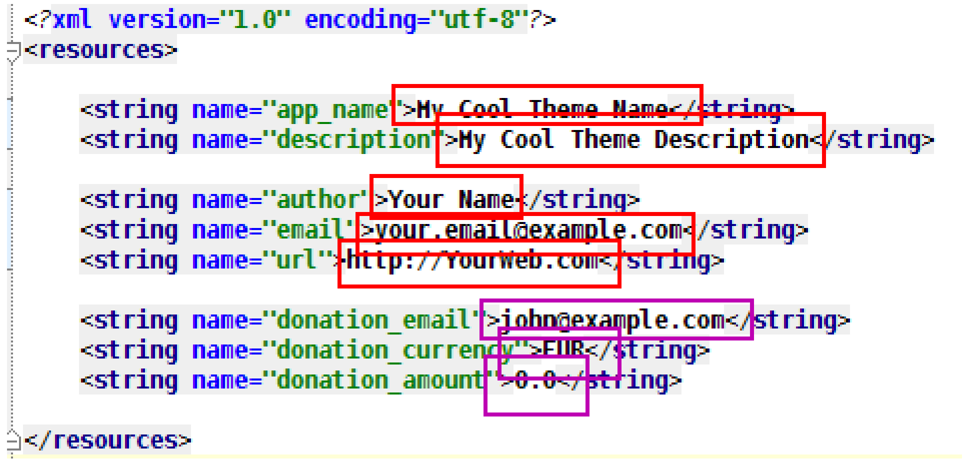
CS 528 Mobile and Ubiquitous Computing
Lecture 02b: Android UI Design

Emmanuel Agu

Resources

Android Resources

- Resources? Images, strings, dimensions, layout files, menus, etc that your app uses
- Basically app elements declared in other files
  - Easier to update, maintain code



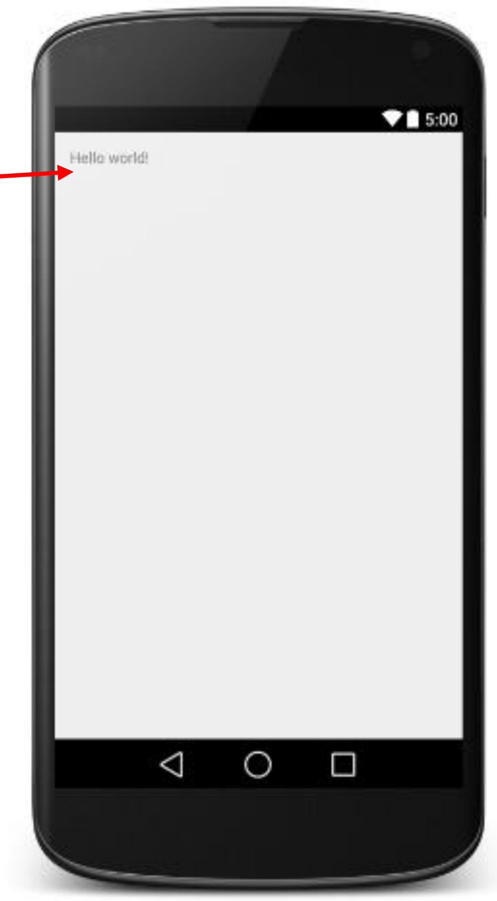
```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">My Cool Theme Name</string>
    <string name="description">My Cool Theme Description</string>

    <string name="author">Your Name</string>
    <string name="email">your.email@example.com</string>
    <string name="url">http://YourWeb.com</string>

    <string name="donation_email">john@example.com</string>
    <string name="donation_currency">EUR</string>
    <string name="donation_amount">0.0</string>

</resources>
```

Declaring Strings in Strings.xml

- Can declare all strings in strings.xml

String declaration in strings.xml

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

  <string name="app_name">EmPubLite</string>
  <string name="hello_world">Hello world!</string>

</resources>
```



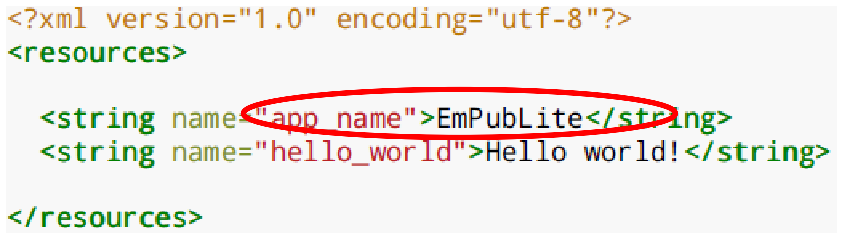
- Then reference in any of app’s xml files

```
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".EmPubLiteActivity">

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

</RelativeLayout>
```

Strings in AndroidManifest.xml

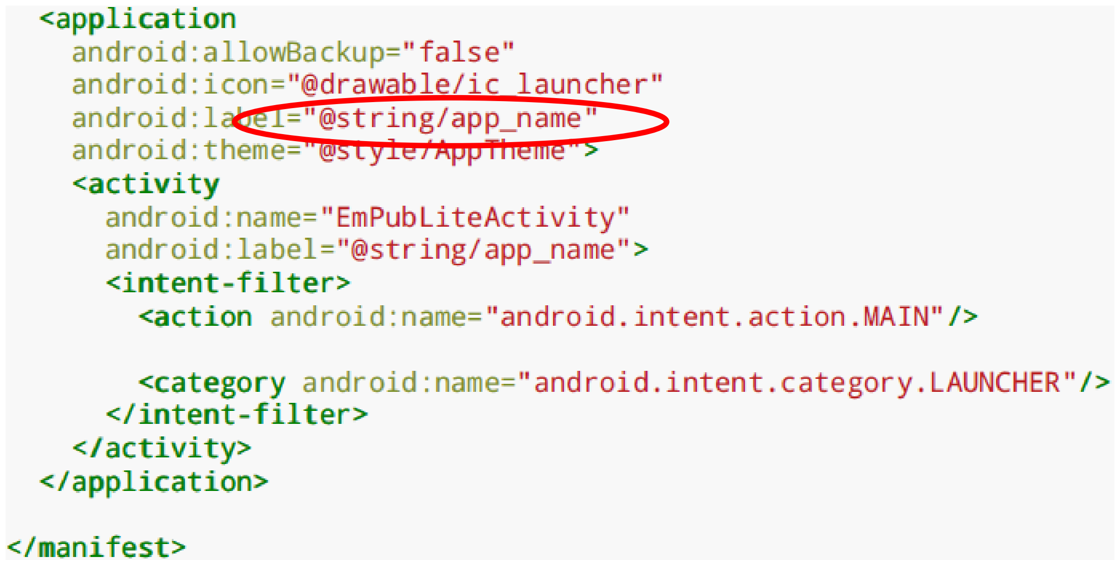
- Strings declared in strings.xml can be referenced by all other XML files (activity_my.xml, AndroidManifest.xml)

String declaration in strings.xml



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

  <string name="app_name">EmPubLite</string>
  <string name="hello_world">Hello world!</string>

</resources>
```

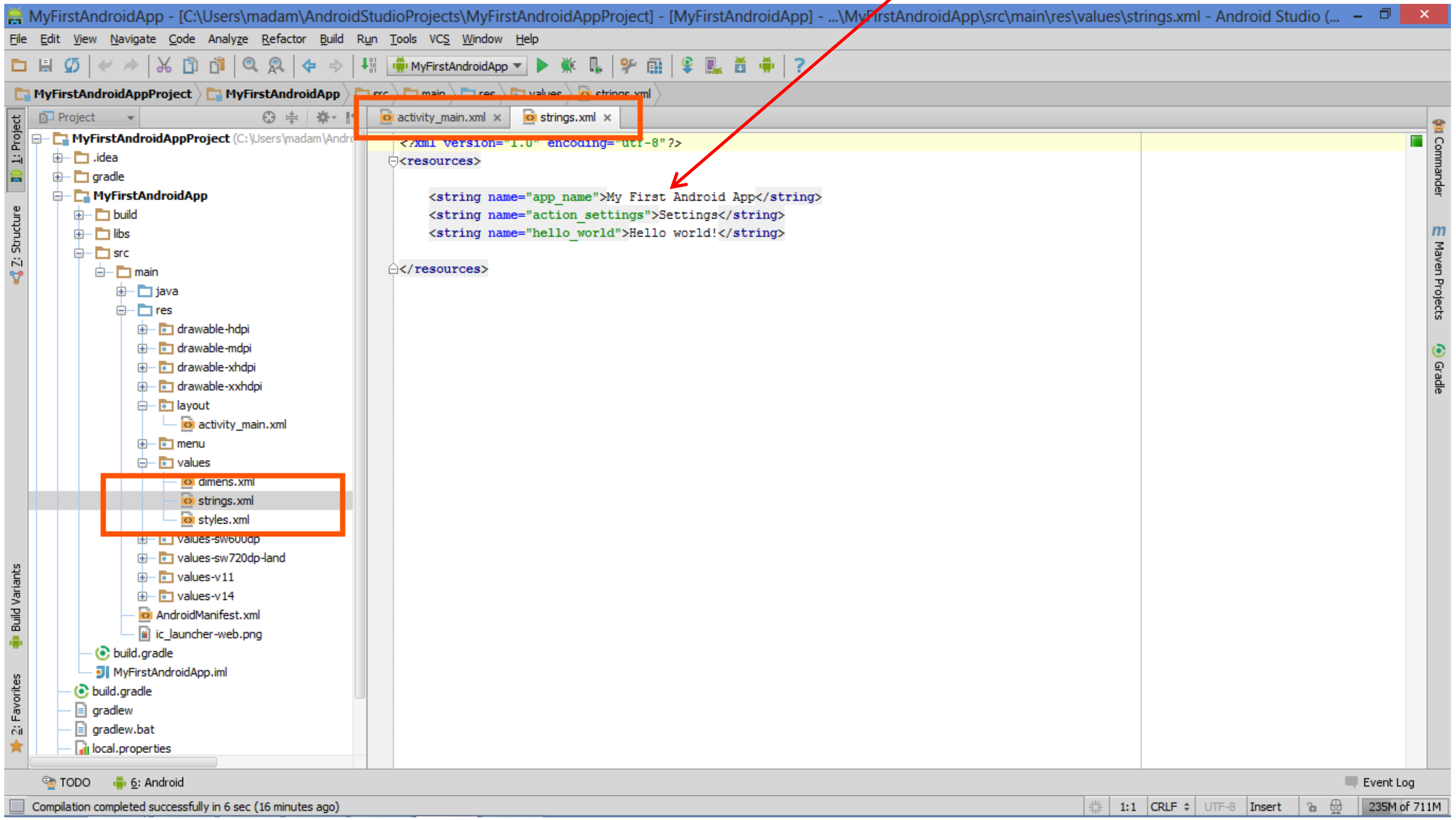
String usage in AndroidManifest.xml



```
  <application
    android:allowBackup="false"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme">
    <activity
      android:name="EmPubLiteActivity"
      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>
```

Where is strings.xml in Android Studio?

Editting any string in strings.xml changes it wherever it is displayed



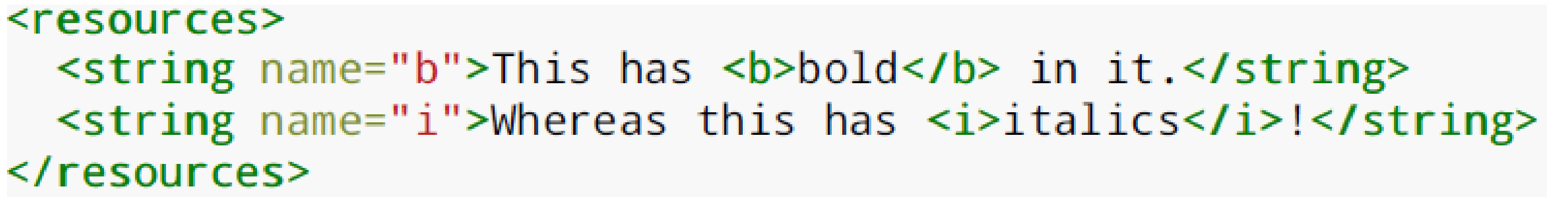
```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">My First Android App</string>
    <string name="action_settings">Settings</string>
    <string name="hello_world">Hello world!</string>

</resources>
```

Compilation completed successfully in 6 sec (16 minutes ago)

Styled Text

- In HTML, tags can be used for italics, bold, etc
  - E.g. <i> Hello </i> makes text *Hello*
  - <b> Hello <b> makes text **Hello**

- Can use the same HTML tags to add style (italics, bold, etc) to Android strings



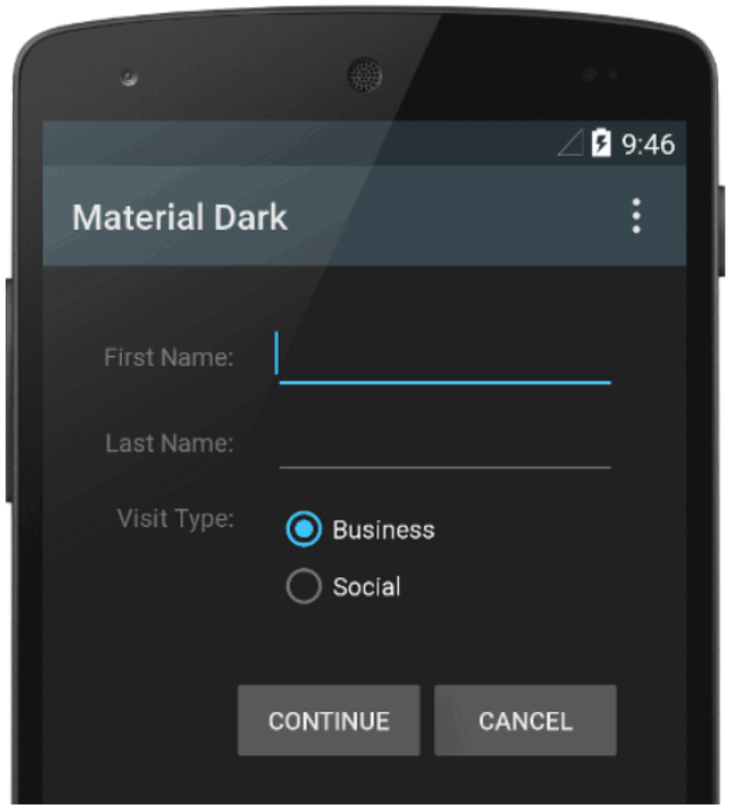
```
<resources>
  <string name="b">This has <b>bold</b> in it.</string>
  <string name="i">Whereas this has <i>italics</i>!</string>
</resources>
```

Android Themes

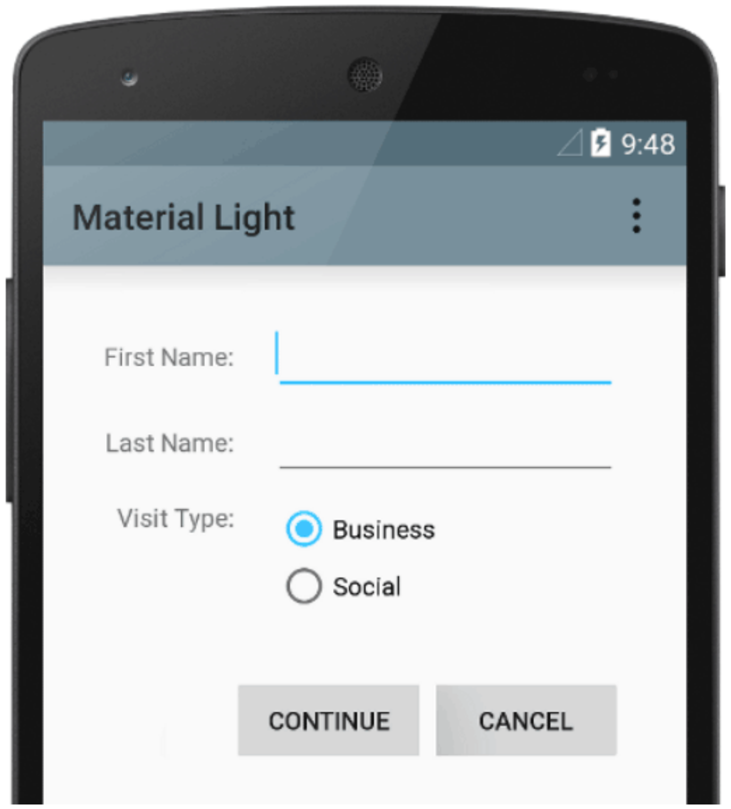
Styles

- Android widgets have properties
  - E.g. Foreground color = red
- **Styles in Android:** specifies properties for **multiple attributes** of **1 widget**
  - E.g. height, padding, font color, font size, background color
- Similar to Cascaded Style Sheets (CSS) in HTML
- Themes apply styles to **all widgets on an app screen (Activity)**
  - E.g. all widgets on a screen can adopt the same font

Examples: Different Themes Applied to Same Screen



Theme.AppCompat

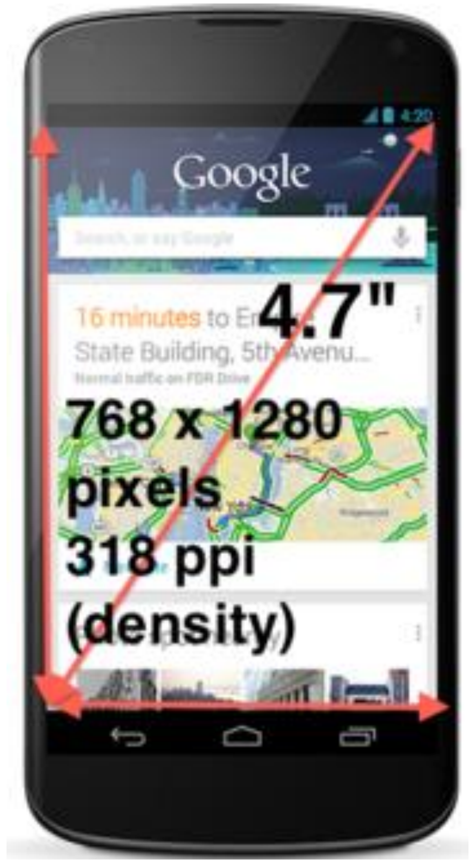


Theme.AppCompat.Light

Adding Pictures in Android

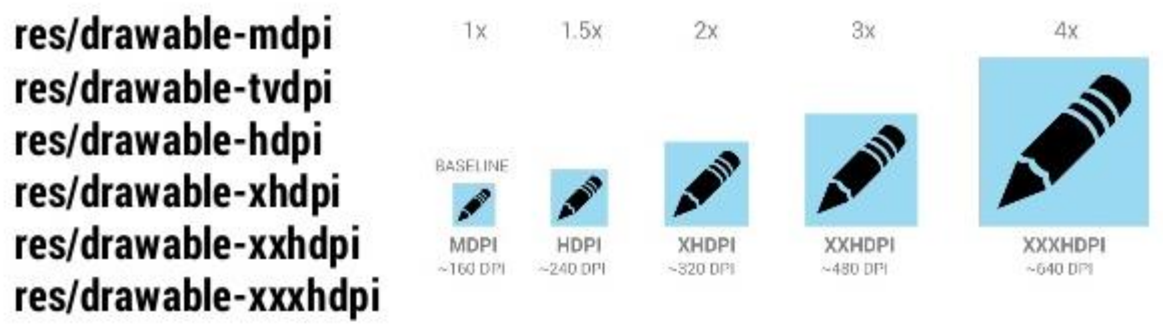
Phone Dimensions Used in Android UI

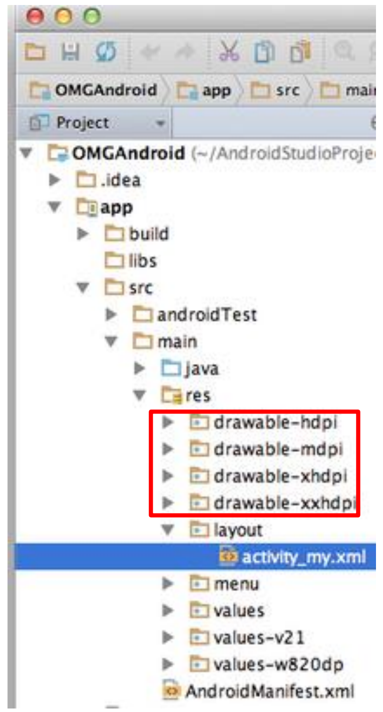
- Physical dimensions (inches) diagonally
  - E.g. Nexus 4 is 4.7 inches diagonally
- Resolution in pixels
  - E.g. Nexus 4 resolution 768 x 1280 pixels
  - No. of pixels diagonally: Sqrt[(768 x 768) + (1280 x 1280)]
- Pixels per inch (PPI) on diagonal =
  - Sqrt[(768 x 768) + (1280 x 1280)] / 4.7= 318



Adding Pictures

- Android supports images in PNG, JPEG and GIF formats
- Put different resolutions of **same image** into different directories
  - **res/drawable-ldpi:** low dpi images (~ 120 dpi of dots per inch)
  - **res/drawable-mdpi:** medium dpi images (~ 160 dpi)
  - **res/drawable-hdpi:** high dpi images (~ 240 dpi)
  - **res/drawable-xhdpi:** extra high dpi images (~ 320 dpi)
  - **res/drawable-xxhdpi:** extra extra high dpi images (~ 480 dpi)
  - **res/drawable-xxxhdpi:** high dpi images (~ 640 dpi)

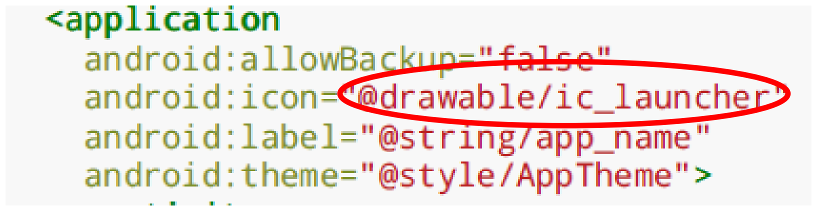

res/drawable-mdpi
res/drawable-tvdpi
res/drawable-hdpi
res/drawable-xhdpi
res/drawable-xxhdpi
res/drawable-xxxhdpi



```
OMGAndroid (~/AndroidStudioProje
  .idea
  app
    build
    libs
    src
      androidTest
      main
        java
        res
          drawable-hdpi
          drawable-mdpi
          drawable-xhdpi
          drawable-xxhdpi
          layout
            activity_my.xml
          menu
          values
          values-v21
          values-w820dp
        AndroidManifest.xml
```

Adding Pictures

- Use generic picture name in code (no .png, .jpg, etc)
  - E.g. to reference an image **ic_launcher.png**



```
<application
  android:allowBackup="false"
  android:icon="@drawable/ic_launcher"
  android:label="@string/app_name"
  android:theme="@style/AppTheme">
```

- At run-time, Android chooses appropriate resolution/directory (e.g. –mdpi) based on phone resolution

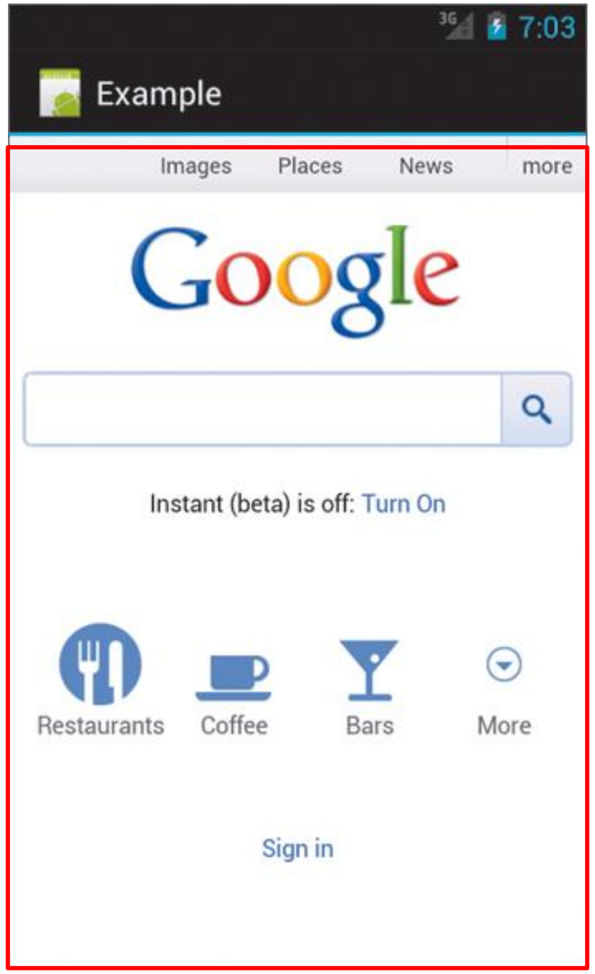
- **Image Asset Studio:** generates icons in various densities from original image

  Ref: https://developer.android.com/studio/write/create-app-icons

WebView Widget

WebView Widget

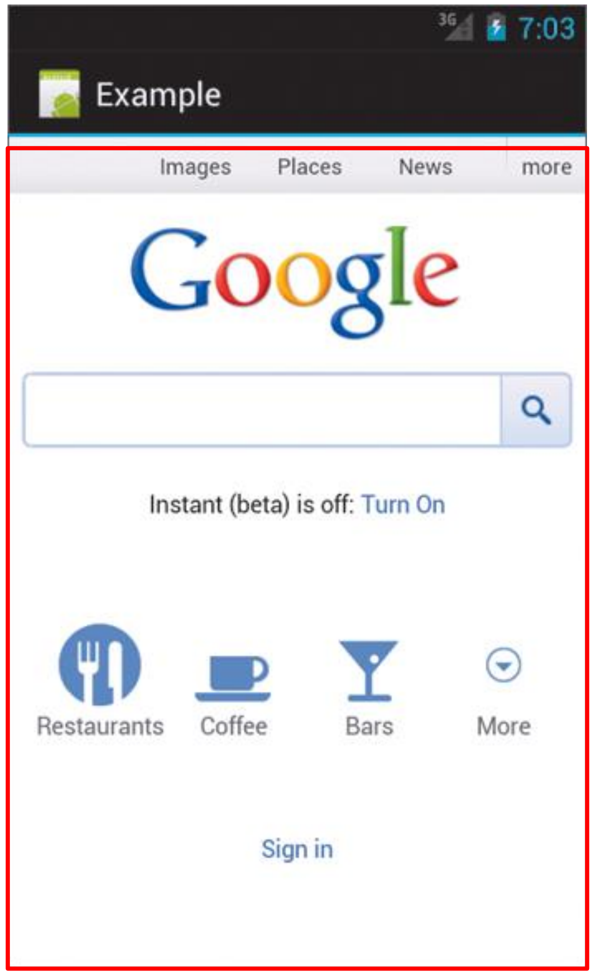
- A View that displays web pages
  - Can be used for creating your own web browser
  - OR just display some web content inside your app



WebView Widget

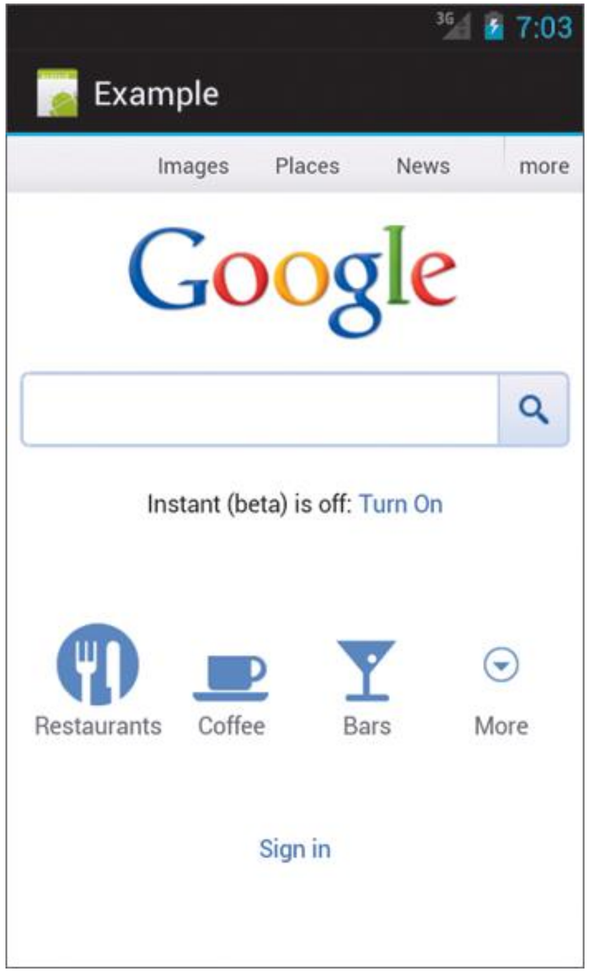
- Since Android 4.4, webviews rendered using:
  - Chromium open source project, engine used in Google Chrome browser (http://www.chromium.org/)

- Webviews on earlier Android versions supported webkit, which is used in many web browsers including Safari



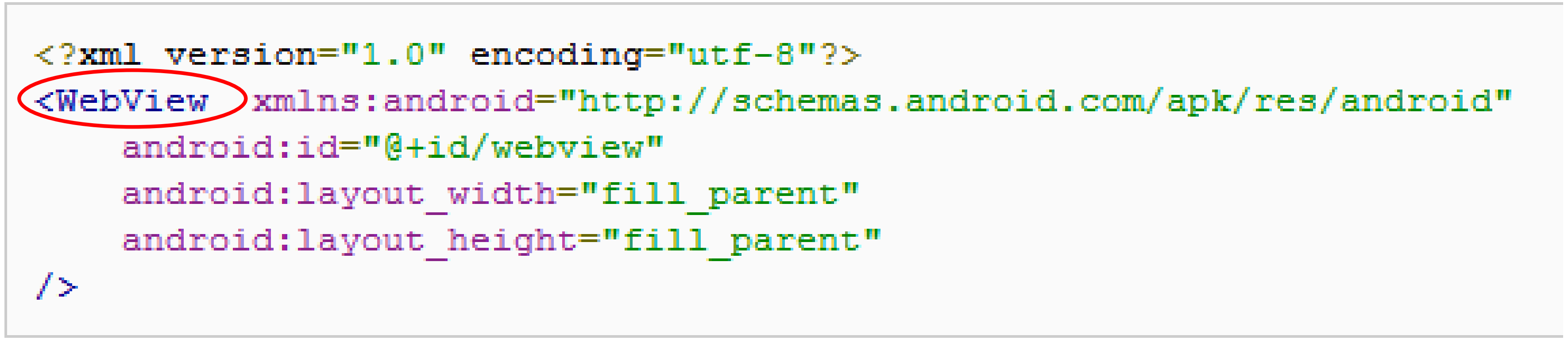
WebView Widget Functionality

- Supports HTML5, CSS3 and JavaScript
- Navigate previous URLs (back and forward)
- zoom in and out
- perform searches
- Can also:
  - Embed images in page
  - Search page for strings
  - Handle cookies



WebView Example

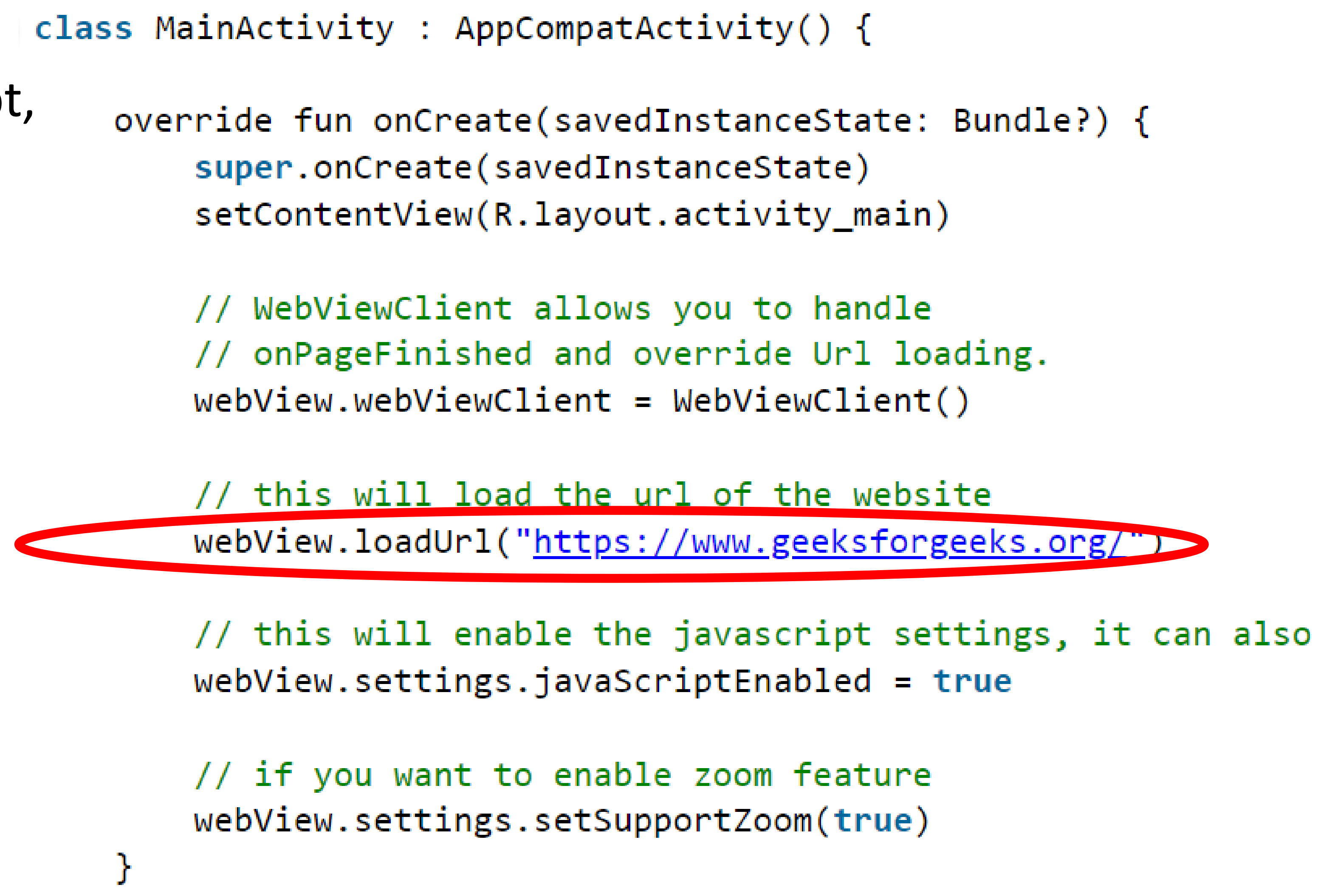
- Simple app to view and navigate web pages
- XML code (e.g in res/layout/main.xml) to declare WebView rectangle



```
<?xml version="1.0" encoding="utf-8"?>
<WebView  xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/webview"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
/>
```

WebView Activity

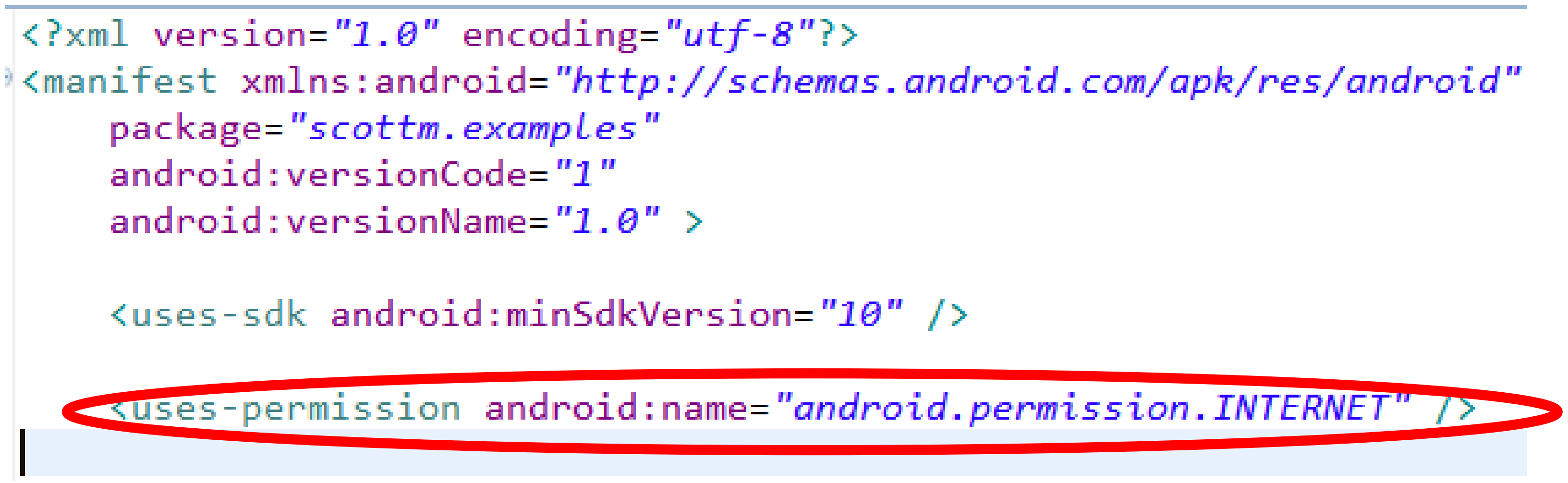
- In onCreate, use loadURL to specify website to load
- If website contains Javascript, enable Javascript
- loadUrl( ) can also load files on Android local filesystem (file://)



```
class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        // WebViewClient allows you to handle
        // onPageFinished and override Url loading.
        webView.webViewClient = WebViewClient()

        // this will load the url of the website
        webView.loadUrl("https://www.geeksforgeeks.org/")

        // this will enable the javascript settings, it can also
        webView.settings.javaScriptEnabled = true

        // if you want to enable zoom feature
        webView.settings.setSupportZoom(true)
    }
```

WebView: Request Internet Access

- In AndroidManifest.xml, request owner of phone to grant **permission to use Internet**

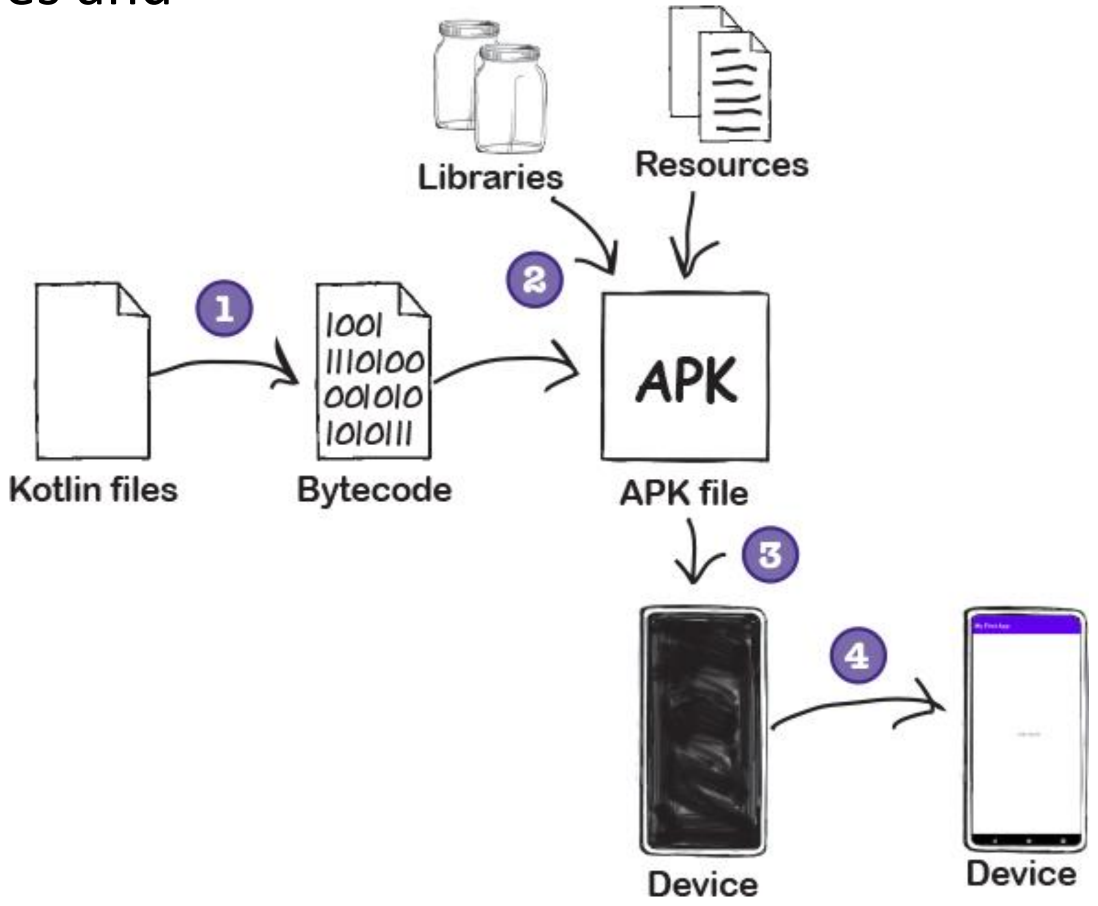


```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="scottm.examples"
    android:versionCode="1"
    android:versionName="1.0" >

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

    <uses-permission android:name="android.permission.INTERNET" />
```

Android Compilation Process
(In more detail)

Android: Compile, Package, Deploy, Run

1. Kotlin files get compiled into bytecode
2. APK file gets created from bytecode, libraries and resources
3. APK is installed (copied) on device
4. Device starts app's main activity



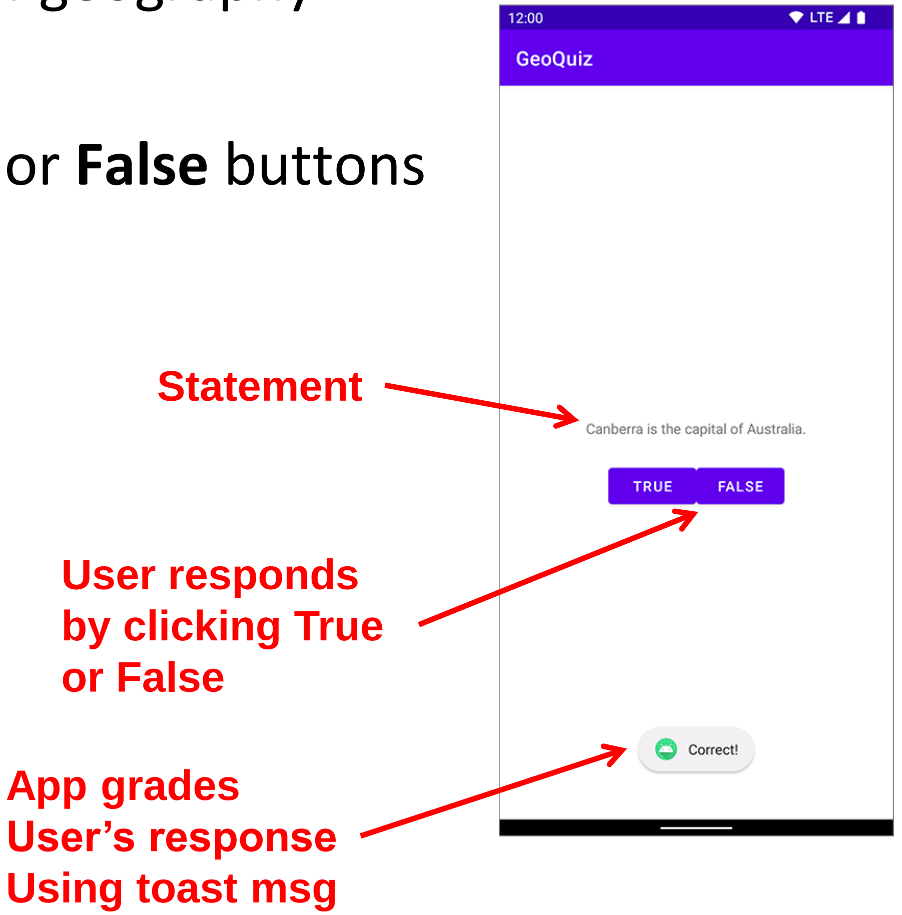
Ref: HFAD (3rd edition), page 25

Android UI Design Example

GeoQuiz App

Ref: Android Nerd Ranch (5th edition), pgs 1-32

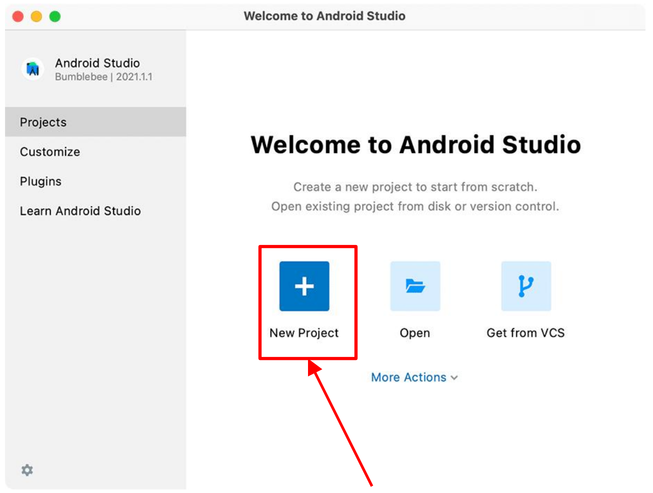
- App makes statements about geography, with goal to test user's knowledge of geography
- User answers by pressing **True** or **False** buttons
- How to get this book?


Statement
User responds by clicking True or False
App grades User's response Using toast msg
12:00
LTE
GeoQuiz
TRUE
FALSE

Canberra is the capital of Australia.

Correct!

GeoQuiz App: Getting Started in Android Studio


Welcome to Android Studio

Android Studio
Bumblebee | 2021.1.1

Projects
Customize
Plugins
Learn Android Studio

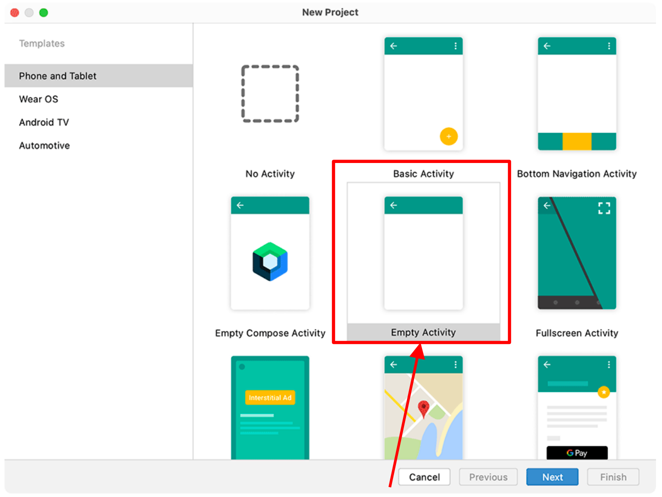
Welcome to Android Studio

Create a new project to start from scratch.
Open existing project from disk or version control.

New Project   Open   Get from VCS

More Actions

1. Start Android app, select “New Project”


New Project

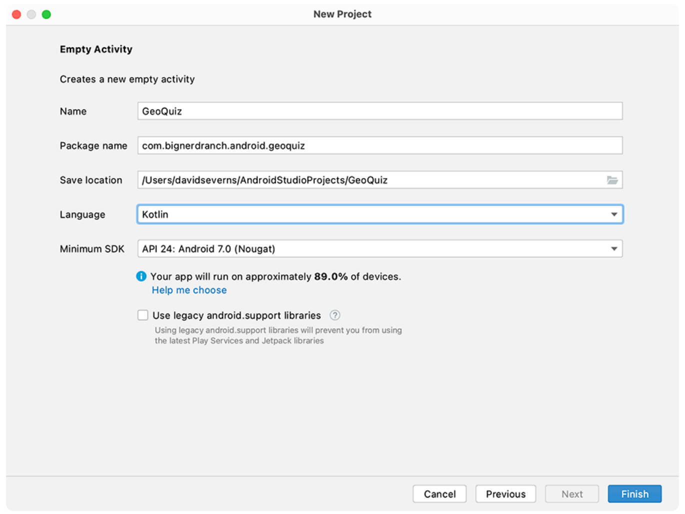
Templates

Phone and Tablet
Wear OS
Android TV
Automotive

No Activity   Basic Activity   Bottom Navigation Activity
Empty Compose Activity   Empty Activity   Fullscreen Activity

Cancel   Previous   Next   Finish

2. Select Android “Empty Activity” template


New Project

Empty Activity

Creates a new empty activity

Name	GeoQuiz
Package name	com.bignerdranch.android.geoquiz
Save location	/Users/davidseverns/AndroidStudioProjects/GeoQuiz
Language	Kotlin
Minimum SDK	API 24: Android 7.0 (Nougat)

Your app will run on approximately **89.0%** of devices.
Help me choose

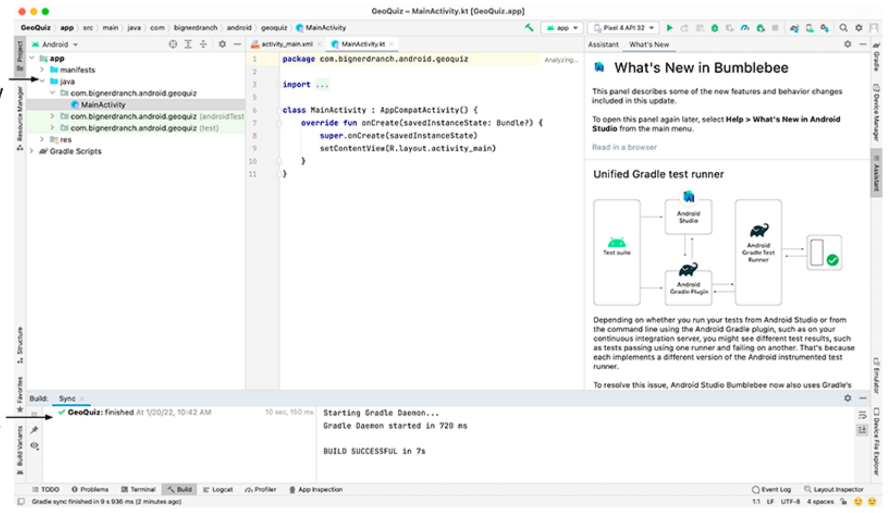
☐ Use legacy android.support libraries
Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

Cancel   Previous   Next   Finish

3. Configure New Project, select kotlin as language

Project
tool window

Build
tool window


GeoQuiz – MainActivity.kt [GeoQuiz.app]

GeoQuiz › app › src › main › java › com › bignerdranch › android › geoquiz › MainActivity

Android
- app
  - manifests
  - java
    - com.bignerdranch.android.geoquiz
      - MainActivity
    - com.bignerdranch.android.geoquiz (androidTest)
    - com.bignerdranch.android.geoquiz (test)
  - res
- Gradle Scripts

activity_main.xml   MainActivity.kt

```
package com.bignerdranch.android.geoquiz

import ...

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

Assistant   What's New

What's New in Bumblebee

This panel describes some of the new features and behavior changes included in this update.

To open this panel again later, select **Help > What's New in Android Studio** from the main menu.

Read in a browser

Unified Gradle test runner

Depending on whether you run your tests from Android Studio or from the command line using the Android Gradle plugin, such as on your continuous integration server, you might see different test results, such as tests passing using one runner and failing on another. That's because each implements a different version of the Android instrumented test runner.

To resolve this issue, Android Studio Bumblebee now also uses Gradle's

Build: Sync

✓ GeoQuiz: finished At 1/20/22, 10:42 AM   10 sec, 150 ms

```
Starting Gradle Daemon...
Gradle Daemon started in 720 ms

BUILD SUCCESSFUL in 7s
```

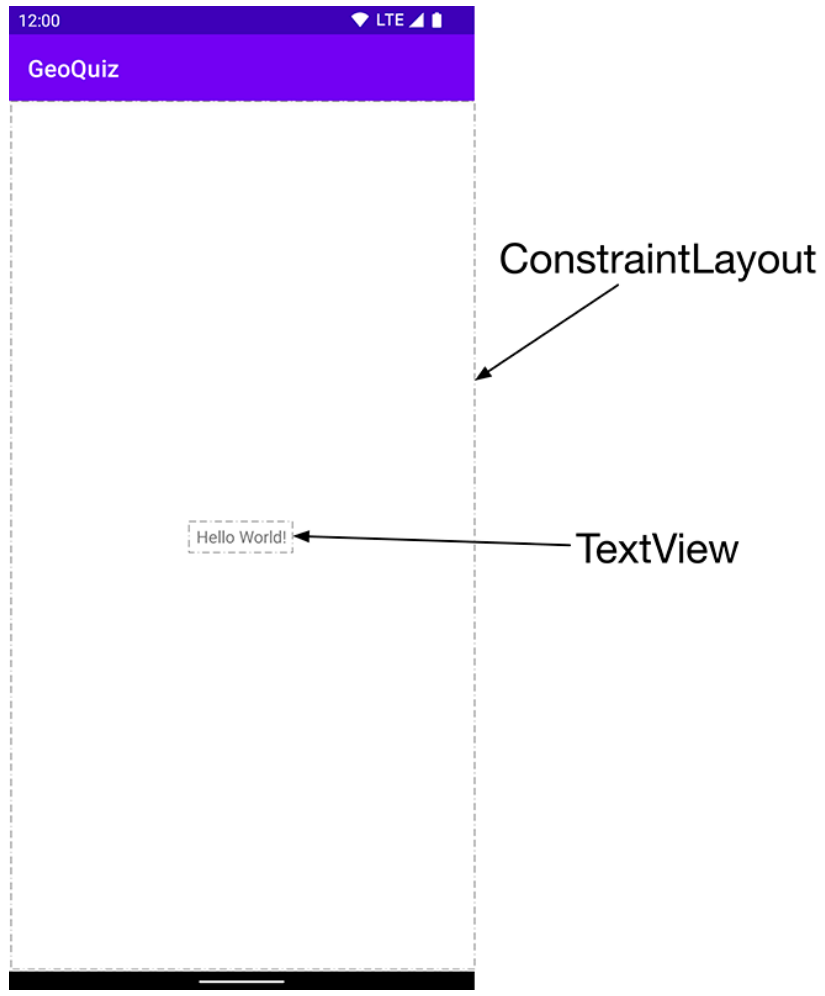
TODO   Problems   Terminal   Build   Logcat   Profiler   App Inspection   Event Log   Layout Inspector

Gradle sync finished in 9 s 936 ms (2 minutes ago)   1:1   LF   UTF-8   4 spaces

4. Land in Android fresh project window

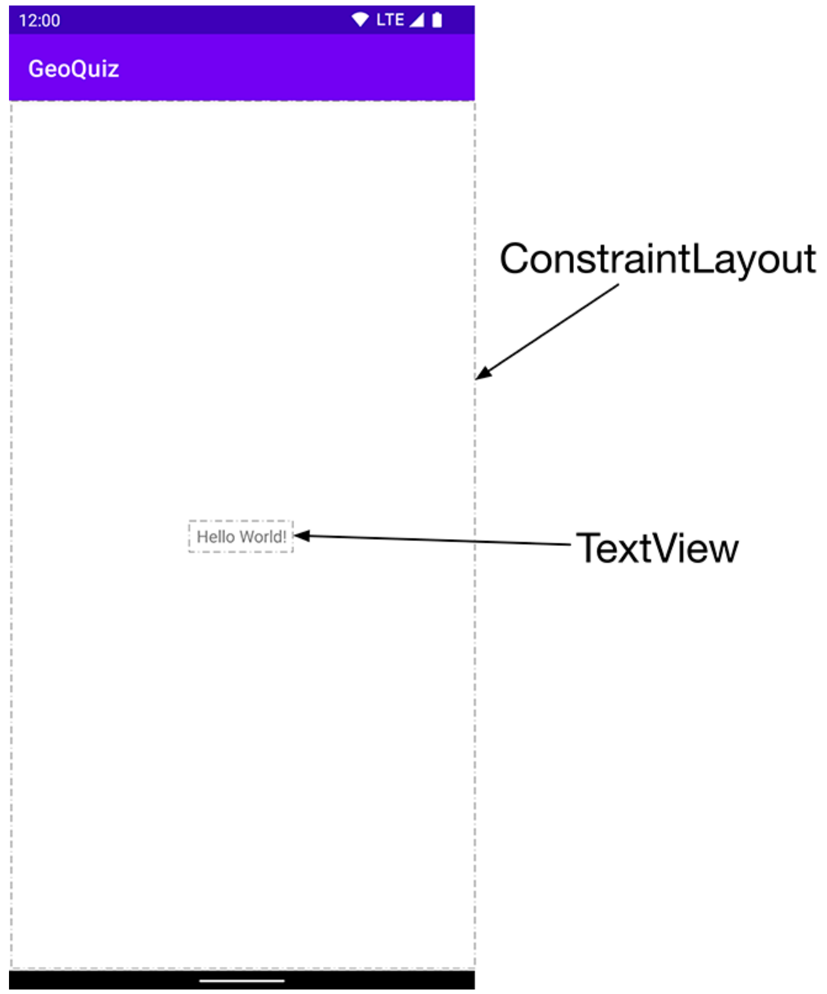
QeoQuiz: Default views

- Default XML generates:
  - ConstraintLayout
  - Textview (with string “Hello World!”)



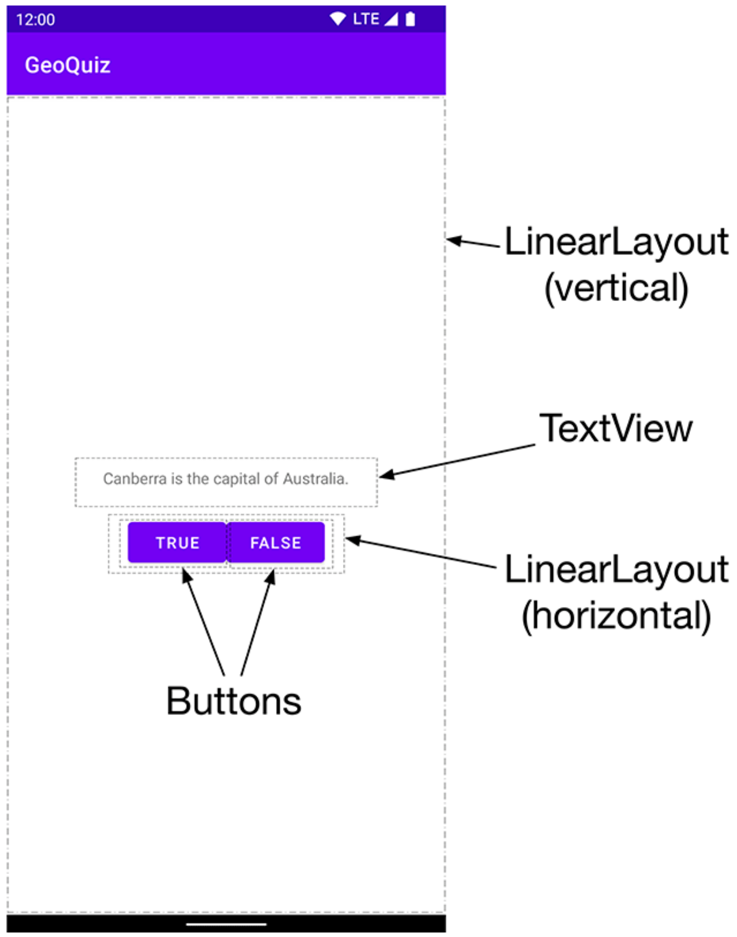
GeoQuiz App Files

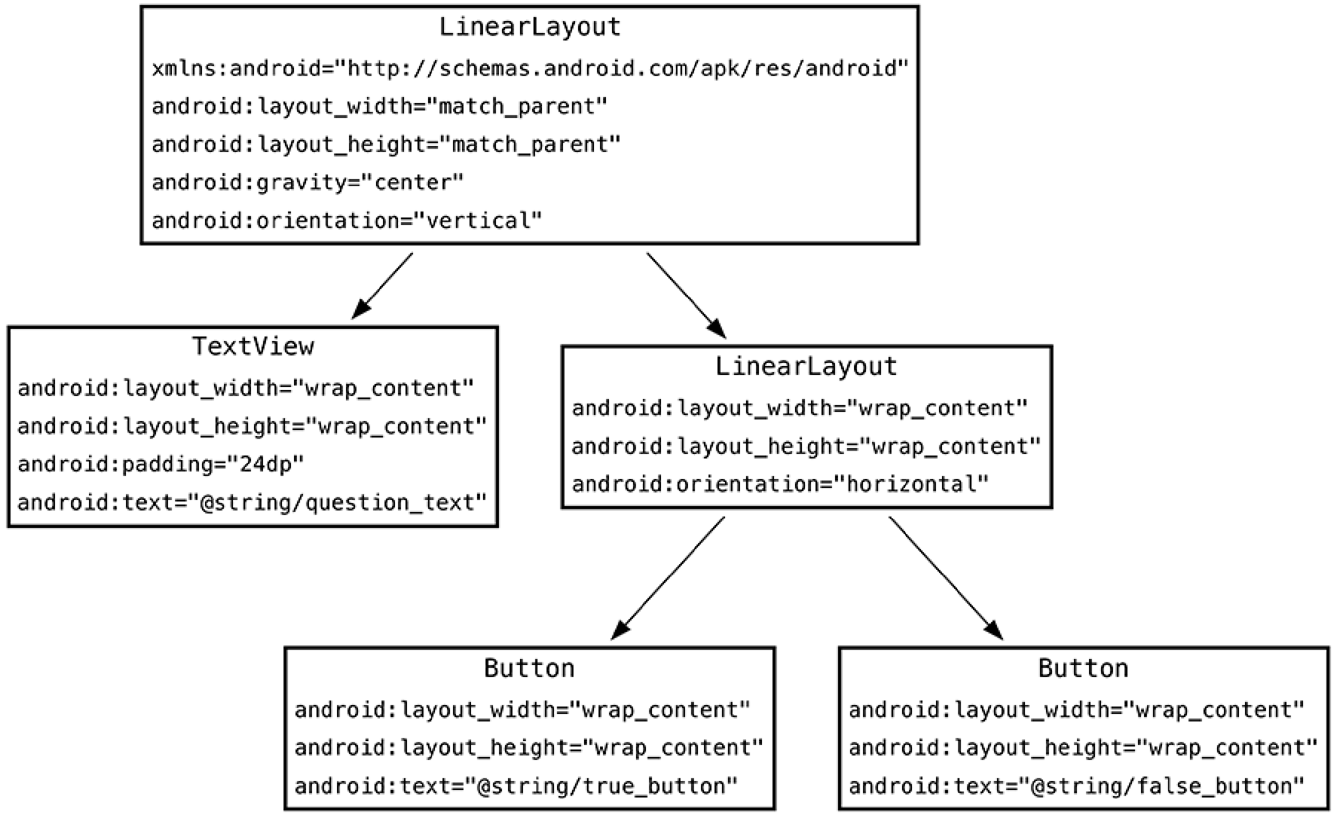
- 2 main files:
  - **activity_quiz.xml:** to format app look
  - **MainActivity** (Kotlin/java file) to manage UI, present question, accept True/False response
- **AndroidManifest.xml** lists all app components, auto-generated



GeoQuiz: Plan Out App Widgets (in activity_quiz.xml)

- 5 Widgets arranged hierarchically


12:00
LTE
GeoQuiz
Canberra is the capital of Australia.
TRUE
FALSE
LinearLayout (vertical)
TextView
LinearLayout (horizontal)
Buttons



```
LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:gravity="center"
android:orientation="vertical"
```

```
TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:padding="24dp"
android:text="@string/question_text"
```

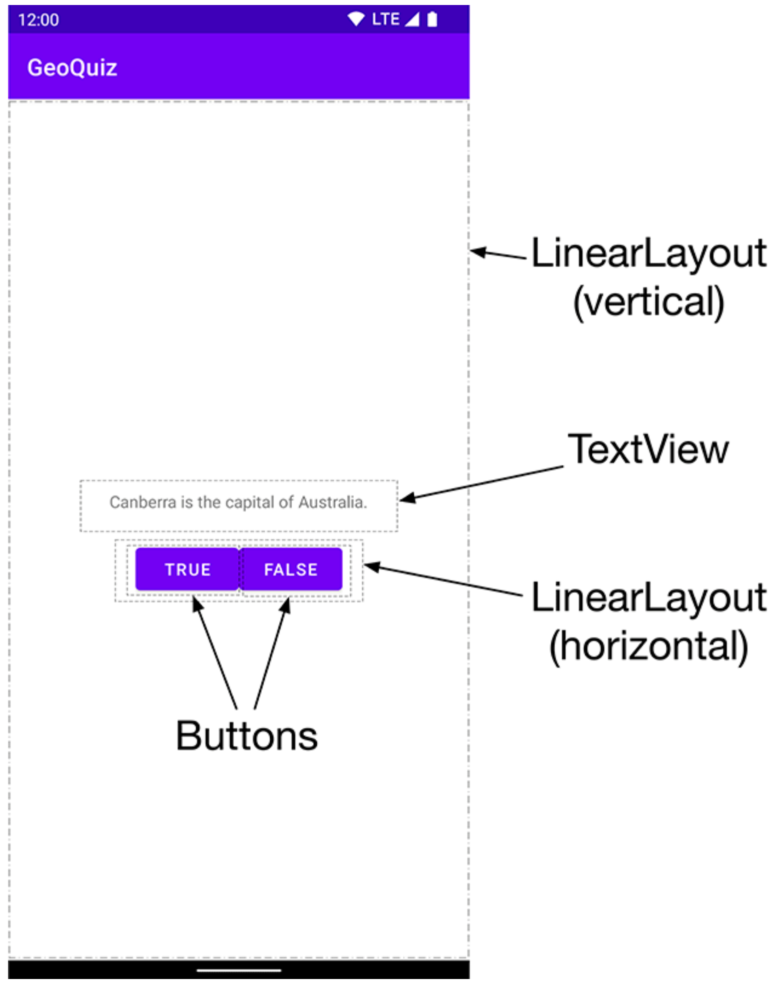
```
LinearLayout
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:orientation="horizontal"
```

```
Button
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="@string/true_button"
```

```
Button
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="@string/false_button"
```

GeoQuiz: activity_quiz.xml File listing

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="24dp"
        android:text="@string/question_text" />

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/true_button" />

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/false_button" />

    </LinearLayout>

</LinearLayout>
```

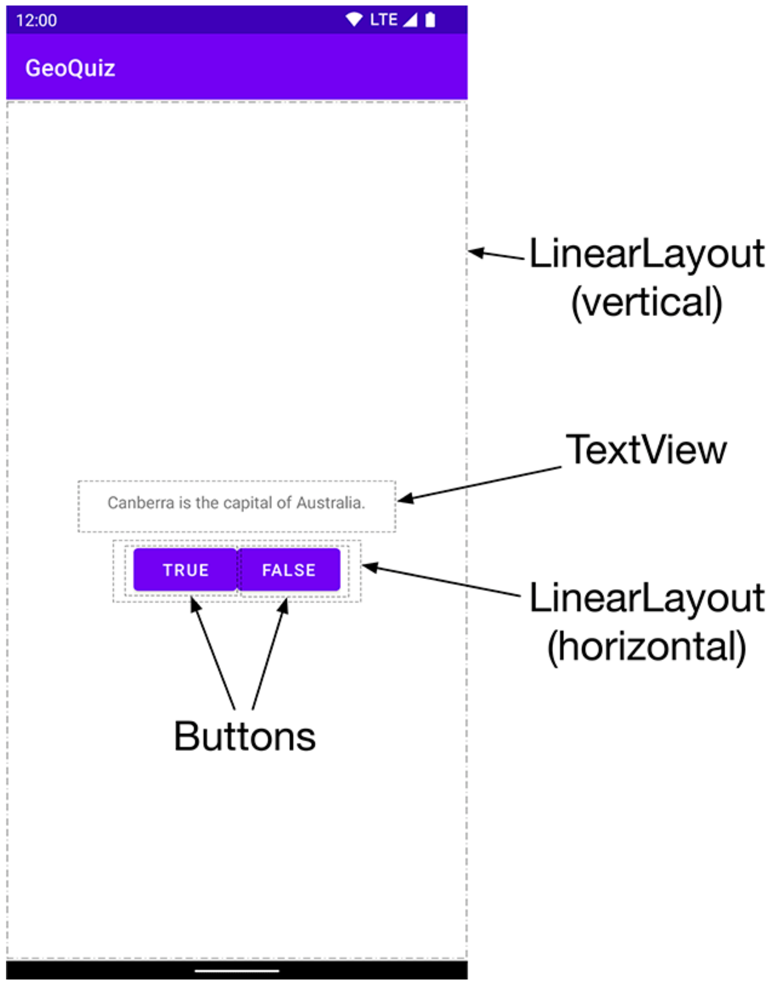

12:00
LTE
GeoQuiz
Canberra is the capital of Australia.
TRUE
FALSE
LinearLayout (vertical)
TextView
LinearLayout (horizontal)
Buttons

GeoQuiz: strings.xml File listing

- Define all strings app will use
  - Question: “Canberra is the capital …. “
  - True
  - False

res/values/strings.xml

```
<resources>
    <string name="app_name">GeoQuiz</string>
    <string name="question_text">Canberra is the capital of Australia.</string>
    <string name="true_button">True</string>
    <string name="false_button">False</string>
</resources>
```


12:00
LTE
GeoQuiz
LinearLayout (vertical)
TextView
Canberra is the capital of Australia.
TRUE
FALSE
LinearLayout (horizontal)
Buttons

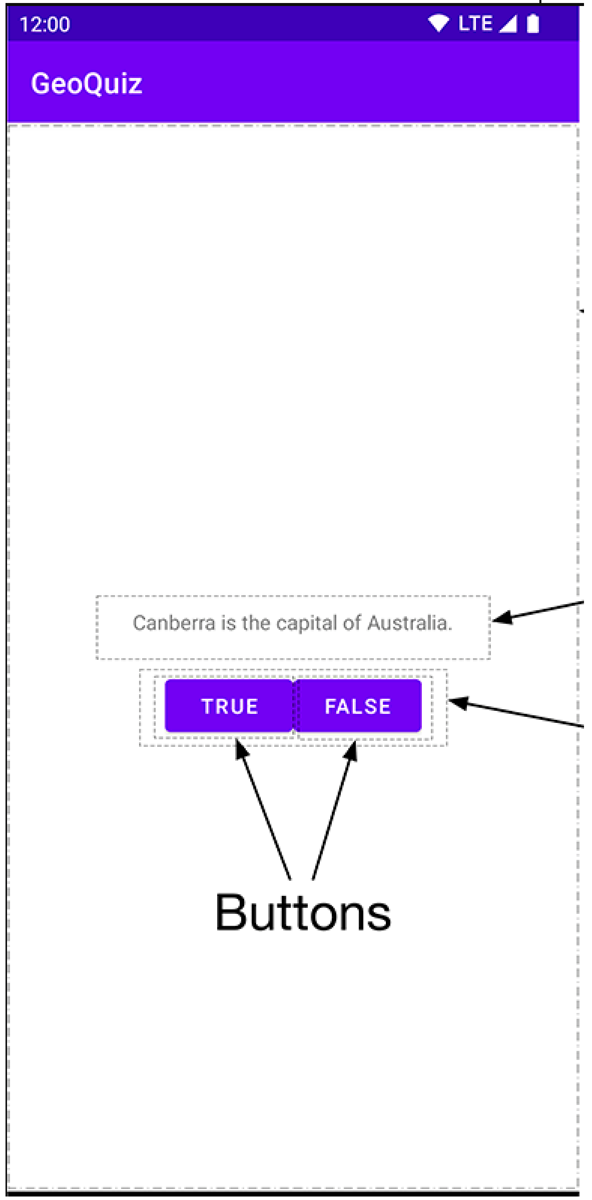
Initial QuizActivity.kt Code (in ../java Directory)

- MainActivity derived from Android AppCompatActivity class, ensures compatibility with older Android versions

```
package com.bignerdranch.android.geoquiz

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

onCreate Method is called once Activity is created (like a constructor?)

specify layout XML file (**activity_quiz.xml**)



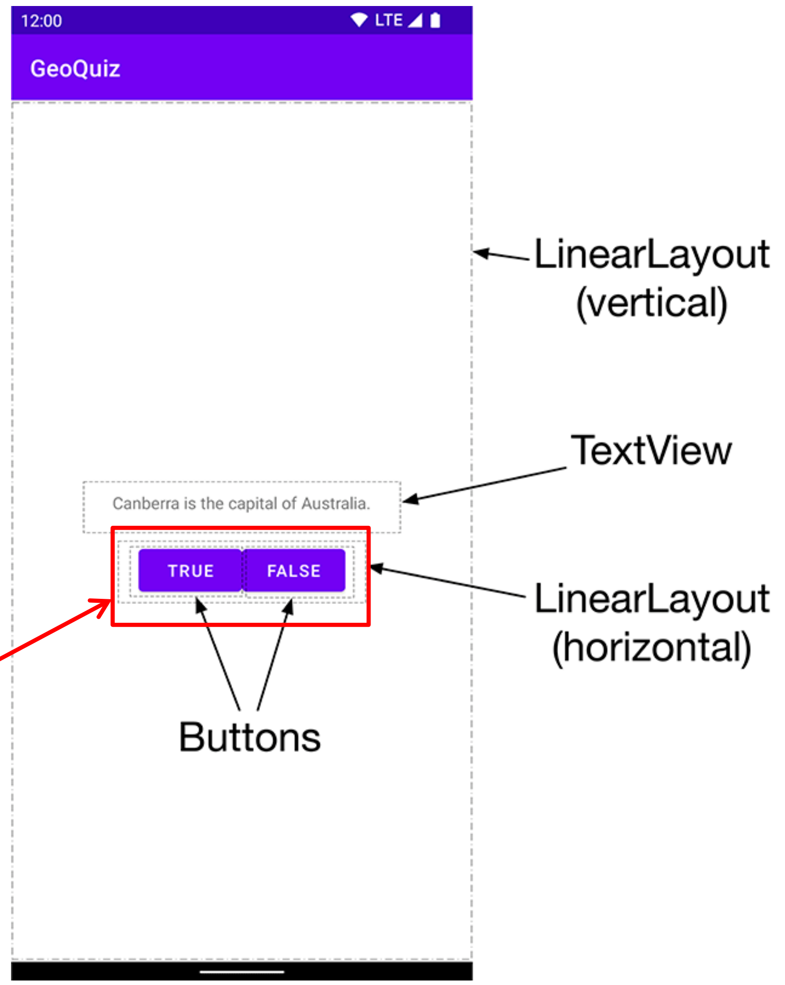
- Would like kotlin code to respond to True/False buttons being clicked

Responding to True/False Buttons in Kotlin File

XML file

```
<LinearLayout ... >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="24dp"
        android:text="@string/question_text" />

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <Button
            android:id="@+id/true_button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/true_button" />

        <Button
            android:id="@+id/false_button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/false_button" />

    </LinearLayout>

</LinearLayout>
```

Write code in Kotlin file to specify app's response when True/False buttons are clicked



12:00
LTE
GeoQuiz
Canberra is the capital of Australia.
TRUE
FALSE
LinearLayout (vertical)
TextView
LinearLayout (horizontal)
Buttons

2 Alternative Ways to Respond to Button Clicks

1. In XML: set android:onClick attribute (already seen this!!)

2. In kotlin, create a ClickListener object, override onClick method

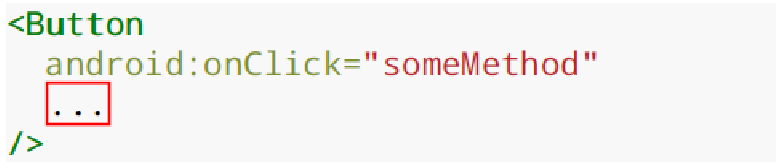
Recall: Approach 1: Responding to Button Clicks

- May want Button press to trigger some action
- How?

1. In XML file (e.g. Activity_my.xml), set android:onClick attribute to specify method to be invoked

2. In Kotlin file (e.g. MainActivity.kt) declare method/handler to take desired action

Activity_my.xml



```
<Button
  android:onClick="someMethod"
  ...
/>
```

MainActivity.kt

… declare someMethod function

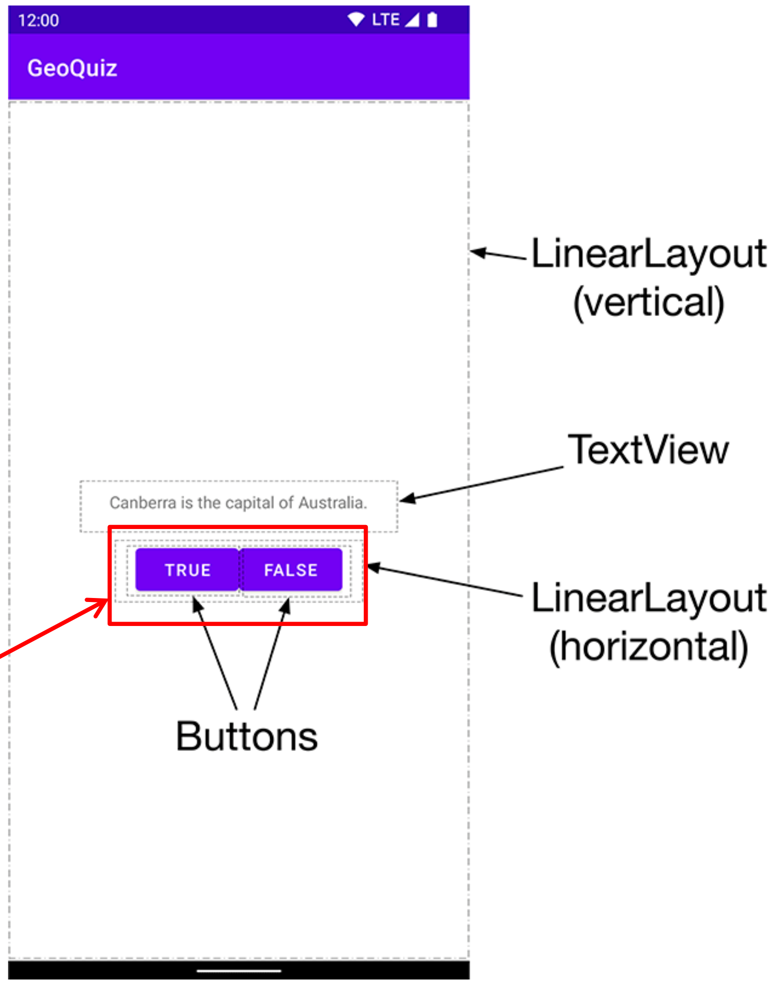
Approach 2: Create a ClickListener object, override onClick

- First, get reference to Button in our kotlin file. How?

```
<Button
  android:id="@+id/true_button"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="@string/true_button" />
```

```
<Button
  android:id="@+id/false_button"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="@string/false_button" />
```

Need reference to Buttons


12:00
LTE
GeoQuiz
Canberra is the capital of Australia.
TRUE
FALSE
LinearLayout (vertical)
TextView
LinearLayout (horizontal)
Buttons

QuizActivity.kt: Getting References to Buttons

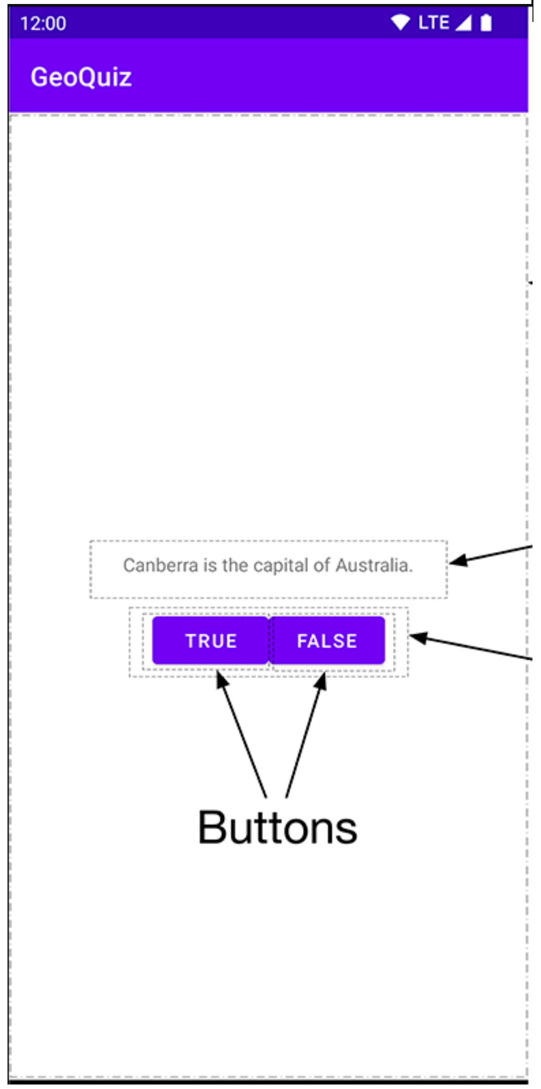
- Compiler assigns each resource an ID
- Use **findViewByID** to find ID of true, false buttons

```
class MainActivity : AppCompatActivity() {

    private lateinit var trueButton: Button
    private lateinit var falseButton: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        trueButton = findViewById(R.id.true_button)
        falseButton = findViewById(R.id.false_button)
    }
}
```

Declaration in XML

```
<Button
  android:id="@+id/true_button"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="@string/true_button" />
```

```
<Button
  android:id="@+id/false_button"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="@string/false_button" />
```



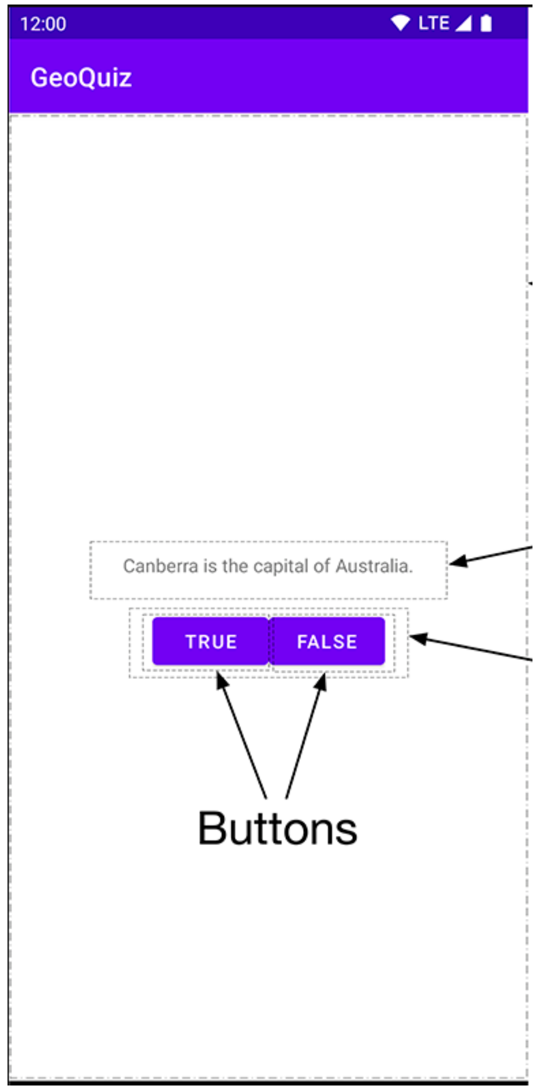
Canberra is the capital of Australia.

QuizActivity.kt: Setting Listeners

- Set listeners for **True** and **False** button clicks
- Implements **View.onClickListener** interface
  - Has one method: **onClick(View)**

```
trueButton.setOnClickListener { view: View ->
    // Do something in response to the click here
}
```

```
falseButton.setOnClickListener { view: View ->
    // Do something in response to the click here
}
```



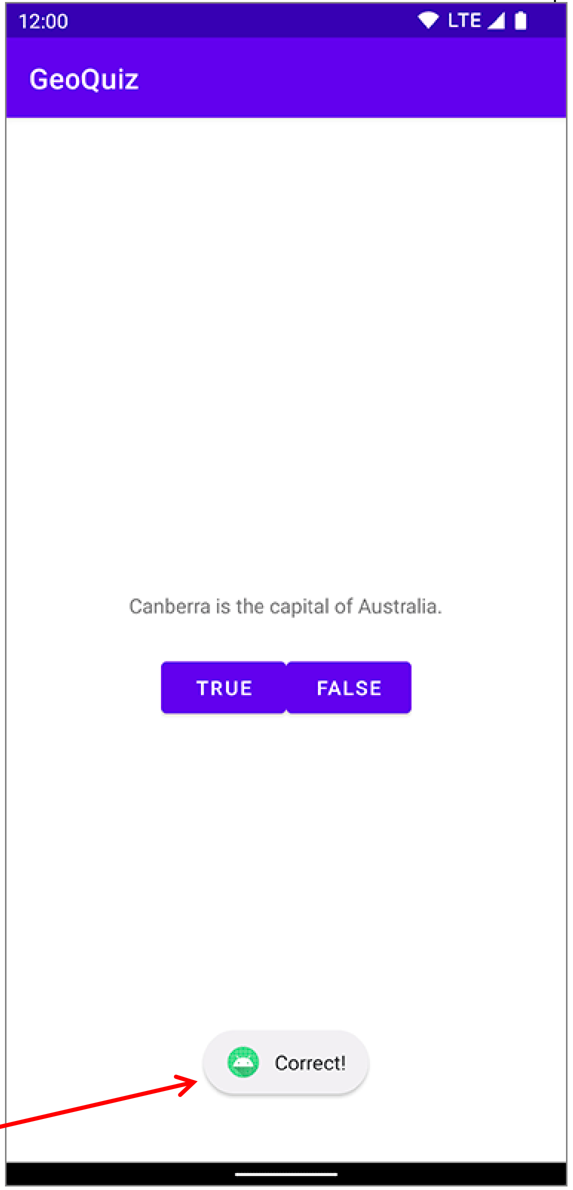
QuizActivity.kt: Adding a Toast

- A toast is a short pop-up message
- Does not require any input or action
- After user clicks True or False button, our app will pop-up a toast to inform the user if they were right or wrong

- First, we need to add toast strings (Correct, Incorrect) to strings.xml

```
<resources>
    <string name="app_name">GeoQuiz</string>
    <string name="question_text">Canberra is the capital of Australia.</string>
    <string name="true_button">True</string>
    <string name="false_button">False</string>
    <string name="correct_toast">Correct!</string>
    <string name="incorrect_toast">Incorrect!</string>
</resources>
```

A toast



QuizActivity.java: Adding a Toast

- To create a toast, call the method:

```
makeText(context: Context, resId: Int, duration: Int)
```

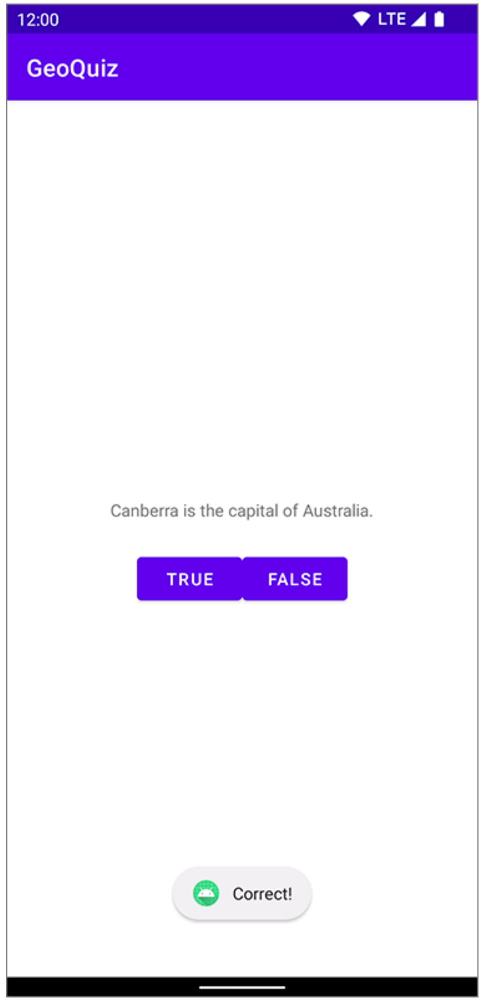
Instance of Activity (Activity is a subclass of context)

Resouce ID of the string that toast should display

Constant to specifiy how long toast should be visible

- After creating toast, call **toast.show( )** to display it.
- E.g, code to wire up trueButton

```
Toast.makeText(
    this,
    R.string.correct_toast,
    Toast.LENGTH_SHORT
).show()
```



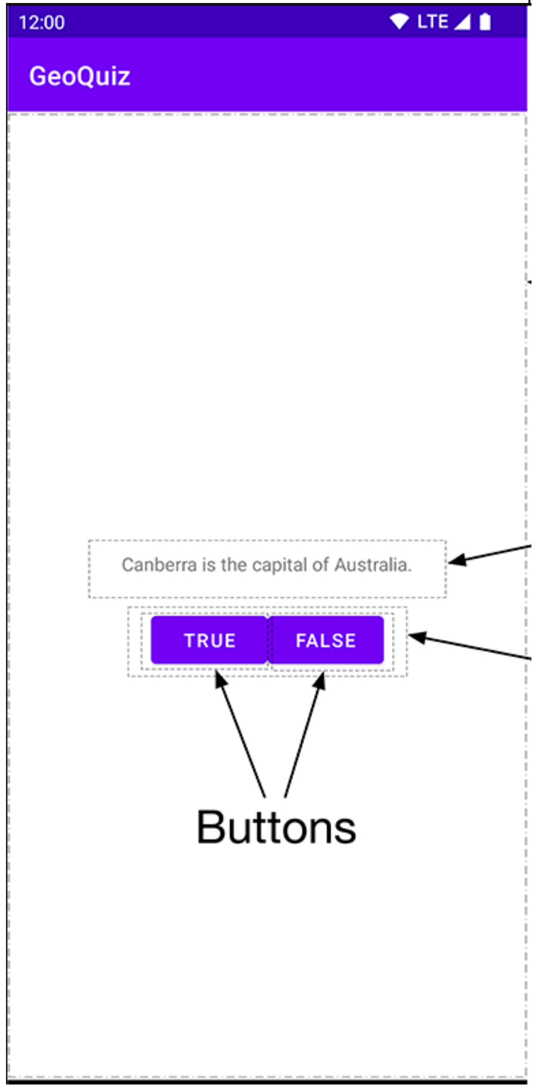
QuizActivity.java: Adding a Toast

- Code for adding a toast to both buttons

Display "correct" toast

Display "Incorrect" toast

```
override fun onCreate(savedInstanceState: Bundle?) {
    ...
    trueButton.setOnClickListener { view: View ->
        ~~// Do something in response to the click here~~
        Toast.makeText(
            this,
            R.string.correct_toast,
            Toast.LENGTH_SHORT
        ).show()
    }

    falseButton.setOnClickListener { view: View ->
        ~~// Do something in response to the click here~~
        Toast.makeText(
            this,
            R.string.incorrect_toast,
            Toast.LENGTH_SHORT
        ).show()
    }
}
```



MainActivity.kt: Complete Listing

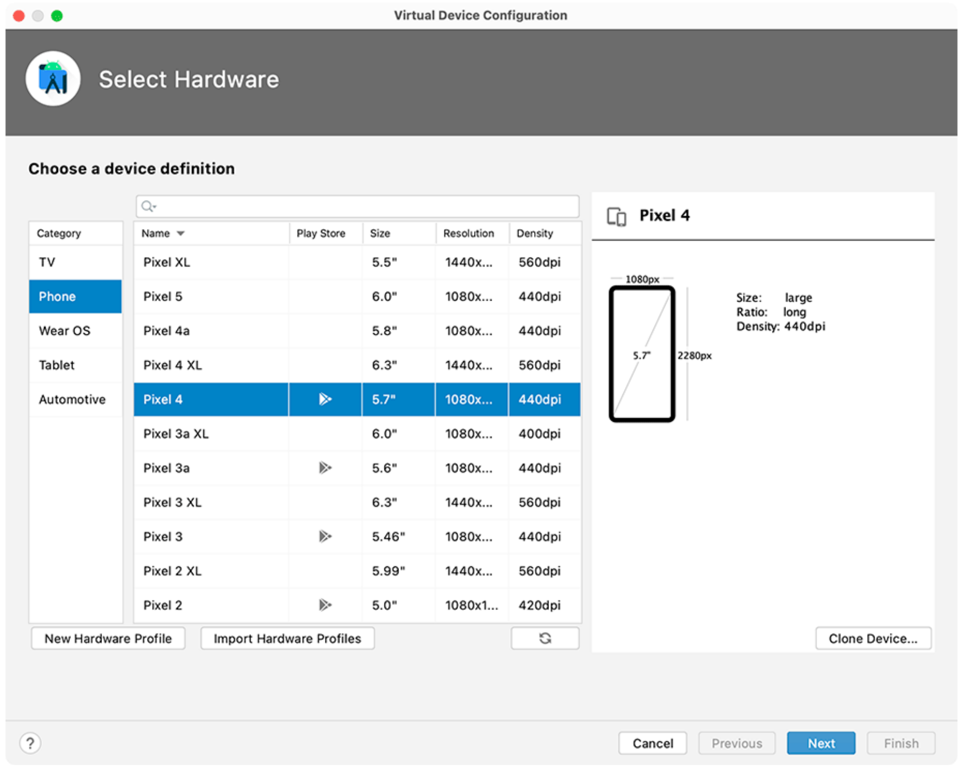
```
package com.bignerdranch.android.geoquiz

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.Button
import android.widget.Toast

class MainActivity : AppCompatActivity() {

    private lateinit var trueButton: Button
    private lateinit var falseButton: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        trueButton = findViewById(R.id.true_button)
        falseButton = findViewById(R.id.false_button)
```

MainActivity.kt: Complete Listing

```
  trueButton.setOnClickListener { view: View ->
            Toast.makeText(
                this,
                R.string.correct_toast,
                Toast.LENGTH_SHORT)
                .show()
        }

        falseButton.setOnClickListener { view: View ->
            Toast.makeText(
                this,
                R.string.incorrect_toast,
                Toast.LENGTH_SHORT)
                .show()
        }
    }
}
```

Create Android Virtual Device to Run Code

- In Android Studio, select Tools -> AVD Manager, click +Create Virtual Device


Virtual Device Configuration

Select Hardware

Choose a device definition

Category
TV
Phone
Wear OS
Tablet
Automotive

Name	Play Store	Size	Resolution	Density
Pixel XL		5.5"	1440x...	560dpi
Pixel 5		6.0"	1080x...	440dpi
Pixel 4a		5.8"	1080x...	440dpi
Pixel 4 XL		6.3"	1440x...	560dpi
Pixel 4	▶	5.7"	1080x...	440dpi
Pixel 3a XL		6.0"	1080x...	400dpi
Pixel 3a	▶	5.6"	1080x...	440dpi
Pixel 3 XL		6.3"	1440x...	560dpi
Pixel 3	▶	5.46"	1080x...	440dpi
Pixel 2 XL		5.99"	1440x...	560dpi
Pixel 2	▶	5.0"	1080x1...	420dpi

Pixel 4

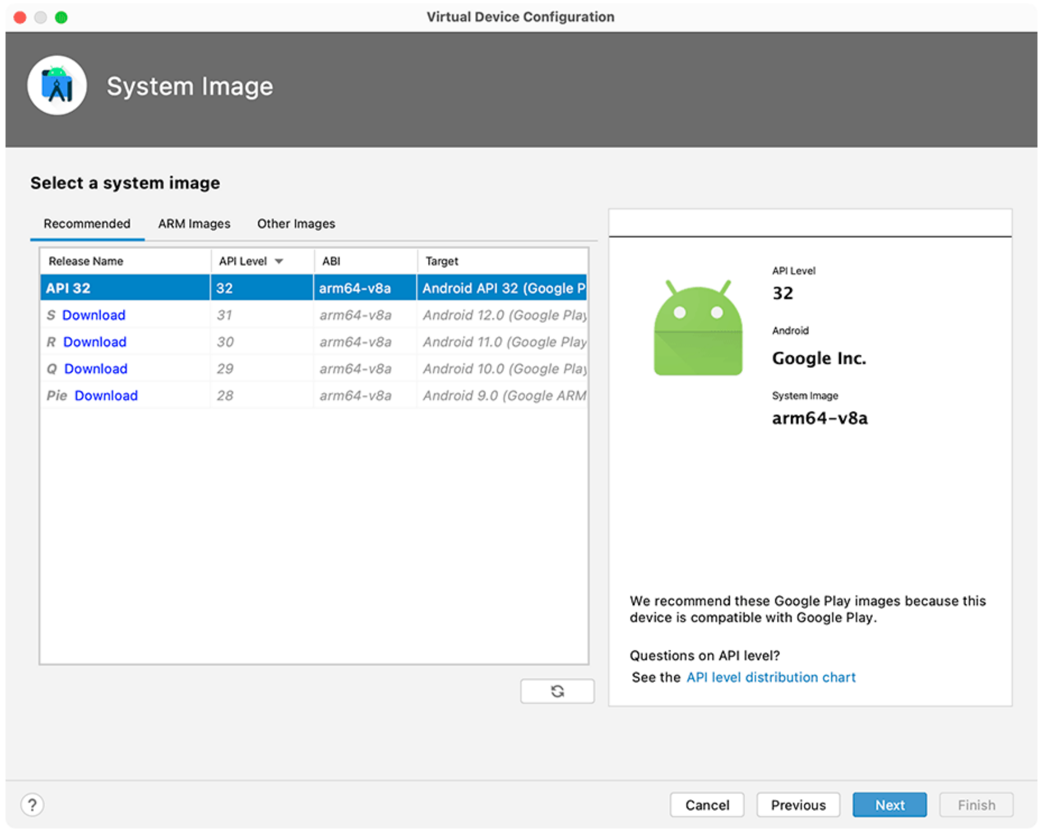
1080px
5.7"
2280px

Size: large
Ratio: long
Density: 440dpi

New Hardware Profile | Import Hardware Profiles | Clone Device...

Cancel | Previous | Next | Finish

Select Phone Hardware


Virtual Device Configuration

System Image

Select a system image

Recommended | ARM Images | Other Images

Release Name	API Level	ABI	Target
API 32	32	arm64-v8a	Android API 32 (Google P
S Download	31	arm64-v8a	Android 12.0 (Google Play
R Download	30	arm64-v8a	Android 11.0 (Google Play
Q Download	29	arm64-v8a	Android 10.0 (Google Play
Pie Download	28	arm64-v8a	Android 9.0 (Google ARM

API Level
32

Android
Google Inc.

System Image
arm64-v8a

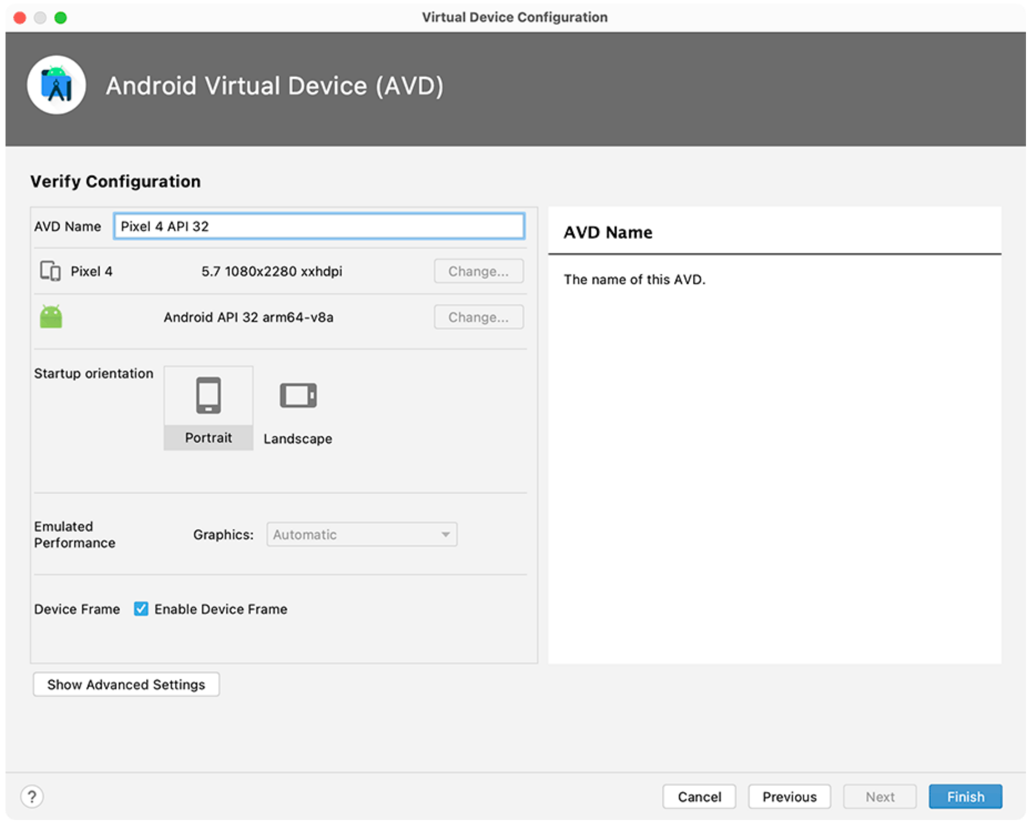
We recommend these Google Play images because this device is compatible with Google Play.

Questions on API level?
See the API level distribution chart

Cancel | Previous | Next | Finish

Select System Image

Create Android Virtual Device to Run Code


Virtual Device Configuration

Android Virtual Device (AVD)

Verify Configuration

AVD Name: Pixel 4 API 32

Pixel 4 — 5.7 1080x2280 xxhdpi — Change...

Android API 32 arm64-v8a — Change...

Startup orientation: [✓ Portrait] Landscape

Emulated Performance — Graphics: Automatic

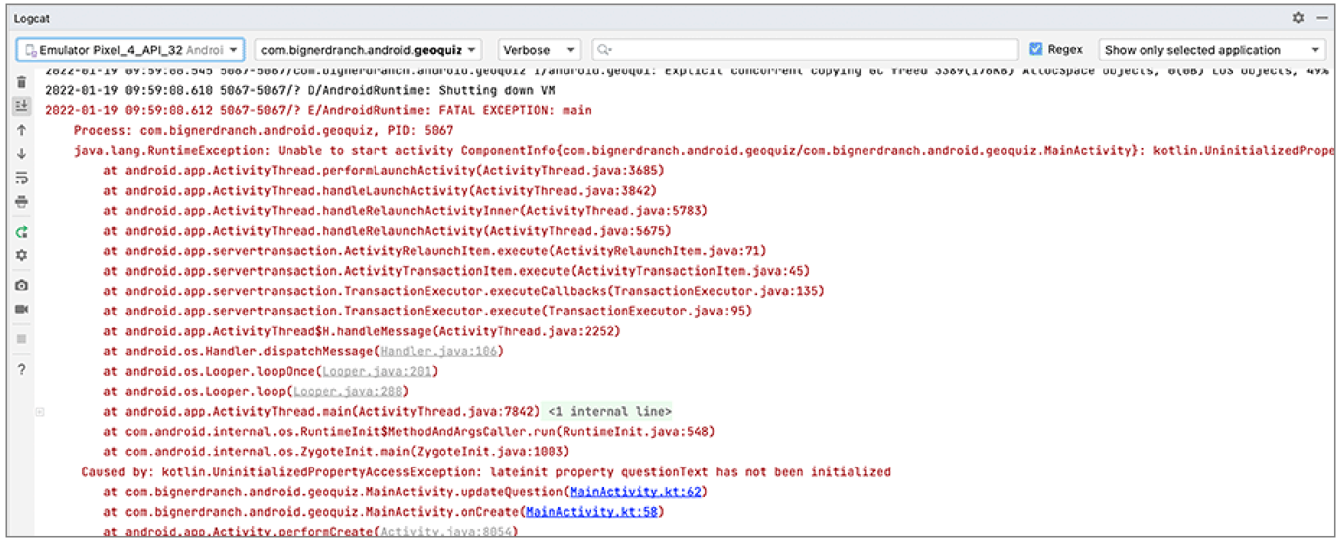
Device Frame — ✓ Enable Device Frame

Show Advanced Settings

AVD Name

The name of this AVD.

Configure emulator


Logcat

Emulator Pixel_4_API_32 Androi — com.bignerdranch.android.geoquiz — Verbose — ✓ Regex — Show only selected application

```
2022-01-19 09:59:08.545 5067-5067/com.bignerdranch.android.geoquiz I/android.geoqui: Explicit concurrent copying GC freed 3389(176KB) AllocSpace objects, 0(0B) LOS objects, 49%
2022-01-19 09:59:08.610 5067-5067/? D/AndroidRuntime: Shutting down VM
2022-01-19 09:59:08.612 5067-5067/? E/AndroidRuntime: FATAL EXCEPTION: main
    Process: com.bignerdranch.android.geoquiz, PID: 5067
    java.lang.RuntimeException: Unable to start activity ComponentInfo{com.bignerdranch.android.geoquiz/com.bignerdranch.android.geoquiz.MainActivity}: kotlin.UninitializedPrope
        at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:3685)
        at android.app.ActivityThread.handleLaunchActivity(ActivityThread.java:3842)
        at android.app.ActivityThread.handleRelaunchActivityInner(ActivityThread.java:5783)
        at android.app.ActivityThread.handleRelaunchActivity(ActivityThread.java:5675)
        at android.app.servertransaction.ActivityRelaunchItem.execute(ActivityRelaunchItem.java:71)
        at android.app.servertransaction.ActivityTransactionItem.execute(ActivityTransactionItem.java:45)
        at android.app.servertransaction.TransactionExecutor.executeCallbacks(TransactionExecutor.java:135)
        at android.app.servertransaction.TransactionExecutor.execute(TransactionExecutor.java:95)
        at android.app.ActivityThread$H.handleMessage(ActivityThread.java:2252)
        at android.os.Handler.dispatchMessage(Handler.java:106)
        at android.os.Looper.loopOnce(Looper.java:201)
        at android.os.Looper.loop(Looper.java:288)
        at android.app.ActivityThread.main(ActivityThread.java:7842) <1 internal line>
        at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run(RuntimeInit.java:548)
        at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:1003)
     Caused by: kotlin.UninitializedPropertyAccessException: lateinit property questionText has not been initialized
        at com.bignerdranch.android.geoquiz.MainActivity.updateQuestion(MainActivity.kt:62)
        at com.bignerdranch.android.geoquiz.MainActivity.onCreate(MainActivity.kt:58)
        at android.app.Activity.performCreate(Activity.java:8054)
```

Run Code, output in LogCat window

Data-Driven Layouts

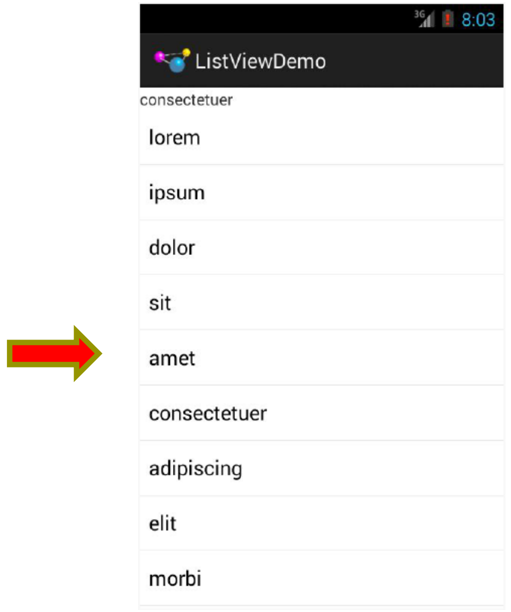
Data-Driven Layouts

- LinearLayout, RelativeLayout, TableLayout, GridLayout useful for positioning UI elements
  - UI data is **hard coded**

- Other layouts dynamically composed from data (e.g. database)
  - ListView, GridView, GalleryView
  - Tabs with TabHost, TabControl

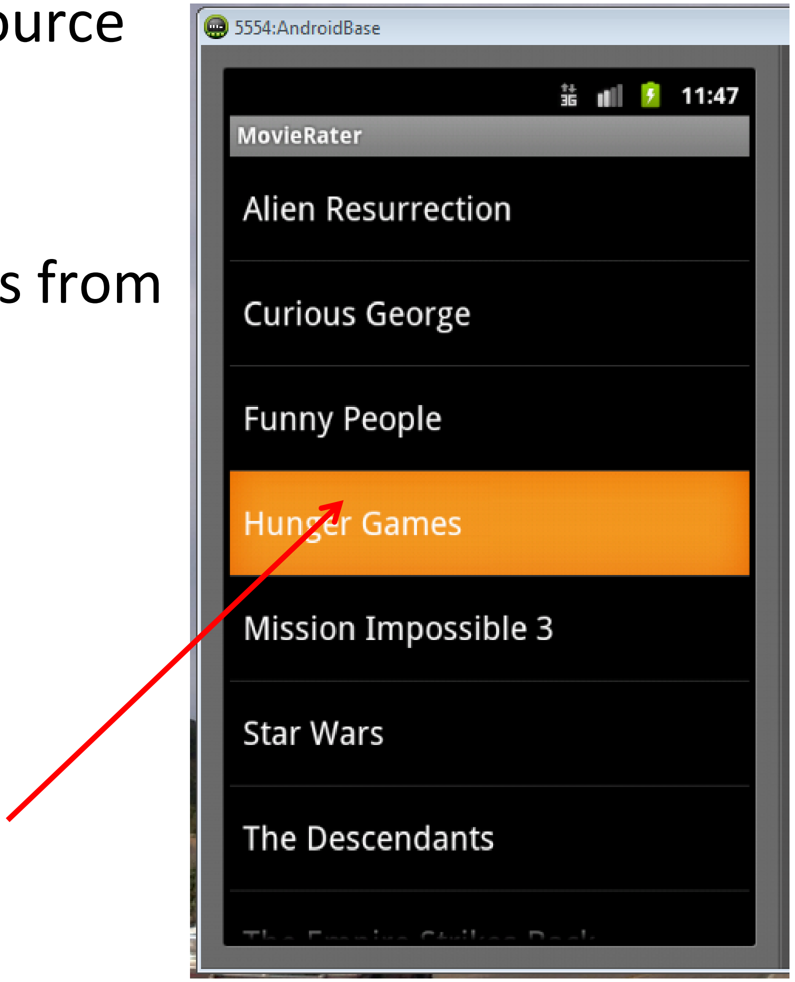
Generate widgets from data source

lorem
ipsum
dolor
amet
consectetuer
adipiscing
elit
morbi



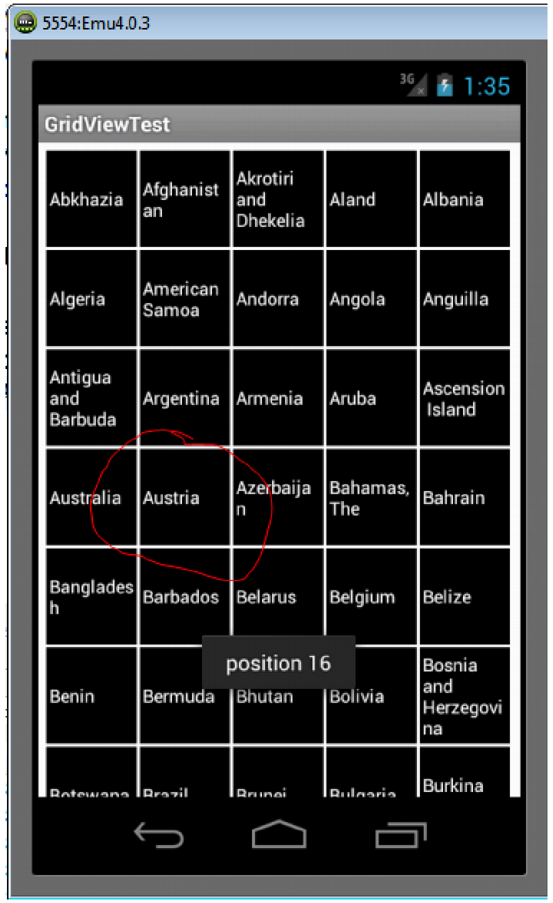
Data Driven Layouts

- May want to populate views from a data source (XML file or database)
- Layouts that display repetitive child widgets from data source
  - ListView
  - GridView
  - GalleryView
- ListView
  - Rows of entries, pick item, vertical scroll

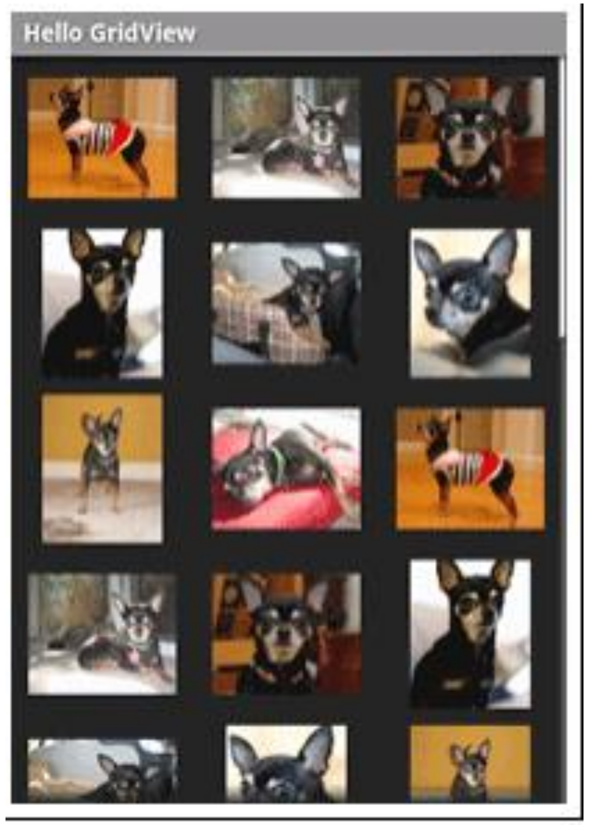


Data Driven Containers

- GridView
  - List of items arranged in rows and columns
- GalleryView
  - List with horizontal scrolling, typically images

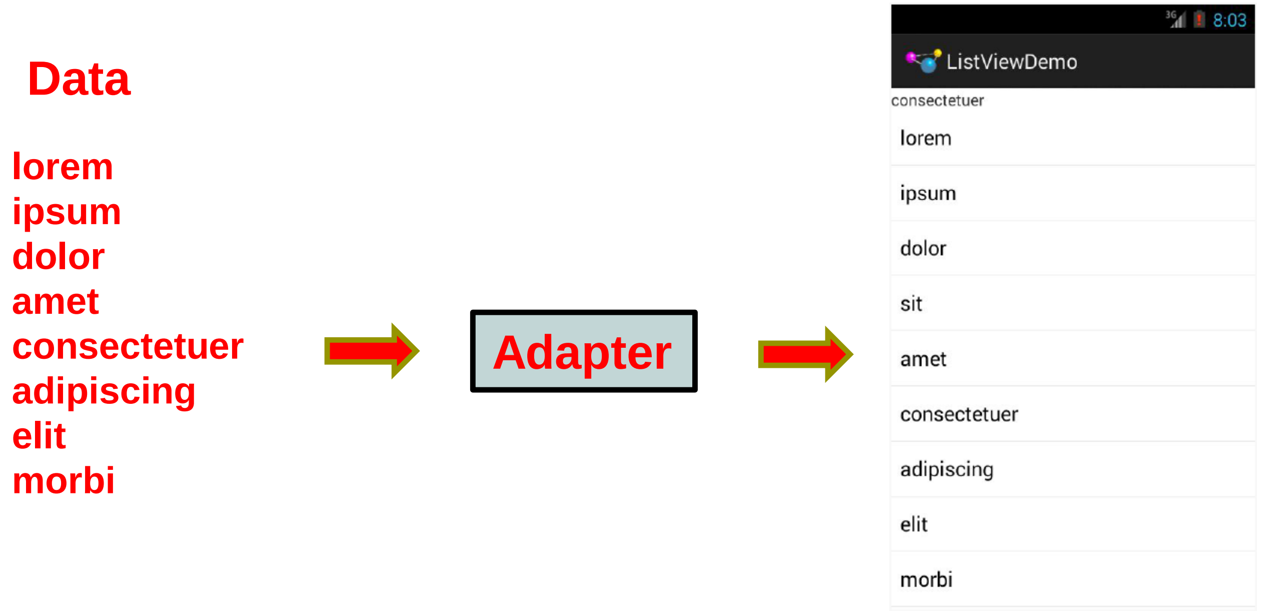


Abkhazia	Afghanistan	Akrotiri and Dhekelia	Aland	Albania
Algeria	American Samoa	Andorra	Angola	Anguilla
Antigua and Barbuda	Argentina	Armenia	Aruba	Ascension Island
Australia	Austria	Azerbaijan	Bahamas, The	Bahrain
Bangladesh	Barbados	Belarus	Belgium	Belize
Benin	Bermuda	Bhutan	Bolivia	Bosnia and Herzegovina
Botswana	Brazil	Brunei	Bulgaria	Burkina



AdapterView

- ListView, GridView, and GalleryView are sub classes of AdapterView (variants)
- **Adapter:** generates widgets from a data source, populates layout
  - E.g., Data is adapted into cells of ListView



- Most common Adapter types:
  - **CursorAdapter:** read from database
  - **ArrayAdapter:** read from resource (e.g., XML file)

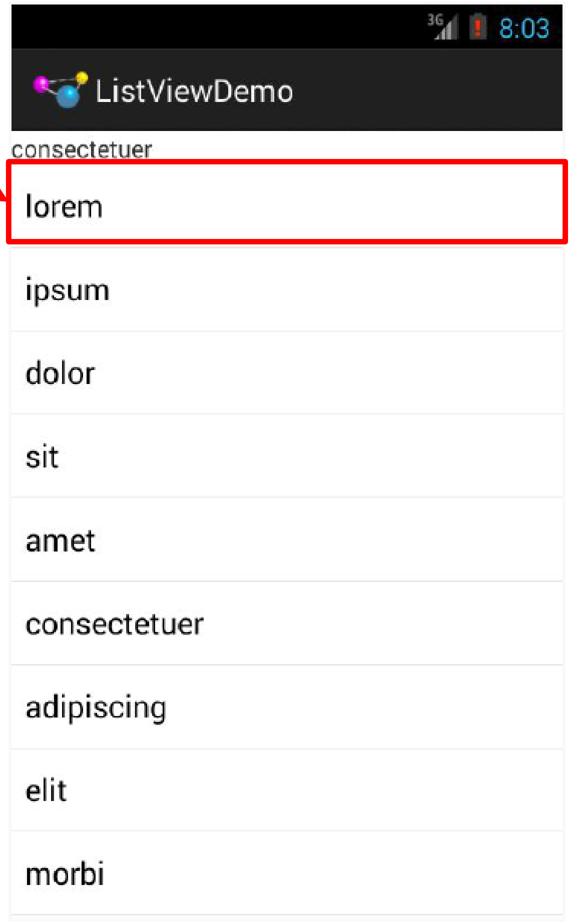
Adapters

- When using Adapter, a layout (XML format) is defined for each child element (View)

- The adapter
  - Reads in data (list of items)
  - Creates Views (widgets) using layout for each element in data source
  - Fills the containing layout (List, Grid, Gallery) with the created Views

- Child widgets can be as simple as a TextView or more complex layouts / controls
  - simple views can be declared in a layout XML file

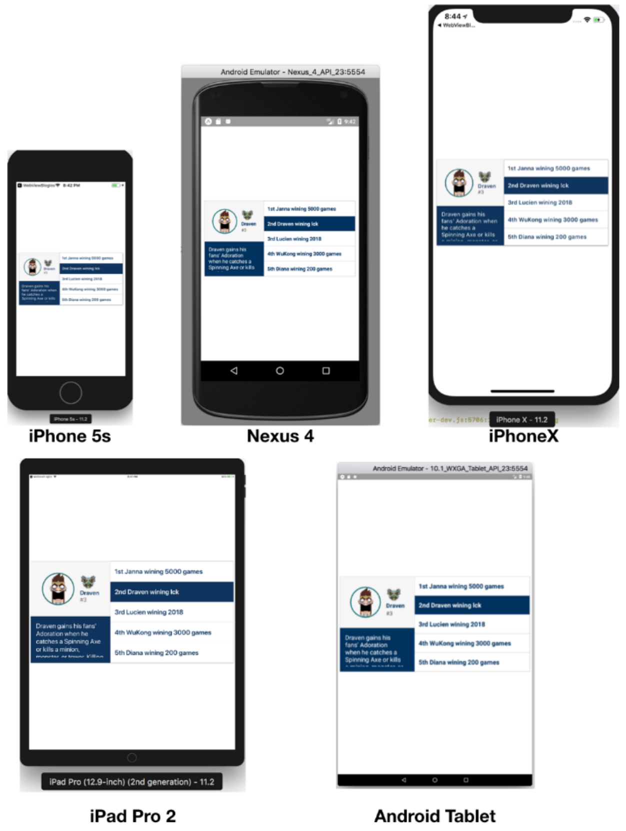
    (e.g. android.R.layout)



Mobile HCI

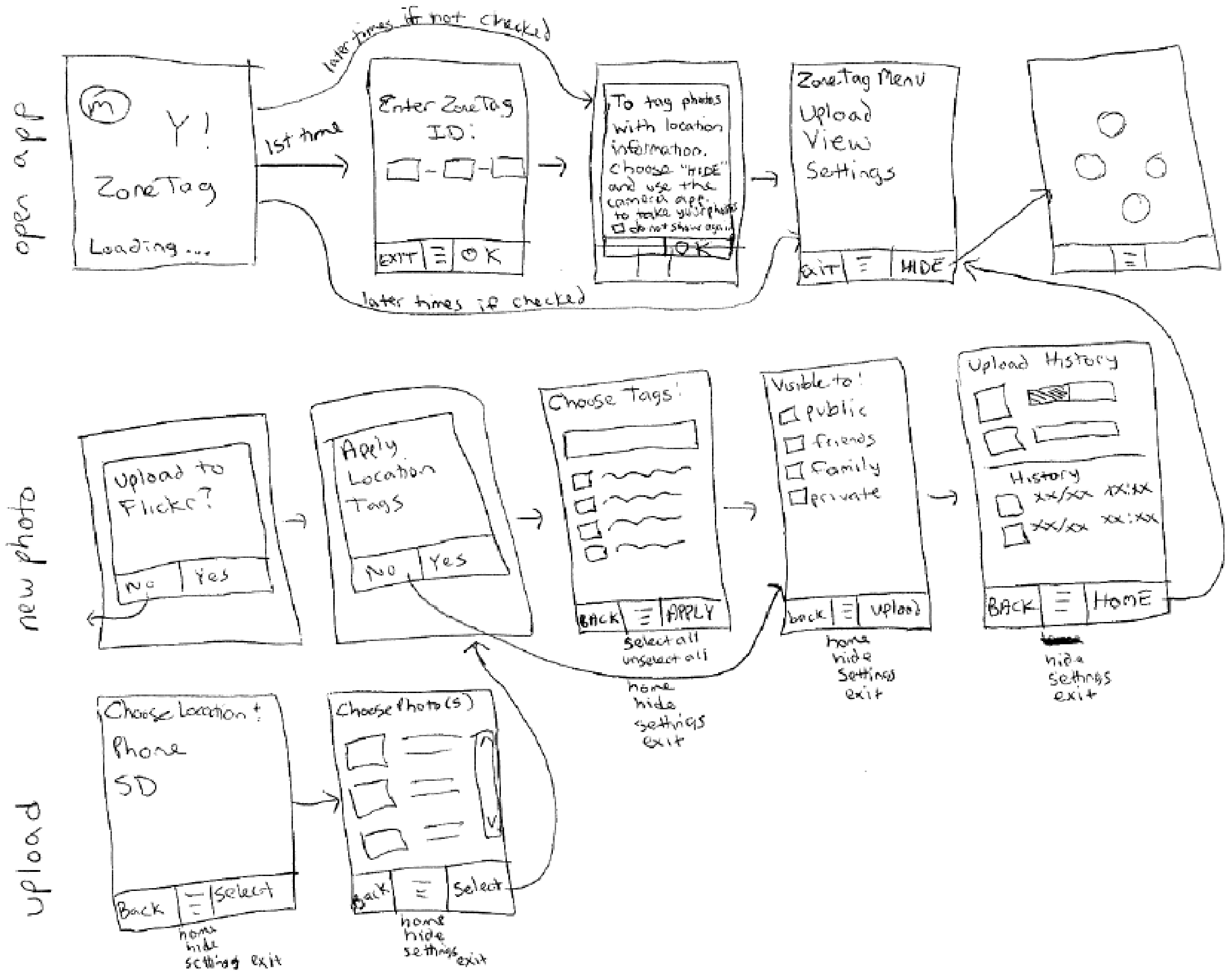
Mobile HCI

- Mobile HCI is important for an enjoyable user experience
- Excerpts from:
  - Bentley, F. and Barrett, E., 2012. *Building mobile experiences*. MIT Press.
- Can’t just reuse screens originally designed for desktops. Why?
  1. Mobile screen is small, need to manage space better
  2. Does your screen look good on wide variety of mobile screen sizes?
  3. Can users reach buttons with one hand on different resolutions?
  4. Mobile device will be carried into varied, adverse conditions. E.g.
     1. Do colors work well indoor vs outdoor, bright vs dim light
     2. Are buttons big enough for frozen hands during winter vs summer



Mobile HCI: Plan out Interaction Flow on Paper

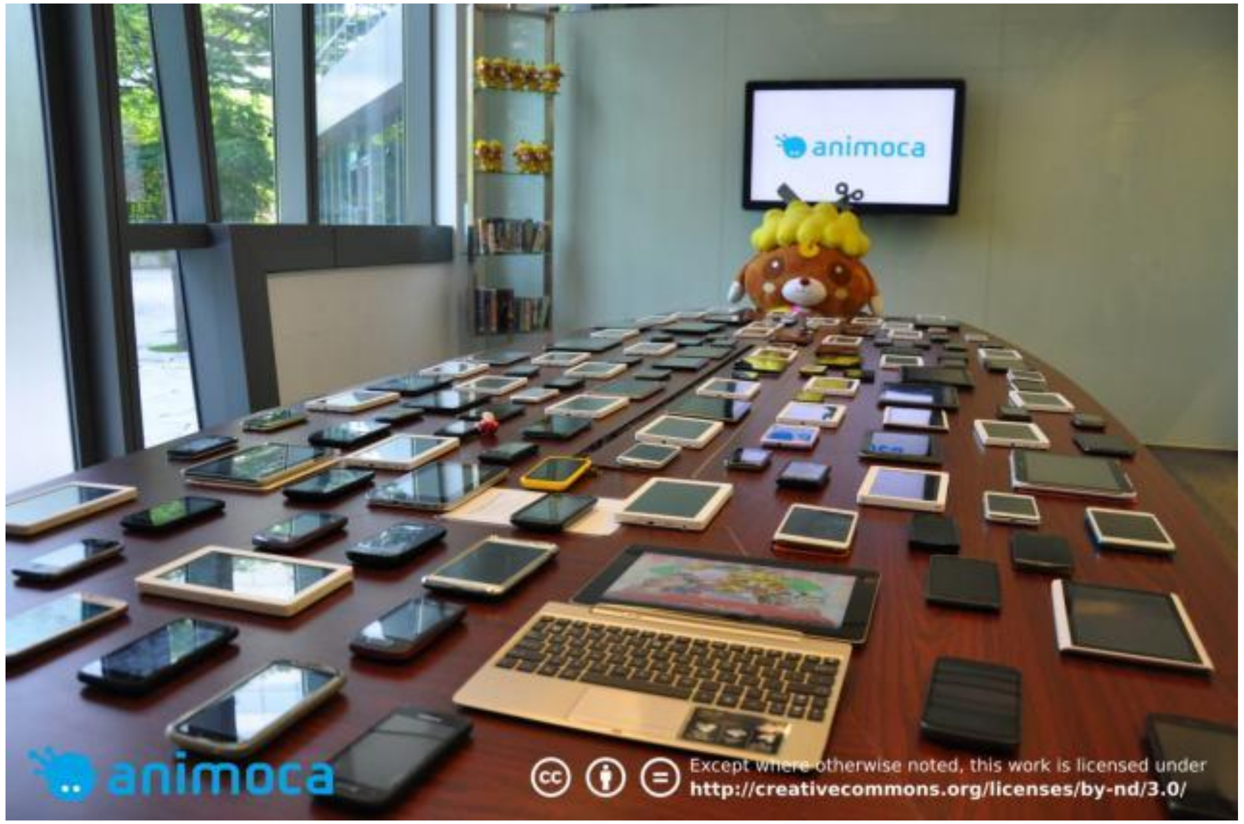
- Example interaction flow of ZoneTag app on paper
  - Ref: Bentley, F. and Barrett, E., 2012. *Building mobile experiences*. MIT Press.



Mobile HCI: Evaluation

- App evaluation: iterative, user-centered
  - In lab (small) then in the field (large)
  - Test on on wide variety of devices
    - Most poor ratings on Google Play app store are "doesn't work on my device"
- Example: Android mobile developer tests each game on over 400 different smartphones and tablets
  - Screens
  - Aspect ratios
  - Form factors
  - Controls
  - OS versions
  - CPU/GPU
  - OpenGL/DirectX

  versions……. etc


animoca
animoca
Except where otherwise noted, this work is licensed under http://creativecommons.org/licenses/by-nd/3.0/

Android UI Youtube Tutorials
by Bucky Roberts theNewBoston
Gentle but a bit Old?

Tutorials from YouTube Android Development Tutorials 1-8 by Bucky Roberts (Completely Optional)

- **Tutorials 1 & 2 (Optional):** Installing Java, Android Studio on your own machine
  - **Tutorial 1:** Install Java (Android studio needs this at least ver. 1.8)
  - **Tutorial 2:** Install Android Studio
- **Tutorial 3:** Setting up your project
  - How to set up a new Android Project, add new Activity (App screen)
- **Tutorial 4:** Running a Simple App
  - How to select, run app on a virtual device (AVD)
- **Tutorial 5:** Tour of Android Studio Interface
  - Intro to Android Studio menus, toolbars and Drag-and-drop widget palette

Review this tutorial only if you feel you need gentler intro
Note: It’s in java, not Kotlin

YouTube Tutorial 11 & 12 by Bucky Roberts

- Tutorial 11: Designing the User Interface [6:19 mins]
  - https://www.youtube.com/watch?v=72mf0rmjNAA
  - Designing the UI
  - Adding activity (screen)
  - Dragging in widgets
  - Changing the text in widgets

- Tutorial 12: More on User Interface [10:24 mins]
  - https://www.youtube.com/watch?v=72mf0rmjNAA
  - Changing text in widgets
  - Changing strings from hardcoded to string resources (variables)

EML: Cooperative Based Groups

EML: Cooperative Based Groups

- Japanese students visiting Boston for 2 week vacation
- Speak little English, need help to find
  - Attractions to visit, where to stay (cheap, central), meet Americans, getting around, eat (Japanese, some Boston food), weather info, events, ….. Anything
  - **New!:** One of them is worried they have COVID. What apps could help. E.g. minimize risk of infection? determine if positive? Find nearest hospital? Testing center? Buy masks/PPE?
- Your task: Search android market for helpful apps (6 mins)
  - **Location-aware:** 5 points
  - **Ubicomp (e.g. uses sensor) or smartwatch:** 10 points
- Also **IoT** devices they can buy that would help them (5 points)

References

- Android App Development for Beginners videos by Bucky Roberts (thenewboston)
- Head First Android, 2nd and 3rd edition
- Android Nerd Ranch, Fifth Edition
- Ask A Dev, Android Wear: What Developers Need to Know, https://www.youtube.com/watch?v=zTS2NZpLyQg
- Ask A Dev, Mobile Minute: What to (Android) Wear, https://www.youtube.com/watch?v=n5Yjzn3b_aQ