Android: Guts and GUIs

What Is Inside an Android App?

- Components
  - Activities
    - Primary unit of user interface
  - Services
  - Content Providers
- Intents
  - Message bus among components, OS

Creating a Project

- Eclipse
  - File > New > Project
  - Choose Android Project
- Command Line
  - **`android create project`**

Notable Project Contents

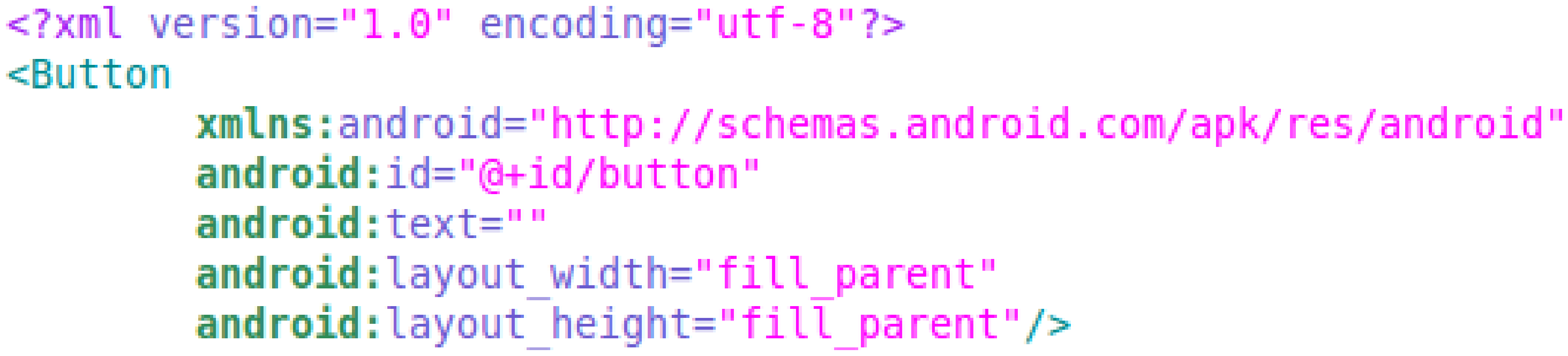
- `AndroidManifest.xml`: what the app is about
- `libs/` : JARs to add to your project
- `res/` : resources, such as:
  - `res/drawable/`
  - `res/layout/`
  - `res/values/`
- `src/` : the Java source tree

Inside an Activity

- Public Java class
- Extends Activity (or an Activity subclass)
- Implements one or more callback methods
  - e.g., `onCreate()`
- Resides in your application's Java package
- `setContentView()` to declare the UI

Inside the XML Layout

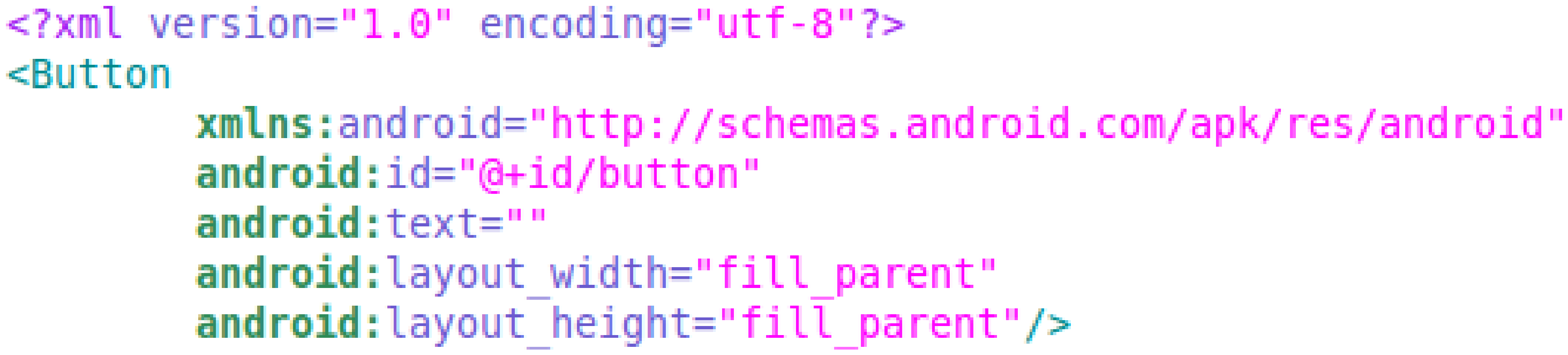
- XML Declaration
- Namespace Declaration
  - Only needed in root element



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

Inside the XML Layout

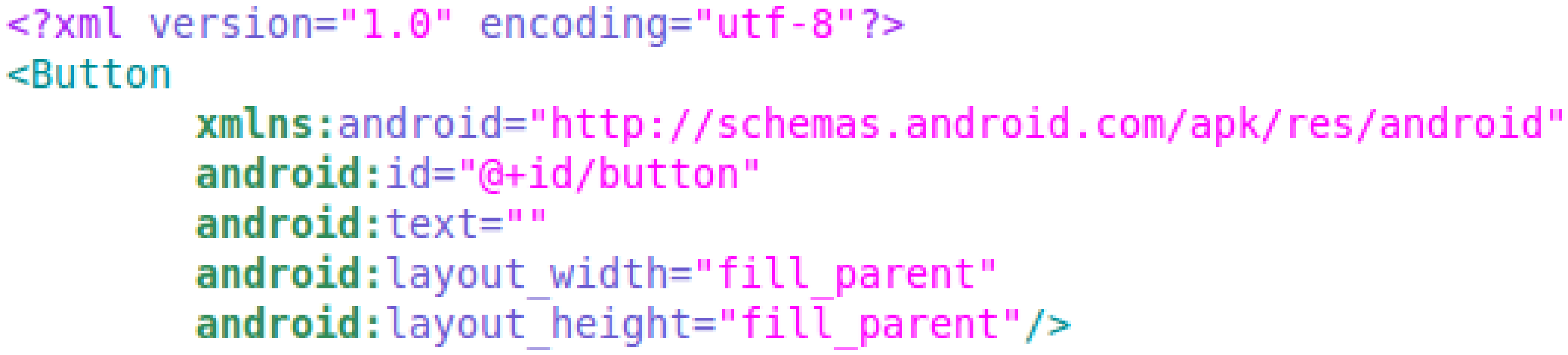
- Elements are Views
  - Can be fully-qualified class name
    - com.commonsware.android.sample.MyOwnWidget
  - Usually a shorthand name for built-in widget



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

Inside the XML Layout

- Attributes are properties of the View
  - `android:id`
    - `@+id/...` syntax for your own names
    - Sometimes need a “magic name” to work with Android code



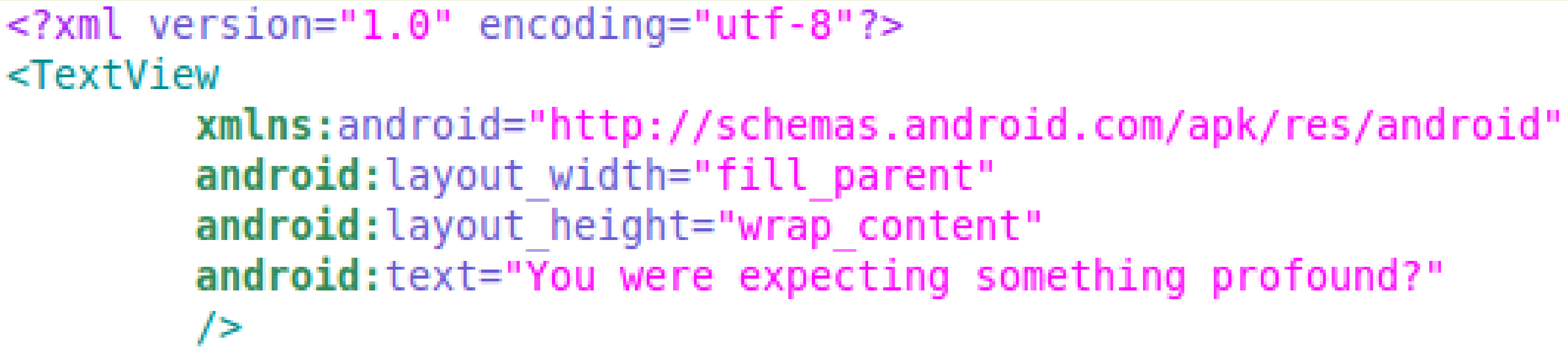
```
<?xml version="1.0" encoding="utf-8"?>
<Button
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/button"
        android:text=""
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"/>
```

Common Widgets

- TextView
- EditText
- Button
- CheckBox
- RadioButton

Widget Dimensions

- Atrributes: android:layout_width, android:layout_height
- Values: fill_parent/match_parent, wrap_content, or specific dimension



```
<?xml version="1.0" encoding="utf-8"?>
<TextView
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="You were expecting something profound?"
        />
```

Combining Widgets via Layouts

- Layouts are containers for widgets
  - Also help to allocate space to widgets
  - Get own height, width
- Simplest = LinearLayout
  - Puts widgets in a row or column
    - `android:orientation=“vertical”`

Using LinearLayout

- Space
  - Widgets inside the layout ask for what they want
  - If 2+ ask for fill_parent, use android:layout_weight on widgets to determine how the available space is divided amongst them
  - Or, use 0px for size and use weight for apportioning space
  - Only single pass made through widgets!

Other Popular Containers

- RelativeLayout
  - Directly specifies positions of widgets relative to the container and relative to each other
  - More explicit control than LinearLayout
- TableLayout
  - Akin to HTML <table> system
  - Lays widgets out in a grid-like structure
  - TableLayout with nested TableRows

DDMS

- Dalvik Debug Monitor Service
- Where To Get It
  - `tools/ddms` in your SDK directory
  - DDMS perspective in Eclipse

DDMS: Devices

- Upper-left tree-table of identified devices
  - Running emulators
  - Hardware devices attached via USB
- Click on the one you wish to manipulate to enable the rest of the tools
  - Also shows running processes

DDMS: Error Logs

- Android Logging
  - android.util.Log plus ones logged by system
  - Five log levels
    - Verbose
    - Debug
    - Information
    - Warning
    - Error

DDMS: Error Logs

- DDMS Bottom Pane = Log Display
  - Filter via toolbar button corresponding to log level
  - Save selected log entries to file for later inspection
  - Clear log entries
    - Only clears DDMS
    - Does not affect logs in emulator/device

DDMS: Simulating Incoming Calls

- Goal: Pretend a Call Is Coming In
- Steps
  - Choose Emulator Control tab
  - Scroll to Telephony Actions
  - Enter in number for call to be from
  - Click Call
  - Optional: click Hang Up

DDMS: Simulating Your Location

- Goal: Pretend the Emulator Is Somewhere
- Steps
  - – Choose Emulator Control tab
  - – Scroll to Location Controls
  - – Enter longitude and latitude
  - – Click Send
  - – Device will get a “fix” on GPS
  - – Alternatives: GPX, KML for routes

DDMS: Other Features

- Screen capture
- Browse filesystems
- Examine running threads
- Force garbage collection
- Track memory usage
- Terminate processes
- ...and more!

About the Exercises

- Work Through Tutorials
  - Will point out the tutorial(s) pertaining to a particular module
  - Options
    - Follow directions and do work step-by-step
    - Download answers and review
- Work on Own Projects