less than 0.1% distribution are not shown.*

# Screen Densities as of August 2015

	~120dpi	~160dpi	~240dpi	~320dpi	~480dpi	~640dpi	
	ldpi	mdpi	tvdpi	hdpi	xhdpi	xxhdpi	Total
Small	3.6%						3.6%
Normal		6.9%	0.1%	40.6%	20.4%	15.8%	83.8%
Large	0.3%	4.9%	2.3%	0.6%	0.6%		8.7%
Xlarge		3.0%		0.3%	0.6%		3.9%
Total	3.9%	14.8%	2.4%	41.5%	21.6%	15.8%	



Data collected during a 7-day period ending on August 3, 2015.

Any screen configurations with less than 0.1% distribution are not shown.

# Android Runtime

# Android Runtime: Dalvik VM

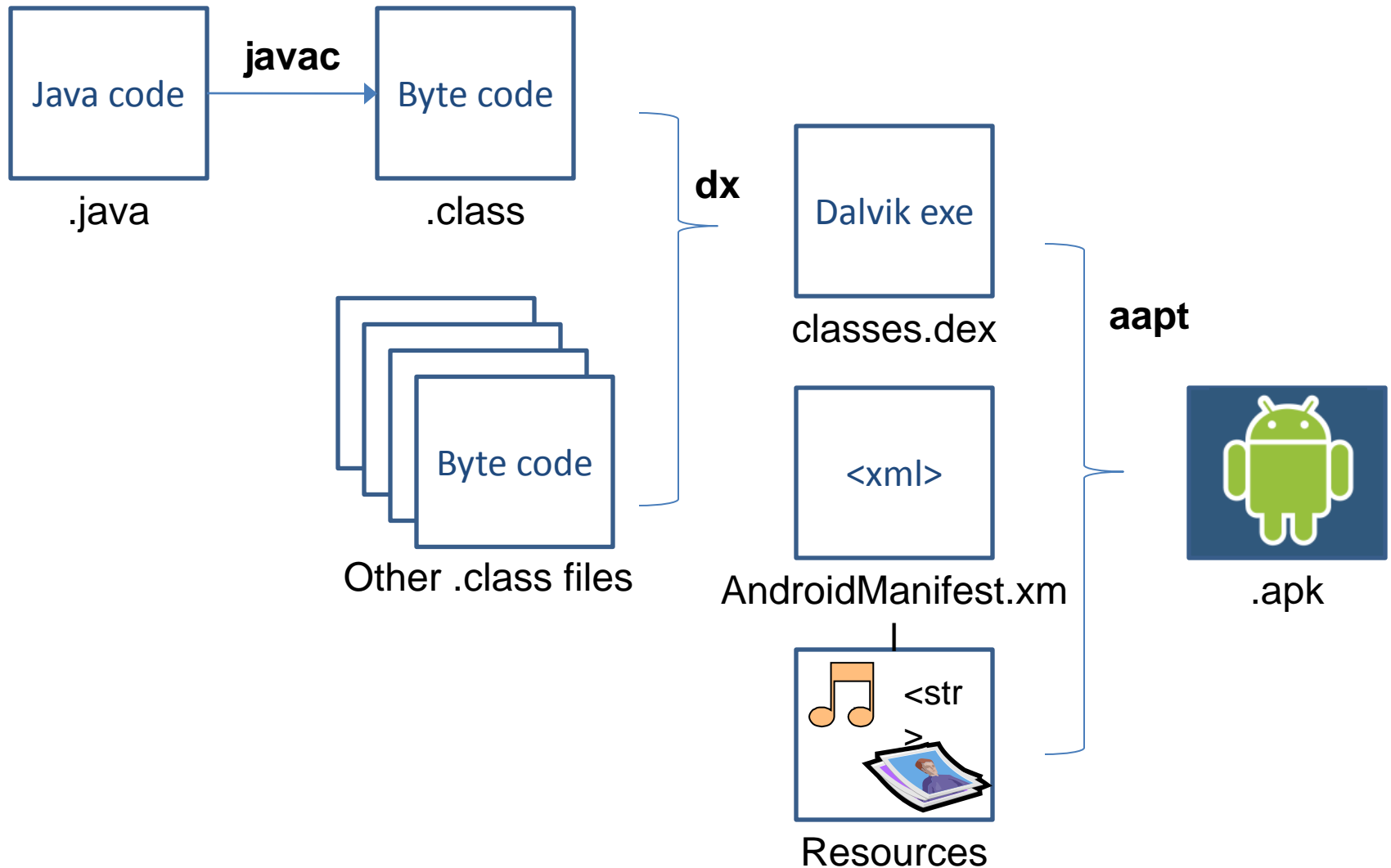
- Subset of Java developed by Google
- Optimized for mobile devices (better memory management, battery utilization, etc.)
- Dalvik runs .dex files that are compiled from .class files
- Introduces new libraries
- Does not support some Java libraries like AWT, Swing
- <http://developer.android.com/reference/packages.html>



# Applications Are Boxed

- By default, each app is run in its own Linux process
  - Process started when app's code needs to be executed
  - Threads can be started to handle time-consuming operations
- Each process has its own Dalvik VM
- By default, each app is assigned unique Linux ID
  - Permissions are set so app's files are only visible to that app

# Producing an Android App



Emulator

# Emulator Basics

- Host computer's keyboard works
- Host's mouse acts as finger
- Uses host's Internet connection
- Other buttons work: Home, Menu, Back, Search, volume up and down, etc.
- Ctrl-F11 toggle landscape → portrait
- Alt-Enter toggle full-screen mode
- More info at <http://developer.android.com/guide/developing/devices/emulator.html>

# Emulator Limitations

- No support for placing or receiving actual phone calls
  - Simulate phone calls (placed and received) through the emulator console
- No support for USB connections
- No support for camera/video capture (input)
- No support for device-attached headphones
- No support for determining connected state
- No support for determining battery charge level and AC charging state
- No support for determining SD card insert/eject
- No support for Bluetooth
- No support for simulating the accelerometer
  - Use OpenIntents's Sensor Simulator

# Android Emulator or AVD

- Emulator is essential to testing app but is not a substitute for a real device
- Emulators are **Android Virtual Devices** (AVDs)
- Android SDK and AVD Manager allows you to create AVDs that target any Android API level
- AVD have configurable resolutions, RAM, SD cards, skins, and other hardware

# Live Lab

## Creating Hello World App

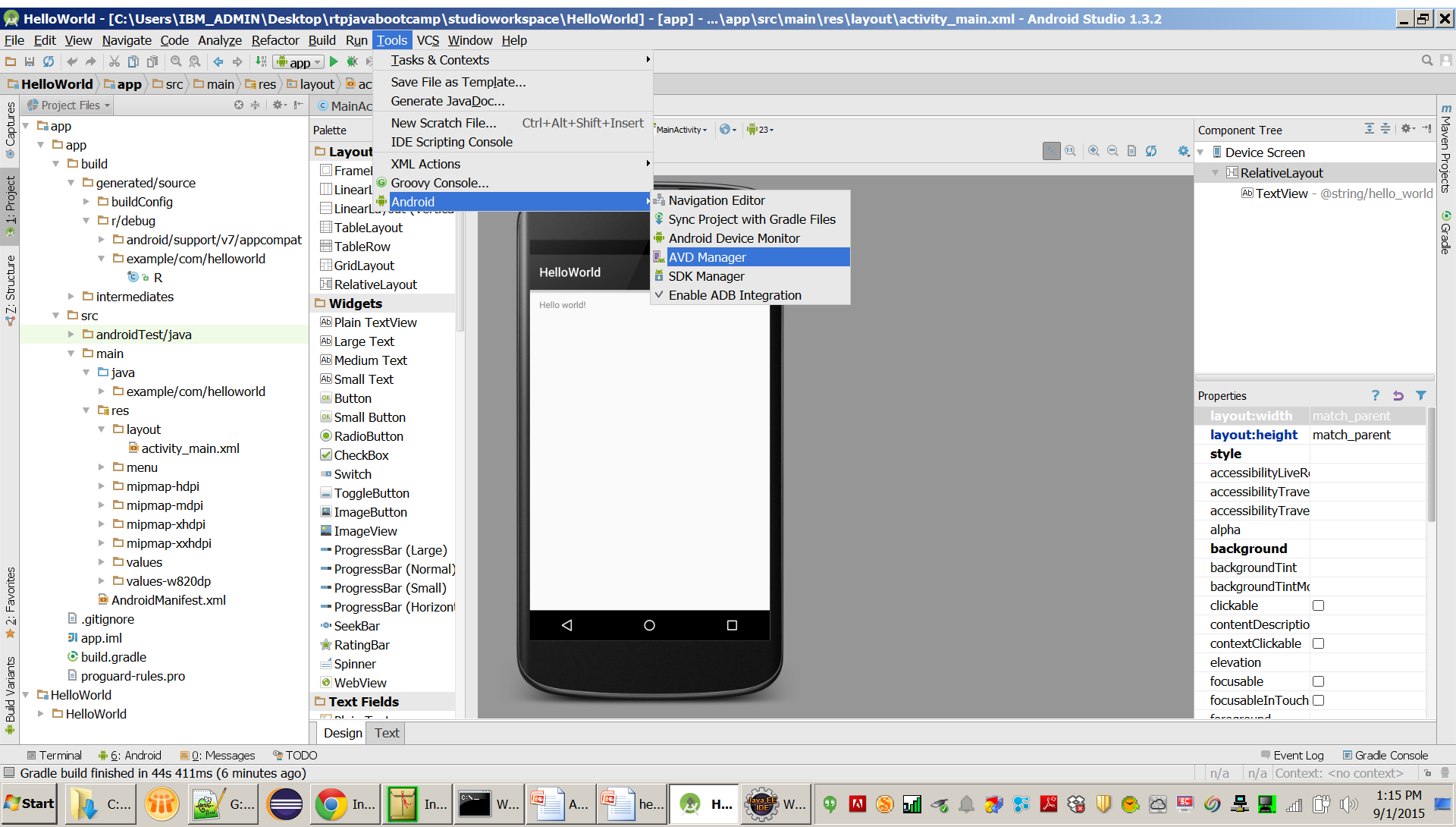



1. Create an Activity
2. Demonstrate resources created
3. show the Activity lifecycle within the Android OS
4. show the various debugging tools available
5. show how to start one Activity from another


# Pre-Req for Mobile Lab







Ensure the AVD and DSK are  
configured well






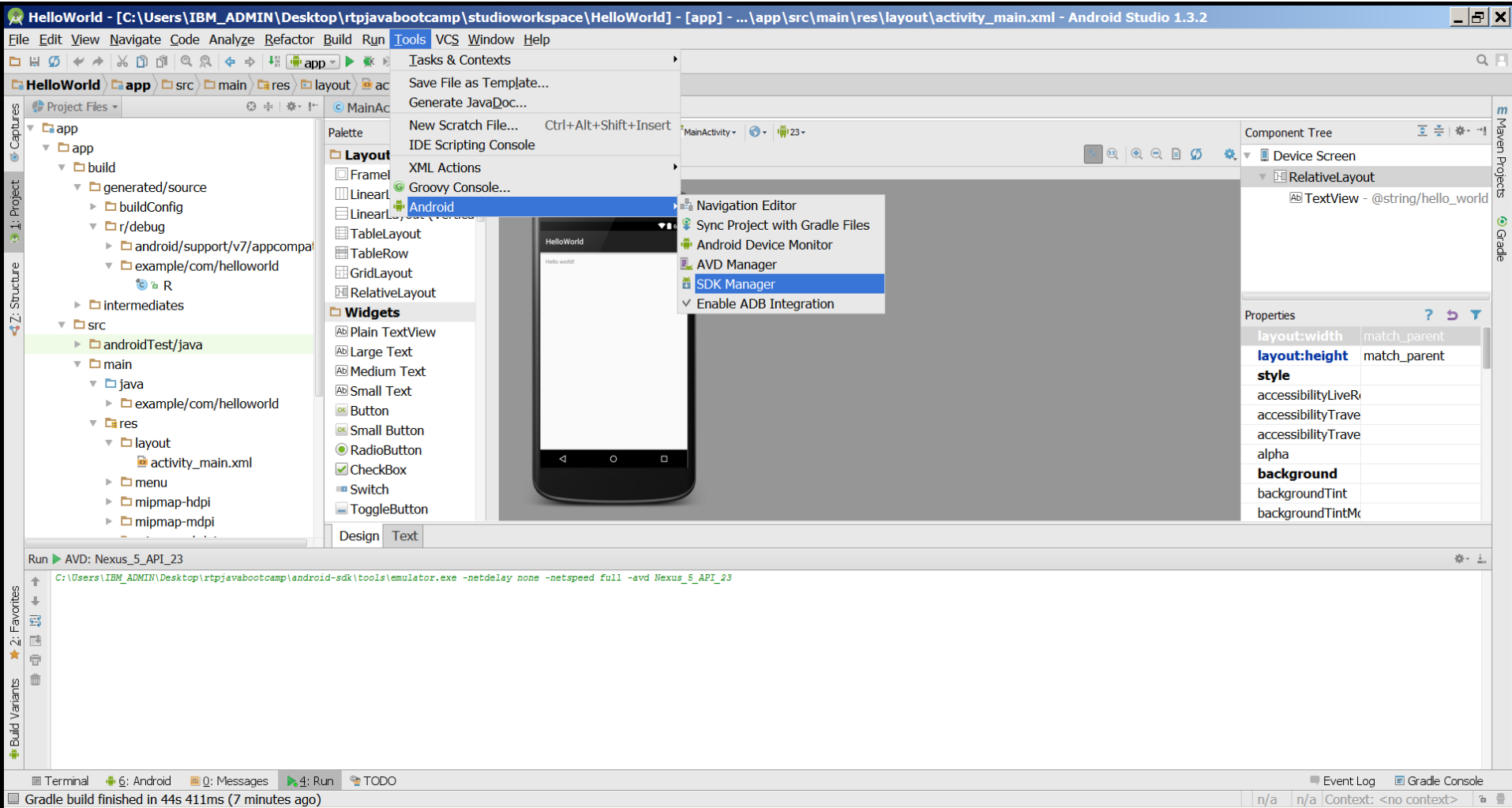
 Android Virtual Device Manager

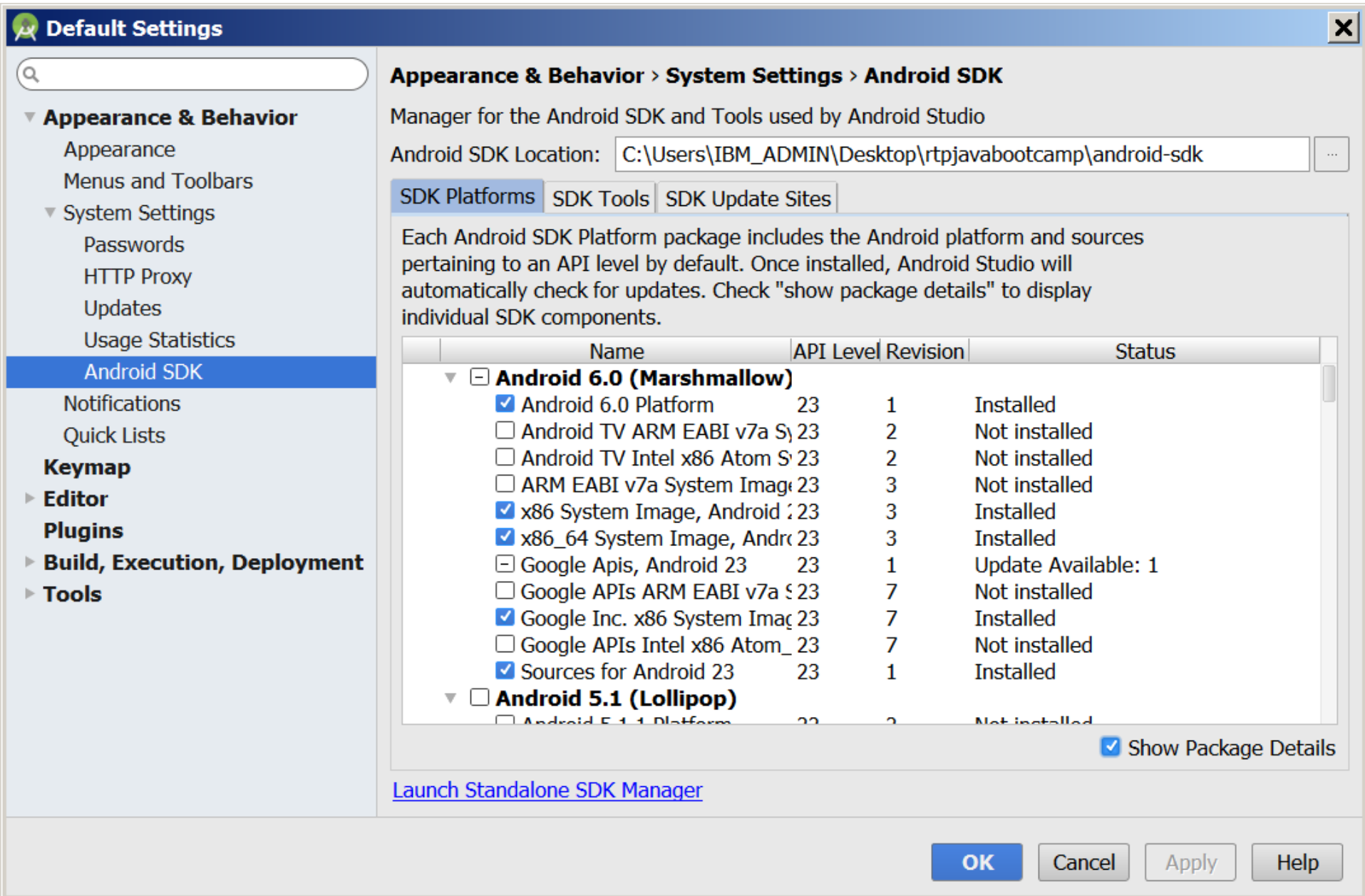
 Your Virtual Devices  
Android Studio

Type	Name	Resolution	API	Target	CPU/...	Size on Disk	Actions
	AndroidVirtualDevice	480 × 800: hdpi	N/A	N/A	arm	1 GB	 Failed to load
	Nexus 5 API 23	1080 × 1920: xxhdpi	23	Google APIs	x86	1 GB	  

+ Create Virtual Device...







# Mobile Labs

1-7

# Lab 1

Getting Hello World App Running



## New Project

Android Studio

### Configure your new project

Application name:

Company Domain:

Package name:  [Edit](#)

Project location:  ...

Previous

**Next**

Cancel

Finish



## Target Android Devices

## Select the form factors your app will run on

Different platforms may require separate SDKs

☒ Phone and Tablet

Minimum SDK

API 16: Android 4.1 (Jelly Bean)

Lower API levels target more devices, but have fewer features available. By targeting API 16 and later, your app will run on approximately **88.7%** of the devices that are active on the Google Play Store.

[Help me choose](#)

☐ Wear

Minimum SDK

API 21: Android 5.0 (Lollipop)

☐ TV

Minimum SDK

API 21: Android 5.0 (Lollipop)

☐ Android Auto

☐ Glass (Not Installed) [Download](#)

Minimum SDK

Previous

Next

Cancel

Finish





## Add an activity to Mobile



Add No Activity



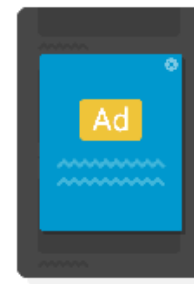
Blank Activity



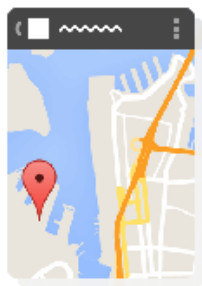
Blank Activity with Fragment



Fullscreen Activity



Google AdMob Ads Activity



Google Maps Activity



Google Play Services Activity



Login Activity



Master/Detail Flow



Navigation Drawer Activity



Previous

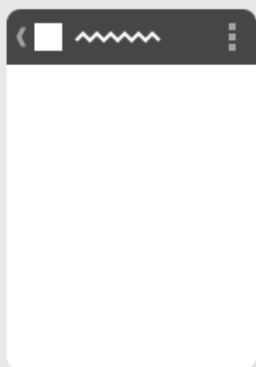
Next

Cancel

Finish



## Customize the Activity



Blank Activity

Creates a new blank activity with an action bar.

Activity Name:

Layout Name:

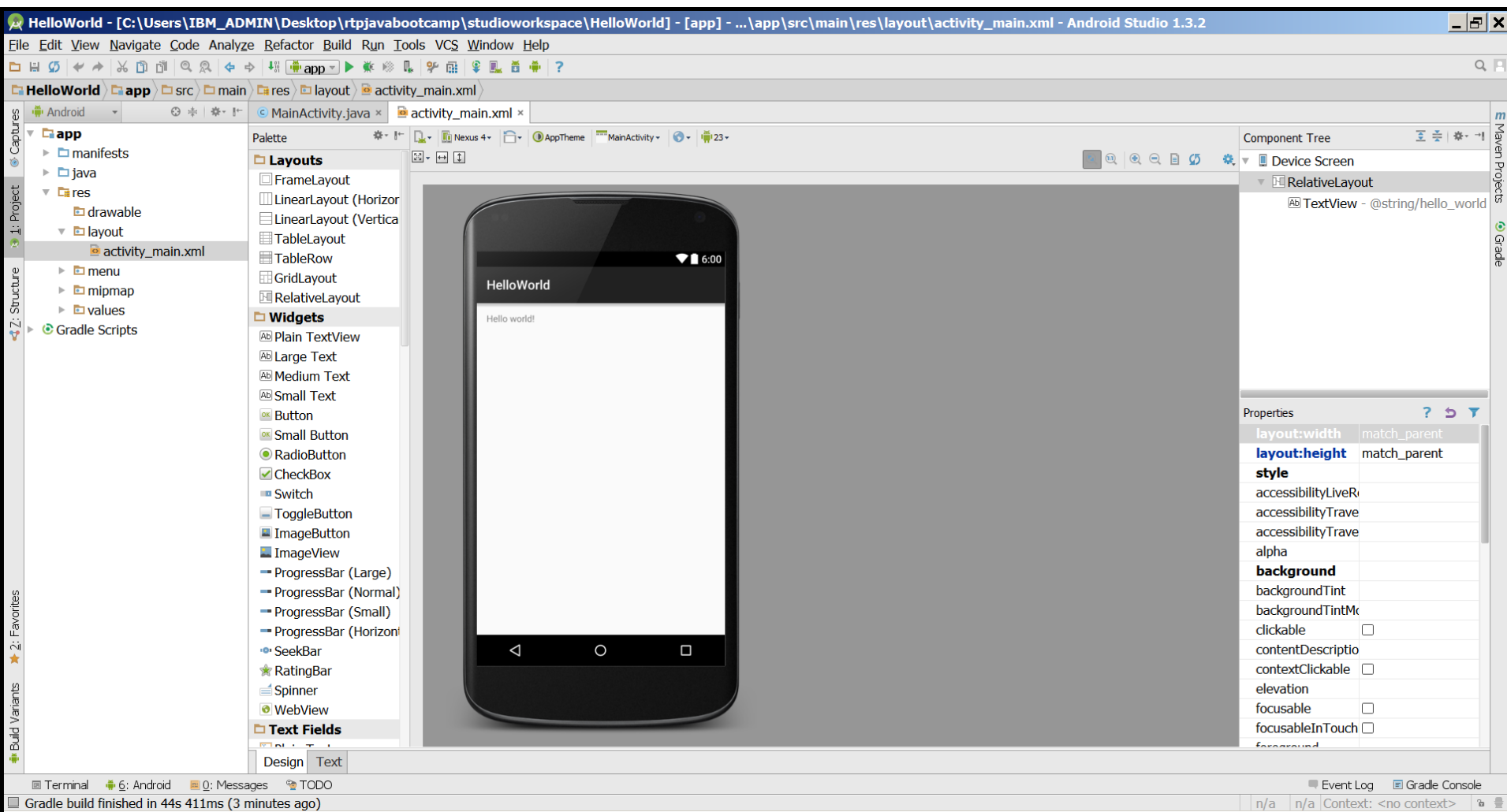
Title:

Menu Resource Name:

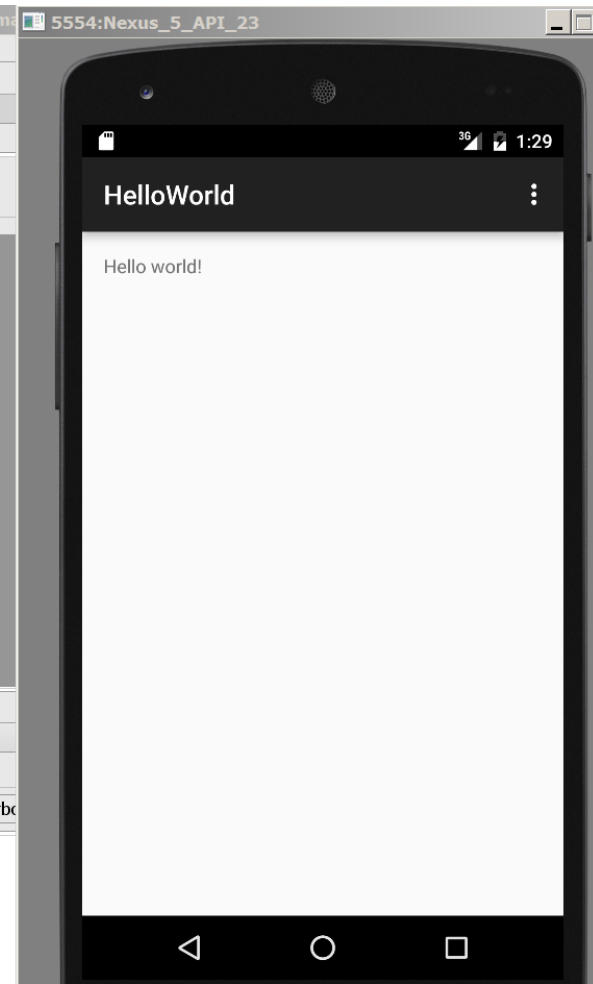
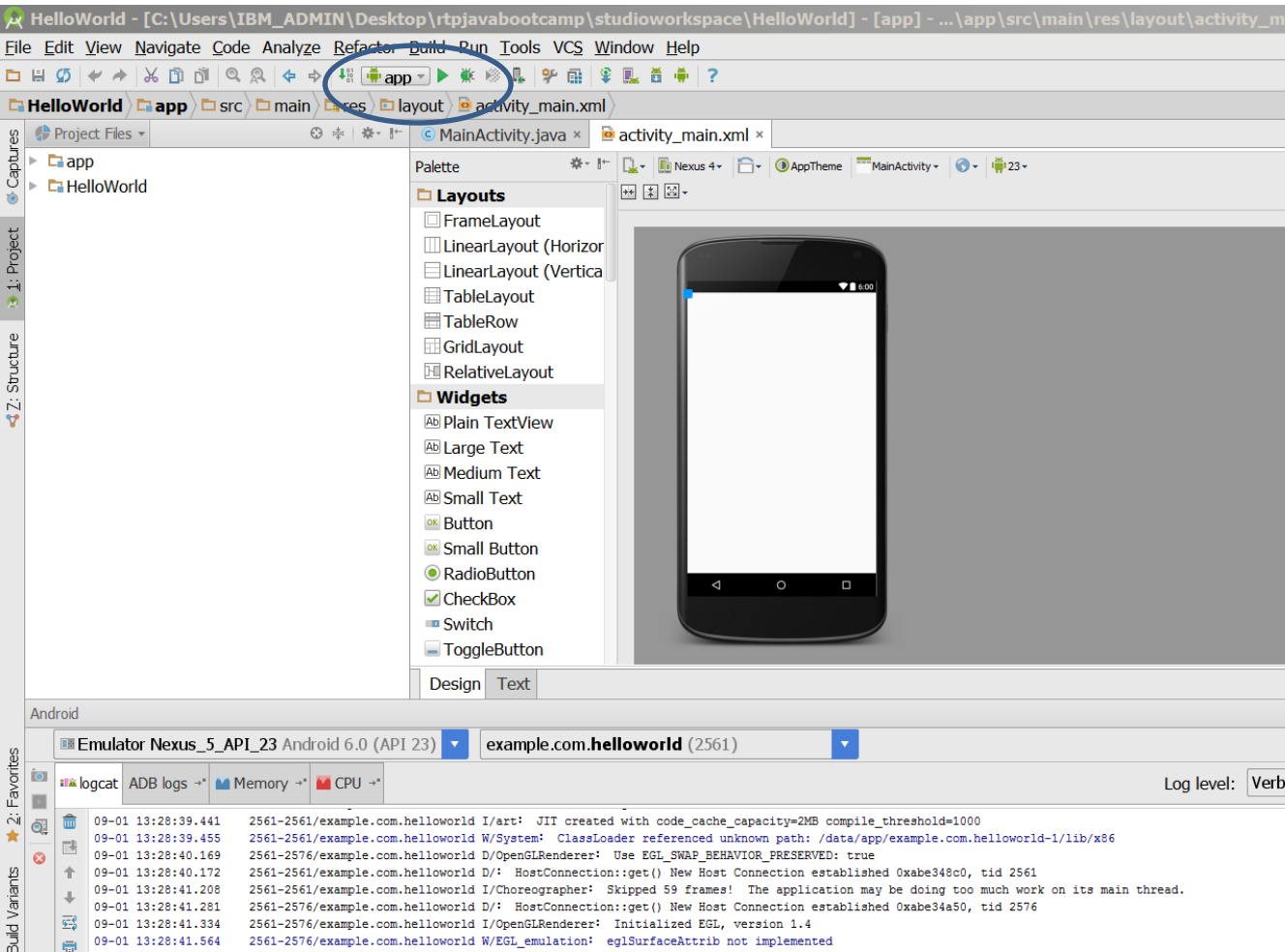
The name of the activity class to create

[Previous](#)[Next](#)[Cancel](#)[Finish](#)

# Ready to Launch !



# Hurray !



# Important Files

1. **src/HelloWorld.java**
  1. Activity which is started when app executes
2. **gen/R.java** (DO NOT MODIFY!)
  1. Auto-generated, auto-updated file with identifiers from main.xml, strings.xml, and elsewhere
3. **res/layout/main.xml**
  1. Defines & lays out widgets for the activity
4. **res/values/strings.xml**
  1. String constants used by app
5. **AndroidManifest.xml**
  1. Declares all the app's components
  2. Names libraries app needs to be linked against
  3. Identifies permissions the app expects to be granted

# 1 - src/HelloWorld.java

- Activity which is started when app executes

## 2 - gen/R.java

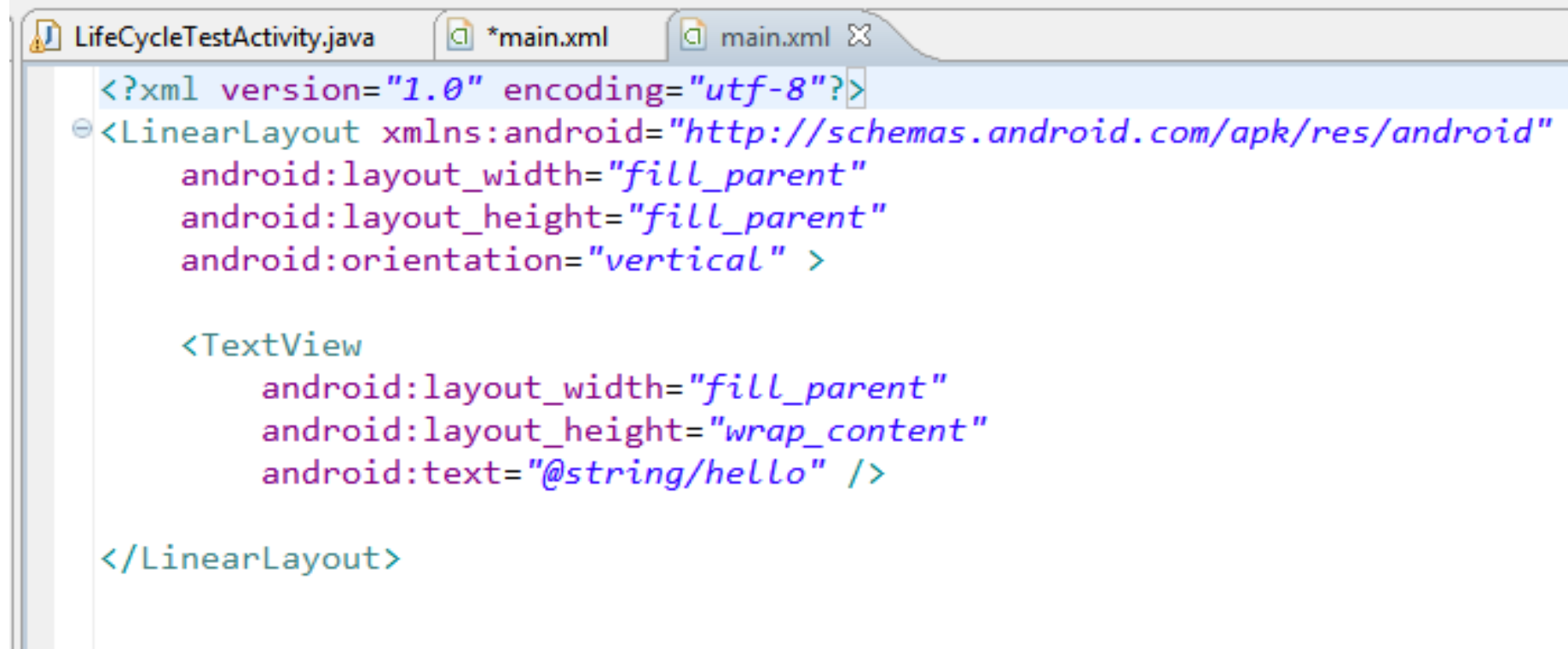
- Auto-generated file with identifiers from main.xml,

```
public final class R {  
    public static final class attr {  
    }  
    public static final class drawable {  
        public static final int icon=0x7f020000;  
    }  
    public static final class id {  
        public static final int hello_button=0x7f050001;  
        public static final int my_button=0x7f050003;  
        public static final int my_check_box=0x7f050002;  
        public static final int name=0x7f050000;  
    }  
    public static final class layout {  
        public static final int main=0x7f030000;  
        public static final int second=0x7f030001;  
    }  
    public static final class string {  
        public static final int app_name=0x7f040001;  
        public static final int hello=0x7f040000;  
    }  
}
```

Do not  
modify!

# 3 - res/layout/main.xml

- layout of main activity
- xml view



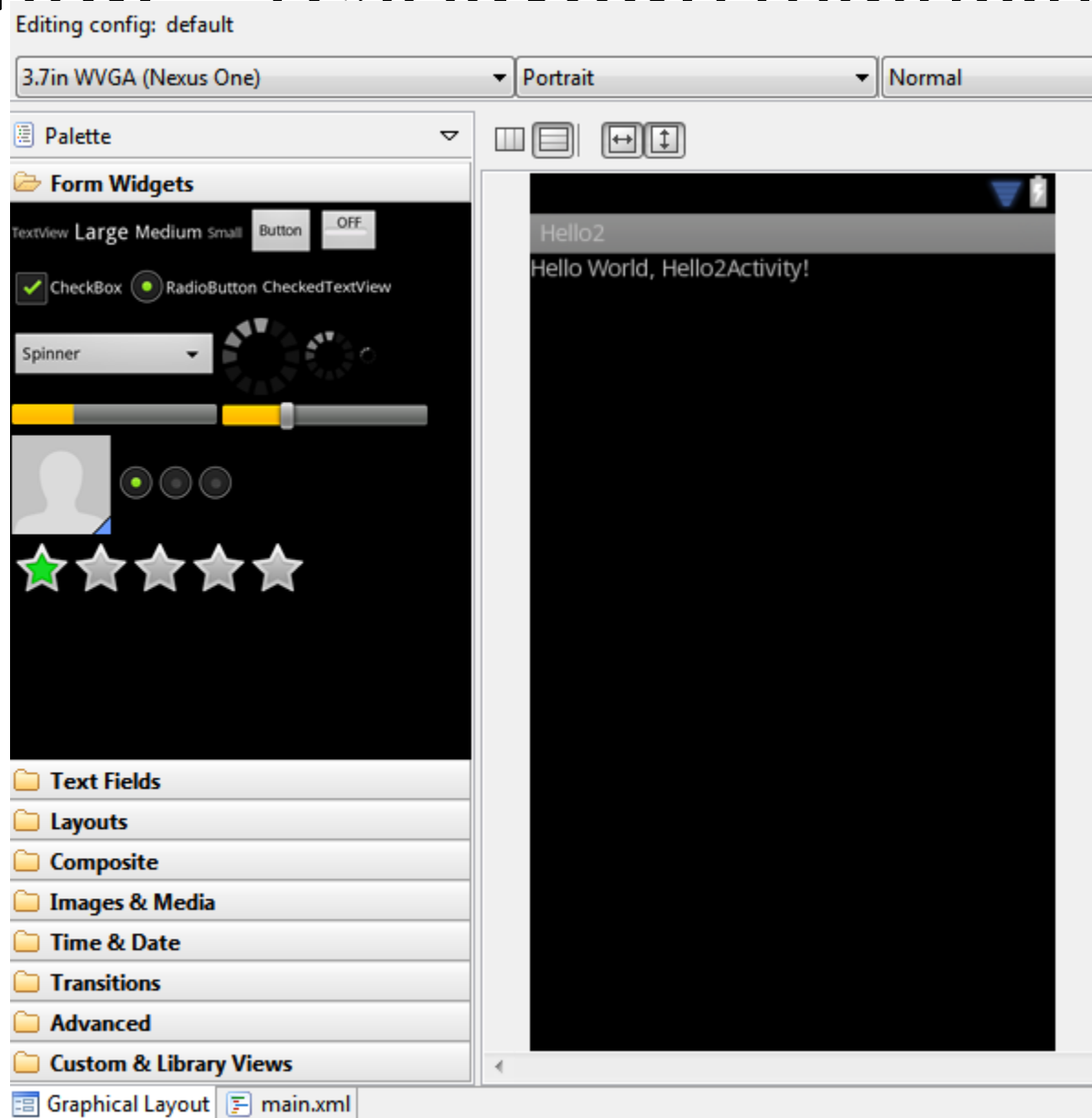
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello" />

</LinearLayout>
```



# 3 contd. - res/layout/main.xml



# 3 contd. - res/layout/main.xml

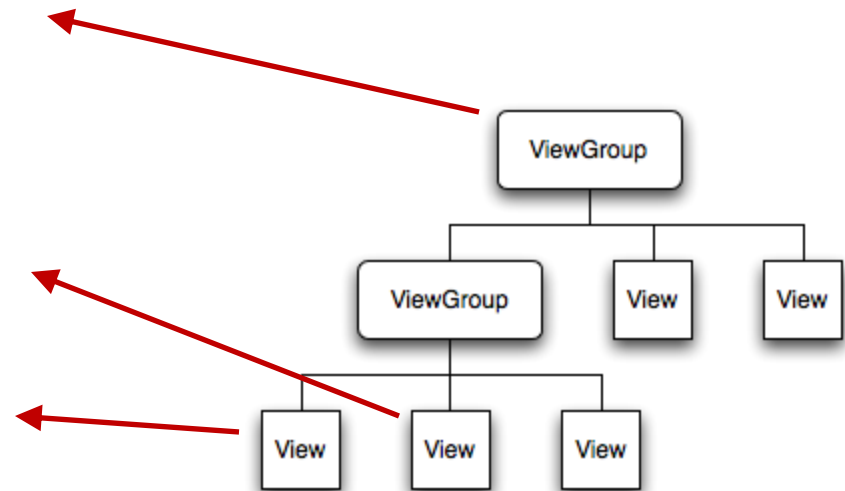
- Declares layouts & widgets for the activity

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <EditText
        android:id="@+id/name"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello" />

    <Button
        android:id="@+id/hello_button"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:text="Press Me" />

</LinearLayout>
```

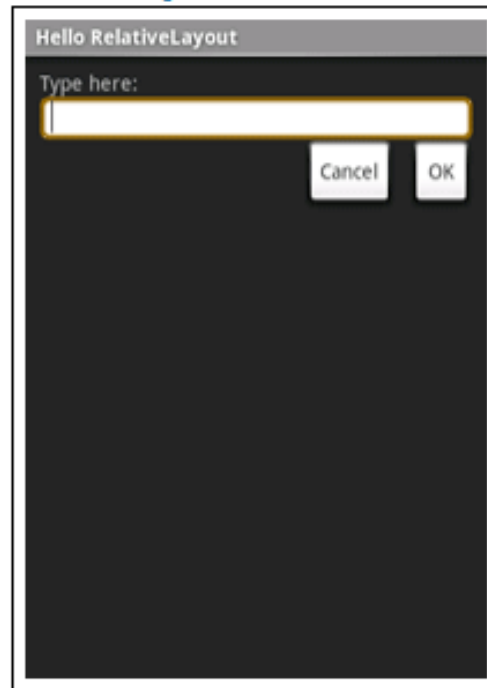


# 3 Contd - Available Layouts

LinearLayout



RelativeLayout

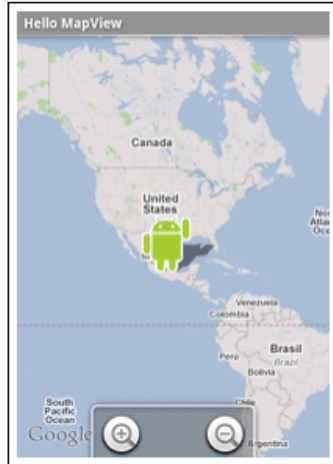


TableLayout



# 3 contd - Available Widgets

MapView



WebView



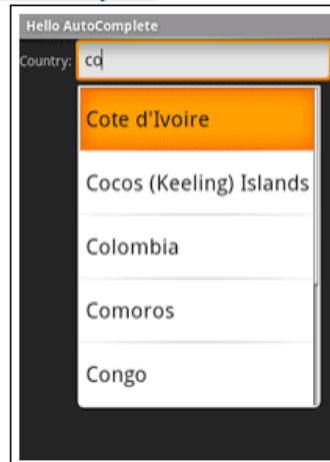
DatePicker



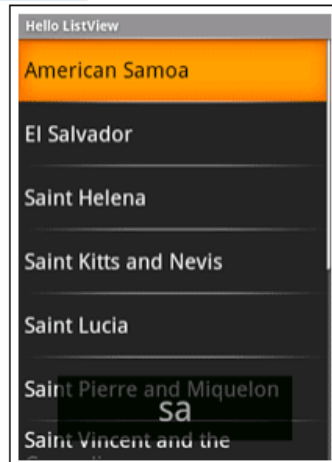
Spinner



AutoComplete



ListView



# 4 - res/values/strings.xml

- String constants used by app

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Android</string>
    <string name="app_name">Hello Android</string>
</resources>
```

- Used for supporting Localization
  - res/values-es/values/strings.xml to support Spanish
  - res/values-fr/values/strings.xml to support French
  - Etc.

# 5 - AndroidManifest.xml

- Declares all the app's components
- Names libraries app needs to be linked against

```
<?xml version="1.0" encoding="utf-8"?>
```

- ```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="scott.examples.hello2"
    android:versionCode="1"
    android:versionName="1.0" >
```

```
<uses-sdk android:minSdkVersion="10" />
```

← min sdk version

```
<application
```

```
    android:icon="@drawable/ic_launcher"
```

```
    android:label="@string/app_name" >
```

```
    <activity
```

```
        android:name=".Hello2Activity"
```

```
        android:label="@string/app_name" >
```

```
        <intent-filter>
```

```
            <action android:name="android.intent.action.MAIN" />
```

```
            <category android:name="android.intent.category.LAUNCHER" />
```

```
        </intent-filter>
```

```
    </activity>
```

```
</application>
```

```
</manifest>
```

# 5 contd - AndroidManifest.xml

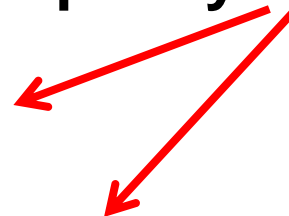
All Activities that are part of application must be registered in M

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="scott.examples.lifeCycleTest"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="10" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:name=".LifeCycleTestActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".NameGetter"
            android:label="@string/getName"/>
    </application>
```

Specify Activity to start w



# Lab 2

Add some Code to onCreate method



# Application Components List

five primary components - different purposes

## 1. Activity

- single screen with a user interface, app may have several activities, subclass of Activity
- Most of early examples will be activities

## 2. Intents

- used to pass information between applications

## 3. Service

- Application component that performs long-running operations in background with no UI
- example, an application that automatically responds to texts when driving

# Application Components List..contd

## 4. Content Providers

- a bridge between applications to share data
- for example the devices contacts information
- we tend to use these, but not create new ones

## 5. Broadcast Receivers

- component that responds to system wide announcements
- battery low, screen off, date changed
- also possible to initiate broadcasts from within an application

# Understanding Activity Stack

# Activity Stack

Most recently  
created is at  
Top

Activity 1

User currently interacting with  
me

Activity 2

Pressing Back or destroying  
A1 will bring me to the top

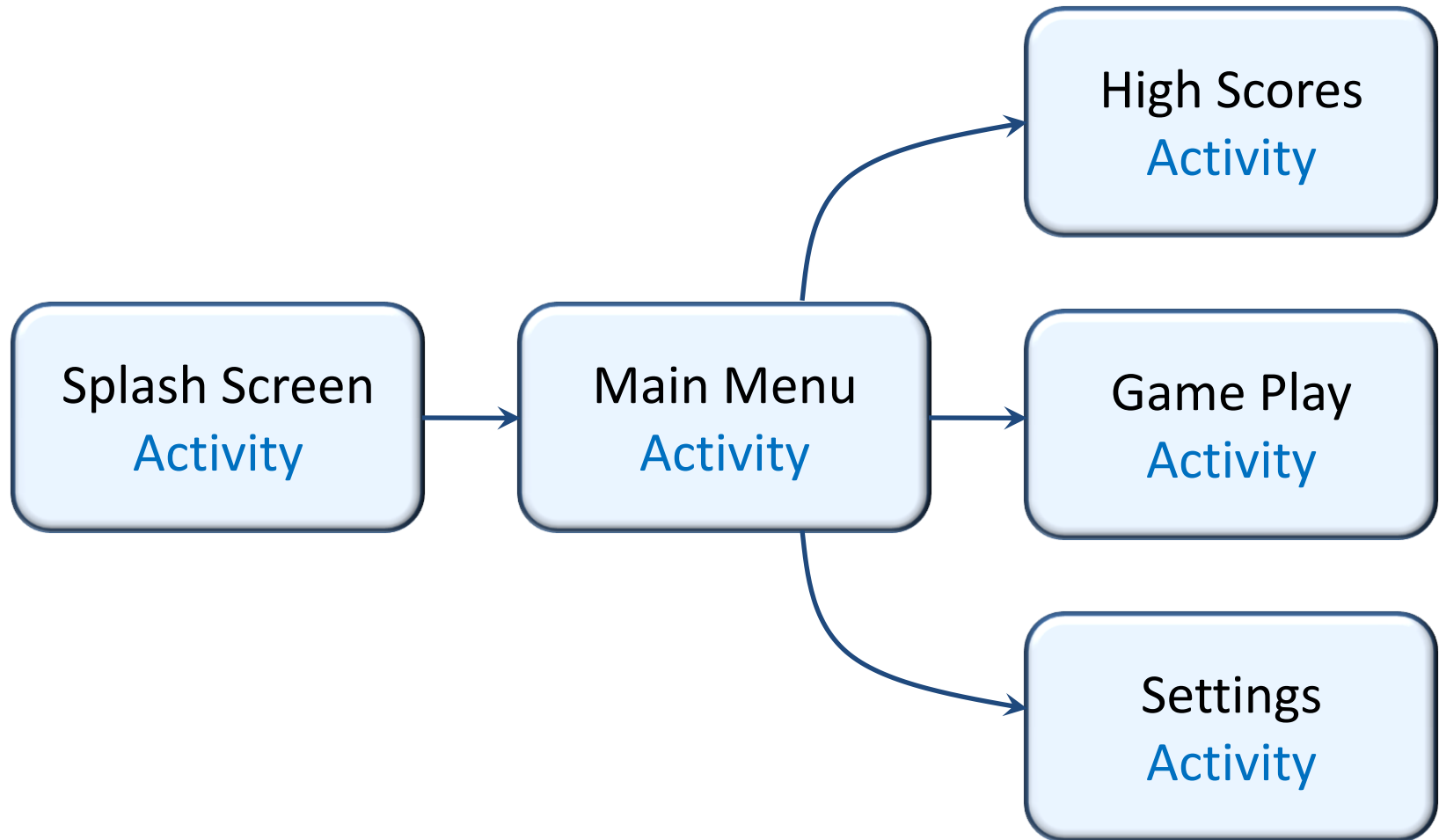
Activity 3

⋮

Activity  $N$

If Activities above me use too  
many resources, I'll be  
destroyed!

# Typical Game



# Understanding the Essence of Lifecycle

Necessary to overload callback methods so you app behaves well:

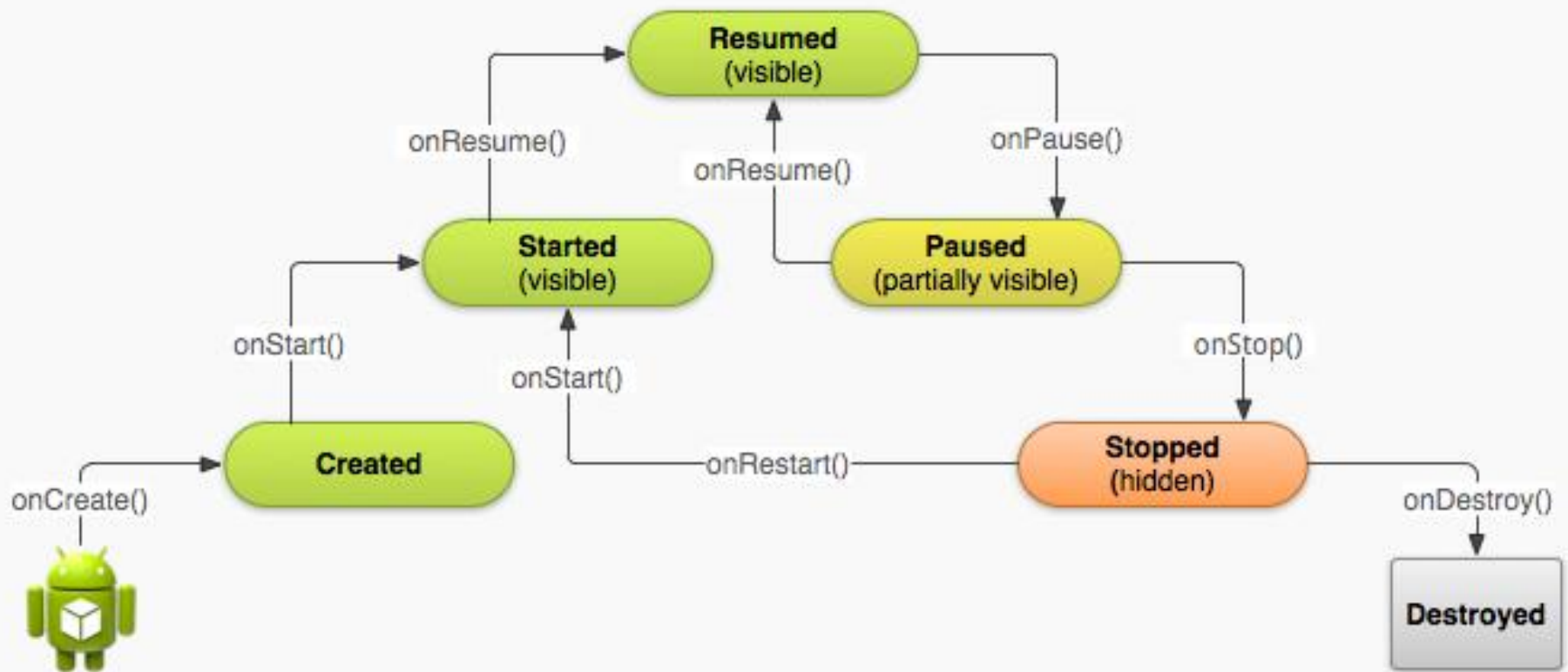
1. App should not crash if the user receives a phone call or switches to another app while using your app.
2. App should not consume valuable system resources when the user is not actively using it.
3. App should not lose the user's progress if they leave your app and return to it at a later time.
4. App should not crash or lose the user's progress when the screen rotates between landscape and portrait orientation.

# Starting Activities

- Android applications don't start with a call to `main(String[])`
- instead a series of callback methods are invoked
- each corresponds to specific stage of the Activity / application lifecycle
- callback methods also used to tear down Activity / application

# Simplified Lifecycle Diagram

ready to interact  
with user





# Primary States

- Active
  - activity is in the foreground and user can interact with it
- Paused
  - activity partially obscured by another activity and user cannot interact with it (for example when working with a menu or dialog)
- Stopped
  - activity completely hidden and not visible to user. It is in the background.
  - Activity instance and variables are retained but no code is being executed by the activity
- Dead, activity terminated (or never started)
- Two other states, Created and Started, but they are transitory onCreate -> onStart -> onResume

# Change MainActivity.java to this

MainActivity.java × activity\_main.xml ×

```
package example.com.helloworld;
```

```
import android.support.v7.app.AppCompatActivity;
```

```
import android.os.Bundle;
```

```
import android.view.Menu;
```

```
import android.view.MenuItem;
```

```
import android.widget.TextView; // LAB 2 NEW
```

```
public class MainActivity extends AppCompatActivity {
```

```
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
```

```
        //setContentView(R.layout.activity_main); // LAB 2 COMMENTED OUT
```

```
        // LAB 2 THREE NEW LINES ADDED
```

```
        TextView textView = new TextView(this);
```

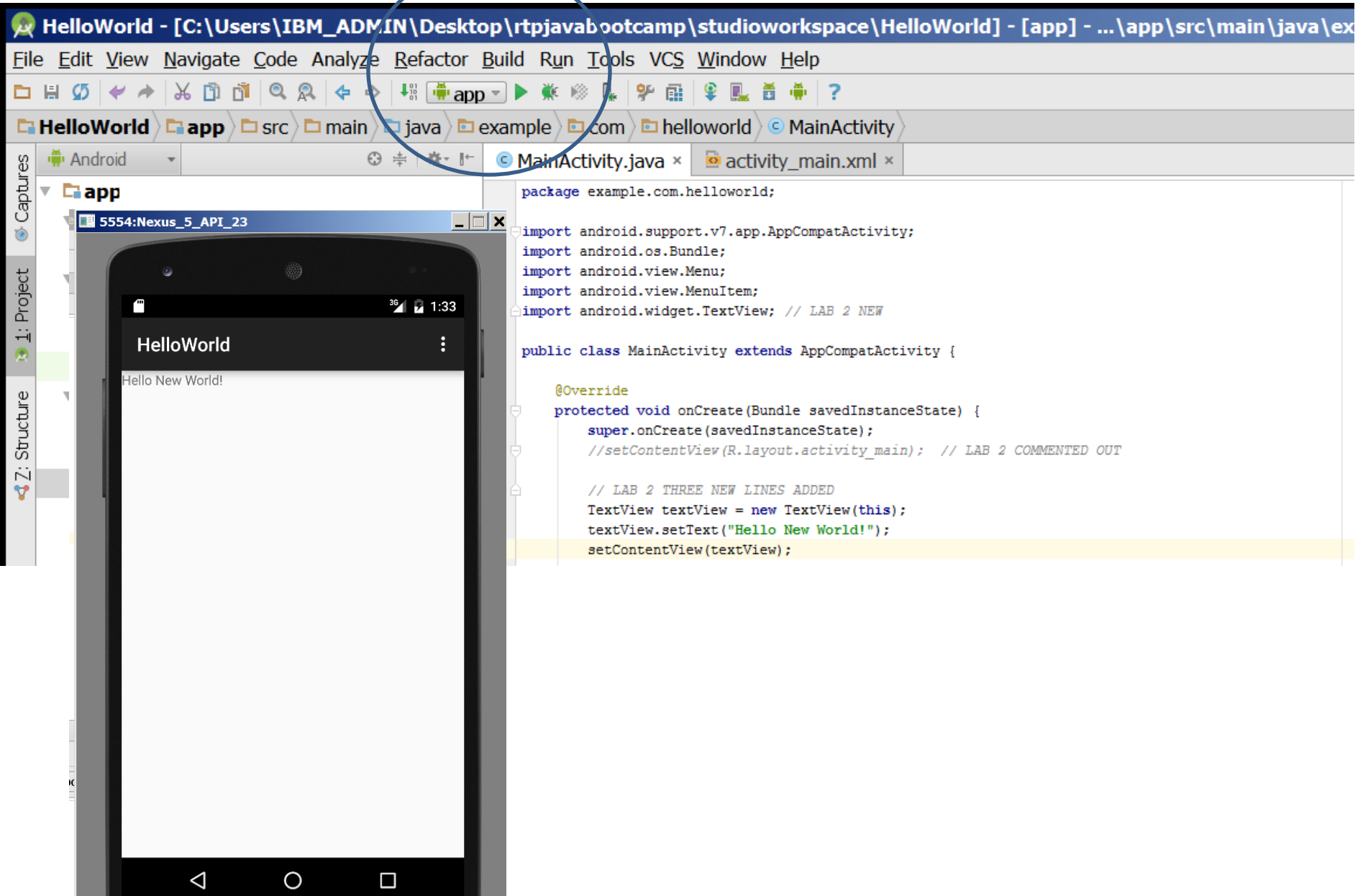
```
        textView.setText("Hello New World!");
```

```
        setContentView(textView);
```

```
    }
```

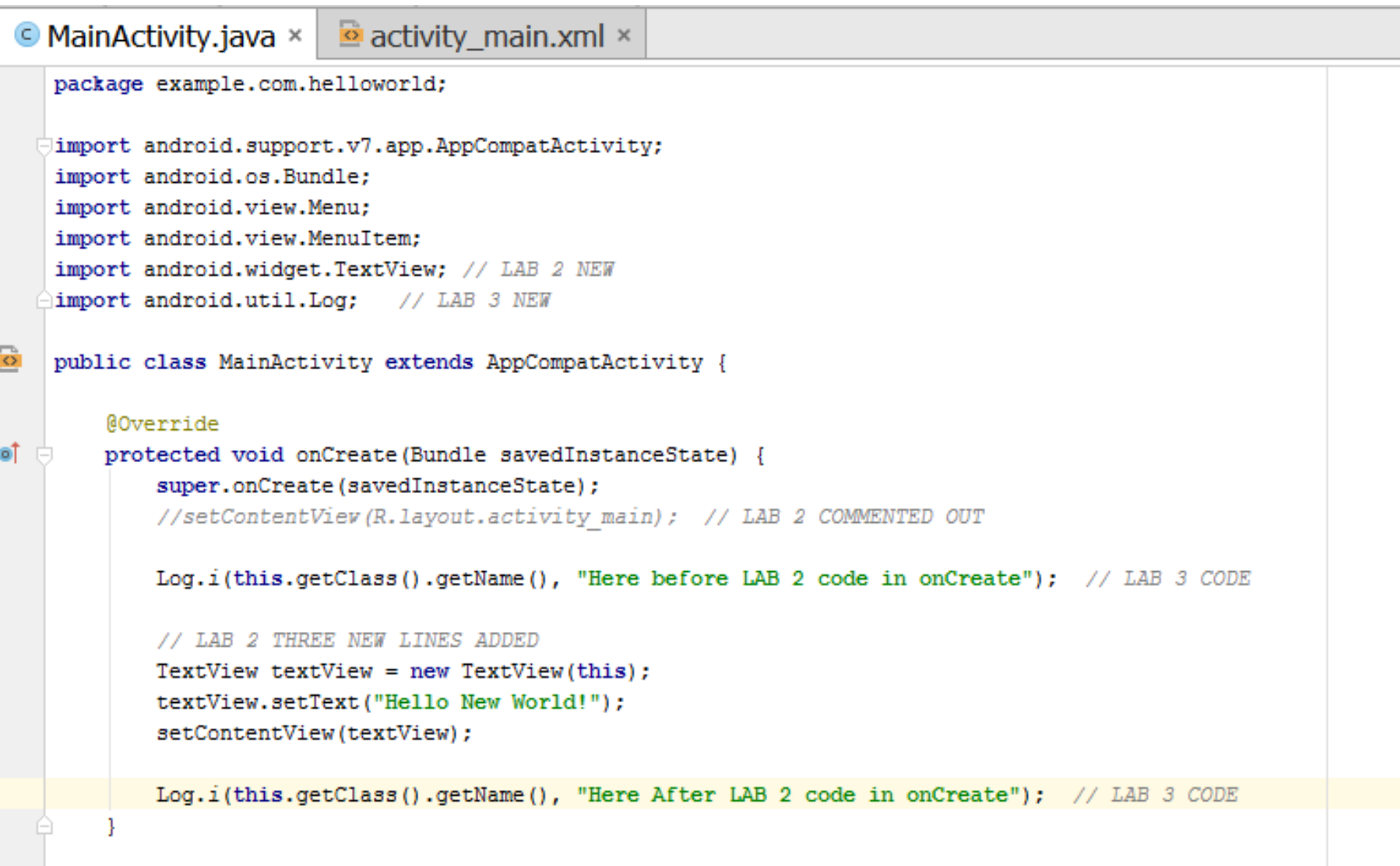
# Lab 2 Changes to Lab 1 Code

- Add  
`import android.widget.TextView;`
- In the onCreate Method
  1. Comment out THIS LINE  
`// setContentView(R.layout.activity_main);`
  2. Add the lines
    - `TextView textView = new TextView(this);`
    - `textView.setText("Hello New World!");`
    - `setContentView(textView);`



# Lab 3

# Change MainActivity.java to this



```
package example.com.helloworld;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView; // LAB 2 NEW
import android.util.Log; // LAB 3 NEW

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //setContentView(R.layout.activity_main); // LAB 2 COMMENTED OUT

        Log.i(this.getClass().getName(), "Here before LAB 2 code in onCreate"); // LAB 3 CODE

        // LAB 2 THREE NEW LINES ADDED
        TextView textView = new TextView(this);
        textView.setText("Hello New World!");
        setContentView(textView);

        Log.i(this.getClass().getName(), "Here After LAB 2 code in onCreate"); // LAB 3 CODE
    }
}
```

# Lab 3 Changes to Lab 2 Code

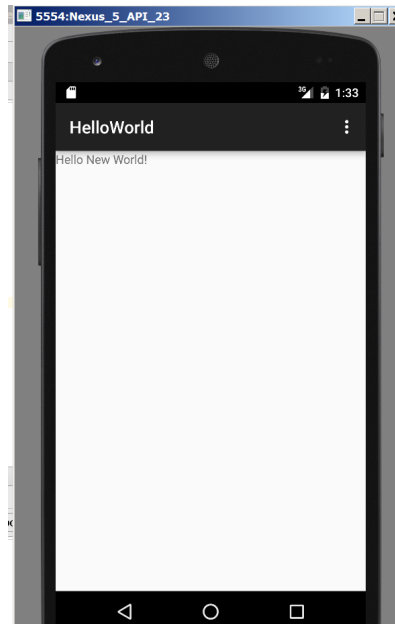
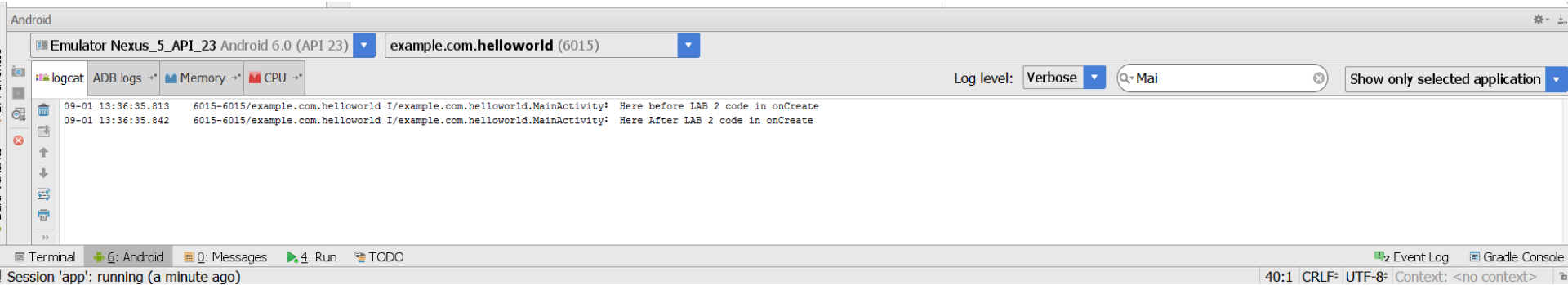
- Add to Import  
import android.util.Log;
- ADD THESE LINES to OnCreate Method

```
Log.i(this.getClass().getName(), "TEXT1");
```

```
TextView textView = new TextView(this); // 3 lines already exists from lab2  
textView.setText("Hello New World!");  
setContentView(textView);
```

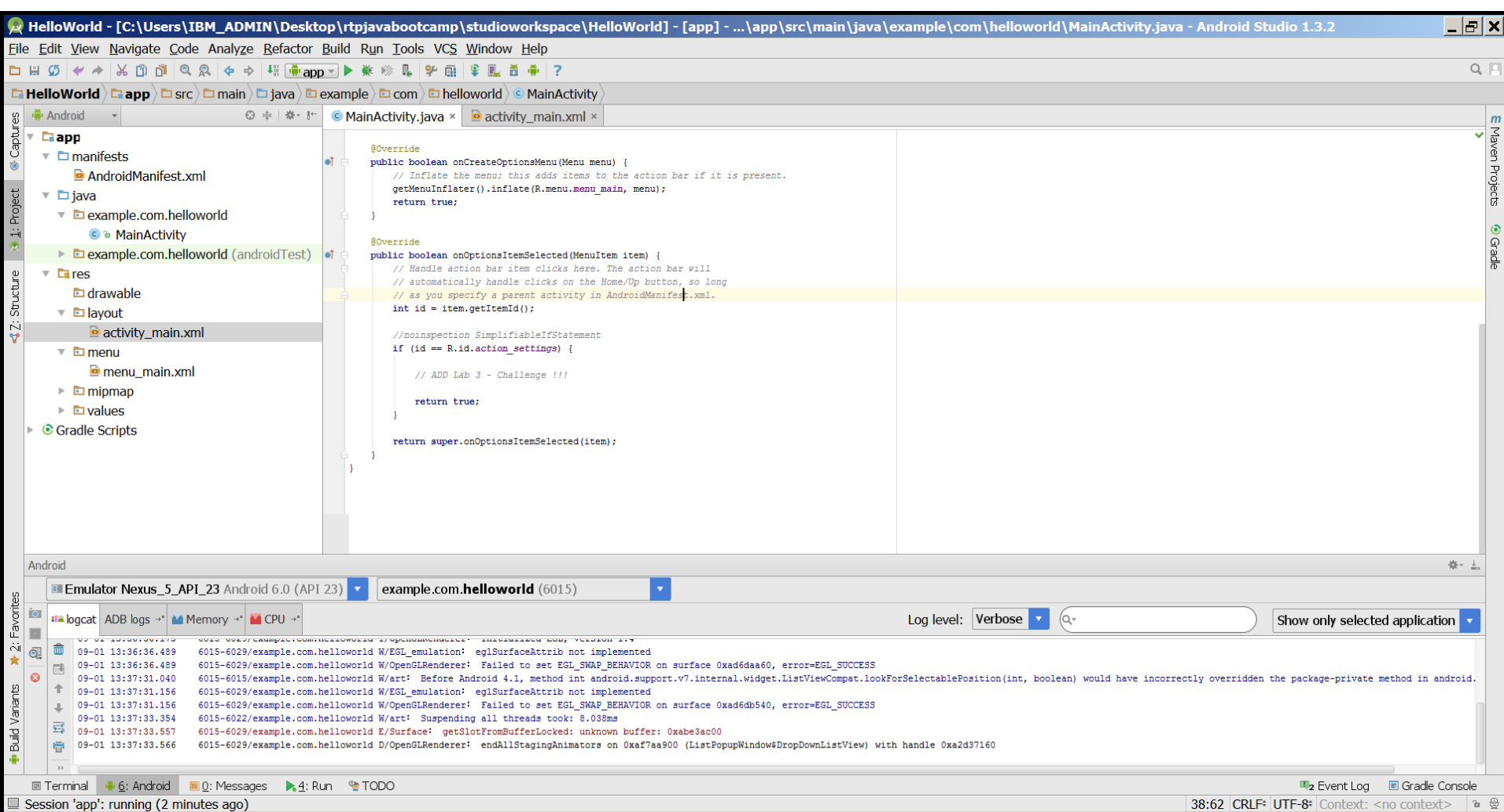
```
Log.i(this.getClass().getName(), "TEXT2");
```

# Run and Check the Logcat Filter





# See if you can Find Method for Options Settings menu via Logging



# Lab 4

Creating ERROR

# Lab 4 – MainActivity.java

The screenshot shows the Android Studio IDE with the following components:

- Top Bar:** File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
- Project Explorer (Left):** Shows the project structure for 'HelloWorld'. The 'MainActivity' file is selected under 'example.com.helloworld'.
- Code Editor (Center):** Displays the code for 'MainActivity.java'. The code includes methods for creating the menu and handling item selections. A yellow highlight is visible on the line `int k = i / j;`.
- Android Studio Bottom Bar:** Shows the emulator status as 'Emulator Nexus\_5\_API\_23 Android 6.0 (API 23)' and the package name 'example.com.helloworld (8041)'. The logcat is visible at the bottom, showing system logs.

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    int id = item.getItemId();

    //noinspection SimplifiableIfStatement
    if (id == R.id.action_settings) {
        // ADD Lab 3 - Challenge !!!

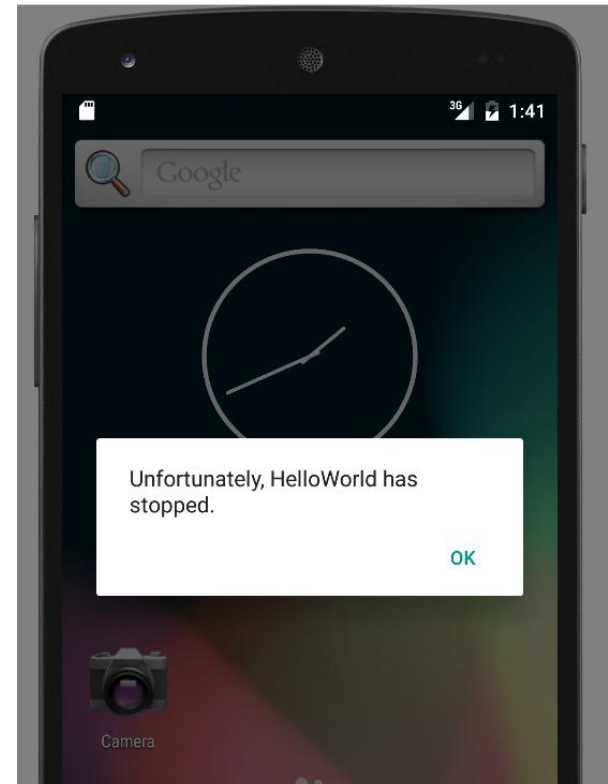
        // LAB 4 CODE
        int i = 0;
        int j = 0;
        int k = i / j;

        return true;
    }

    return super.onOptionsItemSelected(item);
}
```



# Error



```
09-01 13:41:39.073 8041-8041/example.com.helloworld E/InputEventReceiver: Exception dispatching input event.
09-01 13:41:39.073 8041-8041/example.com.helloworld E/MessageQueue-JNI: Exception in MessageQueue callback: handleReceiveCallback
09-01 13:41:39.074 8041-8041/example.com.helloworld E/MessageQueue-JNI: java.lang.ArithmeticException: divide by zero
    at example.com.helloworld.MainActivity.onOptionsItemSelected(MainActivity.java:50)
    at android.app.Activity.onMenuItemSelected(Activity.java:2908)
    at android.support.v4.app.FragmentActivity.onMenuItemSelected(FragmentActivity.java:325)
    at android.support.v7.app.AppCompatActivity.onMenuItemSelected(AppCompatActivity.java:147)
```

# Lab 5

Toasting

# Change onOptionsItemSelected Method

MainActivity.java ×

activity\_main.xml ×

```
// Handle action bar item clicks here. The action bar will
// automatically handle clicks on the Home/Up button, so long
// as you specify a parent activity in AndroidManifest.xml.
int id = item.getItemId();

//noinspection SimplifiableIfStatement
if (id == R.id.action_settings) {

    // ADD Lab 3 - Challenge !!!

    Toast toast = Toast.makeText(this, "CLICKED SETTINGS !!!", Toast.LENGTH_LONG);
    toast.show();

    // LAB 4 CODE - COMMENTED in LAB 5
    //int i = 0;
    //int j = 0;
    //int k = i / j;

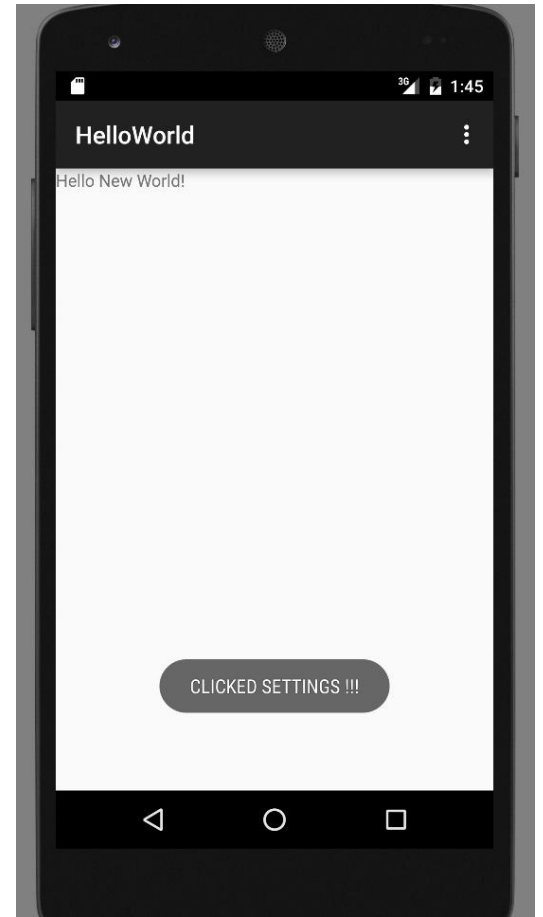
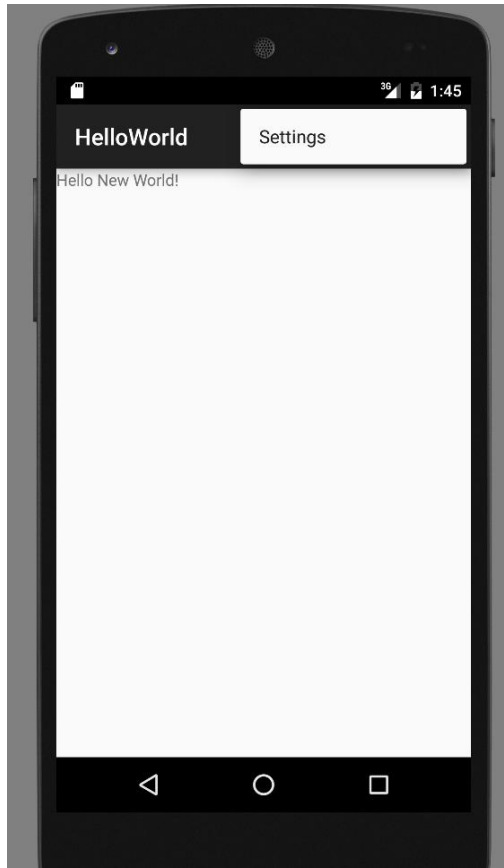
    return true;
}

return super.onOptionsItemSelected(item);
```

# Lab 5 Changes to Lab 4 Code

- Add to Import  
`import android.widget.Toast;`
- REMOVE – int Lines
- ADD THESE LINES

[illegible]





# Lab 6

# Adding DEBUG Breakpoint

The screenshot displays the Android Studio IDE interface. The top toolbar shows the 'Run' button (a green play icon) with a tooltip that reads 'Session \'app\' running'. The 'HelloWorld' project is open, and the 'MainActivity.java' file is selected in the editor. A red circular breakpoint is set on the line `toast.show();`. The left sidebar shows the project structure, with 'example.com.helloworld (androidTest)' selected. The bottom panel shows the 'Debugger' tab, which is currently empty. The 'Logcat' tab shows a list of log messages, including 'onOptionsItemSelected():48, MainActivity (example.com.helloworld)emImpl@4253} "Settings"'. The right side of the image shows a virtual Android device (Nexus 5) running the application. The device screen displays the 'HelloWorld' app with a 'Settings' button and the text 'Hello New World!'. The status bar at the top of the device shows '5554:Nexus\_5\_API\_23'.

```
// Uncomment this line to show the toast message as long
// as you specify a parent activity in AndroidManifest.xml.
int id = item.getItemId(); id: 2131492954 item: "Settings"

//noinspection SimplifiableIfStatement
if (id == R.id.action_settings) { id: 2131492954

    // ADD Lab 3 - Challenge !!!

    Toast toast = Toast.makeText(this, "CLICKED SETTINGS !!!", Toast.LENGTH_LONG); toast: Toast@4256
    toast.show();

    // LAB 4 CODE - COMMENTED in LAB 5
    //int i = 0;
    //int j = 0;
    //int k = i / j;

    return true;
}

return super.onOptionsItemSelected(item);
}
```

Debugger

onOptionsItemSelected():48, MainActivity (example.com.helloworld)emImpl@4253} "Settings"

onOptionsItemSelected():2908, Activity (android.os.Bundle)@4253} "Settings"

onOptionsItemSelected():325, FragmentActivity (android.os.Bundle)@4253} "Settings"

onOptionsItemSelected():147, AppCompatActivity (android.os.Bundle)@4253} "Settings"

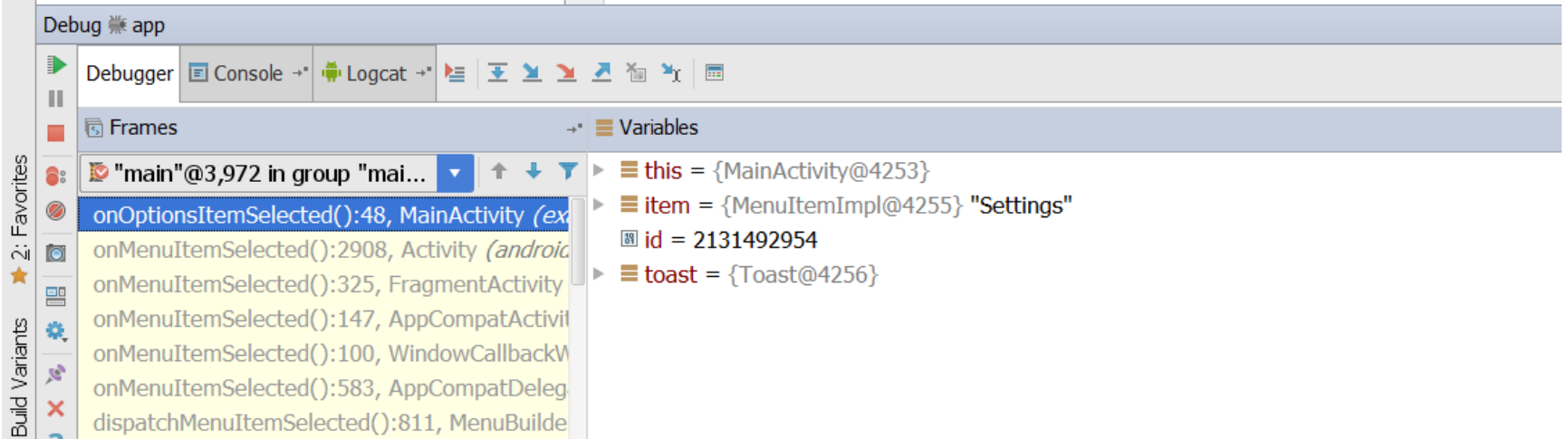
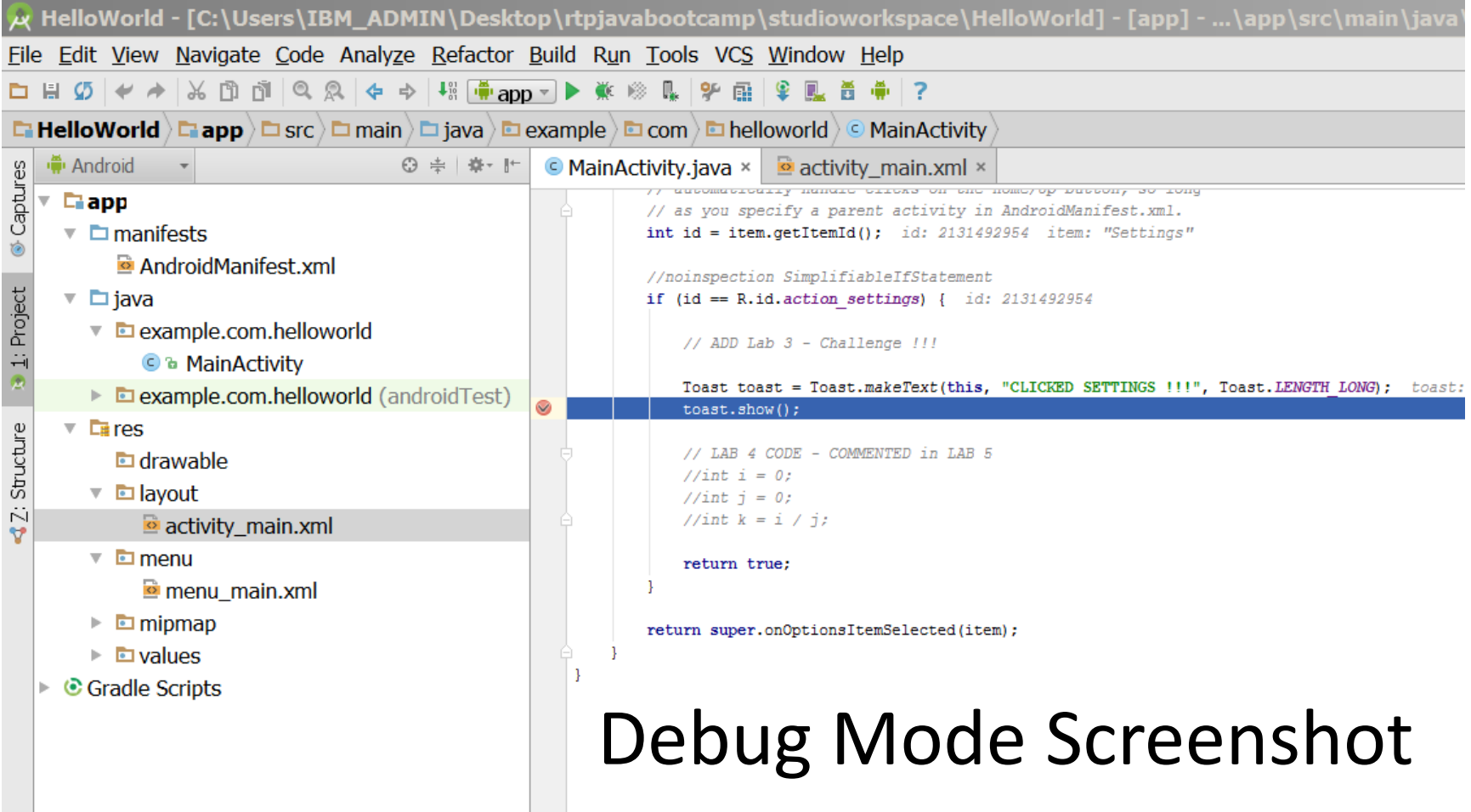
onOptionsItemSelected():100, WindowCallbackWrapper (android.os.Bundle)@4253} "Settings"

onOptionsItemSelected():583, AppCompatActivity (android.os.Bundle)@4253} "Settings"

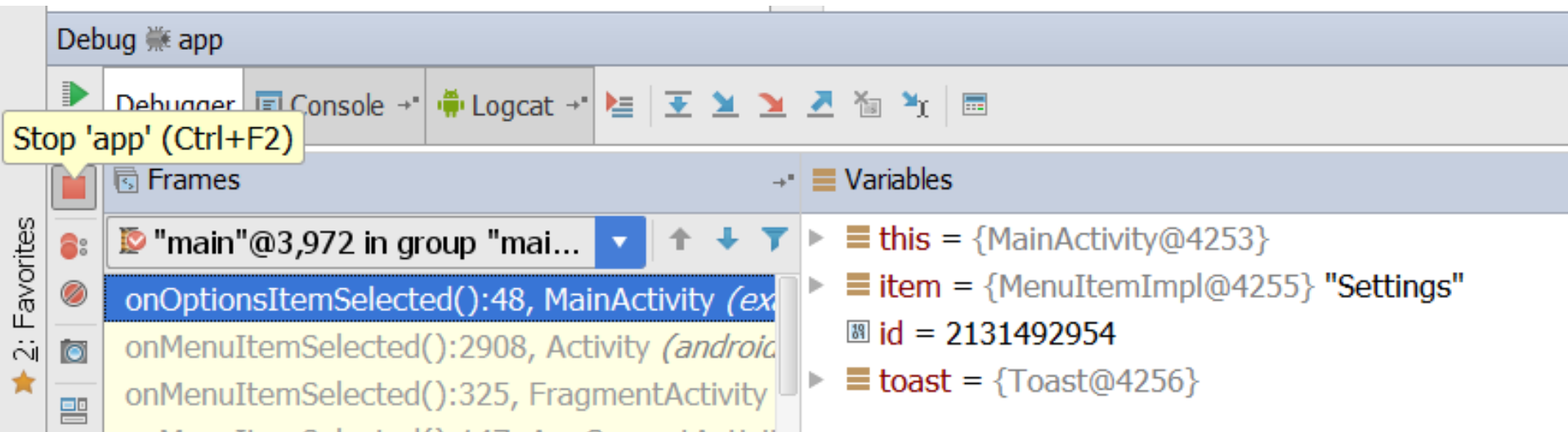
dispatchOptionsItemSelected():811, MenuBuilder (android.os.Bundle)@4253} "Settings"

invoke():153, MenuItemImpl (android.os.Bundle)@4253} "Settings"

Session 'app': running



# Stopping the Debugger



# Dalvik Debug Monitor Server

- DDMS
- debugging tool
- "provides, screen capture on the device, thread and heap information on the device, logcat, process, and radio state information, incoming call and SMS spoofing, location data spoofing, and more."
- can interact with DDMS via Eclipse plugin, another view in Eclipse

# DDMS

DDMS - AndroidTicTacToe-Tutorial2/res/layout/main.xml - Eclipse SDK

File Edit Refactor Run Navigate Search Project Window Help

Devices Threads Heap Allocation Tracker File Explorer

Name

com.android.settings	147	8605
android.process.acore	169	8606
com.android.deskclock	184	8607
com.android.protips	203	8608
com.android.music	214	8609
com.android.quicksearchbo	230	8612
com.android.defcontainer	238	8614
android.process.media	246	8616
com.android.mms	261	8618
com.android.email	282	8620
com.svox.pico	302	8622
scottmd3.tictactoe	322	8624 / 8700
scott.examples.lifeCycleTest	333	8625

Heap updates will happen after every GC for th

ID	Heap Size	Allocated	Free	% Used	# Objects
1	5,254 MB	2,551 MB	2,703 MB	48.56%	48,634

Cause GC

Display: Stats

Type	Count	Total Size	Smallest	Largest	Median	Average
free	5,338	2,691 MB	16 B	78.516 KB	176 B	528 B
data object	33,061	996.391 KB	16 B	672 B	32 B	30 B
class object	2,042	586.086 KB	168 B	26.836 KB	168 B	293 B
1-byte array (byte[], boolean[])	1,563	228.414 KB	24 B	1.977 KB	40 B	149 B
2-byte array (short[], char[])	8,957	564.203 KB	24 B	28.023 KB	48 B	64 B
4-byte array (object[], int[], float[])	2,789	227.836 KB	24 B	16.023 KB	40 B	83 B
8-byte array (long[], double[])	222	9.352 KB	32 B	1.000 KB	32 B	43 B

Allocation count per size

Emulator Control

Telephony Status

Voice: home Speed: Full

Data: home Latency: None

Telephony Actions

Incoming number:

Voice

SMS

Message:

LogCat

Saved Filters

All messages (no filters)

life cycle 2

Search for messages. Accepts Java regexes. Prefix with pid, app, tag; or text: to limit scope.

verbose

L...	Time	PID	Application	Tag	Text
W	01-29 19:31:23.124	61	sys...	InputManagerService	Window already focused, ignoring focus gain of: c...
D	01-29 19:31:24.194	322	scott...	TicTacToeGame	Computer moving to 5 as a random move.
D	01-29 19:31:27.865	322	scott...	TicTacToeGame	Computer moving to 6 to block win.
D	01-29 19:31:29.875	322	scott...	TicTacToeGame	Computer moving to 1 to block win.
D	01-29 19:32:26.682	322	scott...	dalvikvm	+++ active profiler count now 1
I	01-29 19:32:26.682	322	scott...	dalvikvm	TRACE STARTED: '[DDMS]' 8192KB
I	01-29 19:32:32.656	322	scott...	dalvikvm	dvmDdmHandleHpsgChunk(when 1, what 0, heap 0)

# Some Elements covered

- onCreate() in Activity
- Adding Text and Changing Properties from Code
- Use the log to v, d, i, w, e  
VERBOSE, DEBUG, INFO, WARN, ERROR
- Create a TAG so we can filter
- Toast a message
- Debug BreakPoint
- ScreenShot of the APP

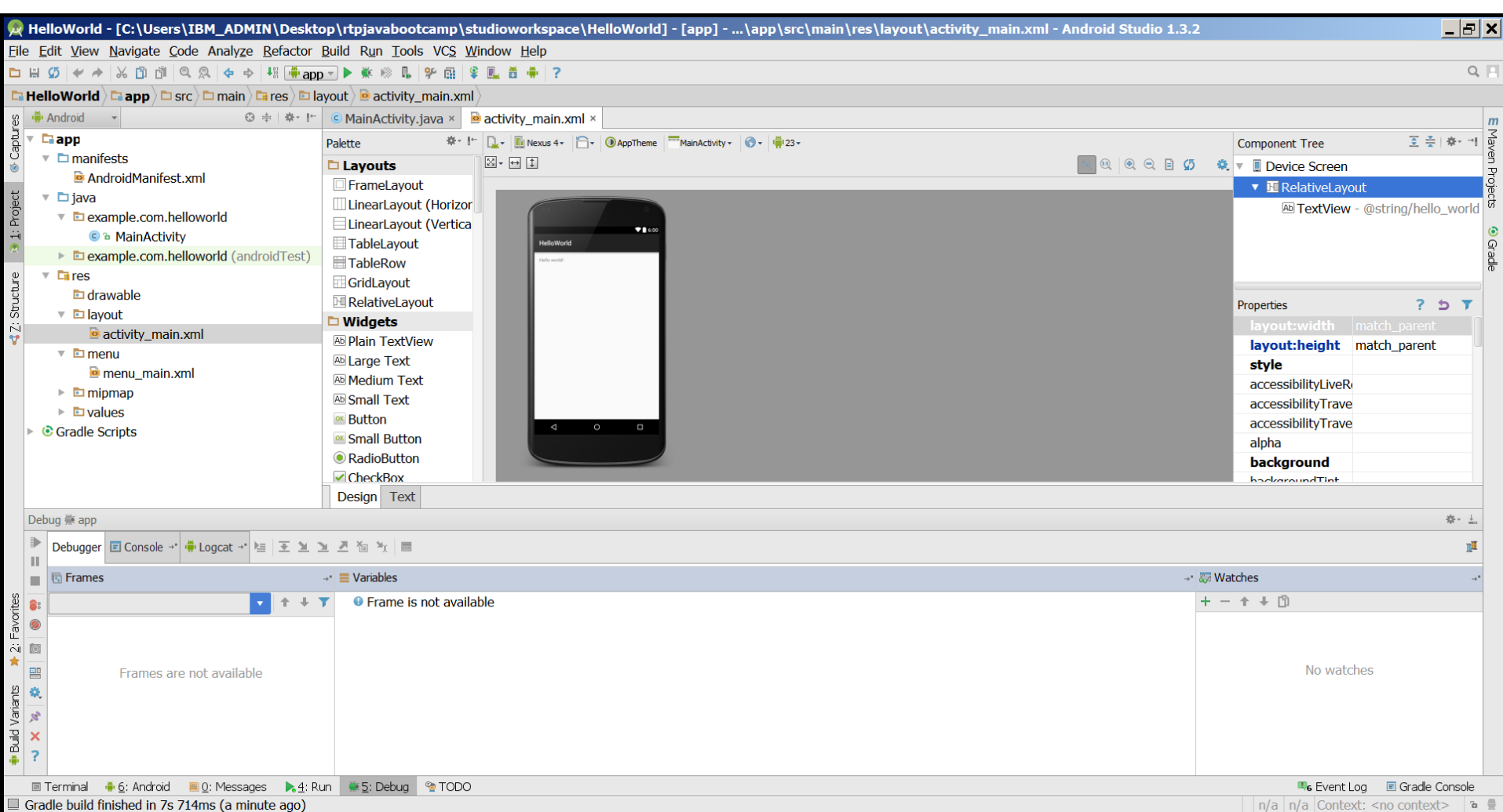
# Lab 7

Adding Button and OnClickEvent



# On Activity\_main.xml - 1

## add a button and change properties to Submit



# On Activity\_main.xml - 2

app src main res layout activity\_main.xml

Android

app

- manifests
  - AndroidManifest.xml
- java
  - example.com.helloworld
    - MainActivity
    - example.com.helloworld (androidTest)
- res
  - drawable
  - layout
    - activity\_main.xml
  - menu
    - menu\_main.xml
  - mipmap
  - values
- Gradle Scripts

Palette

Layouts

- FrameLayout
- LinearLayout (Horizontal)
- LinearLayout (Vertical)
- TableLayout
- TableRow
- GridLayout
- RelativeLayout

Widgets

- Plain TextView
- Button
- Small Button
- RadioButton
- CheckBox

Design Text

Button

Represents a push-button widget.

MainActivity.java activity\_main.xml

Palette

Layouts

- FrameLayout
- LinearLayout (Horizontal)
- LinearLayout (Vertical)
- TableLayout
- TableRow
- GridLayout
- RelativeLayout

Widgets

- Plain TextView
- Large Text
- Medium Text
- Small Text
- Button
- Small Button
- RadioButton
- CheckBox

Design Text

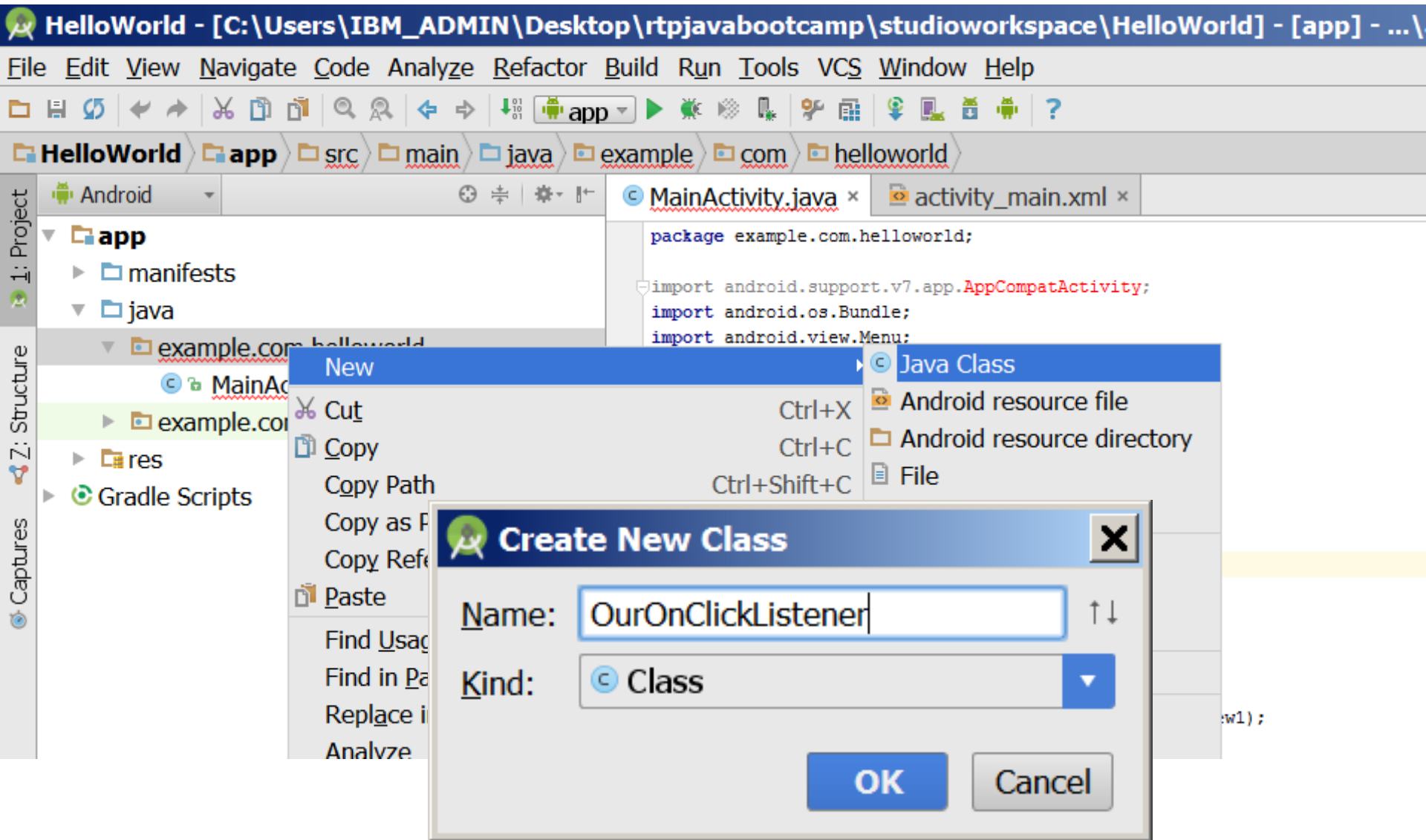
Component Tree

- Device Screen
  - RelativeLayout
    - TextView - @string/hello\_world
    - button - "Submit"

Properties

- stateListAnimator
- text
  - Integer, Enum
- textAlignment
- textAppearance
- textColor
- textColorHighlight
- textColorHint
- textColorLink
- textSize

# Create a new Java Class



# New OnCreate Method

The screenshot shows the Android Studio IDE interface. The top toolbar includes icons for File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, and Help. The breadcrumb navigation at the top indicates the current file path: `example.com.helloworld.MainActivity`. The left sidebar displays the Project, Structure, and Captures panels. The Project panel shows the hierarchy: `app` (manifests, java) > `example.com.helloworld` (MainActivity, OurOnClickListener) > `example.com.helloworld (androidTest)` (res, Gradle Scripts). The main editor area shows the `MainActivity.java` file with the following code:

```
package example.com.helloworld;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView; // LAB 2 NEW
import android.util.Log; // LAB 3 NEW
import android.widget.Toast; // LAB 5 NEW
import android.widget.Button; // LAB 7 NEW

public class MainActivity extends AppCompatActivity {

    // OVER WROTE onCreate Method in LAB 7
    @Override
    protected void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);

        textView = (TextView) findViewById(R.id.textView1);

        button = (Button) findViewById(R.id.button1);
        button.setOnClickListener(new OurOnClickListener(this));
    }
}
```

# New OnCreate Method

```
import android.widget.Button; // LAB 7 NEW

public class MainActivity extends AppCompatActivity {

    // OVER WRITE OnCreate Method in LAB 7
    @Override
    protected void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_main);

        textView = (TextView) findViewById(R.id.textView1);

        button = (Button) findViewById(R.id.button1);
        button.setOnClickListener(new OurOnClickListener(this));
    }
```

# New OurOnClickListener Class

The screenshot shows the Android Studio IDE with the following components:

- Title Bar:** HelloWorld - [C:\Users\IBM\_ADMIN\Desktop\rtpjavabootcamp\studioworkspace\HelloWorld] - [app] - ... \app\src\main\java\example\com\helloworld\OurOnClickListener.java
- Menu Bar:** File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help
- Toolbar:** Standard Android Studio icons for file operations, running, and debugging.
- Breadcrumb:** HelloWorld > app > src > main > java > example > com > helloworld > OurOnClickListener
- Project Structure:**
  - app
    - manifests
    - java
      - example.com.helloworld
        - MainActivity
        - OurOnClickListener** (selected)
      - example.com.helloworld (androidTest)
      - res
      - Gradle Scripts
- Code Editor:**

```
import org.w3c.dom.Document;
import org.xml.sax.SAXException;

import com.alchemyapi.api.AlchemyAPI;
import com.alchemyapi.api.AlchemyAPI_RelationParams;

import android.view.View;
import android.view.View.OnClickListener;

public class OurOnClickListener implements OnClickListener {

    MainActivity caller;
    private int count;

    public OurOnClickListener(MainActivity activity) {
        this.caller = activity;
        this.count = 0;
    }

    @Override
    public void onClick(View v) {
        count++;
        String countstr = Integer.toString(count);
        caller.textView.setText("Clicked " + countstr + " times");
    }
}
```

# NEW CLASS CREATED IN LAB 7

```
package example.com.helloworld;
import java.io.IOException;
import android.view.View;
import android.view.View.OnClickListener;

public class OurOnClickListener implements OnClickListener {

    MainActivity caller;
    private int count;
    public OurOnClickListener(MainActivity activity) {
        this.caller = activity;
        this.count = 0;
    }

    @Override
    public void onClick(View v) {
        count++;
        String countstr = Integer.toString(count);
        caller.textView.setText("Clicked " + countstr + " times");
    }
}
```

# References

- Android Dev Guide

<http://developer.android.com/guide/topics/fundamentals.html>

- <http://developer.android.com/guide/topics/fundamentals/activities.html>