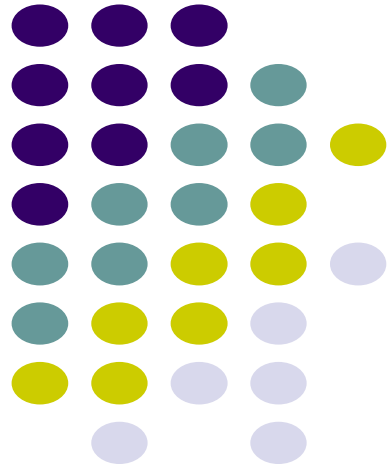
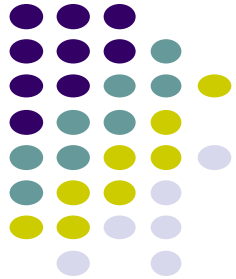
CS 528 Mobile and Ubiquitous Computing
Lecture 02b: Android UI Design

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

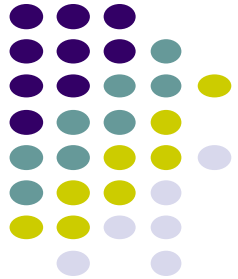
String declaration in strings.xml

```
<?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 your 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

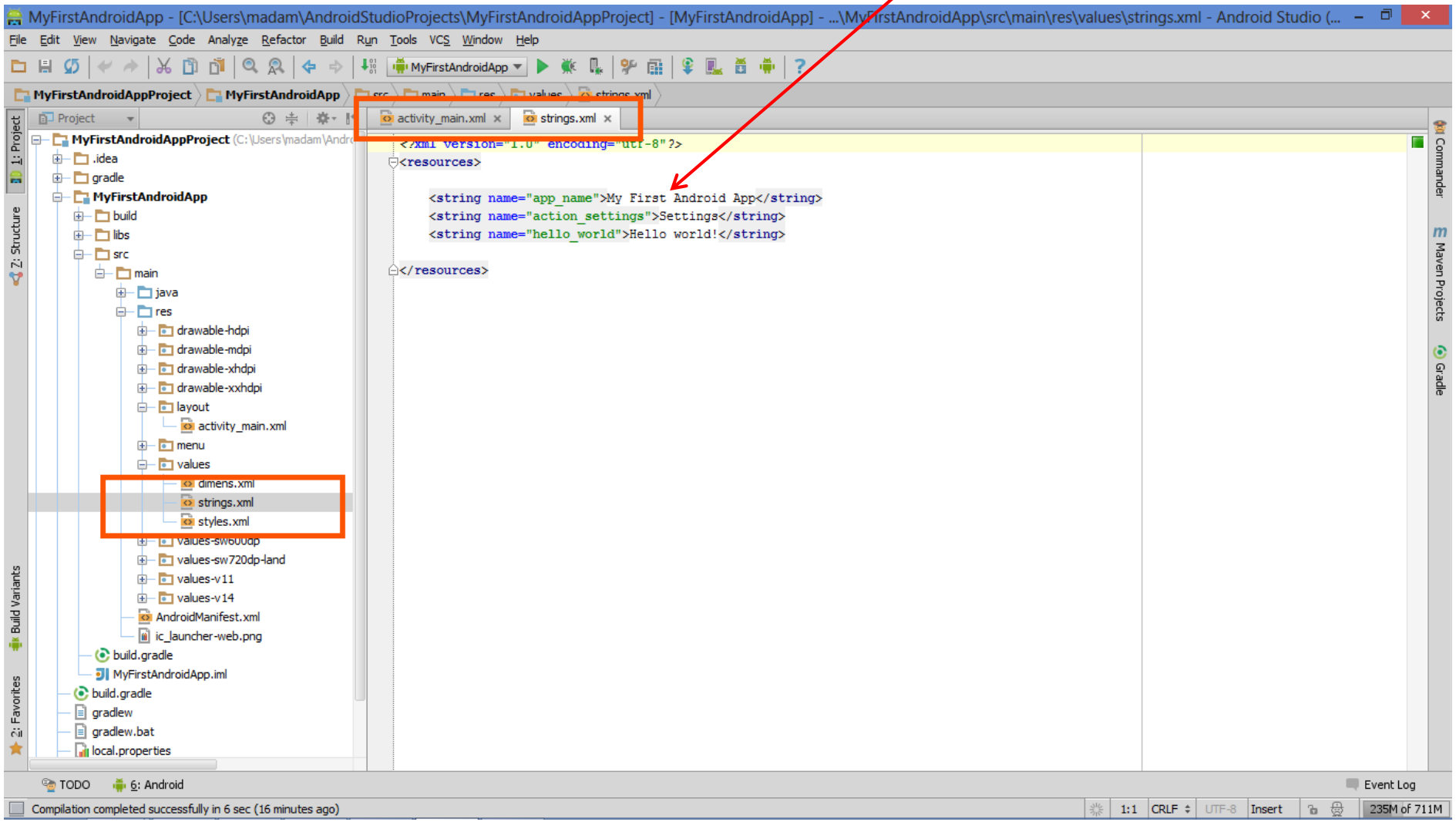
```
<?xml version="1.0" encoding="utf-8"?>
<resources>

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

</resources>
```

String usage in AndroidManifest.xml

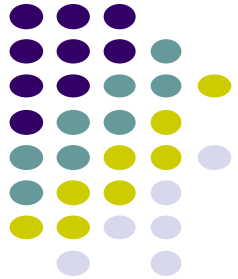
```
  <application
    android:allowBackup="false"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme">
    <activity
      android:name="EmPubLiteActivity"
      android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN"/>

        <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



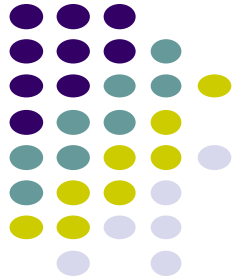
Styled Text



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

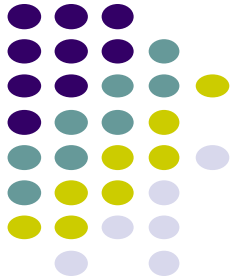
- Can use the same HTML tags to add style (italics, bold, etc) to Android strings

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



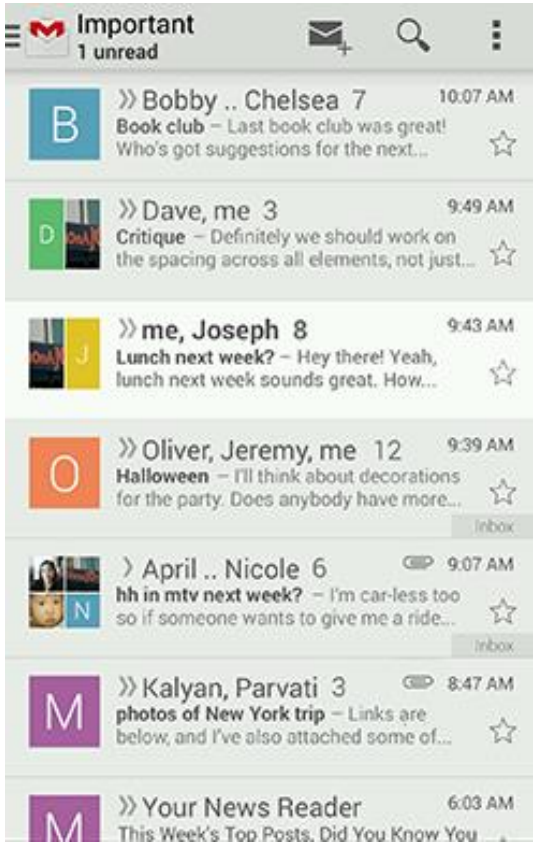
Android Themes

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 in an Activity (screen)**
  - E.g. all widgets on a screen can adopt the same font
- Example Android themes: Theme, Theme.holo and Theme.material)

Examples of Themes in Use

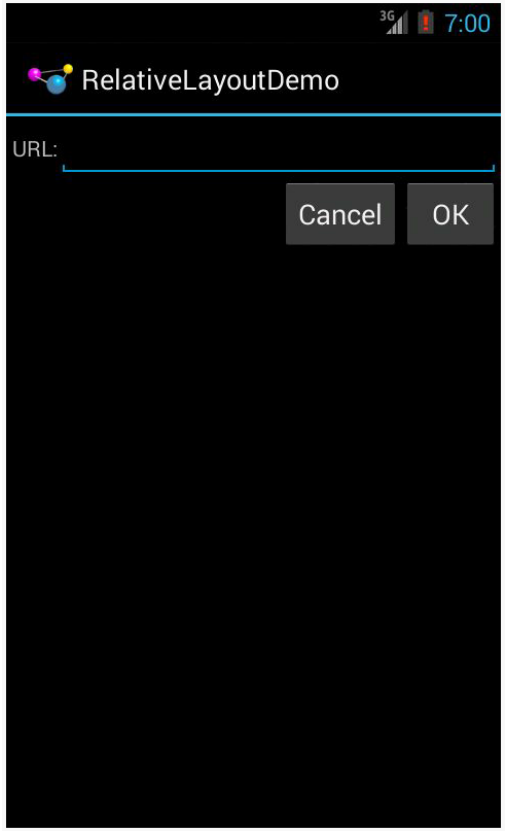


GMAIL in Holo Light

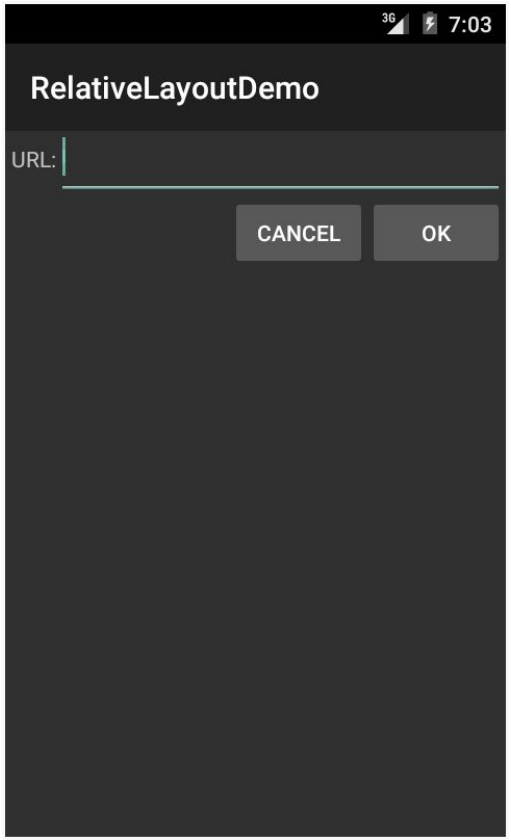
Settings screen in Holo Dark

Default Themes

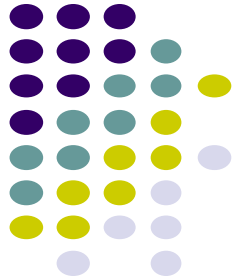
- Many pre-defined themes to choose from
- Android chooses a default theme if you specify none



Theme.Holo: default theme in Android 3.0



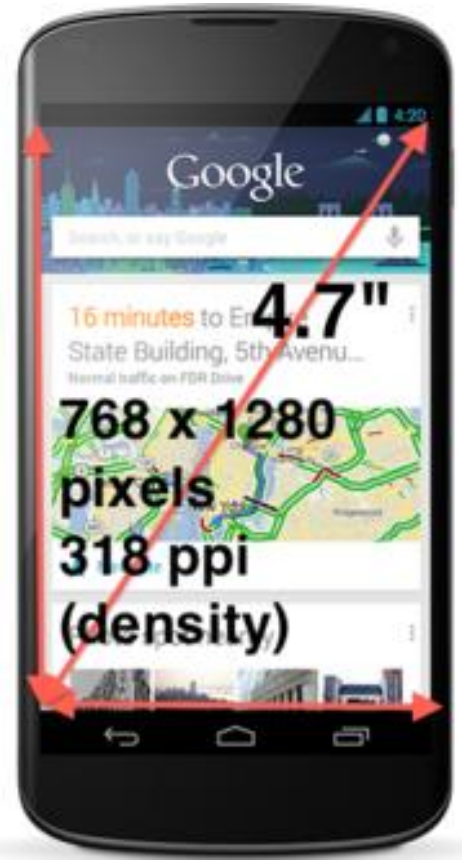
Theme.Material: default theme in Android 5.0



Adding Pictures in Android

Phone Dimensions Used in Android UI

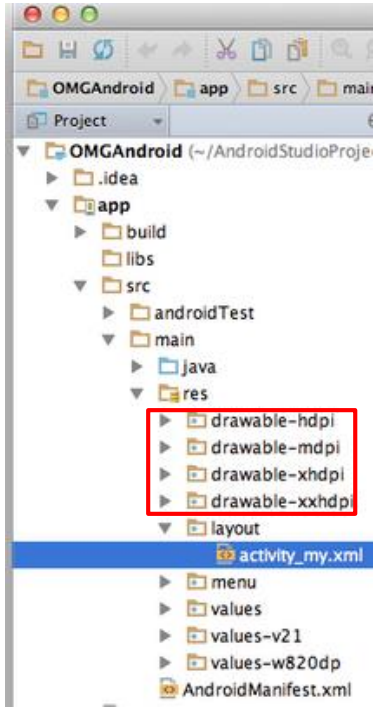
- Physical dimensions (inches) diagonally
  - E.g. Nexus 4 is 4.7 inches diagonally
- Resolution in pixels
  - E.g. Nexus 4 resolution 768 x 1280 pixels
  - Pixels diagonally: Sqrt[(768 x 768) + (1280 x 1280)]
- Pixels per inch (PPI) =
  - Sqrt[(768 x 768) + (1280 x 1280)] / 4.7= 318



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



Adding Pictures

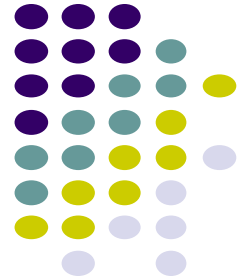
- Use generic picture name in code (no .png, .jpg, etc)
  - E.g. to reference an image **ic_launcher.png**

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

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

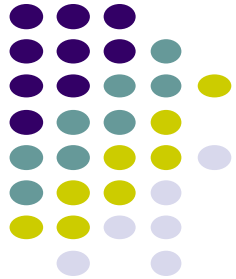
- **Image Asset Studio:** generates icons in various densities from original image

  Ref: https://developer.android.com/studio/write/image-asset-studio.html



Android UI Youtube Tutorials

YouTube Tutorial 11 & 12 from thenewBoston

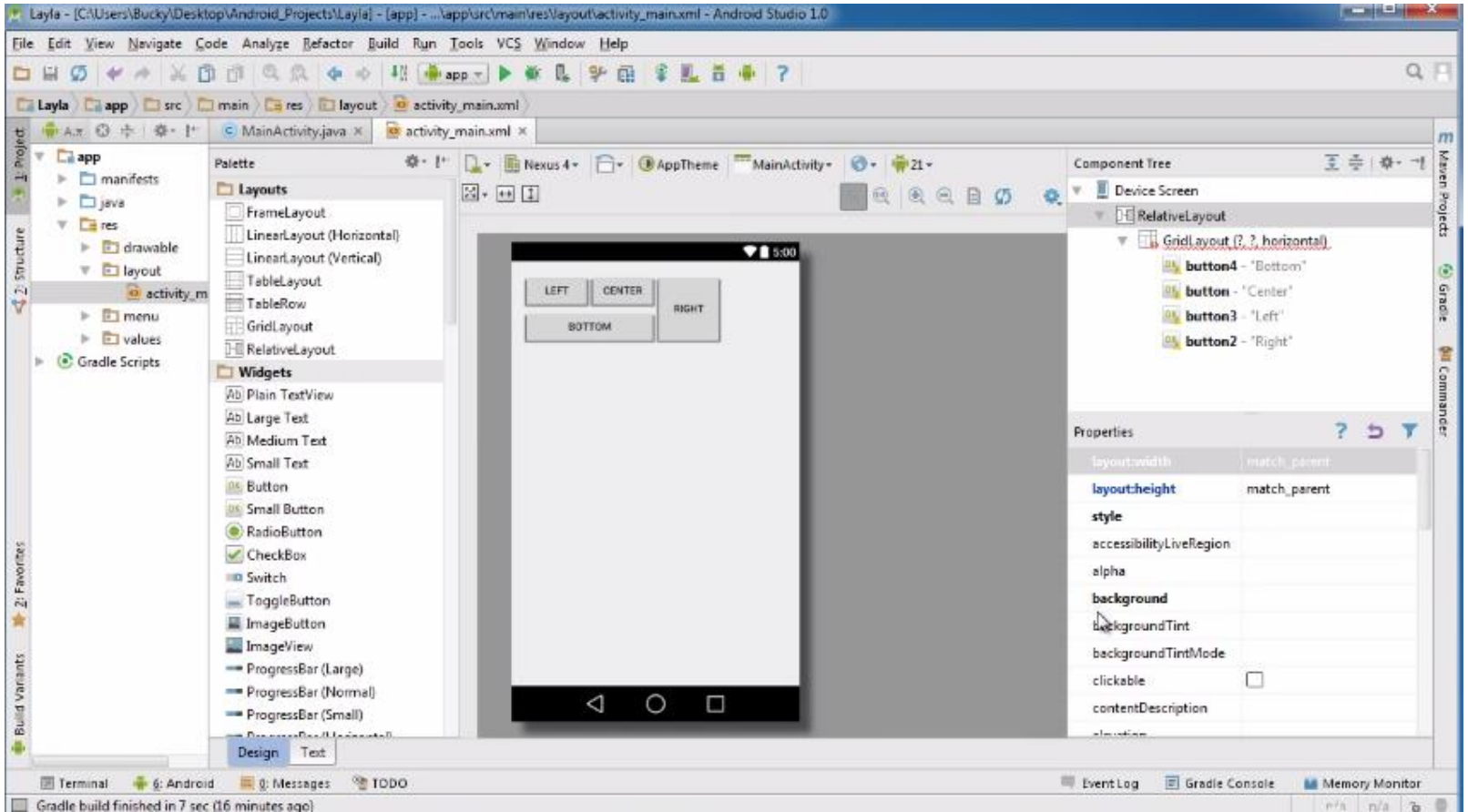


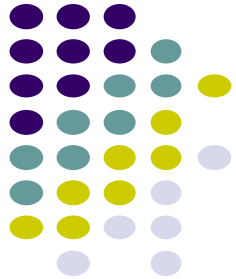
- Tutorial 11: Designing the User Interface [6:19 mins]
  - https://www.youtube.com/watch?v=72mf0rmjNAA
  - Designing the UI
  - Adding activity (screen)
  - Dragging in widgets
  - Changing the text in widgets

- 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)

Tutorial 17: GridLayout

- Tutorial 17: GridLayout [9:40 mins]
  (https://www.youtube.com/watch?v=4bXOr5Rk1dk
  - Creating GridLayout: Layout that places its children in a grid
  - Add widgets (buttons) to GridLayout
  - Format width, height, position of widgets

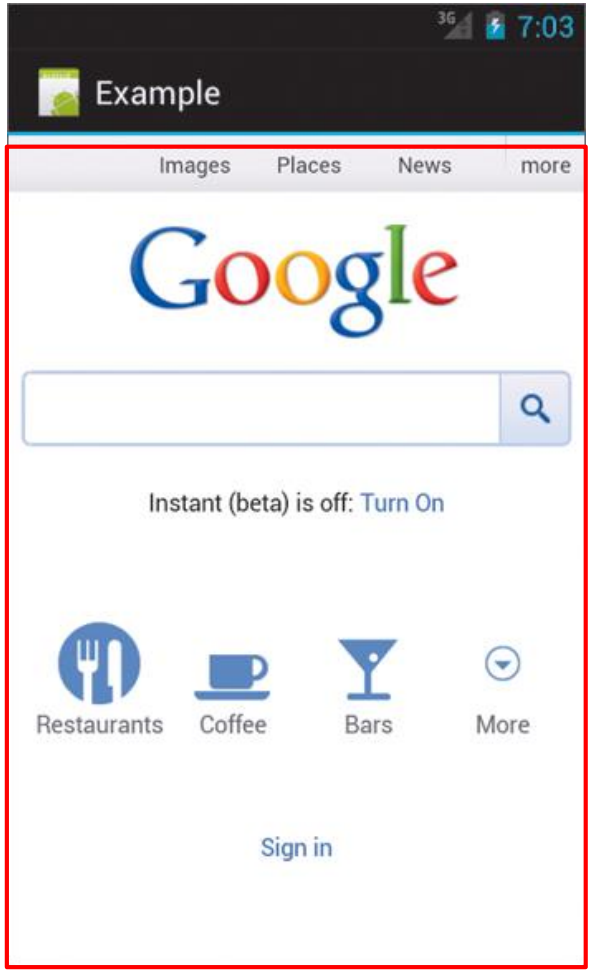




WebView Widget

WebView Widget

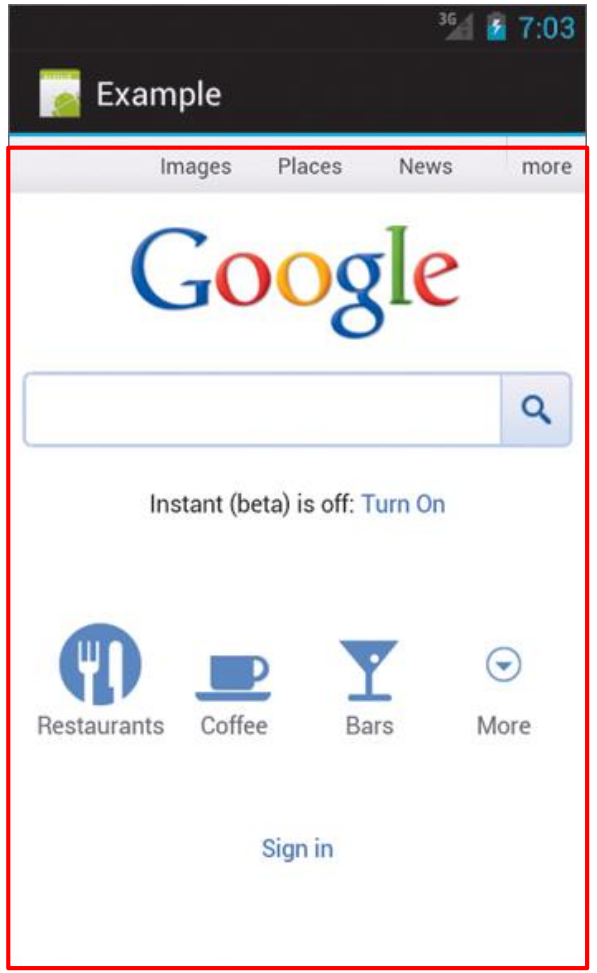
- A View that displays web pages
  - Can be used for creating your own web browser
  - OR just display some online content inside your app



WebView Widget

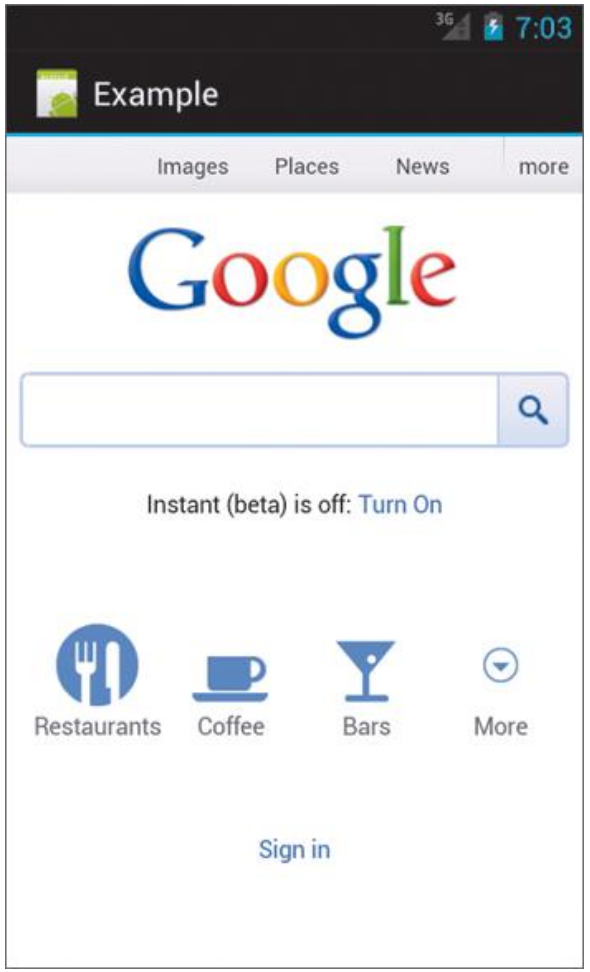
- Since Android 4.4, webviews rendered using:
  - Chromium open source project, engine used in Google Chrome browser (http://www.chromium.org/)

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

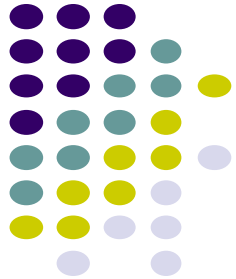


WebView Widget Functionality

- 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



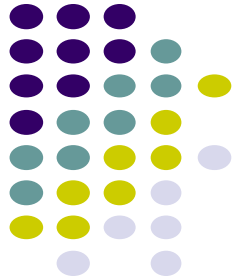
WebView Example



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

```
<?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



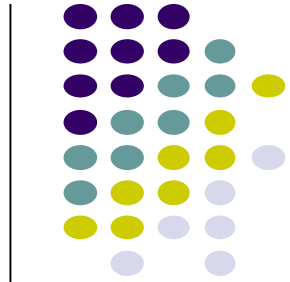
- In onCreate, use loadURL to specify website to load
- If website contains Javascript, enable Javascript
- loadUrl( ) can also load files on Android local filesystem (file://)

```
public class HelloWebView extends Activity {

    private WebView mWebView;

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

        mWebView = (WebView) findViewById(R.id.webview);
        mWebView.getSettings().setJavaScriptEnabled(true);
        mWebView.loadUrl("http://m.utexas.edu");
    }
}
```

WebView: Request Internet Access

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

```
<?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 UI Design Example

GeoQuiz App

Ref: Android Nerd Ranch (3rd edition), pgs 1-30

- App presents questions to test user's knowledge of geography
- User answers by pressing **True** or **False** buttons
- How to get this book?

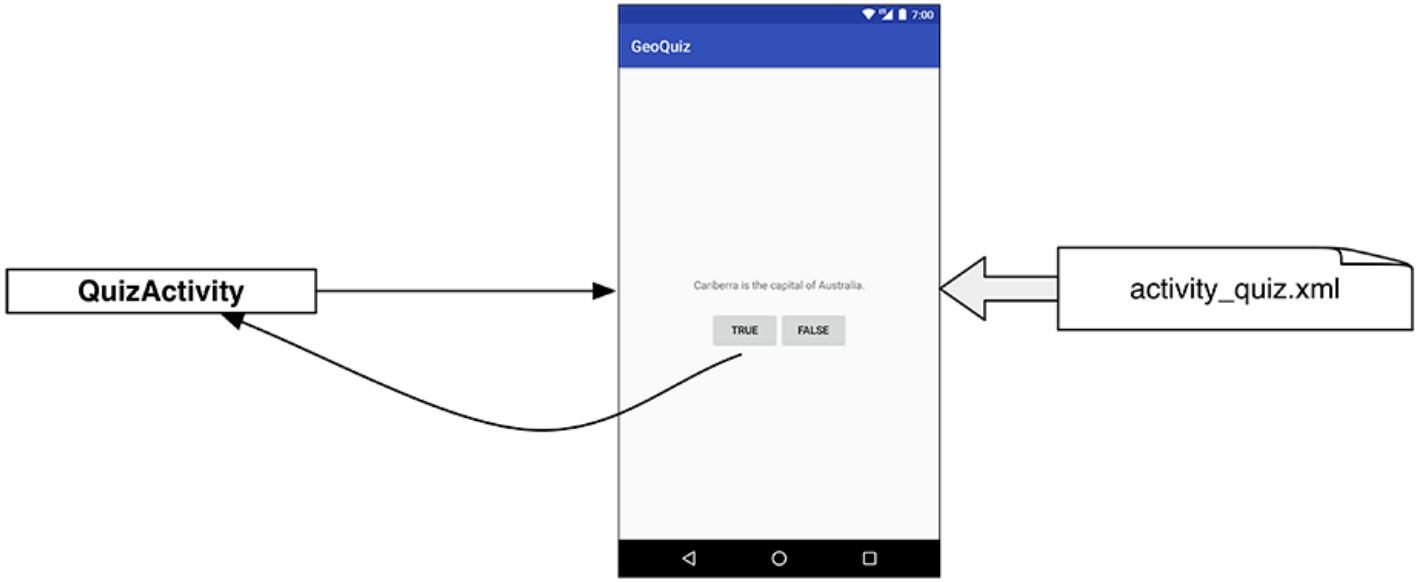
Question

User responds by clicking True or False

App grades User's response Using toast msg

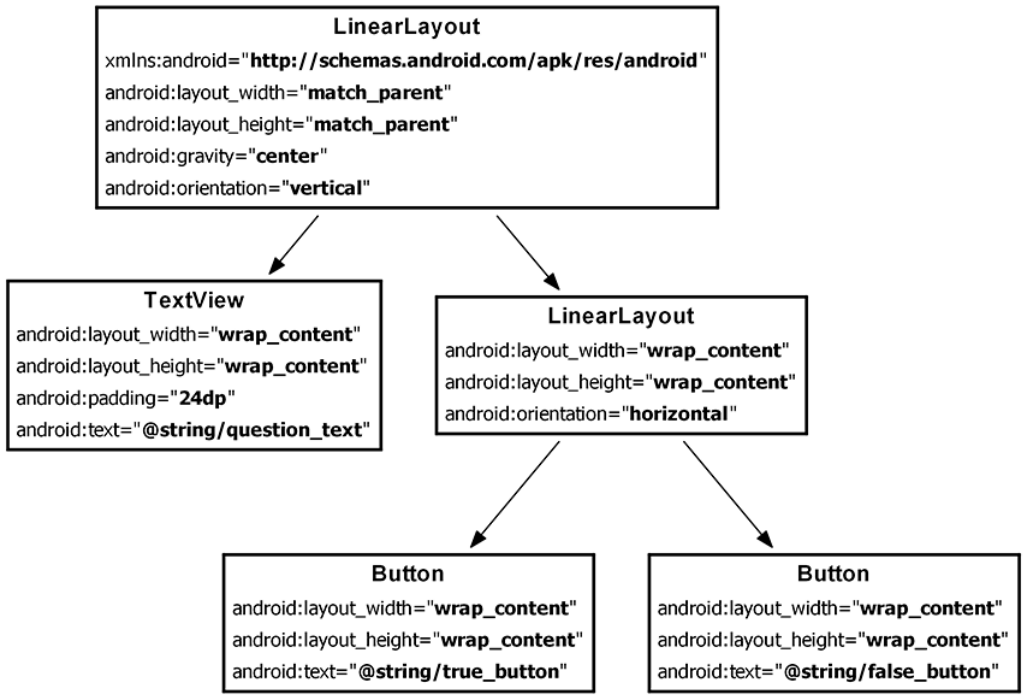
GeoQuiz App

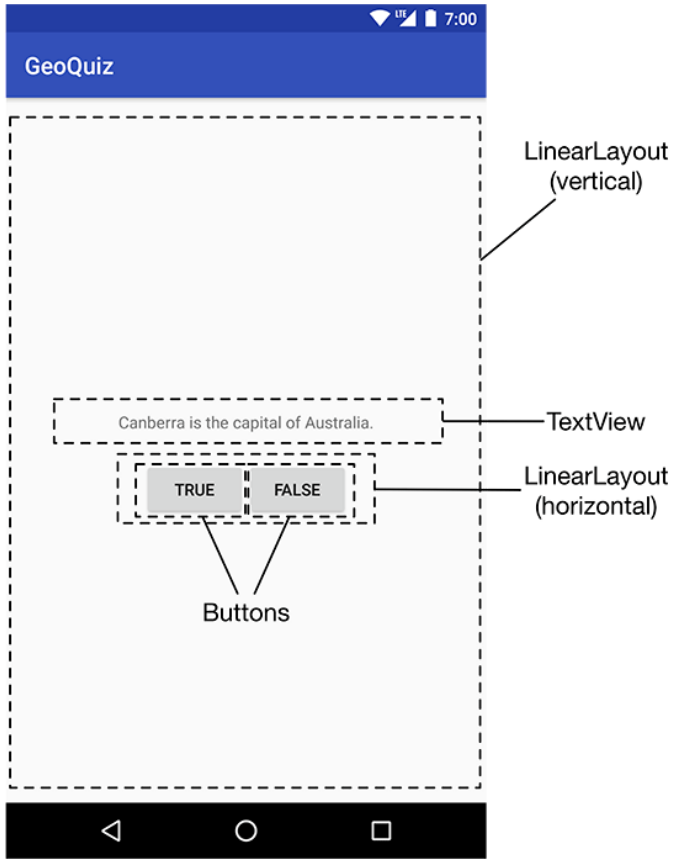
- 2 main files:
  - **activity_quiz.xml:** to format app screen
  - **QuizActivity.java:** To present question, accept True/False response
- **AndroidManifest.xml** lists all app components, auto-generated



GeoQuiz: Plan Out App Widgets

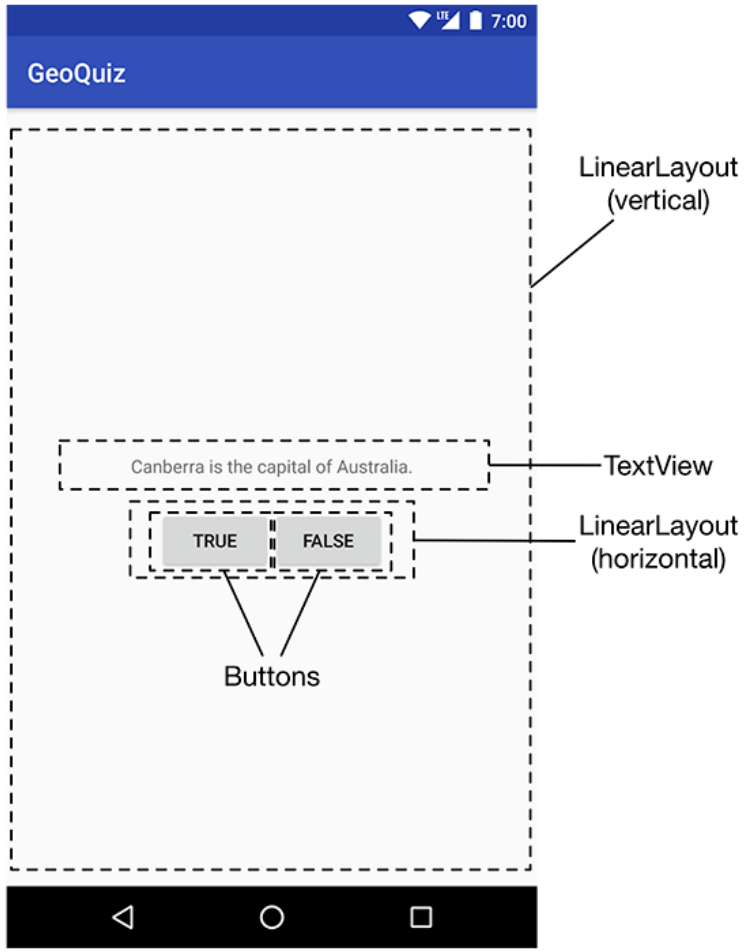
- 5 Widgets arranged hierarchically





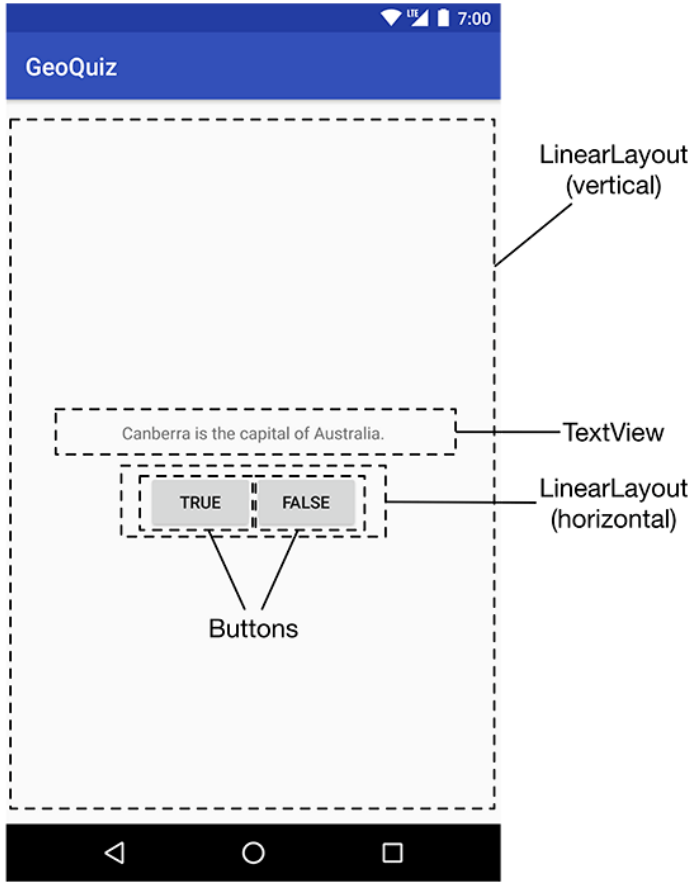
GeoQuiz: activity_quiz.xml File listing

```
<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>
```



GeoQuiz: strings.xml File listing

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



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>
```

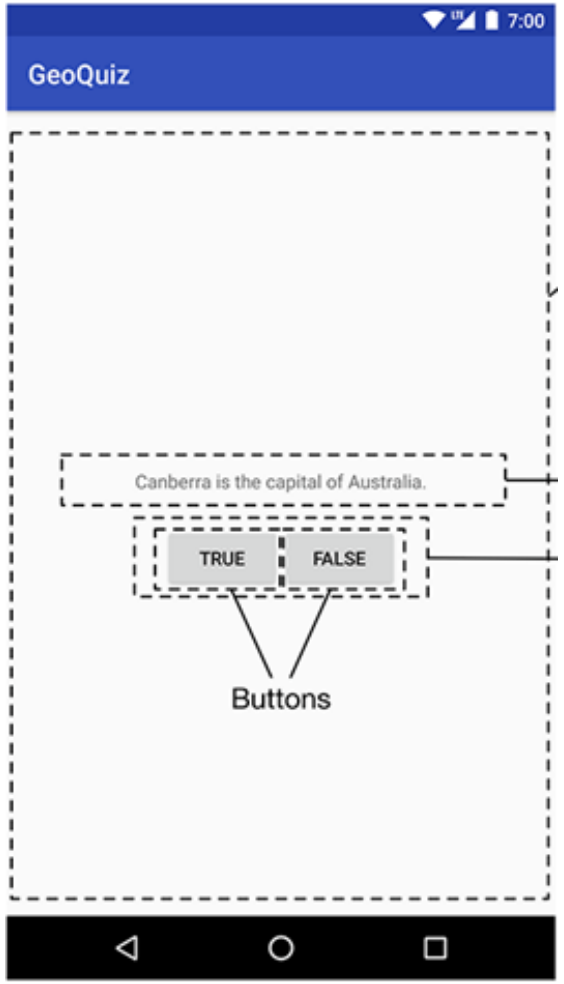
QuizActivity.java

- Initial QuizActivity.java code

```
package com.bignerdranch.android.geoquiz;

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

public class QuizActivity extends Activity {

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

onCreate Method is called once Activity is created

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

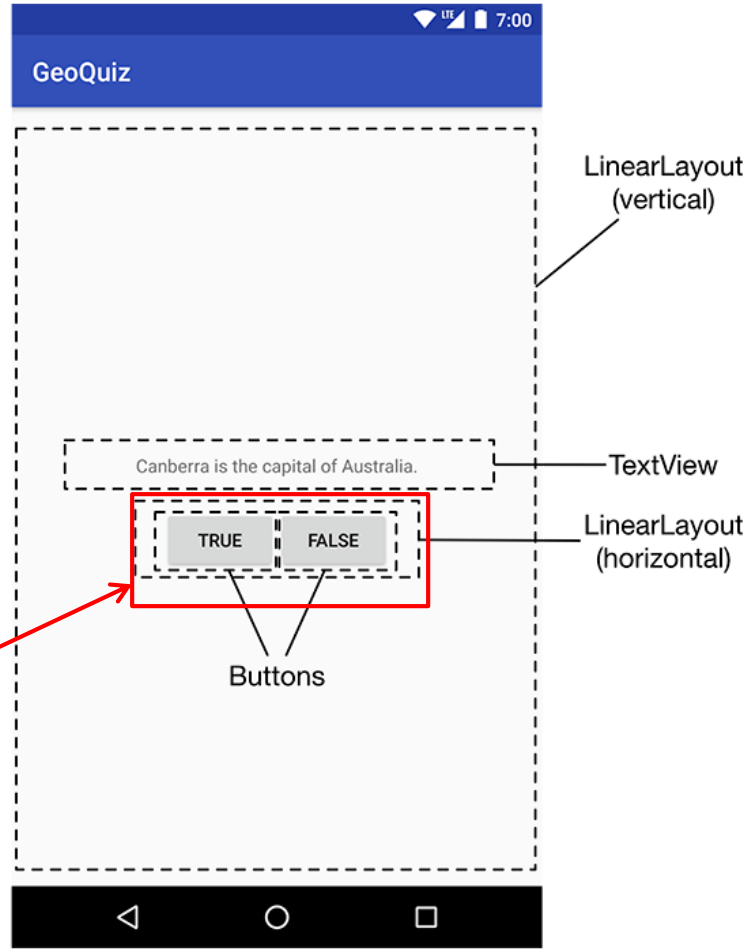


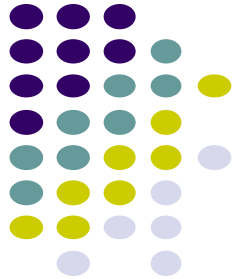
- Would like java code to respond to True/False buttons being clicked

Responding to True/False Buttons in Java

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
... >

  <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 Java file to specify app's response when True/False buttons are clicked



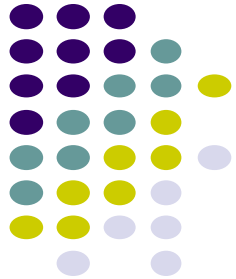


2 Alternative Ways to Respond to Button Clicks

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

2. In java create a ClickListener object, override onClick method
   - typically done with anonymous inner class

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 Java file (e.g. MainActivity.java) declare method/handler to take desired action**

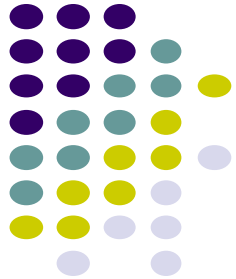
Activity_my.xml

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

MainActivity.java

```
public void someMethod(View theButton) {
  // do something useful here
}
```

Approach 2: Create a ClickListener object, override onClick

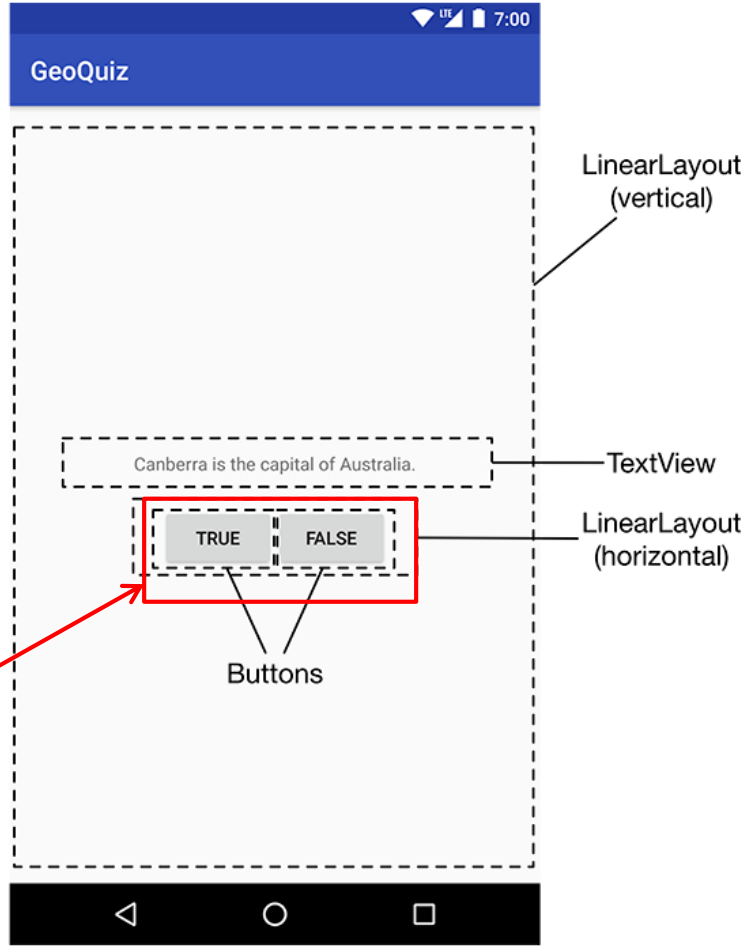


- First, get reference to Button in our Java file. How?

```
<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



QuizActivity.java: Getting References to Buttons

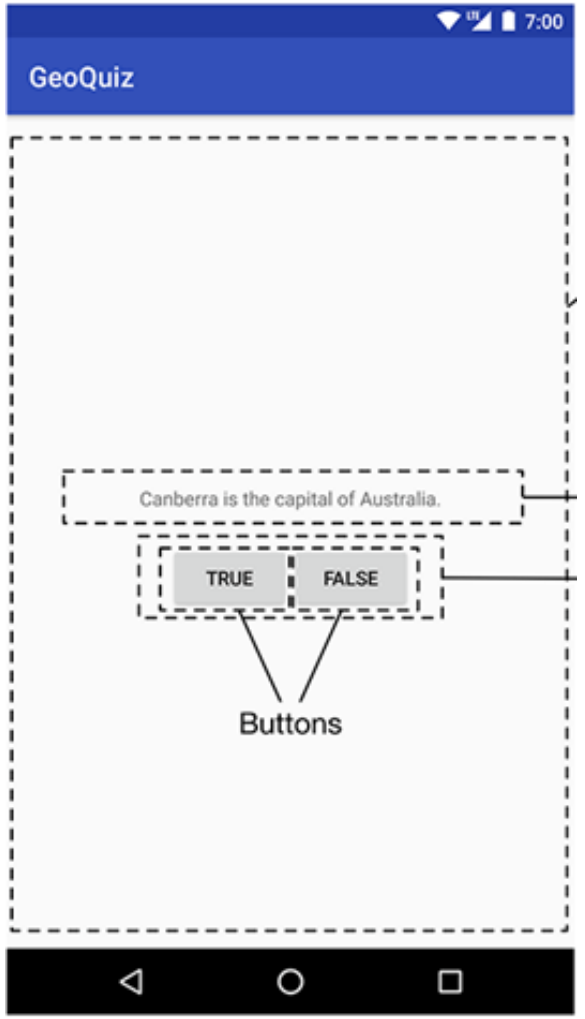
- To get reference to buttons in java code

```
public class QuizActivity extends Activity {

    private Button mTrueButton;
    private Button mFalseButton;

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

        mTrueButton = (Button)findViewById(R.id.true_button);
        mFalseButton = (Button)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" />
```



QuizActivity.java: Setting Listeners

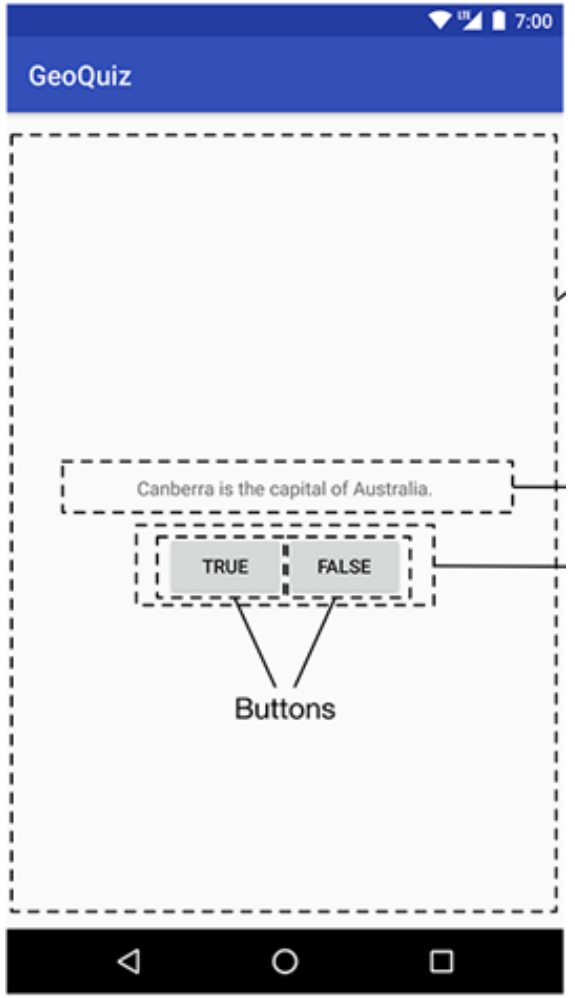
- Set listeners for **True** and **False** button

```
...
    mTrueButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // Does nothing yet, but soon!
        }
    });

    mFalseButton = (Button)findViewById(R.id.false_button);
    mFalseButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // Does nothing yet, but soon!
        }
    });
}
```

2.Set Listener Object For mTrueButton

3. Overide onClick method (insert your code to do whatever you want as mouse response here)

1. Create listener object as anonymous (unnamed) inner object



QuizActivity.java: 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.</str
    <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>
```

QuizActivity.java: Adding a Toast

- To create a toast, call the method:

```
public static Toast makeText(Context context, int resId, int duration)
```

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
- For example to add a toast to our onClick( ) method:

```
public void onClick(View v) {
    Toast.makeText(QuizActivity.this,
                   R.string.incorrect_toast,
                   Toast.LENGTH_SHORT).show();
}
```

QuizActivity.java: Adding a Toast

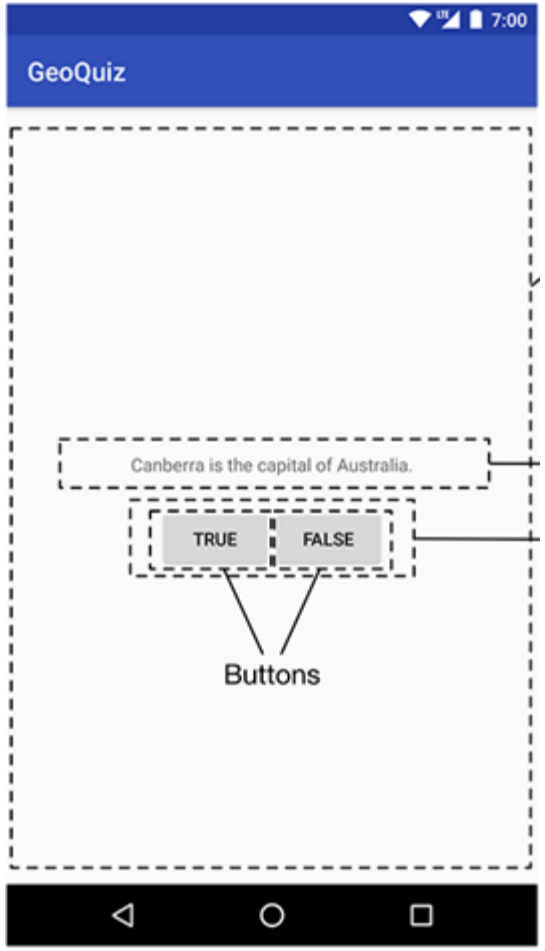
- Code for adding a toast

```
mTrueButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(QuizActivity.this,
                       R.string.correct_toast,
                       Toast.LENGTH_SHORT).show();
            // Does nothing yet, but soon!
    }
});
mFalseButton = (Button) findViewById(R.id.false_button);
mFalseButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(QuizActivity.this,
                       R.string.incorrect_toast,
                       Toast.LENGTH_SHORT).show();
            // Does nothing yet, but soon!
    }
});
```

2.Set Listener Object For mTrueButton

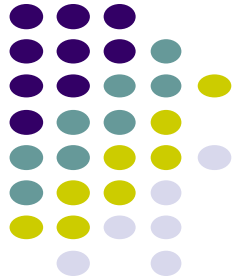
3. Overide onClick method Make a toast

1. Create listener object as anonymous innner object



QuizActivity.java: Complete Listing

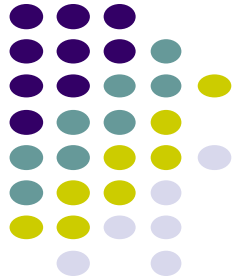
```
package com.bignerdranch.android.geoquiz;

import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

public class QuizActivity extends Activity {

    Button mTrueButton;
    Button mFalseButton;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_quiz);

        mTrueButton = (Button)findViewById(R.id.true_button);
        mTrueButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(QuizActivity.this,
                        R.string.incorrect_toast, Toast.LENGTH_SHORT)
                        .show();
            }
        });
```



```
        mFalseButton = (Button)findViewById(R.id.false_button);
        mFalseButton.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                Toast.makeText(QuizActivity.this,
                        R.string.correct_toast, Toast.LENGTH_SHORT)
                        .show();
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {

     // Inflate the menu;
     // this adds items to the action bar if it is present.

        getMenuInflater().inflate(R.menu.activity_quiz, menu);
        return true;
    }

}
```

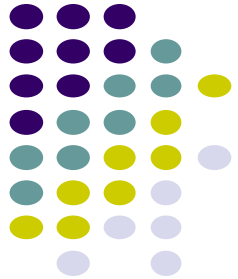
QuizActivity.java: Complete Listing (Contd)

Used if app has an Action bar menu



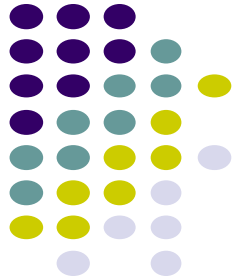
Quiz 1

Quiz 1

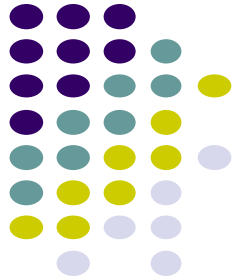


- Quiz next class (Sept 16) before class
- Short answer questions
- Try to focus on understanding, not memorization
- Covers:
  - Lecture slides for lectures 1-2 (including today)
  - YouTube Tutorials (from thenewboston) 1-8, 11,12, 17
  - 3 code examples from books
    - **HFAD examples:** myFirstApp, Beer Advisor
    - **ANR example:** geoQuiz

Groups for Projects 2,3 and Final Project



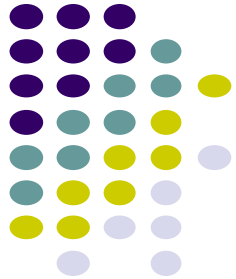
- Projects 2,3 and final project done in teams of 4!
  - A bit large for projects 2,3
  - Great for final project?
- Deadline to form groups: by class time on Wed. Sept 16
- How?
  - Form teams of 4
  - ALL members of the group should email me indicating their group
    - List all team members in their email to me
  - I'll put students who don't have groups into groups
- I'll post list of all students in class. Also set up InstructAssist Forum for discussion



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
- 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



Data-Driven Layouts

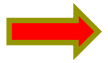
Data-Driven Layouts

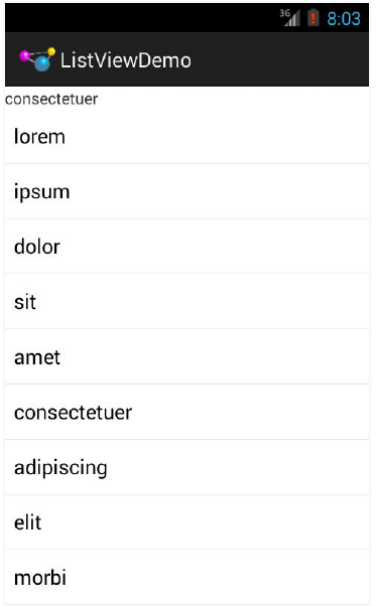
- LinearLayout, RelativeLayout, TableLayout, GridLayout useful for positioning UI elements
  - UI data is **hard coded**

- 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



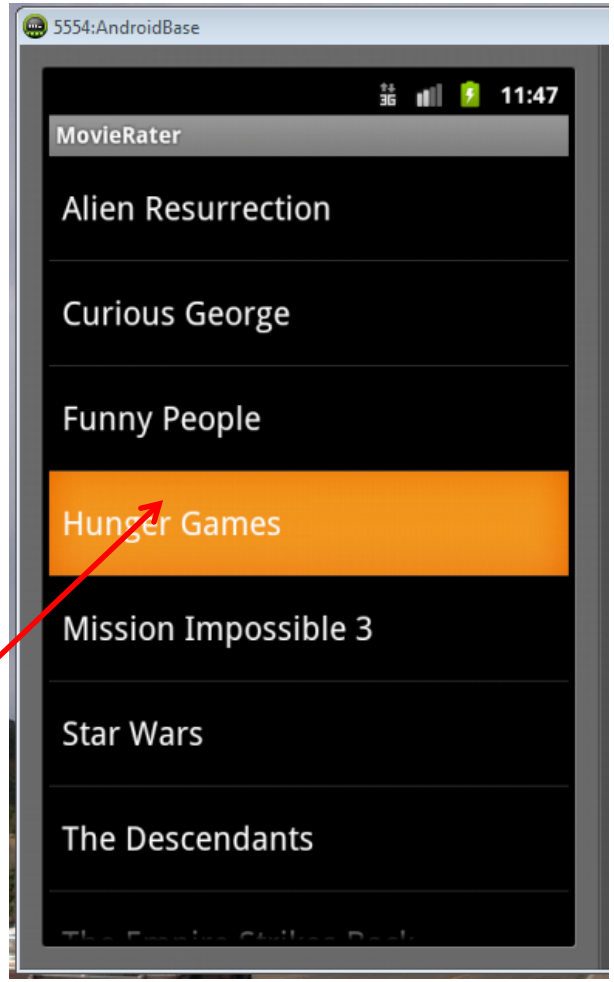


Data Driven Layouts

- May want to populate views from a data source (XML file or database)

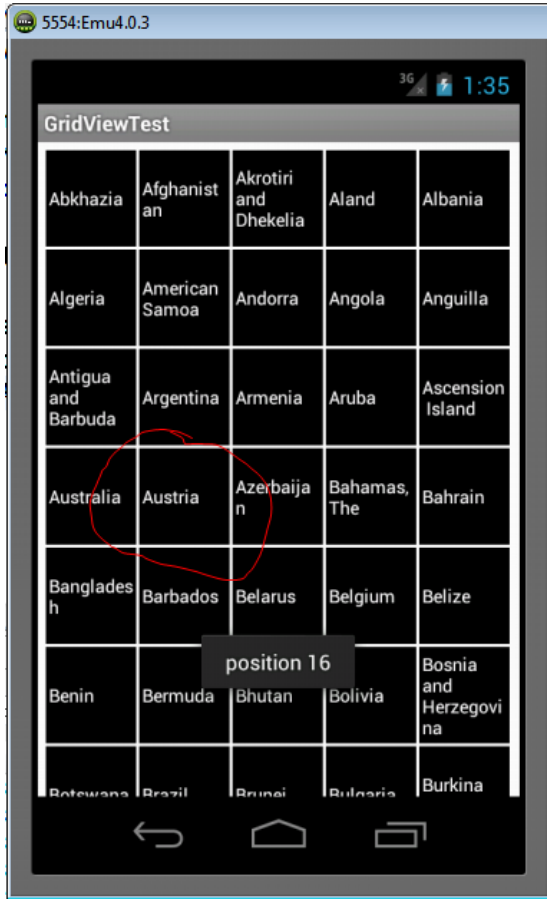
- Layouts that display repetitive child widgets from data source
  - ListView
  - GridView
  - GalleryView

- ListView
  - Rows of entries, pick item, vertical scroll

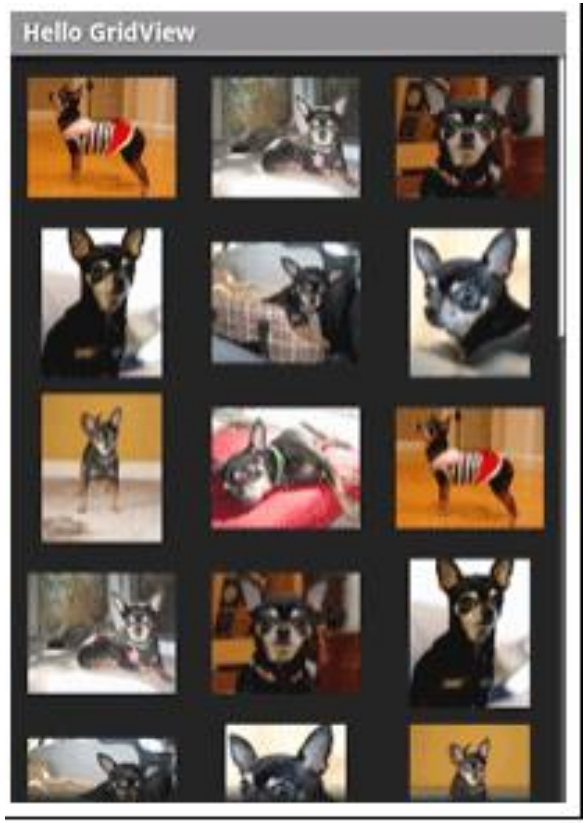


Data Driven Containers

- GridView
  - List of items arranged in rows and columns

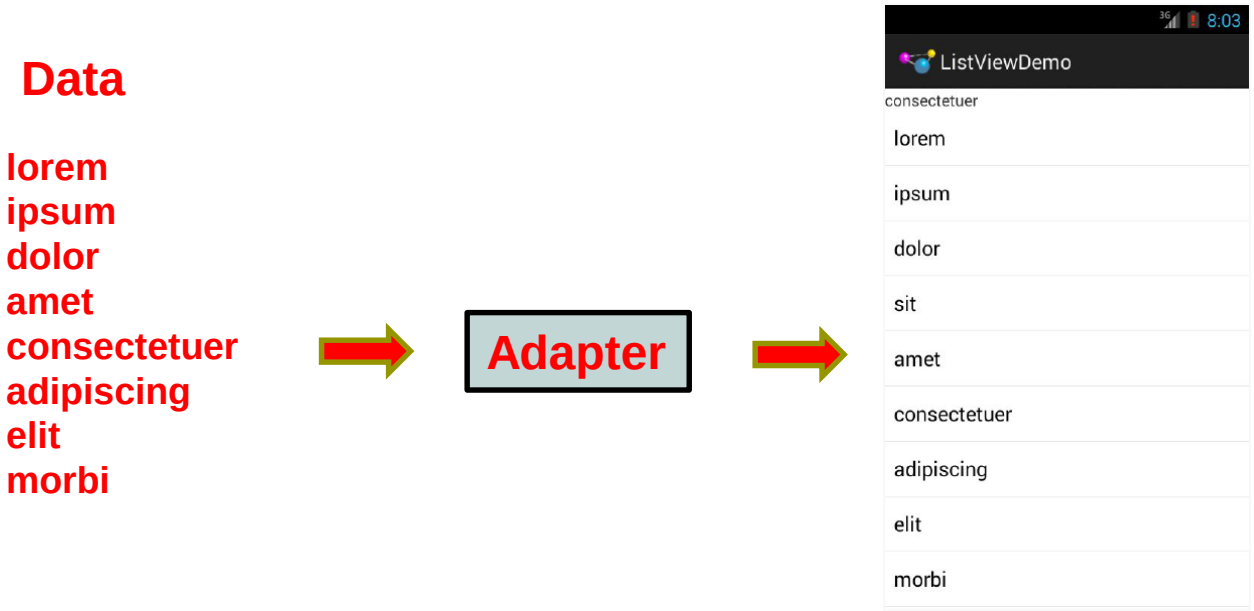


- GalleryView
  - List with horizontal scrolling, typically images



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 GridView



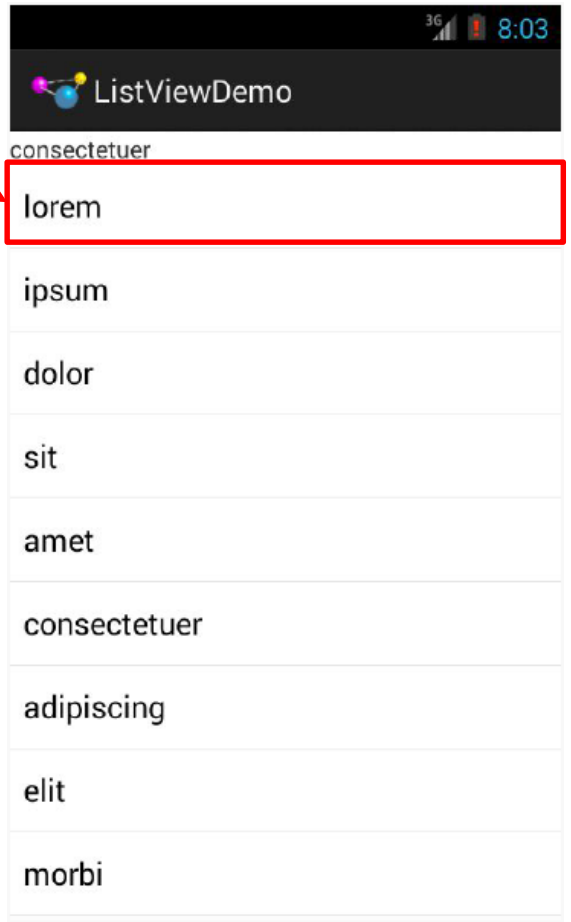
- Most common Adapter types:
  - **CursorAdapter:** read from database
  - **ArrayAdapter:** read from resource (e.g. XML file)

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)

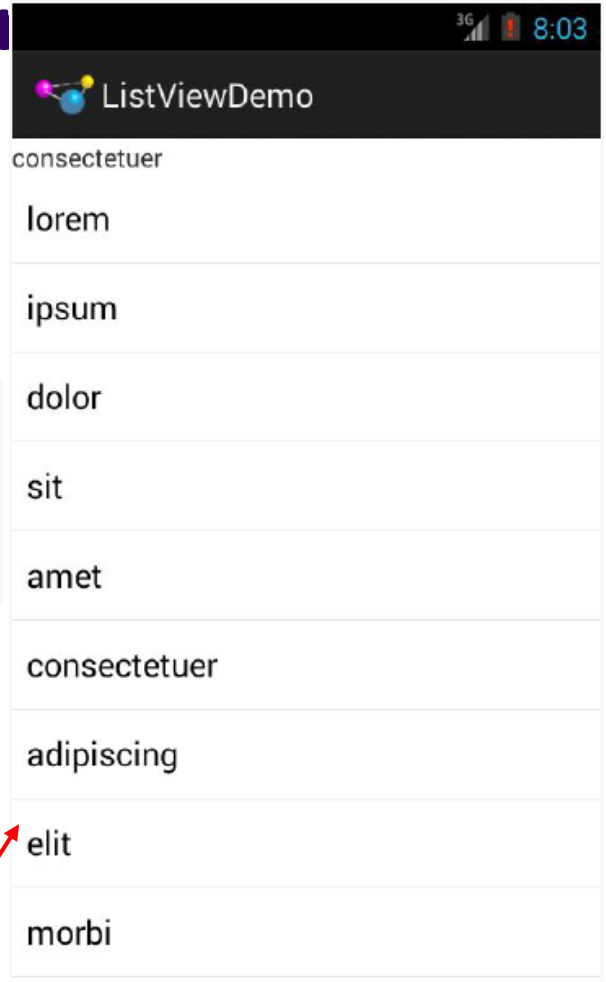


Example: Creating ListView using Adapter

- **Task:** Create listView (on right) from strings below

```
private static final String[] items={"lorem", "ipsum", "dolor",
        "sit", "amet",
        "consectetuer", "adipiscing", "elit", "morbi", "vel",
        "ligula", "vitae", "arcu", "aliquet", "mollis",
        "etiam", "vel", "erat", "placerat", "ante",
        "porttitor", "sodales", "pellentesque", "augue", "purus"};
```

Enumerated list



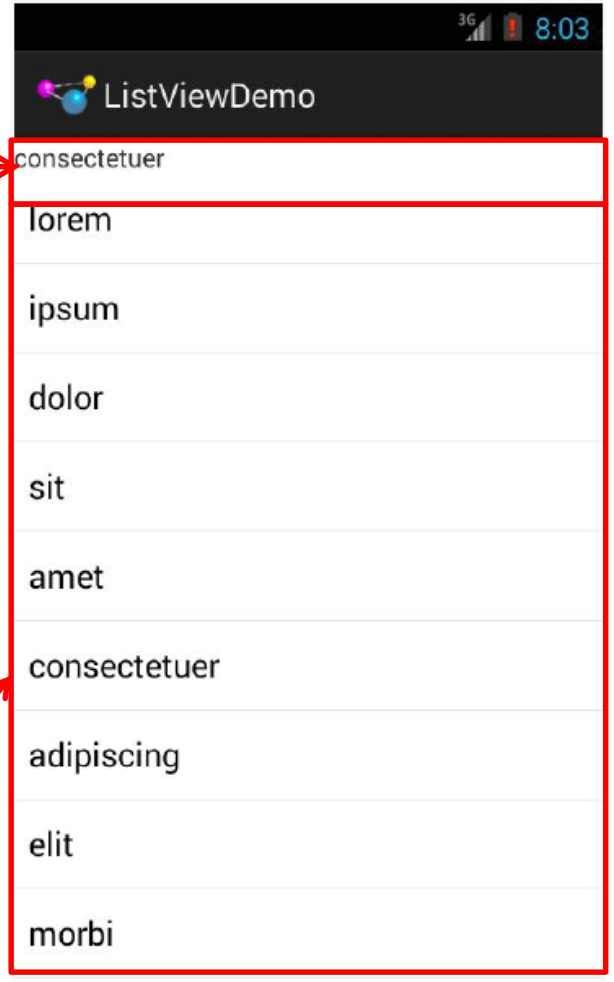
ListView of items

Example: Creating ListView using AdapterArray

- First create Layout file (e.g. LinearLayout)

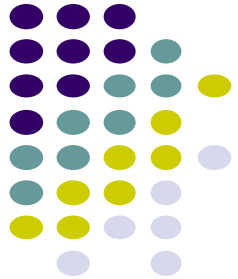
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.co
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="match_parent">
  <TextView
    android:id="@+id/selection"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>
  <ListView
    android:id="@android:id/list"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    />
</LinearLayout>
```

TextView Widget for selected list item



ListView for list of options

Using ArrayAdapter



- Command used to wrap adapter around array of menu items or **java.util.List** instance

```
String[] items={"this", "is", "a", "really", "silly", "list"};
new ArrayAdapter<String>(this,
                         android.R.layout.simple_list_item_1,
                         items);
```

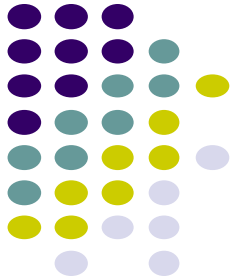
Context to use.
(e.g app's activity)

Array of items to display

Resource ID of View for formatting

- E.g. **android.R.layout.simple_list_item_1** turns strings into textView widgets

Example: Creating ListView using AdapterArray

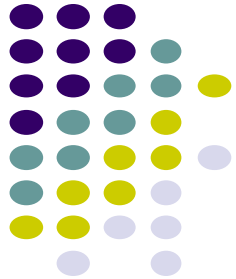


```
package com.commonsware.android.list;

import android.app.ListActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
public class ListViewDemo extends ListActivity {
  private TextView selection;
  private static final String[] items={"lorem", "ipsum", "dolor",
          "sit", "amet",
          "consectetuer", "adipiscing", "elit", "morbi", "vel",
          "ligula", "vitae", "arcu", "aliquet", "mollis",
          "etiam", "vel", "erat", "placerat", "ante",
          "porttitor", "sodales", "pellentesque", "augue", "purus"};

  @Override
  public void onCreate(Bundle icicle) {
    super.onCreate(icicle);
    setContentView(R.layout.main);
    setListAdapter(new ArrayAdapter<String>(this,
                        android.R.layout.simple_list_item_1,
                        items));
    selection=(TextView)findViewById(R.id.selection);
  }

  @Override
  public void onListItemClick(ListView parent, View v, int position,
                              long id) {
    selection.setText(items[position]);
  }
}
```

Set list adapter (Bridge Data source and views)

Get handle to TextView of Selected item

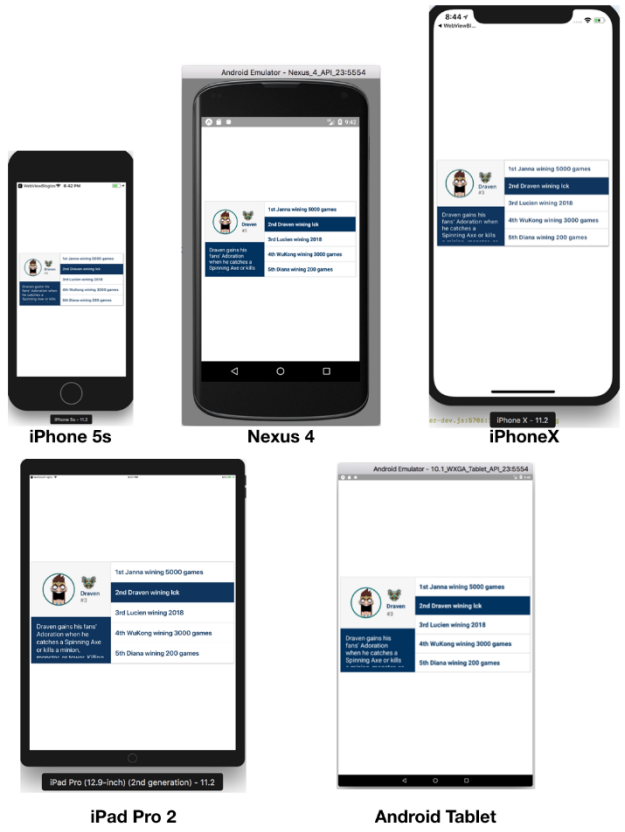
Change Text at top to that of selected view when user clicks on selection



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 summerr



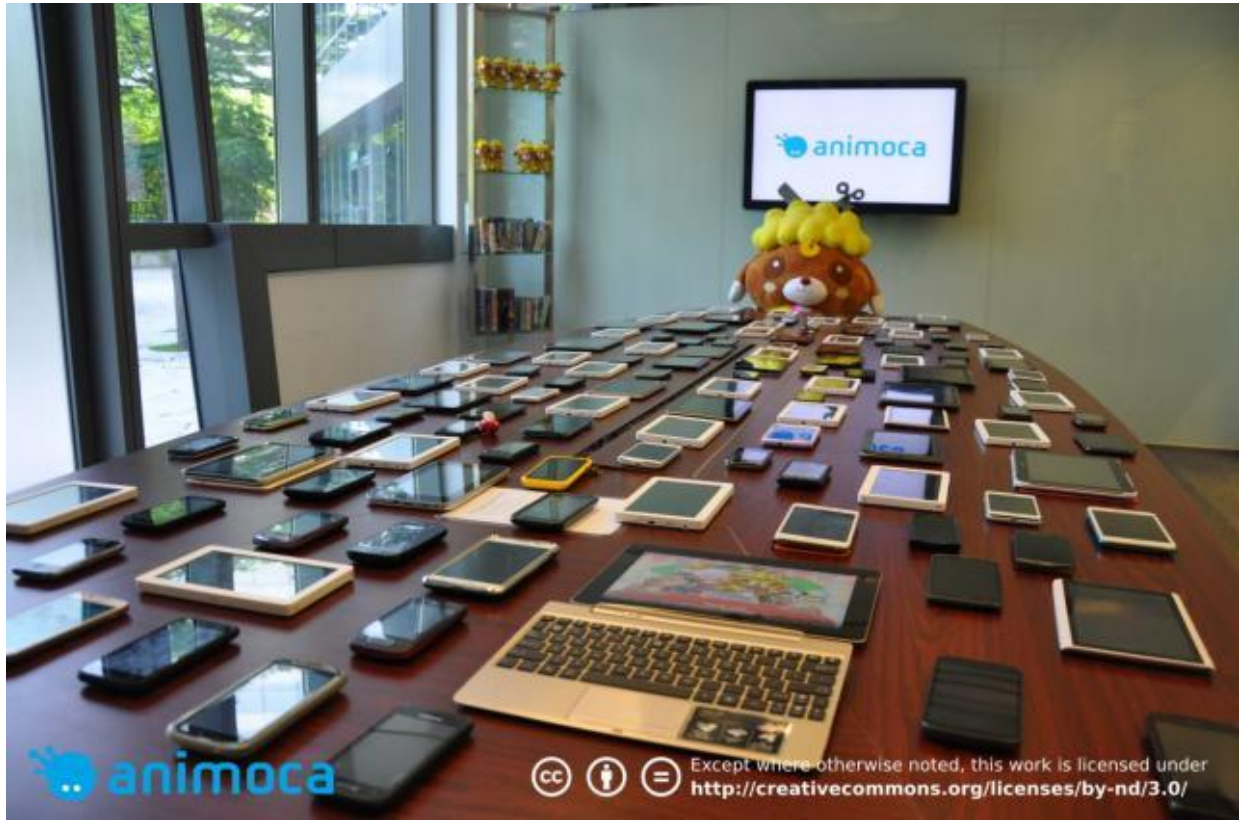
Mobile HCI: Plan out Interaction Flow on Paper

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

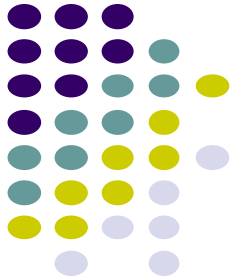
Mobile HCI: Evaluation

- App evaluation: iterative, user-centered
  - In lab (small) then in the field (large)
  - 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

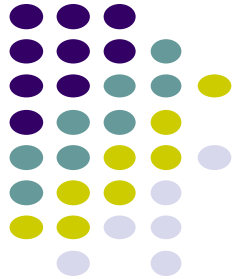


References



- Busy Coder’s guide to Android version 4.4
- CS 65/165 slides, Dartmouth College, Spring 2014
- CS 371M slides, U of Texas Austin, Spring 2014
- Android App Development for Beginners videos by Bucky Roberts (thenewboston)
- Head First Android
- Android Nerd Ranch, Third Edition

References



- Android App Development for Beginners videos by Bucky Roberts (thenewboston)
- 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
- Busy Coder's guide to Android version 4.4
- CS 65/165 slides, Dartmouth College, Spring 2014
- CS 371M slides, U of Texas Austin, Spring 2014