CMSC436: Programming Handheld Systems

Application Fundamentals

Application Components

Activity

Service

BroadcastReceiver

ContentProvider

Applications

Apps are made from components

Android instantiates and runs them as needed

Each component has its own purpose and APIs

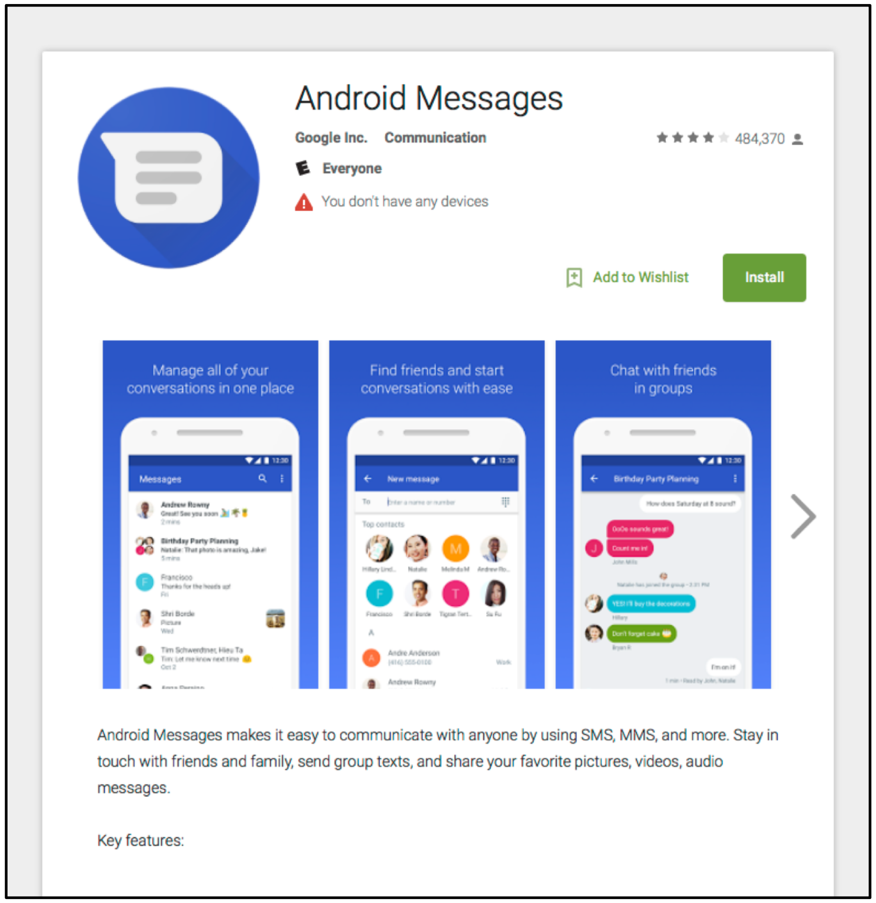
Apps can have multiple “entry points”

Activity

Primary class for user interaction

Usually implements a single, focused task that the user can do

Example App
Android Messages



Android Messages

Google Inc. Communication

484,370

Everyone

You don't have any devices

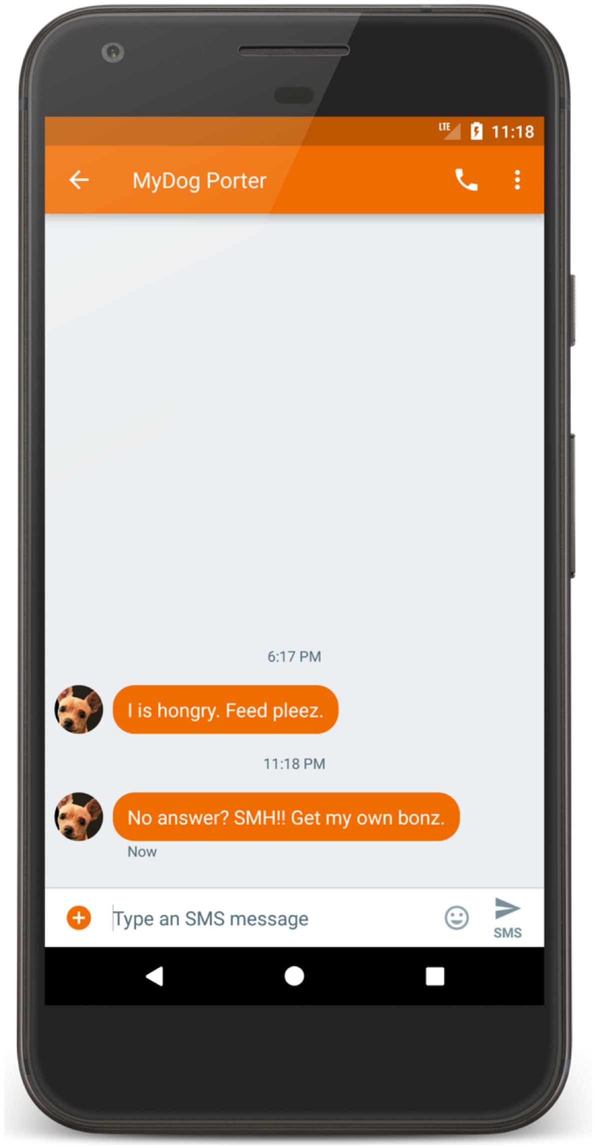
Add to Wishlist

Install

Manage all of your conversations in one place
Find friends and start conversations with ease
Chat with friends in groups

Android Messages makes it easy to communicate with anyone by using SMS, MMS, and more. Stay in touch with friends and family, send group texts, and share your favorite pictures, videos, audio messages.

Key features:



6:17 PM

I is hongry. Feed pleez.

11:18 PM

No answer? SMH!! Get my own bonz.

Now

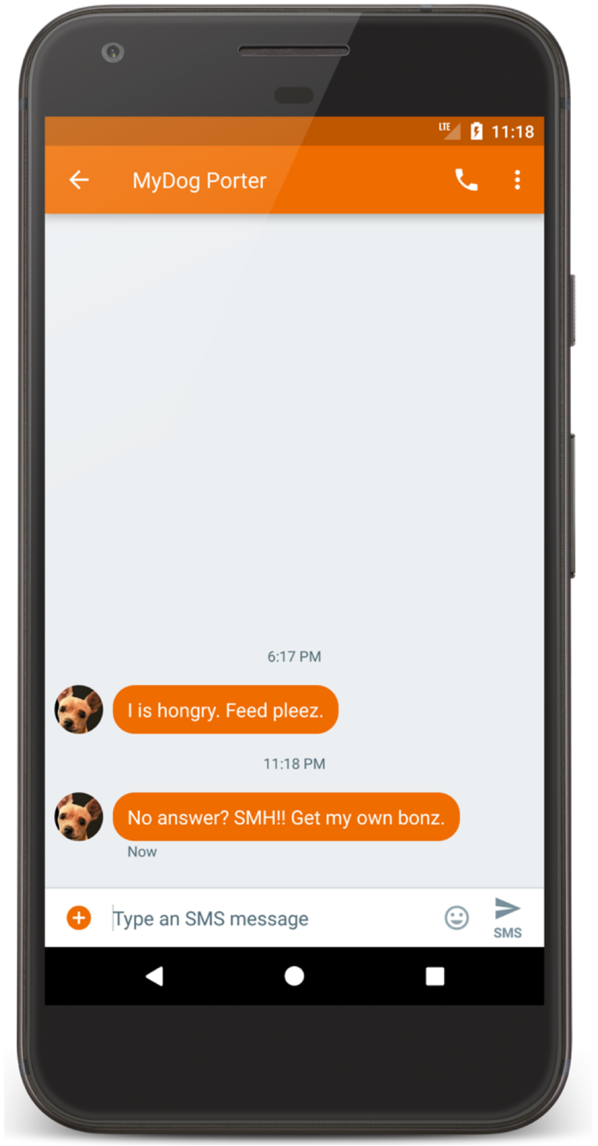
ConversationActivity.java

```
package com.android.messaging.ui.conversation;
…
public class ConversationActivity extends BugleActionBarActivity
            implements ContactPickerFragmentHost,
      ConversationFragmentHost, ConversationActivityUiStateHost {
   …
```

Android source code available at: https://source.android.com

Service

- Runs in the background
    - to perform long-running operations
    - to support interaction with remote processes



11:18

MyDog Porter

6:17 PM

I is hongry. Feed pleez.

11:18 PM

No answer? SMH!! Get my own bonz.

Now

MmsService.java

```
package com.android.mms.service;
...
/**
 * System service to process MMS API requests
 */
public class MmsService extends Service implements
                                        MmsRequest.RequestManager {
...
```

BroadcastReceiver

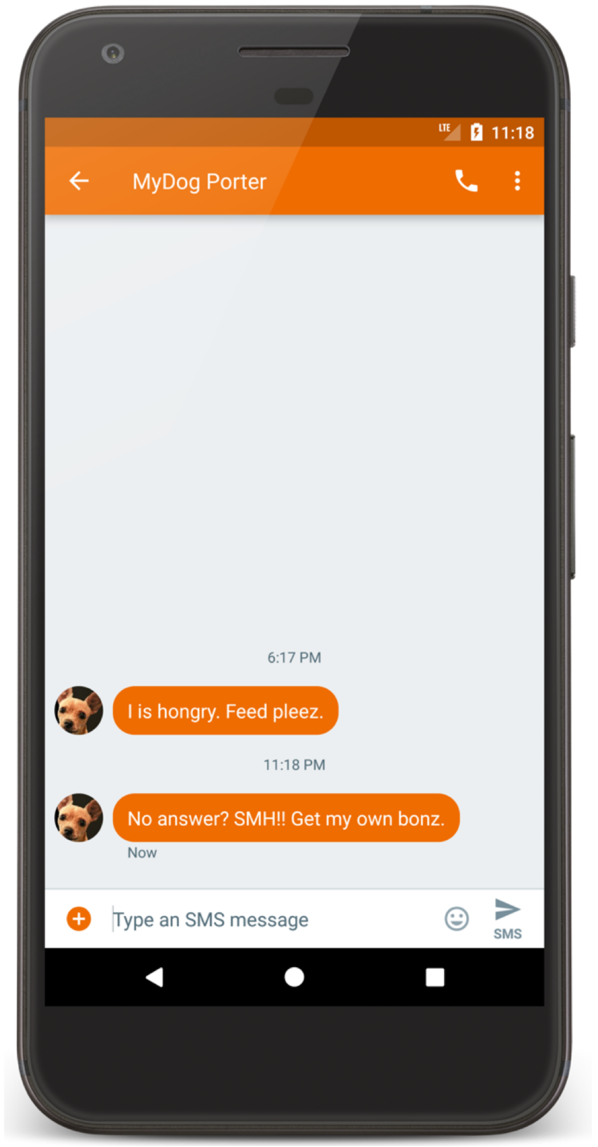
Component that listens for and responds to events

Acts as the subscriber in publish/subscribe pattern

BroadcastReceiver

Events are represented by an Intent and then broadcast by the platform

BroadcastReceivers can receive and respond to to broadcast events



6:17 PM

I is hongry. Feed pleez.

11:18 PM

No answer? SMH!! Get my own bonz.

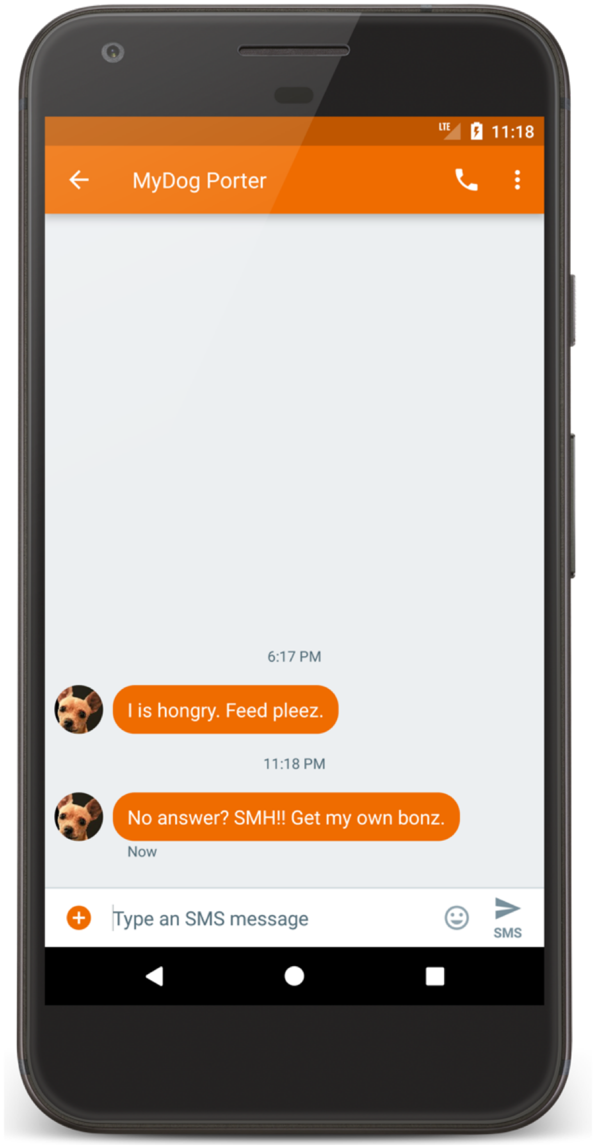
Now

SmsDeliverReceiver.java

```
package com.android.messaging.receiver;
...
/**
 * Class that receives incoming SMS messages on KLP+ Devices.
 */
public final class SmsDeliverReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(final Context context, final Intent intent) {
        SmsReceiver.deliverSmsIntent(context, intent);
    }
}
```

Content Providers

- Store & share data across applications
  - Uses database-style interface
  - Handles interprocess communication



6:17 PM

I is hongry. Feed pleez.

11:18 PM

No answer? SMH!! Get my own bonz.

Now

MessagingContentProvider

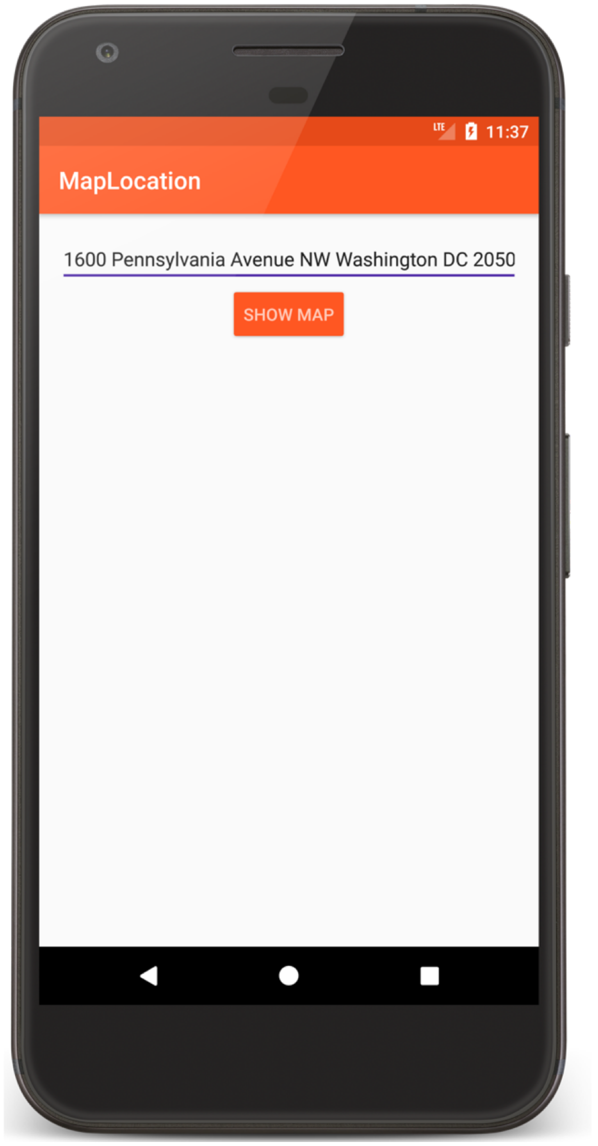
```
package com.android.messaging.datamodel;
…
/**
 * A centralized provider for Uris exposed by Bugle.
 *  */
public class MessagingContentProvider extends ContentProvider {
   …
```

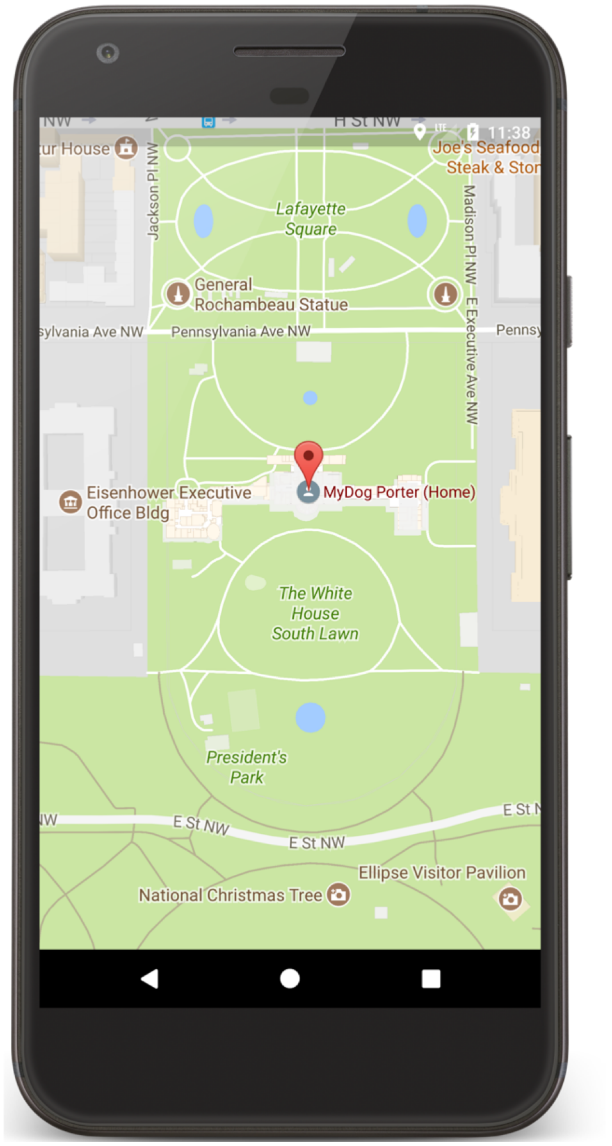
MapLocation

User enters an address

App displays a map of area around the address

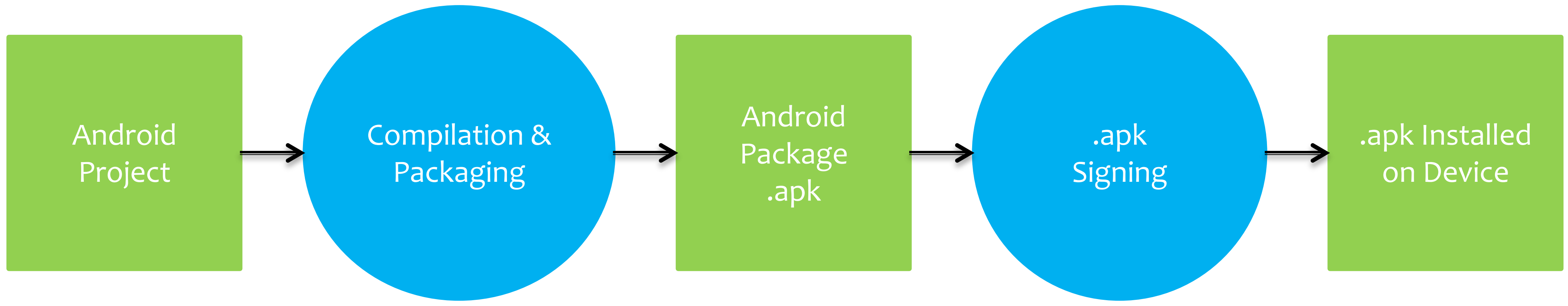
MapLocation





11:38
H St NW
ur House
Joe's Seafood Steak & Ston
Jackson Pl NW
Lafayette Square
Madison Pl NW
General Rochambeau Statue
sylvania Ave NW
Pennsylvania Ave NW
Pennsy
E Executive Ave NW
Eisenhower Executive Office Bldg
MyDog Porter (Home)
The White House South Lawn
President's Park
E St NW
E St NW
E St N
NW
Ellipse Visitor Pavilion
National Christmas Tree

Simplified App Development Workflow



Creating an Android App

Define resources

Implement application classes

Package application

Install & run application

1. Defining Resources

Resources are non-source code entities

Many different resource types, e.g.,

- Layout, strings, images, menus, & animations

Allows apps to be customized for different devices and users

See: https://developer.android.com/
guide/topics/resources/overview.html

Strings

Types: String, String Array, Plurals

Strings

Types: String, String Array, Plurals

Typically stored in res/values/*.xml

Specified in XML, e.g.,

- `<string name="hello">Hello World!</string>`
- Can include formatting and styling codes

Strings

Accessed by other resources as:

    @string/string_name

Accessed in Java as:

    R.string.string_name

MapLocation’s Strings Files

values/strings.xml

```
<resources>
   <string name="show_map_string">Show Map</string>
   <string name="location_string">Enter Location</string>
</resources>
```

values-it/strings/xml

```
<resources>
   <string name="show_map_string">Mostra la mappa</string>
   <string name="location_string">Digita l\'indirizzo</string>
</resources>
```

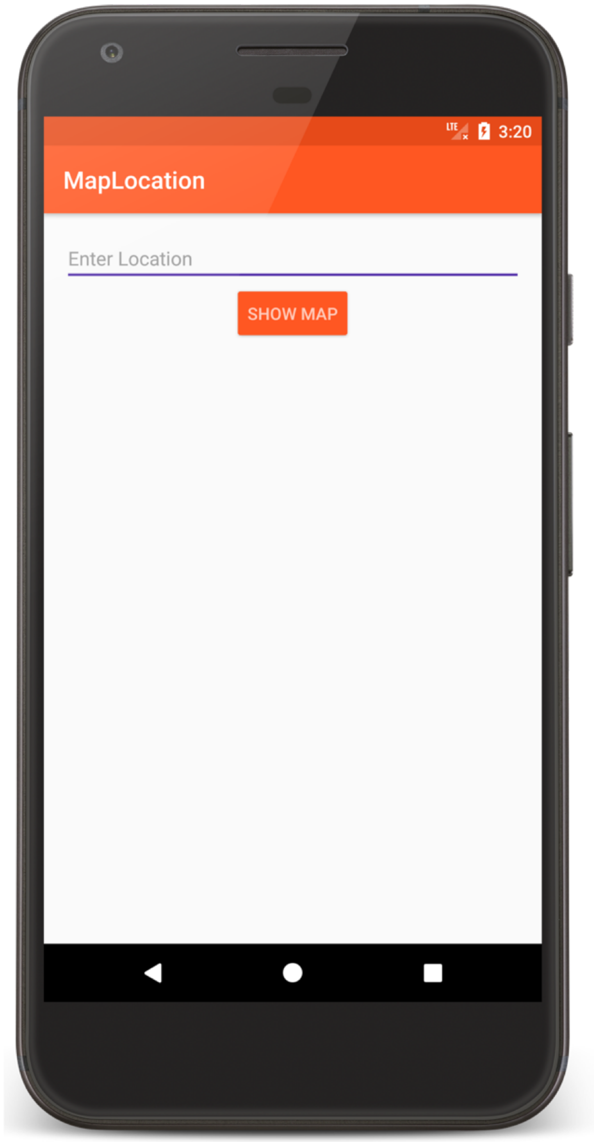
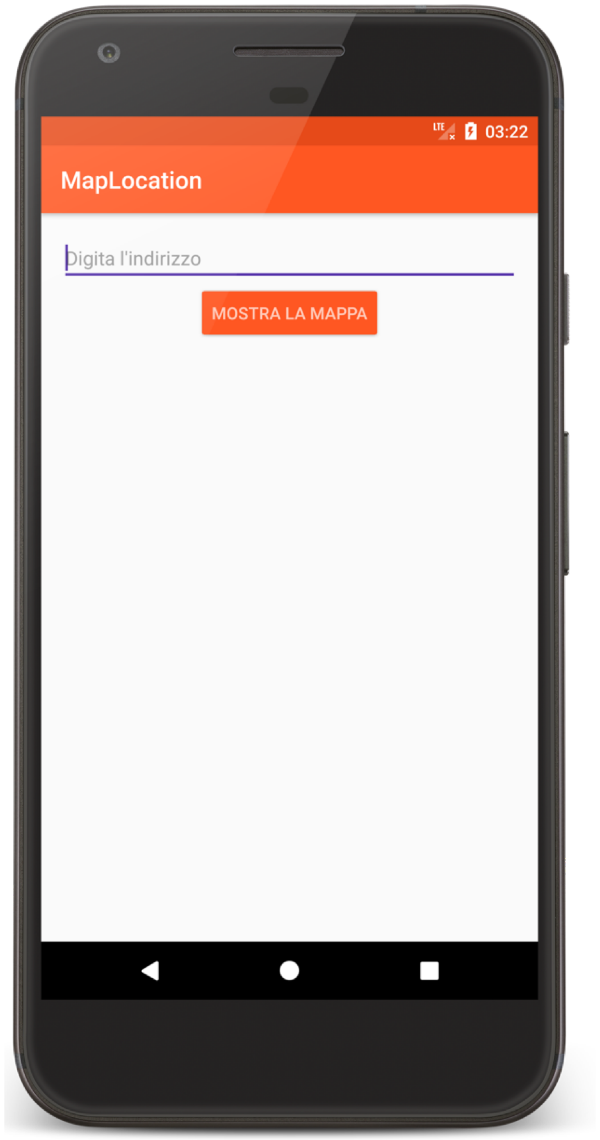
Customized Strings at Runtime

If your default language is Italian, @string/location_string is

- “Digita l’indirizzo”

Otherwise it’s,

- “Enter Location”


3:20
MapLocation
Enter Location
SHOW MAP

03:22
MapLocation
Digita l'indirizzo
MOSTRA LA MAPPA

User Interface Layout

UI layout specified in XML files

- Some tools allow visual layout

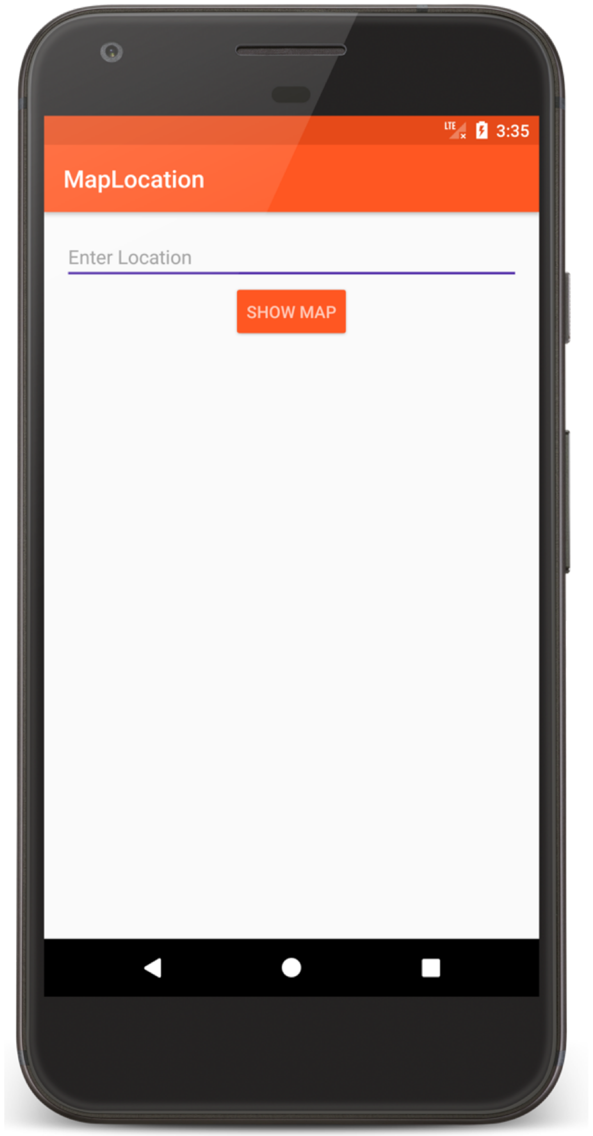
XML files typically stored in res/layout/*.xml

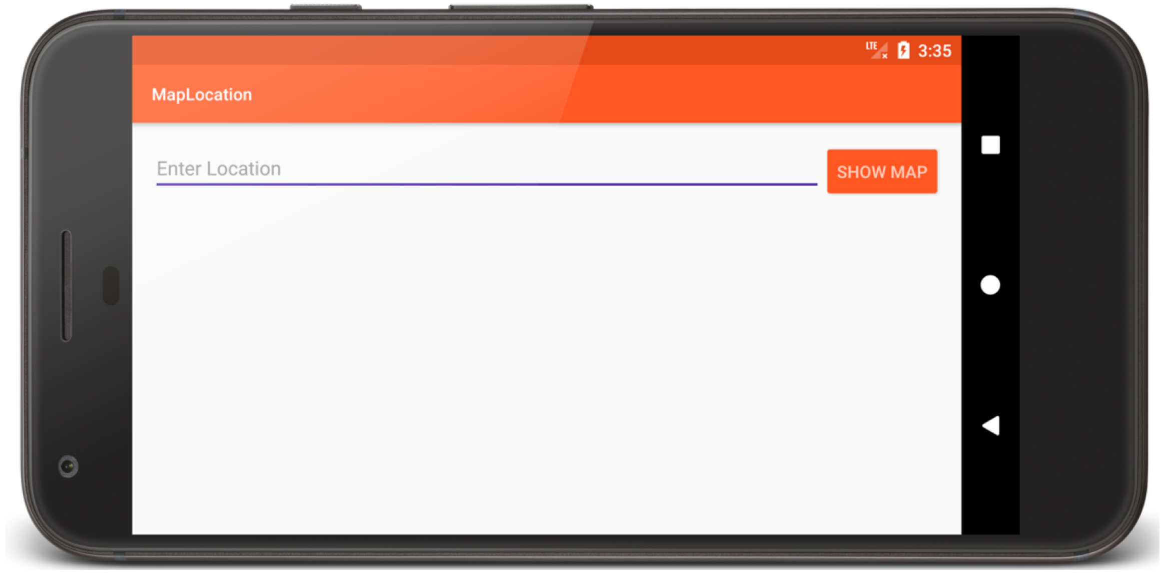
Accessed in Java as: R.layout. layout_name

Accessed by other resources as: @layout/layout_name

Using Multiple Layout Files

Can specify different layout files based on your device’s orientation, screen size, etc.


3:35
MapLocation
Enter Location
SHOW MAP


3:35
MapLocation
Enter Location
SHOW MAP

Portrait Layout

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="@dimen/activity_margin">

    <EditText
        android:id="@+id/location"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:hint="@string/location_string"
        android:inputType="textPostalAddress"
        android:textAppearance="@android:style/TextAppearance.Material.Subhead" />

...
```

```
    ...
    <Button
        android:id="@+id/mapButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/location"
        android:layout_centerHorizontal="true"
        android:text="@string/show_map_string"
        android:textAppearance="@android:style/TextAppearance.Material.Button"
        android:textColor="@color/primary_light" />

</RelativeLayout>
```

Landscape Layout

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="@dimen/activity_margin">
    <EditText
        android:id="@+id/location"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_toStartOf="@+id/mapButton"
        android:ems="10"
        android:hint="@string/location_string"
        android:inputType="textPostalAddress"
        android:textAppearance="@android:style/TextAppearance.Material.Subhead" />
    ...
    ...
    <Button
        android:id="@+id/mapButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignTop="@+id/location"
        android:text="@string/show_map_string"
        android:textAppearance="@android:style/TextAppearance.Material.Button"
        android:textColor="@color/primary_light" />

</RelativeLayout>
```

R.java

At compilation time, resources are used to generate the R.java class

Java code uses the R class to access resources

```
package course.examples.maplocation;
public final class R {
  public static final class color {
    public static final int accent=0x7f010000;
    public static final int edit_text=0x7f010001;
    public static final int primary=0x7f010002;
    public static final int primary_dark=0x7f010003;
    public static final int primary_light=0x7f010004;
    public static final int primary_text=0x7f010005;
    public static final int secondary_text=0x7f010006;
  }
  public static final class dimen {
    public static final int activity_margin=0x7f020000;
  }
  public static final class id {
    public static final int location=0x7f030000;
    public static final int mapButton=0x7f030001;
  }

  …
```

```java
 ...
  public static final class layout {
    public static final int main=0x7f040000;
  }
  public static final class mipmap {
    public static final int ic_launcher=0x7f050000;
  }
  public static final class string {
    public static final int location_string=0x7f060000;
    public static final int show_map_string=0x7f060001;
  }
  public static final class style {
    public static final int MaterialTheme=0x7f070000;
  }
}
```

2. Implement Classes

Usually involves at least one Activity

Activity initialization code usually in onCreate()

2. Implement Classes

Typical onCreate() workflow

- Restore saved state, if necessary
- Set content view
- Initialize UI elements
- Link UI elements to code actions

```java
public class MapLocation extends Activity {
   private final String TAG = "MapLocation";
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      // Required call through to Activity.onCreate()
      // Restore any saved instance state
      super.onCreate(savedInstanceState);
      // Set content view
      setContentView(R.layout.main);
      // Initialize UI elements
      final EditText addrText = findViewById(R.id.location);
      final Button button = findViewById(R.id.mapButton);
      ....
```

```
 ....
// Link UI elements to actions in code
button.setOnClickListener(new OnClickListener() {
   // Called when user clicks the Show Map button
   public void onClick(View v) {
       try {
            // Process text for network transmission
           String address = addrText.getText().toString();
            address = address.replace(' ', '+');

           // Create Intent object for starting Google Maps application
           Intent geoIntent = new Intent(android.content.Intent.ACTION_VIEW,
                                                    Uri.parse("geo:0,0?q=" + address));
           // Use the Intent to start Google Maps application using Activity.startActivity()
           startActivity(geoIntent);
          } catch (Exception e) {
           // Log any error messages to LogCat using Log.e()
           Log.e(TAG, e.toString());
       ...
```

3. Package Application

System packages application components & resources into a .apk file

Developers specify required application information in a file called AndroidManifest.xml

AndroidManifest.xml

Information includes:

- Application name
- Application components
- Other
  - Required permissions
  - Application features
  - etc.

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="course.examples.maplocation">
    <application
        android:allowBackup="false"
        android:icon="@mipmap/ic_launcher"
        android:label="MapLocation"
        android:theme="@style/MaterialTheme">
        <activity android:name="course.examples.maplocation.MapLocation">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

4. Install & Run

From IDE run app in the emulator or device

From command line

- Enable USB Debugging on the device
  - See: https://developer.android.com/studio/debug/dev-options.html
- %adb install <path_to_apk>

Next

The Activity Class

Example Applications

MapLocation