Activities and Intents

Activities

- An ***activity*** presents a visual user interface for one focused endeavor the user can undertake.
- For example, an activity might present a list of menu items users can choose from or it might display photographs along with their captions.
- A text messaging application might have one activity that shows a list of contacts to send messages to, a second activity to write the message to the chosen contact, and other activities to review old messages or change settings.
- Each one is implemented as a subclass of the **Activity** base class.

Activities [Conti.]

- Number of Activity depends on the application and its design.
- One of the activities is marked as the first one that should be presented to the user when the application is launched.
- Moving from one activity to another is accomplished by having the current activity start the next one.
- Each activity is given a default window to draw in. Typically, the window fills the screen, but it might be smaller than the screen and float on top of other windows.

View or UI of Activity

- The visual content of the window is provided by a hierarchy of views — objects derived from the base **View** class.
- Each view controls a particular rectangular space within the window.
- Thus, views are where the activity's interaction with the user takes place.
- Android has a number of ready-made views that you can use — including buttons, text fields, scroll bars, menu items, check boxes, and more.
- A view hierarchy is placed within an activity's window by the **setContentView()** method.
- The *content view* is the View object at the root of the hierarchy.

Understanding Activities

1) Manifest File

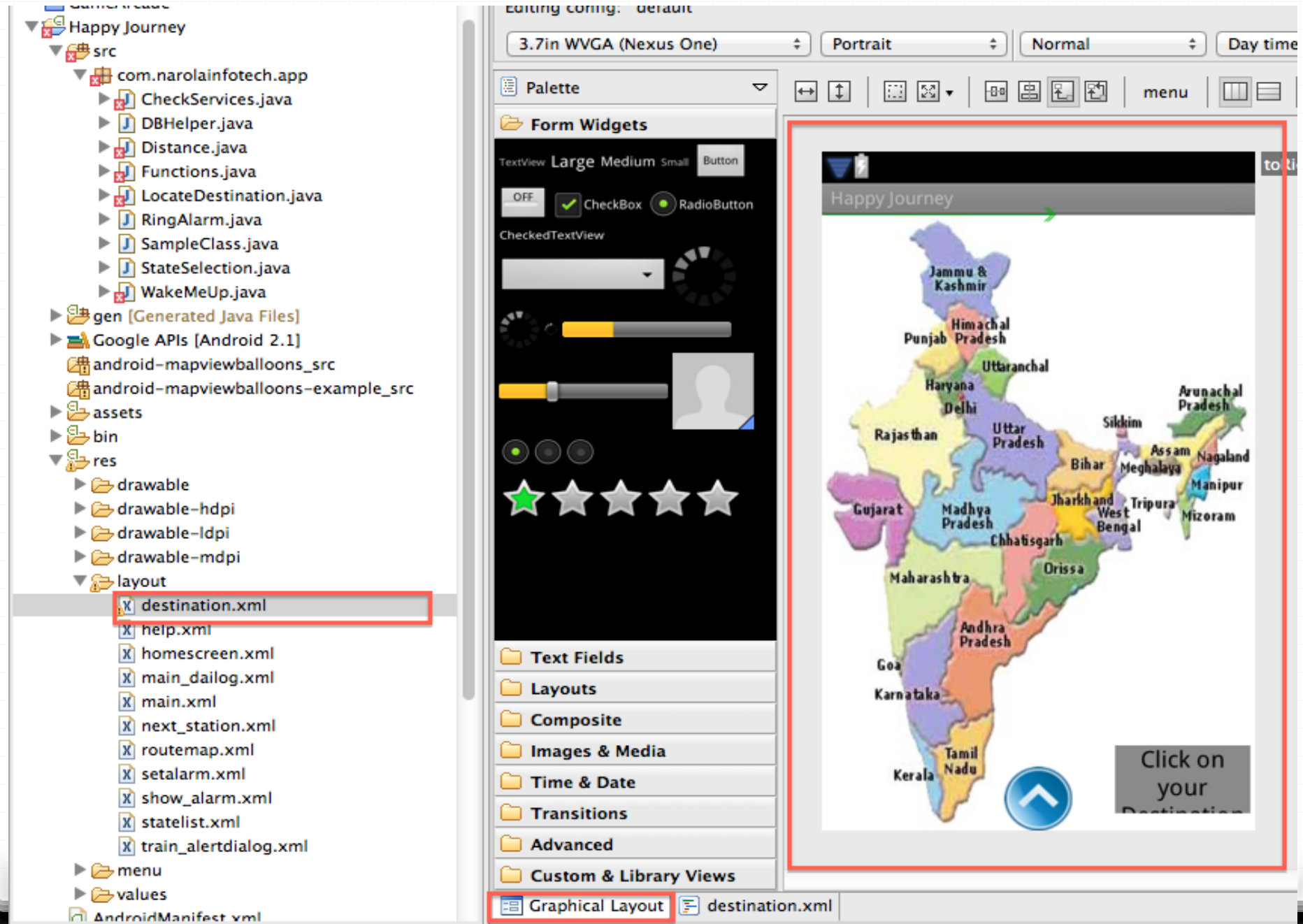
```
<activity android:name="LocateDestination">
    <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
  </intent-filter>
</activity>
```

```
<activity android:name=".AlarmHome">
</activity>

<activity android:name=".StateSelection"></activity>

<activity android:name=".WakeMeUp"
           android:excludeFromRecents="true"
           android:noHistory="true">
 </activity>
```

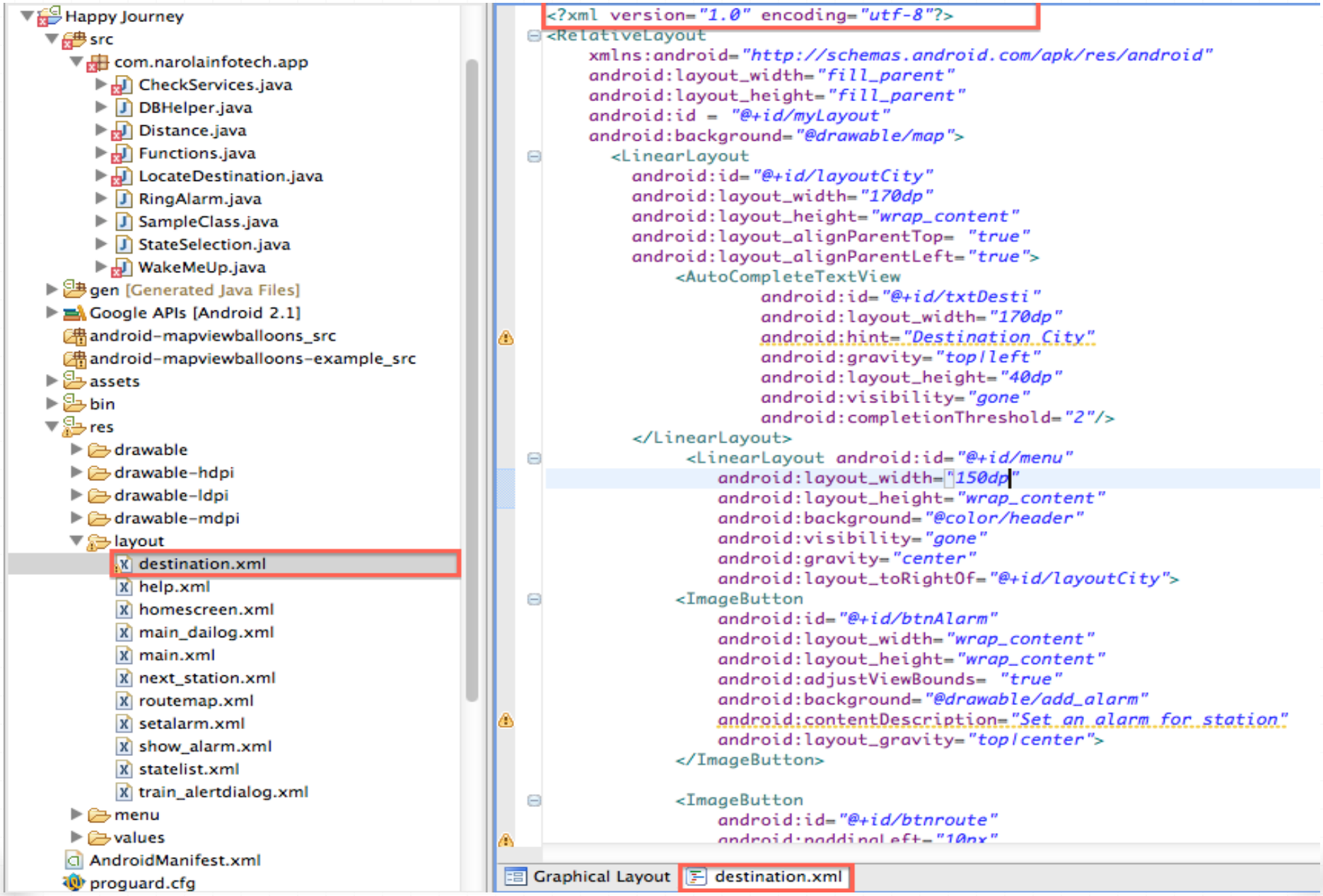
2) Class Structure

```
import android.app.Activity;
import android.os.Bundle;

public class SampleClass extends Activity
{
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.destination);
    }
}
```

3) UI View- Graphical



4) UI View- XML


Happy Journey
src
com.narolainfotech.app
CheckServices.java
DBHelper.java
Distance.java
Functions.java
LocateDestination.java
RingAlarm.java
SampleClass.java
StateSelection.java
WakeMeUp.java
gen [Generated Java Files]
Google APIs [Android 2.1]
android-mapviewballoons_src
android-mapviewballoons-example_src
assets
bin
res
drawable
drawable-hdpi
drawable-ldpi
drawable-mdpi
layout
destination.xml
help.xml
homescreen.xml
main_dailog.xml
main.xml
next_station.xml
routemap.xml
setalarm.xml
show_alarm.xml
statelist.xml
train_alertdialog.xml
menu
values
AndroidManifest.xml
proguard.cfg
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:id = "@+id/myLayout"
android:background="@drawable/map">
<LinearLayout
android:id="@+id/layoutCity"
android:layout_width="170dp"
android:layout_height="wrap_content"
android:layout_alignParentTop= "true"
android:layout_alignParentLeft="true">
<AutoCompleteTextView
android:id="@+id/txtDesti"
android:layout_width="170dp"
android:hint="Destination City"
android:gravity="top|left"
android:layout_height="40dp"
android:visibility="gone"
android:completionThreshold="2"/>
</LinearLayout>
<LinearLayout android:id="@+id/menu"
android:layout_width="150dp"
android:layout_height="wrap_content"
android:background="@color/header"
android:visibility="gone"
android:gravity="center"
android:layout_toRightOf="@+id/layoutCity">
<ImageButton
android:id="@+id/btnAlarm"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:adjustViewBounds= "true"
android:background="@drawable/add_alarm"
android:contentDescription="Set an alarm for station"
android:layout_gravity="top|center">
</ImageButton>
<ImageButton
android:id="@+id/btnroute"
Graphical Layout
destination.xml

Intents

- Three of the core components of an application — activities, services, and broadcast receivers — are activated through messages, called ***intents*.**
- An **Intent object is a bundle of information.**
- It contains information of interest to the component that receives the intent (such as the action to be taken and the data to act on and instructions on how to launch a target activity).

Intents

- An Intent object is passed to **Context.startActivity() or Activity.startActivityForResult() to launch an activity or get an existing activity to do something new.**
- An Intent object is passed to **Context.startService() to initiate a service or deliver new instructions to an ongoing service.**
- Intent objects passed to any of the broadcast methods (such a **Context.sendBroadcast(), Context.sendOrderedBroadcast(), or Context.sendStickyBroadcast()) are delivered to all interested broadcast receivers.**

Working of Intents

- ***Intents*** are asynchronous messages which allow Android components to request functionality from other components of the Android system. For example an Activity can send an *Intents* to the Android system which starts another Activity.
- Therefore *Intents* allow to combine loosely coupled components to perform certain tasks.
- *Intents* can be used to signal to the Android system that a certain event has occurred. Other components in Android can register to this event and will get notified.

Intents

- *Intents* are instances of the **android.content.Intent** class.
- *Intents* are send to the Android system. Depending on how the Intent was constructed the Android system will run an receiver determination and determine what to do.
- Android supports explicit and implicit *Intents*.

Explicit Intent

- **Explicit Intents** explicitly defines the component which should be called by the Android system, by using the Java class as identifier.
- The following shows an Explicit Intent. If that Intent is correctly send to the Android system and the class is accessible, it will start the associated class.

```
Intent i = new Intent(this, ActivityTwo.class);
i.putExtra("Value1", "This value one for ActivityTwo ");
i.putExtra("Value2", "This value two ActivityTwo");
```

Implicit Intents

- **Implicit Intents** do not directly specify the Android components which should be called. They specify the action which should be performed and optionally an URI which should be used for this action.

```
Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.vogella.com"));
```

- If these *Intents* are send to the Android system it searches for all components which are registered for the specific action and the data type.
- If only one component is found, Android starts this component directly. If several components are identifier by the Android system, the user will get an selection dialog and can decide which component should be used for the Intent.

Examples for Implicit Intents

```
final Intent emailIntent = new Intent(android.content.Intent.ACTION_SEND);
emailIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
emailIntent.putExtra(Intent.EXTRA_STREAM, screenshotUri);
emailIntent.setType("image/png");

startActivity(Intent.createChooser(emailIntent, "Send email using"));
```

```
  Intent intentBrowseFiles = new Intent(Intent.ACTION_GET_CONTENT);
  intentBrowseFiles.setType("image/*");
  intentBrowseFiles.setFlag(Intent.FLAG_ACTIVITY_NEW_TASK);
  startActivity(intentBrowseFiles);
```

```
startActivity(new Intent(Intent.ACTION_CALL, Uri.parse("tel:" + mNumber)));
```