Mobile Application Development

Produced by

Eamonn de Leastar (edeleastar@wit.ie)
Dr. Siobhán Drohan (sdrohan@wit.ie)
David Drohan (ddrohan@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

http://www.wit.ie

http://elearning.wit.ie

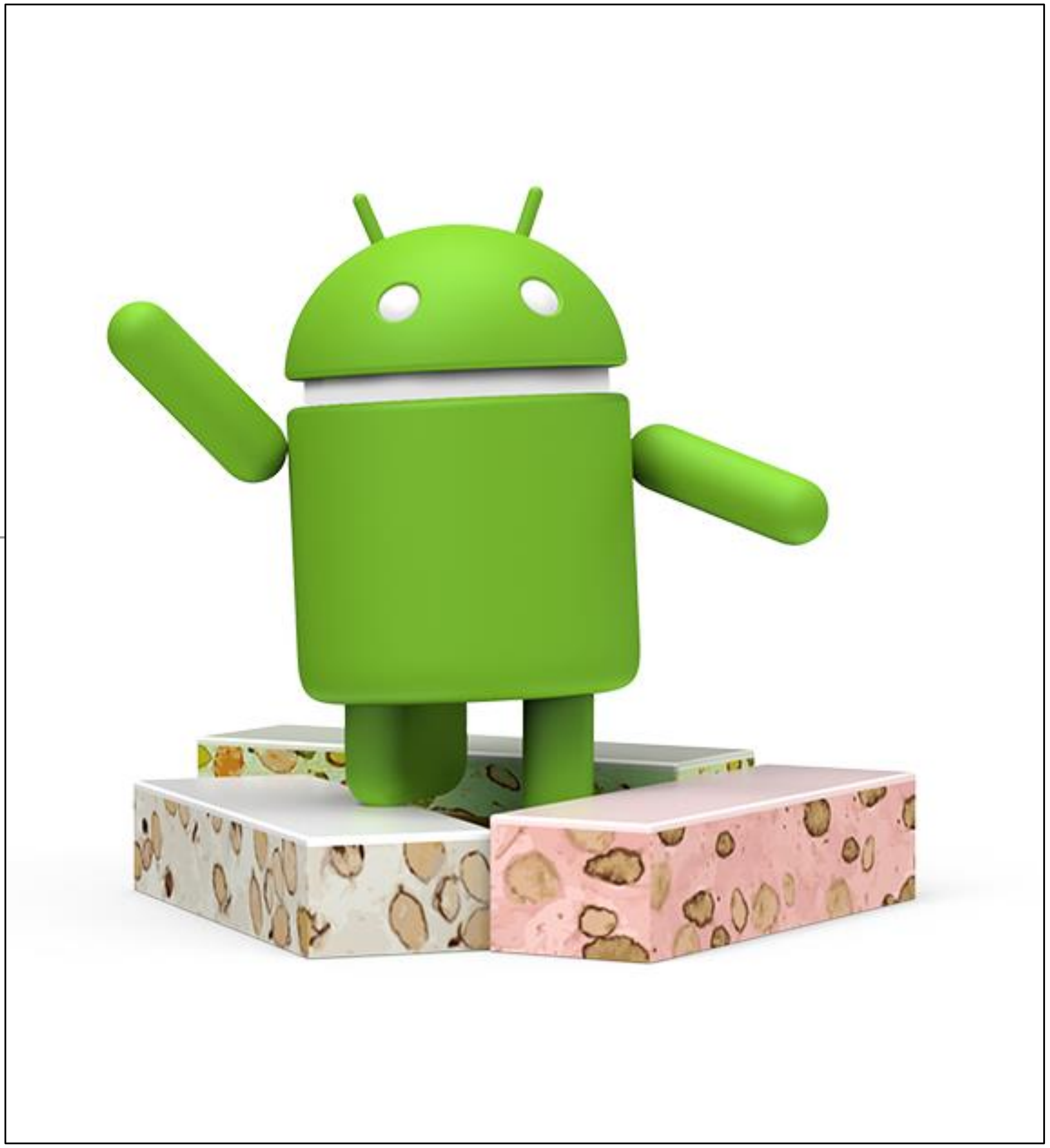
Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

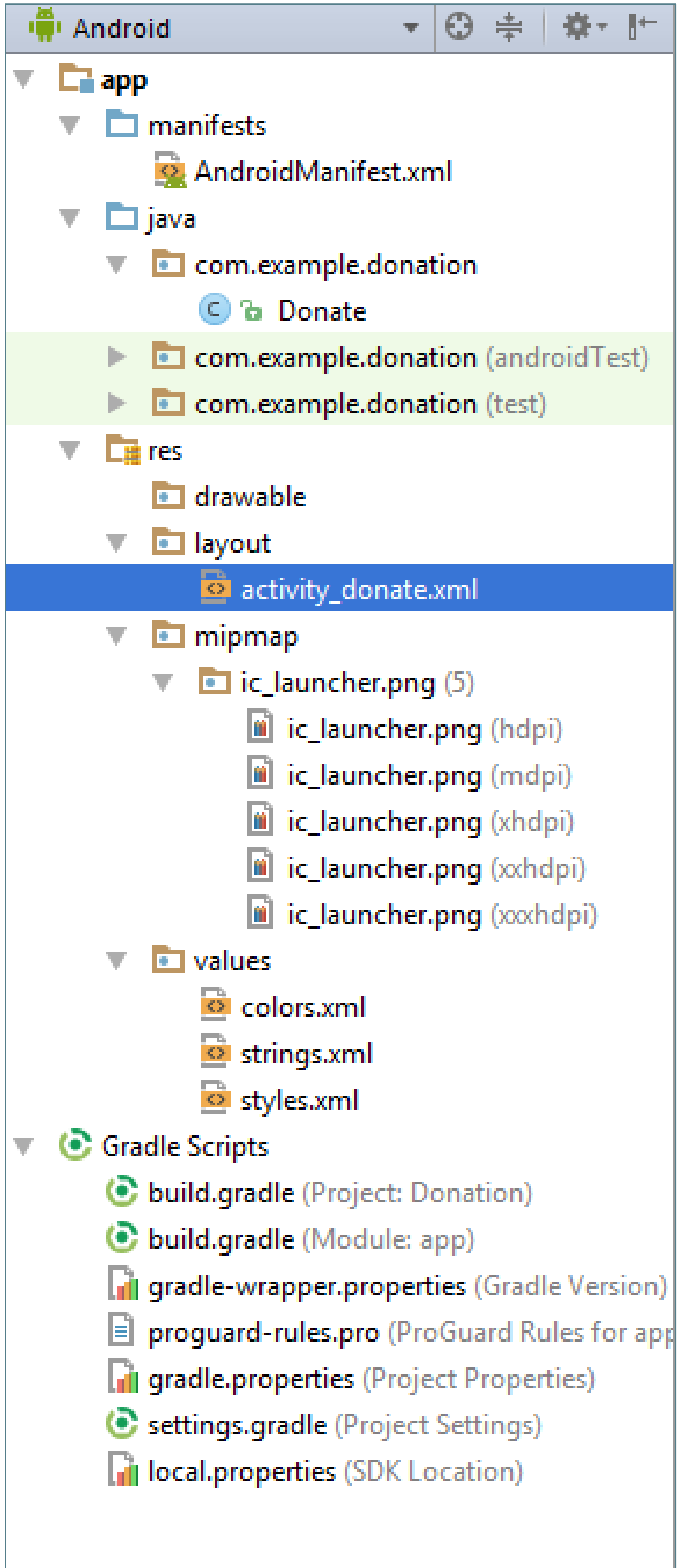
eLearning support unit

A First Android Application

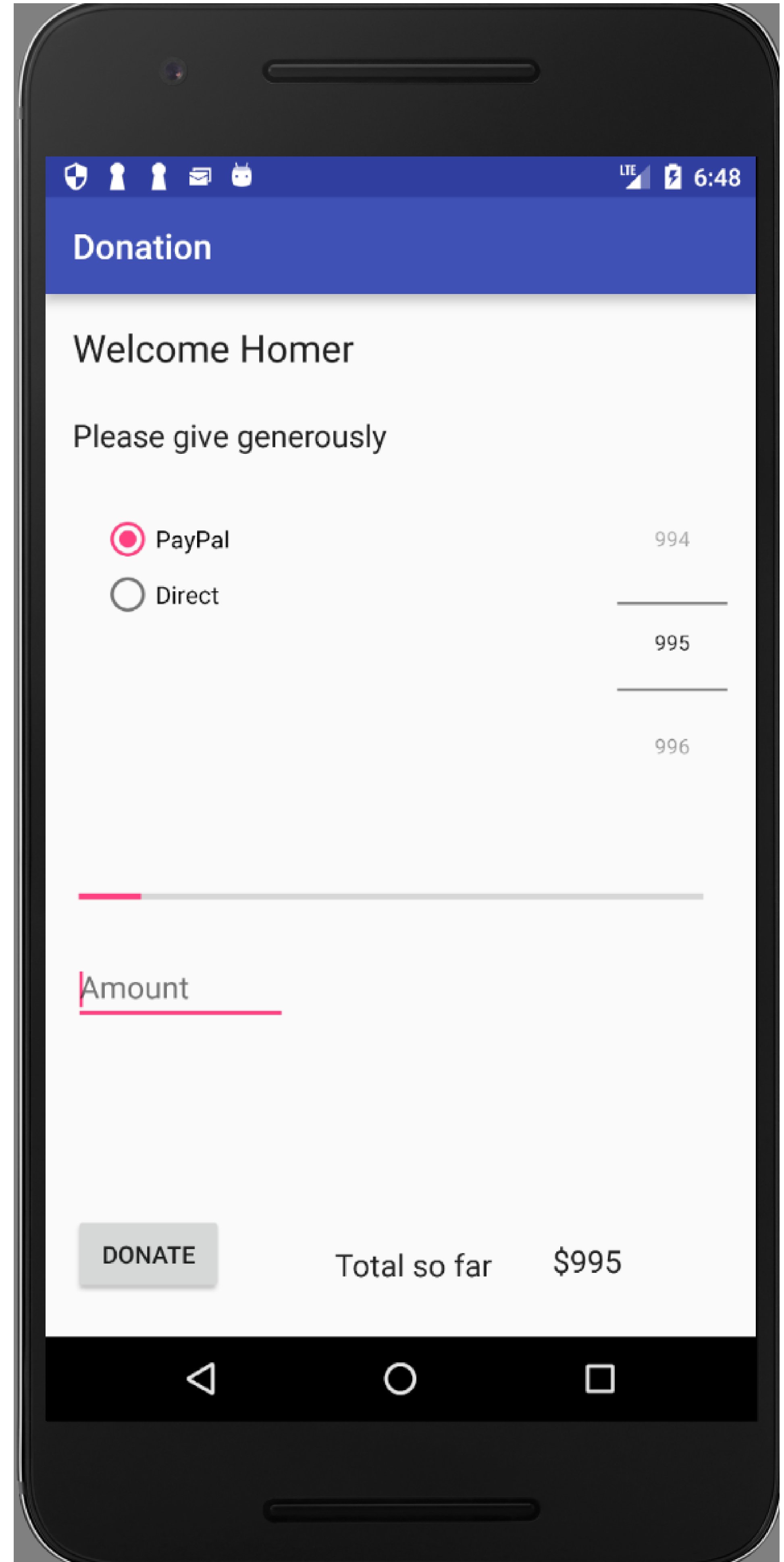
Donation 1.0

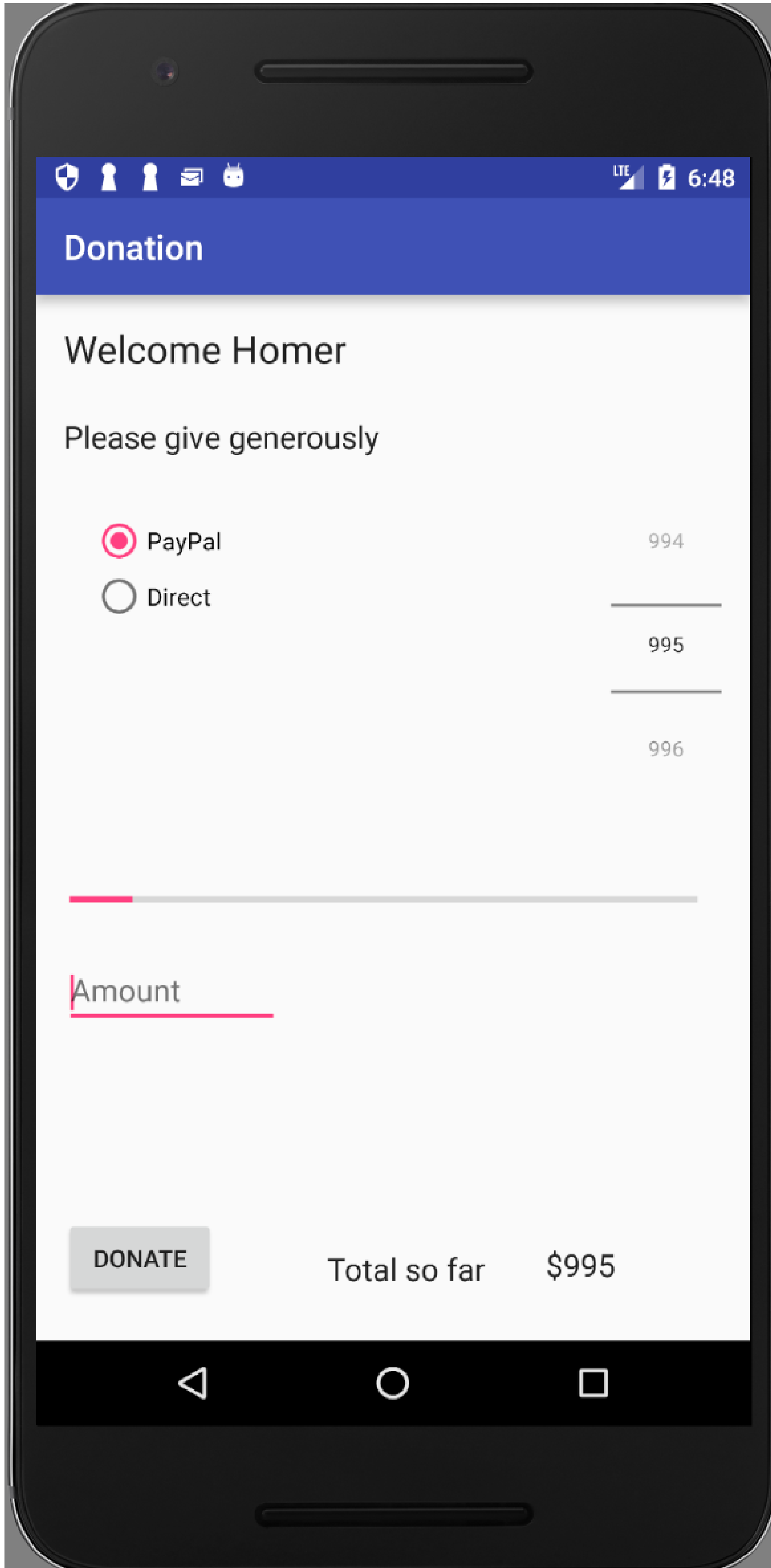
A single activity and layout app.





Donation V1 - a simple activity class and the associated layout.






```java
public class Donate extends AppCompatActivity {

    private int          totalDonated = 0;
    private int          target = 10000;
    private RadioGroup   paymentMethod;
    private ProgressBar  progressBar;
    private NumberPicker amountPicker;
    private EditText     amountText;
    private TextView     amountTotal;

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

        paymentMethod = (RadioGroup)   findViewById(R.id.paymentMethod);
        progressBar   = (ProgressBar)  findViewById(R.id.progressBar);
        amountPicker  = (NumberPicker) findViewById(R.id.amountPicker);
        amountTotal   = (TextView)     findViewById(R.id.amountTotal);
        amountText    = (EditText)     findViewById(R.id.amountText);
        amountPicker.setMinValue(0);
        amountPicker.setMaxValue(1000);
        progressBar.setMax(target);
    }

    public void donateButtonPressed (View view){
        String method = paymentMethod.getCheckedRadioButtonId()
                           == R.id.payPal ? "PayPal" : "Direct";

        int donatedAmount = amountPicker.getValue();
        if (donatedAmount == 0) {
            String text = amountText.getText().toString();
            if (!text.equals(""))
                donatedAmount = Integer.parseInt(text);
        }

        if (totalDonated > target) {
            Toast toast = Toast.makeText(this, "Target Exceeded!", Toast.LENGTH_SHORT);
            toast.show();
            Log.v("Donate","Target Exceeded: " + totalDonated);
        }
        else  {
            totalDonated = totalDonated + donatedAmount;
            progressBar.setProgress(totalDonated);
            Log.v("Donate", amountPicker.getValue() + " donated by " + method
                        + "\nCurrent total " + totalDonated);
        }

        String totalDonatedStr = "$" + totalDonated;
        amountTotal.setText(totalDonatedStr);
    }
}
```

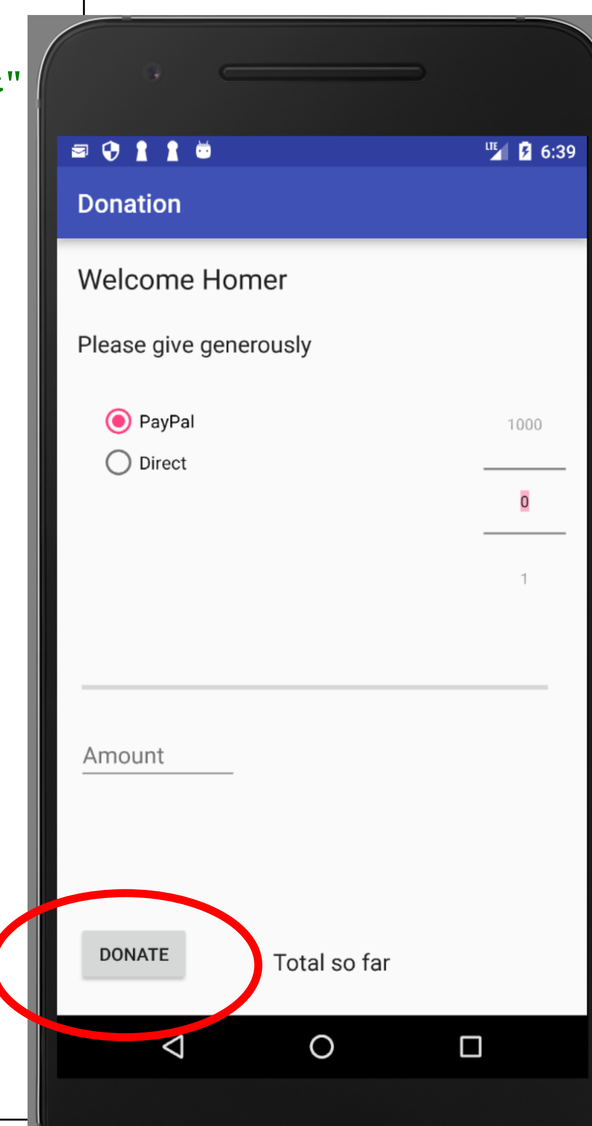
Listeners: Donate button - event handler

activity_donate.xml

```xml
<Button
    android:id="@+id/donateButton"
    android:layout_width="88dp"
    android:layout_height="48dp"
    android:layout_marginBottom="24dp"
    android:text="@string/donateButton"
    app:layout_constraintBottom_toBottomOf="parent"
    android:onClick="donateButtonPressed"
    android:layout_marginLeft="16dp"
    app:layout_constraintLeft_toLeftOf="parent" />
```

```java
public class Donate extends AppCompatActivity {
    private int          totalDonated = 0;
    private int          target = 10000;

    private RadioGroup   paymentMethod;
    private ProgressBar  progressBar;
    private NumberPicker amountPicker;
    private EditText     amountText;
    private TextView     amountTotal;

   protected void onCreate(Bundle savedInstanceState) {
         //code omitted
    }

   public void donateButtonPressed (View view){
        String method = paymentMethod.getCheckedRadioButtonId() == R.id.payPal ? "PayPal" : "Direct"

        int donatedAmount = amountPicker.getValue();
        if (donatedAmount == 0) {
            String text = amountText.getText().toString();
            if (!text.equals(""))
                donatedAmount = Integer.parseInt(text);
        }

        if (totalDonated > target) {
            Toast toast = Toast.makeText(this, "Target Exceeded!", Toast.LENGTH_SHORT);
            toast.show();
            Log.v("Donate","Target Exceeded: " + totalDonated);
        }
        else  {
            totalDonated = totalDonated + donatedAmount;
            progressBar.setProgress(totalDonated);
            Log.v("Donate", amountPicker.getValue() + " donated by " + method
                             + "\nCurrent total " + totalDonated);
        }

        String totalDonatedStr = "$" + totalDonated;
        amountTotal.setText(totalDonatedStr);
    }
}
```

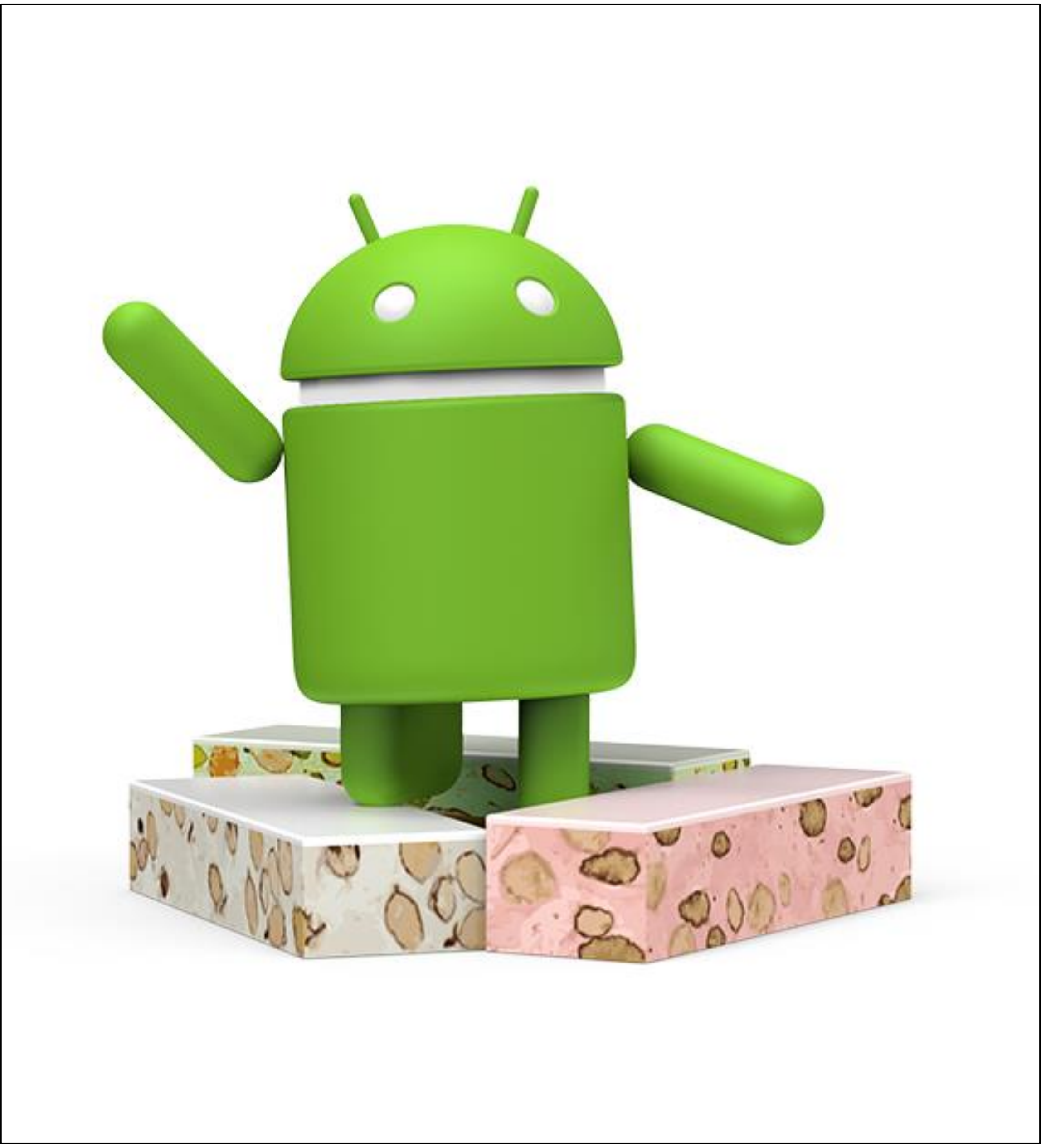
Donate.java



A First Android Application

Donation 2.0

A second activity to display the list of donation made by the user and our first Model class to store these donations.

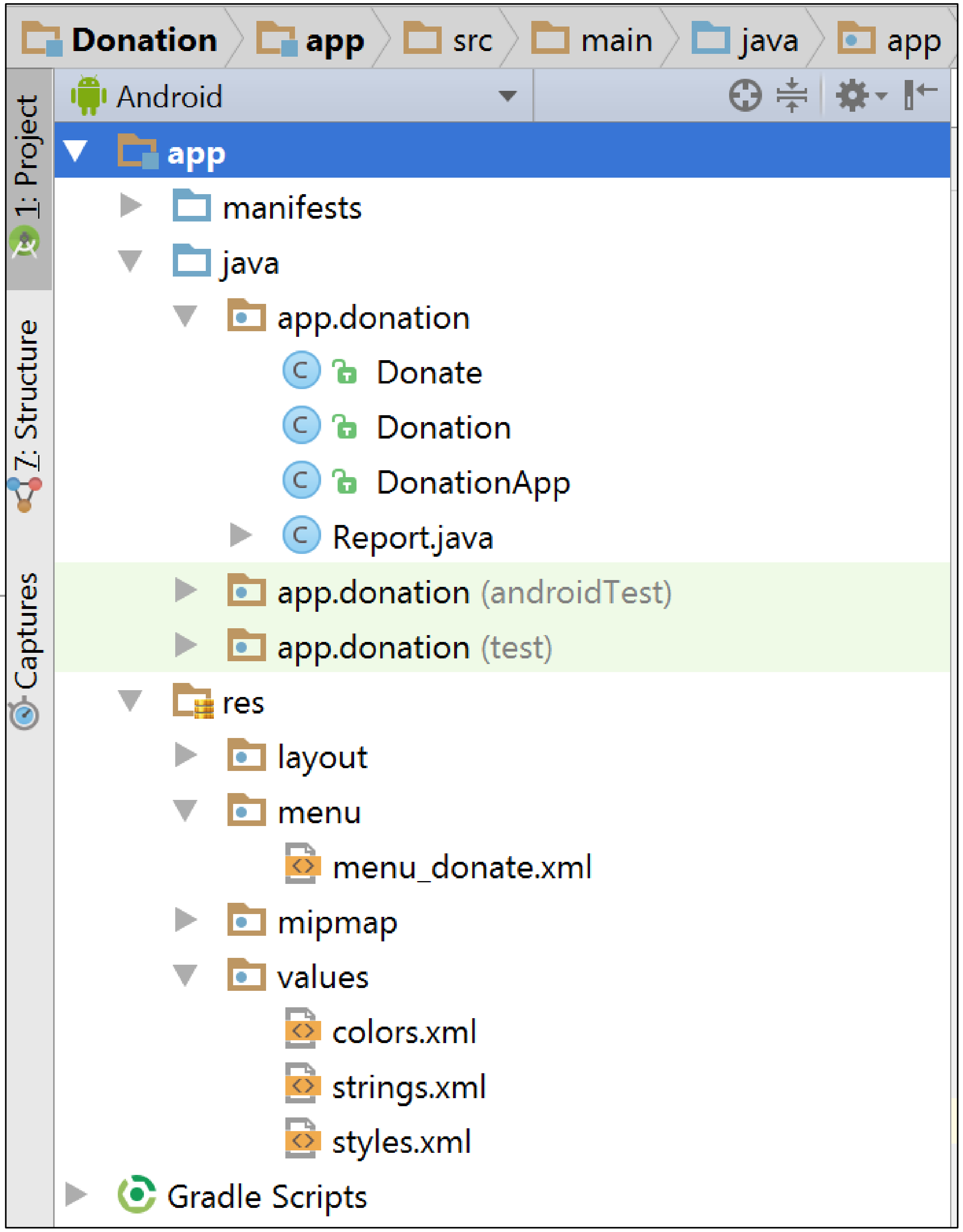


Rename the packages from

com.example.donation

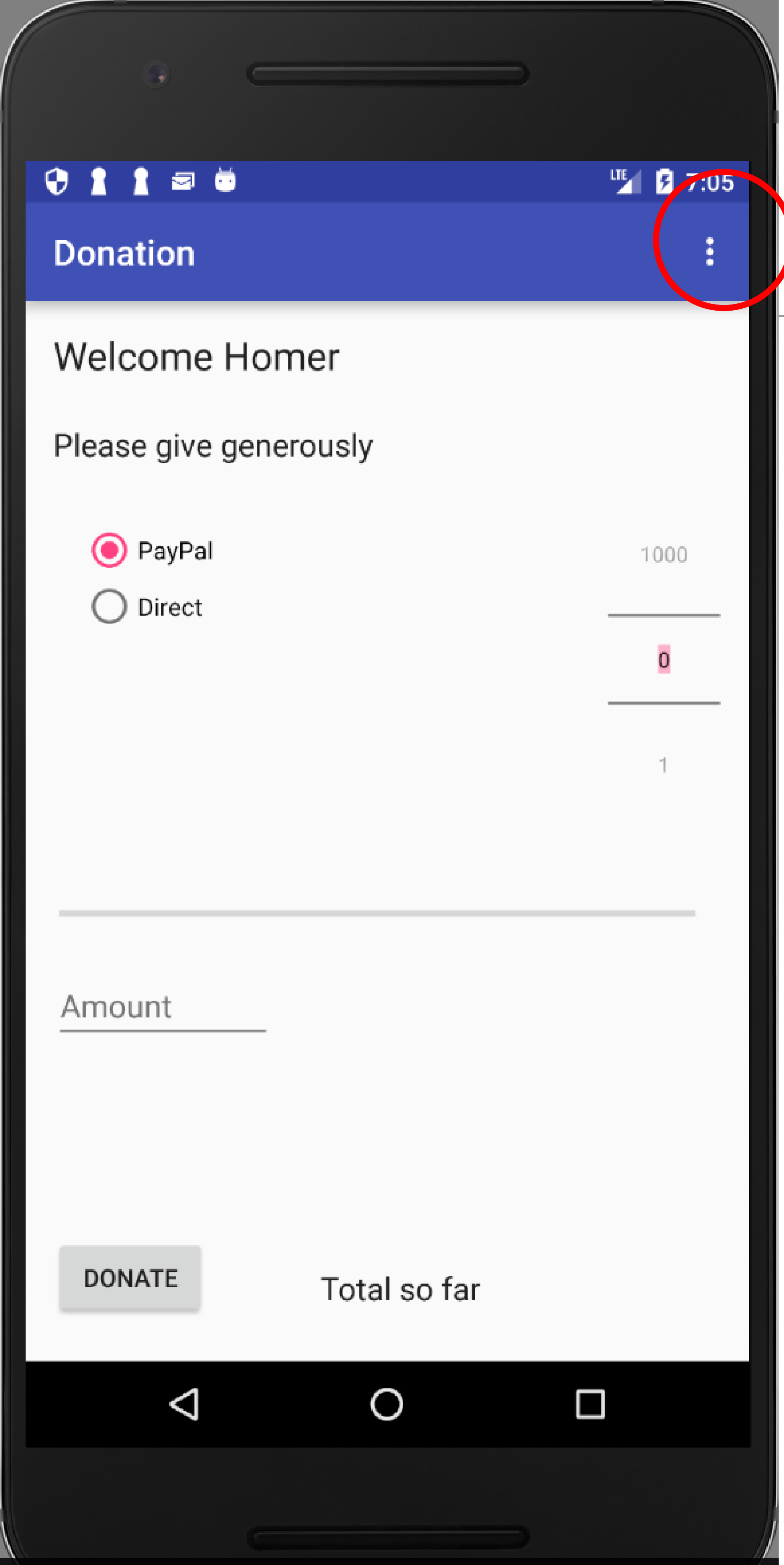
to

app.donation

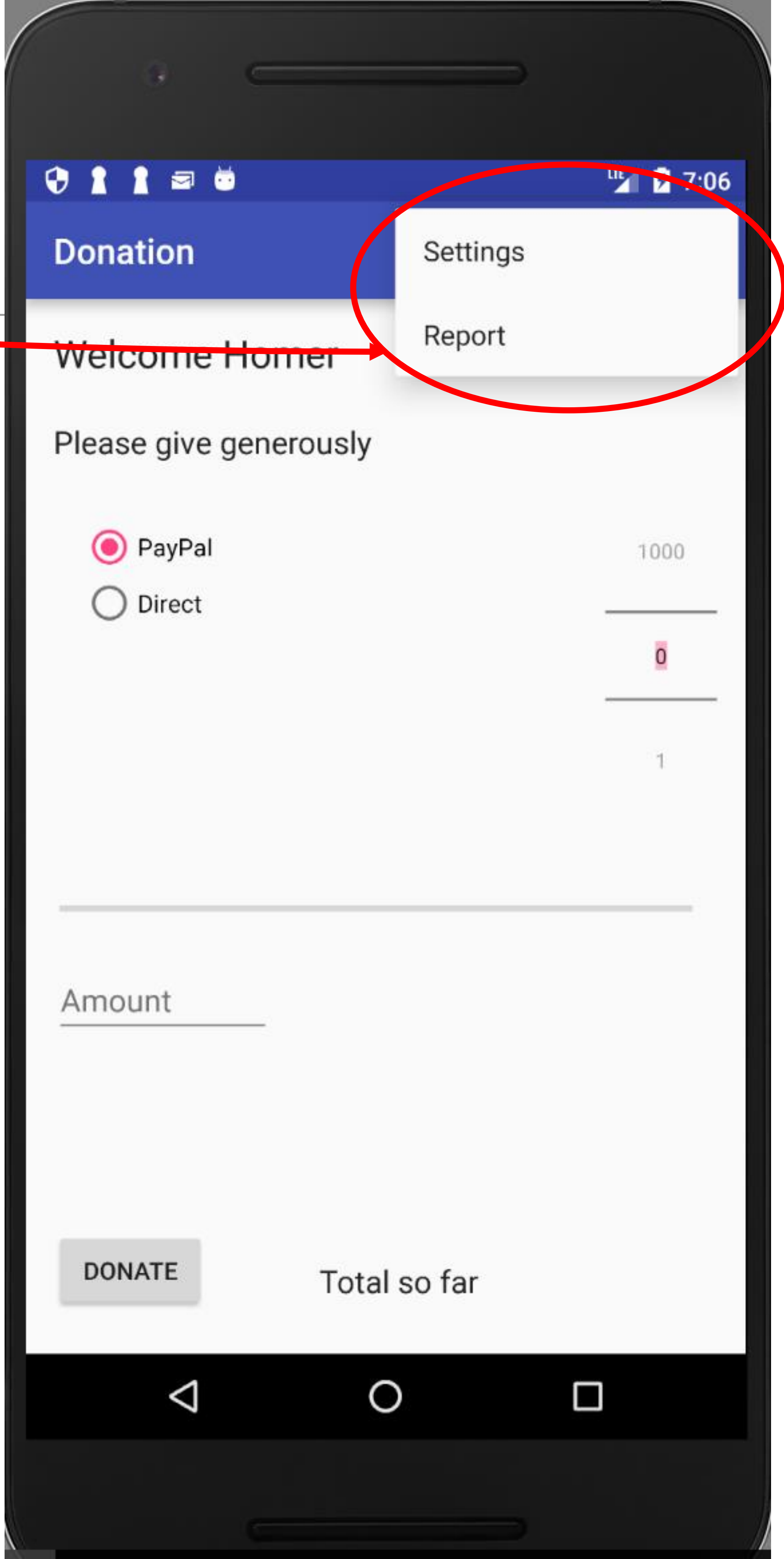


Menus

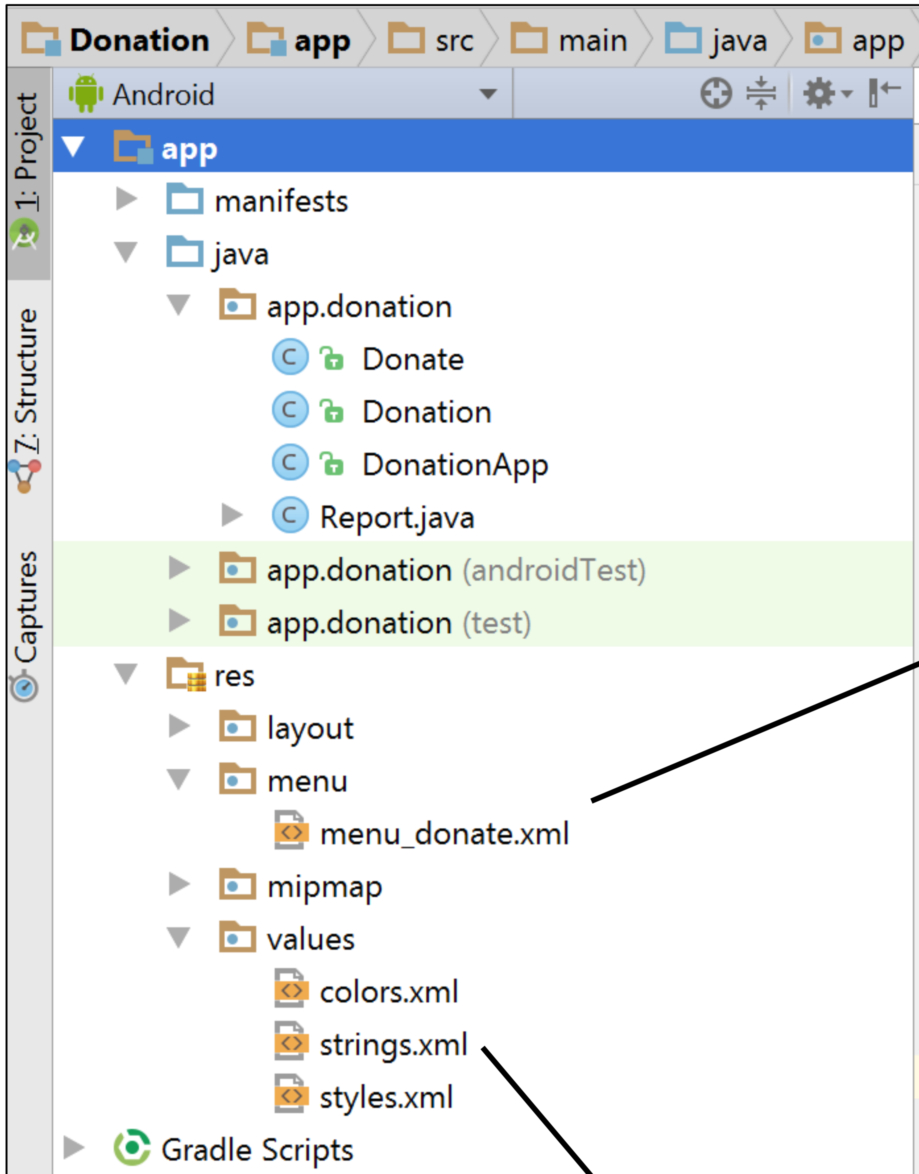
Menus



Pressing the ‘overflow’ icon on the action bar brings up a menu with two entries.



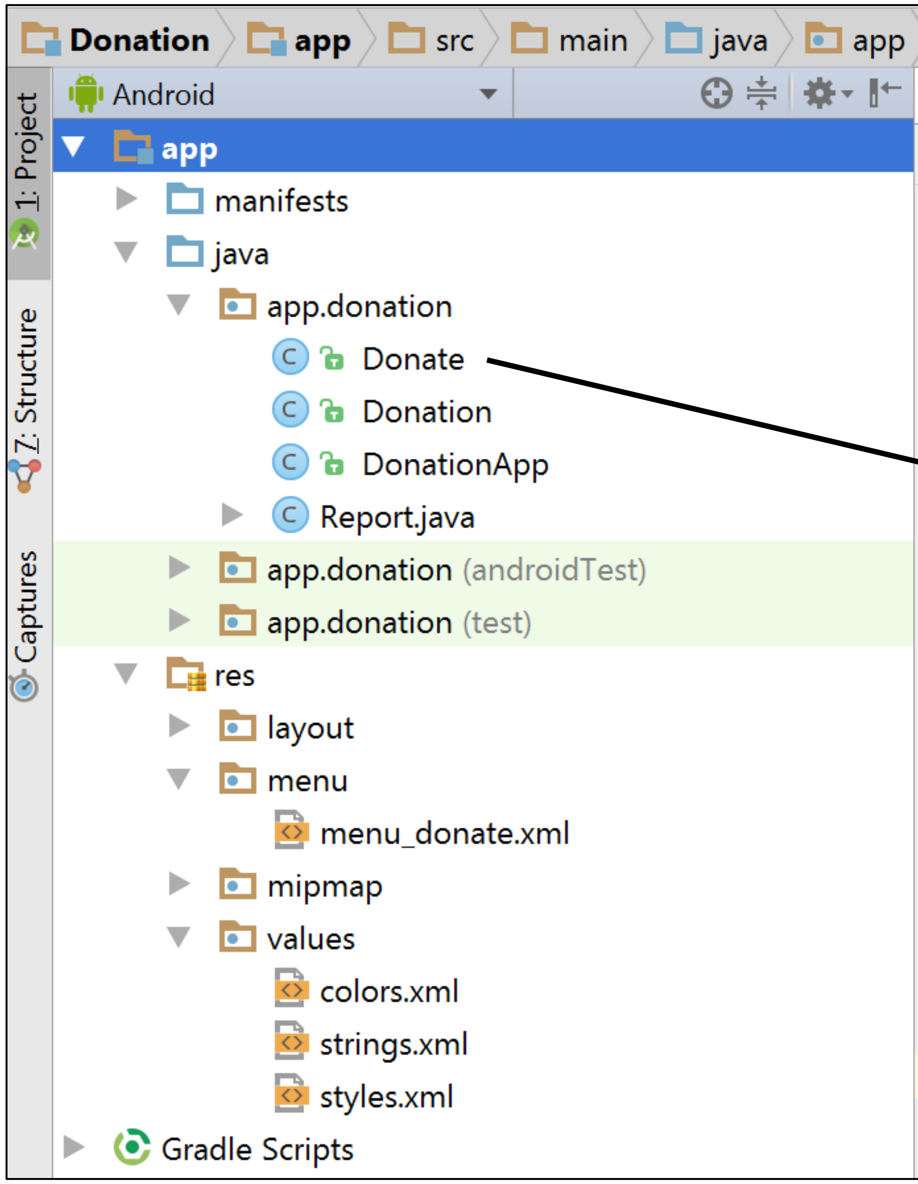
Menu Specification



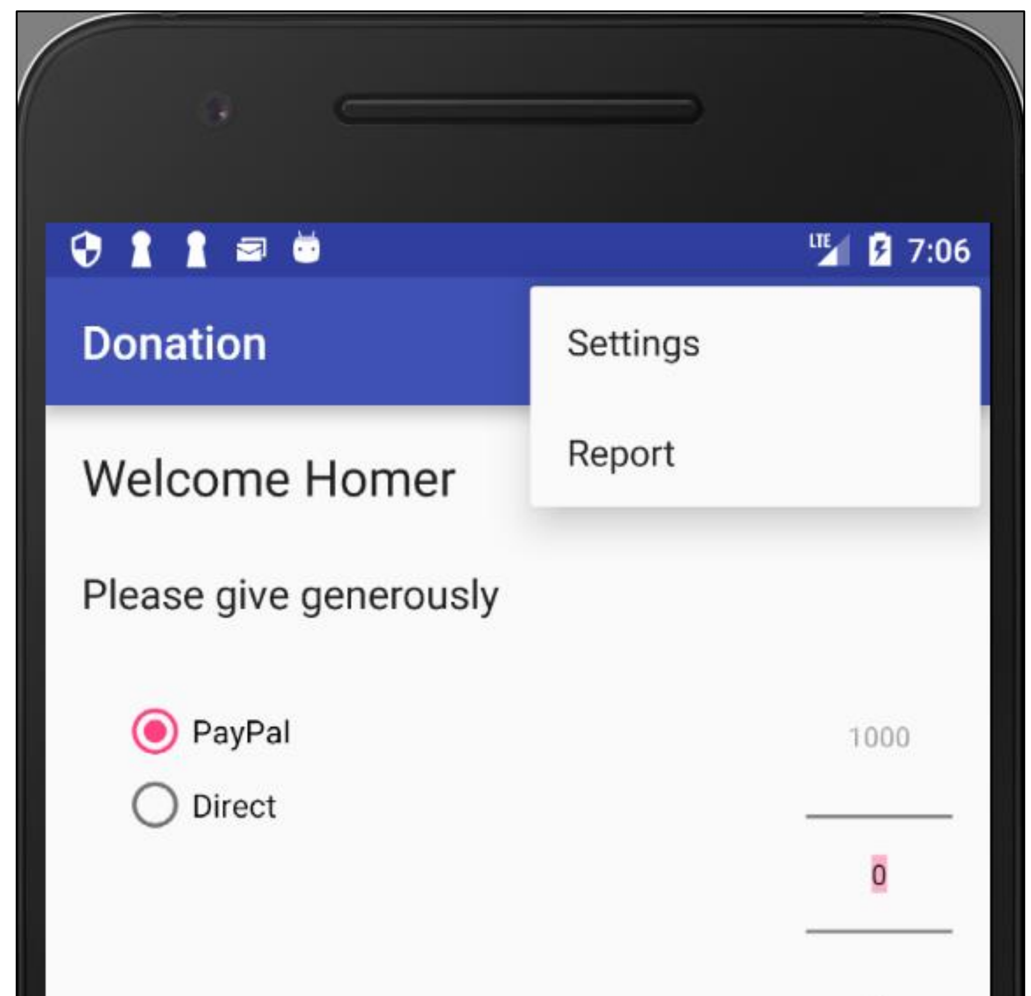

```xml
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <item android:id="@+id/menuSettings"
        android:title="@string/menuSettings"
        android:orderInCategory="100"/>

    <item android:id="@+id/menuReport"
        android:title="@string/menuReport"
        android:orderInCategory="100"/>

</menu>
```

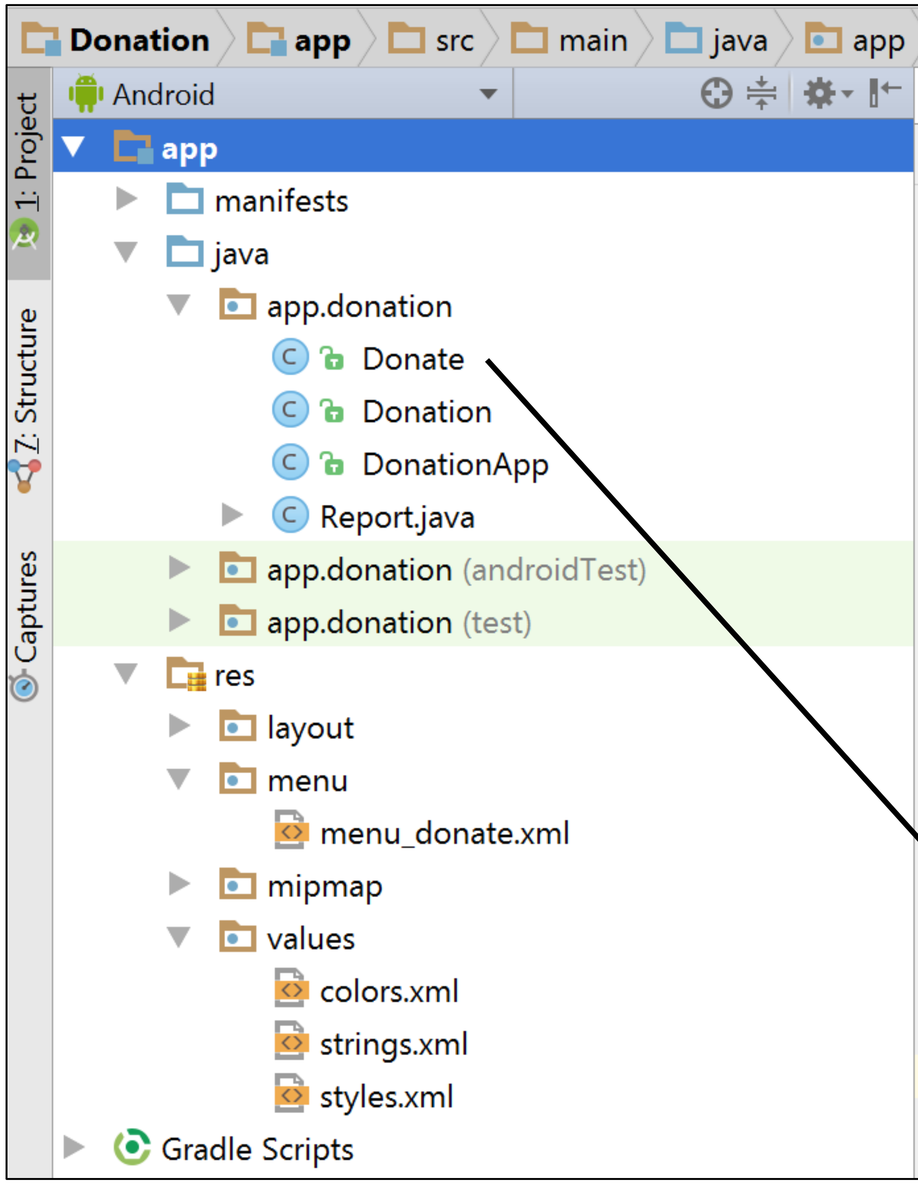
```xml
<resources>
    <string name="app_name">Donation</string>
    <string name="donateTitle">Welcome Homer</string>
    <string name="donateSubtitle">Please give generously</string>
    <string name="donateButton">Donate</string>
    <string name="PayPal">PayPal</string>
    <string name="Direct">Direct</string>
    <string name="Amount">Amount</string>
    <string name="TotalSoFar">Total so far</string>
    <string name="menuSettings">Settings</string>
    <string name="menuReport">Report</string>
    <string name="reportTitle">Report</string>
    <string name="defaultAmount">00</string>
    <string name="defaultMethod">N/A</string>
</resources>
```

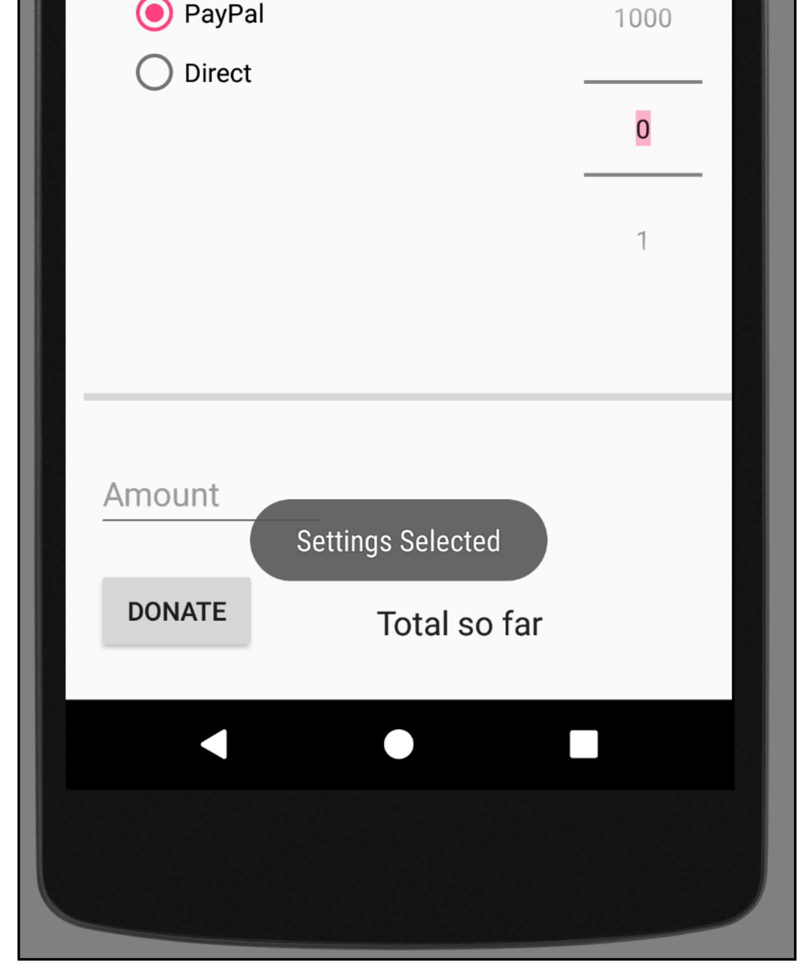
Menu Load




```java
public class Donate extends AppCompatActivity {

//...

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu items for use in the action bar
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu_donate, menu);
        return super.onCreateOptionsMenu(menu);
    }
//...
```







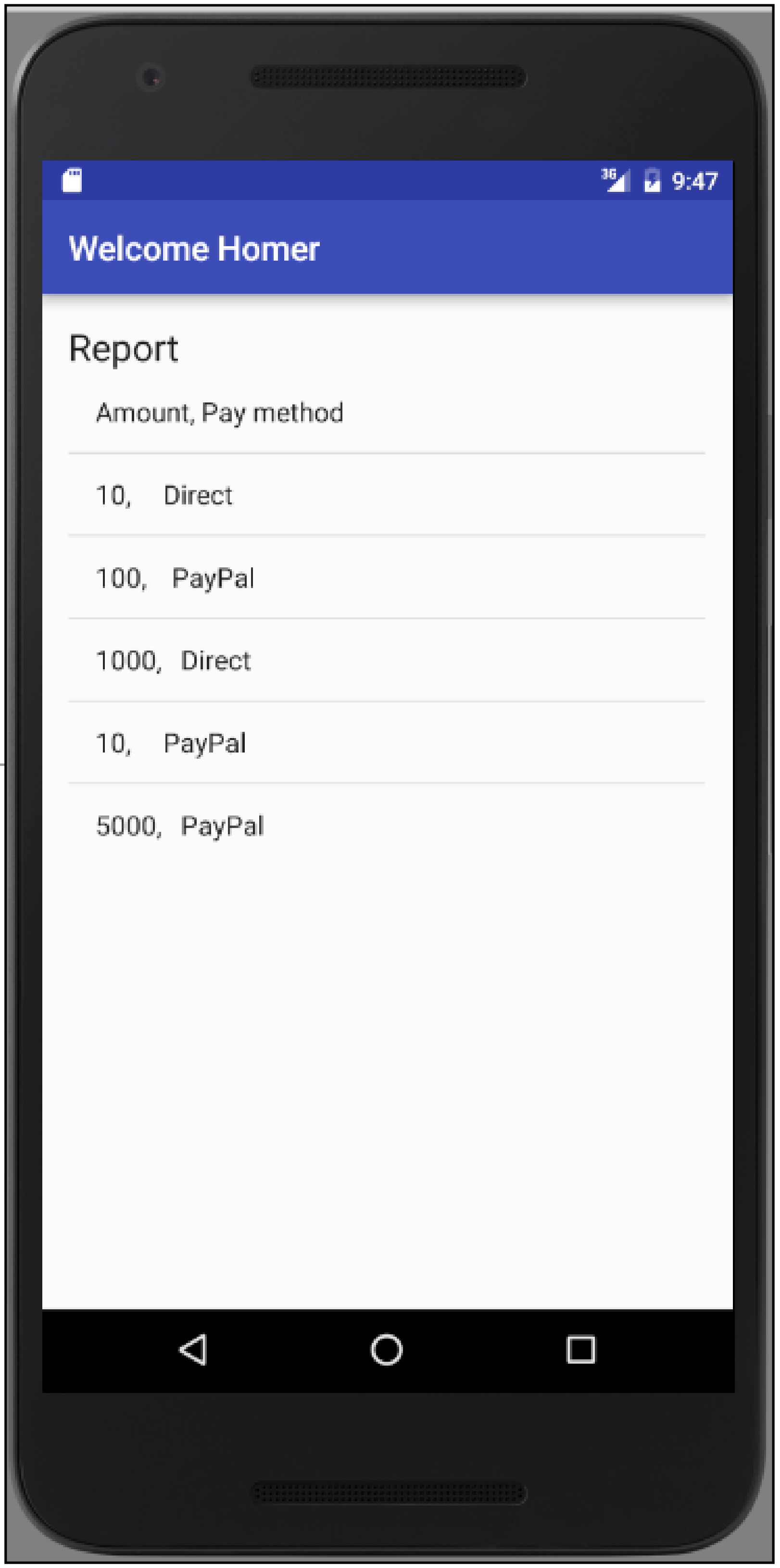
Display ‘Toast’ for a few seconds

Menu Event Handler

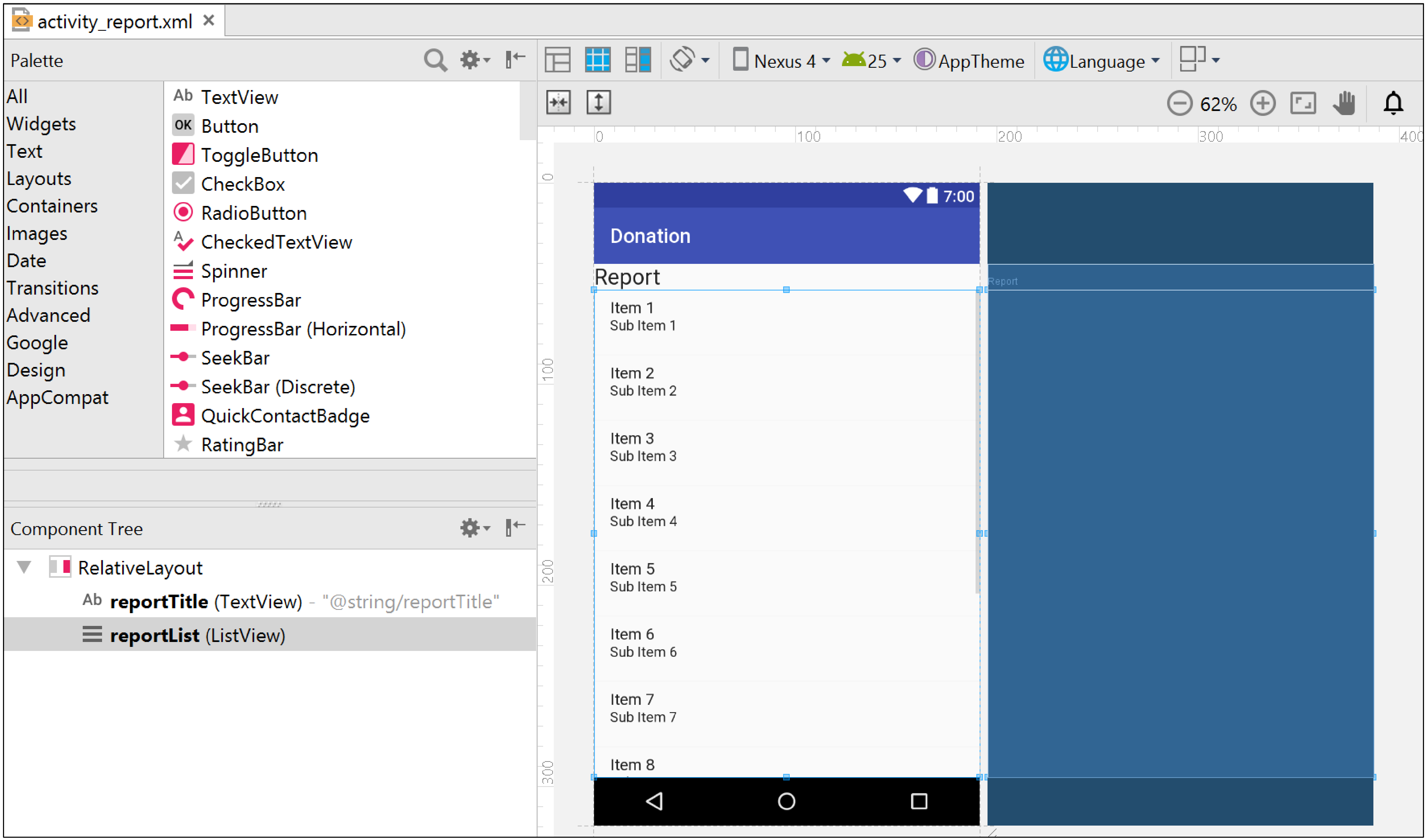
```
public class Donate extends AppCompatActivity {
//…
@Override
public boolean onOptionsItemSelected(MenuItem item){
    switch (item.getItemId()){
        case R.id.menuReport:
            Toast.makeText(this, "Report Selected", Toast.LENGTH_SHORT).show();
        case R.id.menuSettings:
            Toast.makeText(this, "Settings Selected", Toast.LENGTH_SHORT).show();
            break;
    }
    return true;
}
//…
```

New Report Activity – First Draft

…to display a list of hard coded data

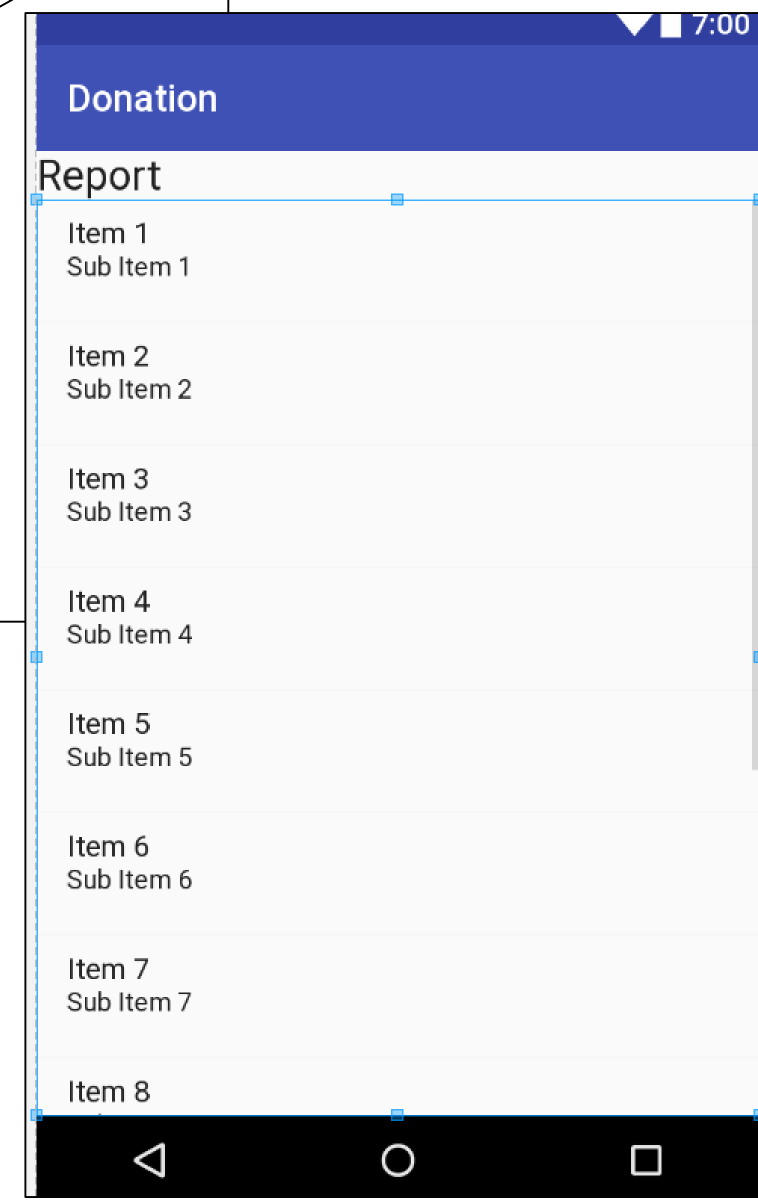


Design for the New Report activity



```xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/reportTitle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentTop="true"
        android:text="@string/reportTitle"
        android:textAppearance="?android:attr/textAppearanceLarge"/>

    <ListView
        android:id="@+id/reportList"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignStart="@+id/reportTitle"
        android:layout_below="@+id/reportTitle" >
    </ListView>

</RelativeLayout>
```

res/layout/activity_report.xml

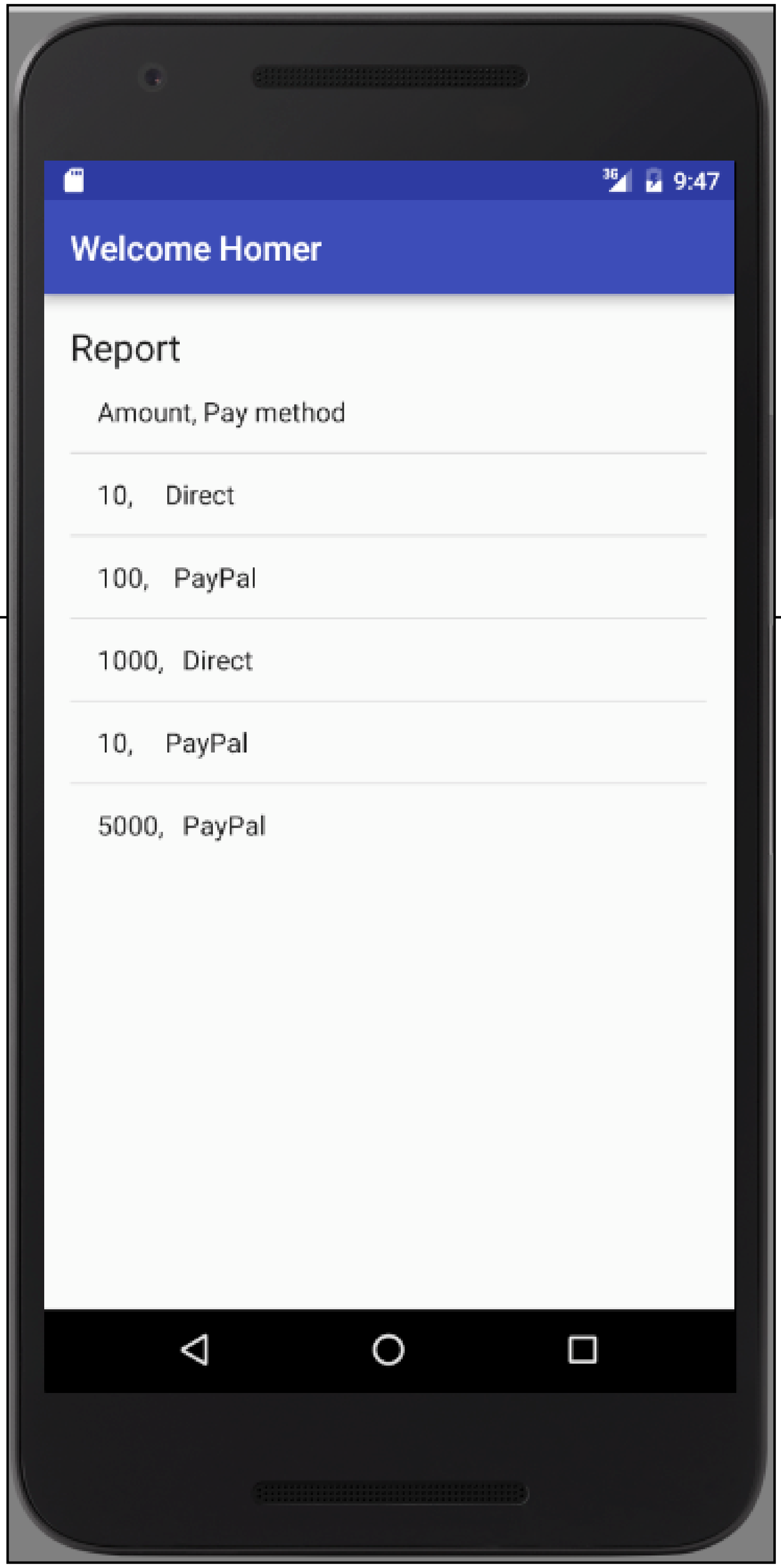


`ListView` is a view group that displays a list of scrollable items.

The list items are automatically inserted to the list using an `Adapter` that pulls content from a source such as an array or database query and converts each item result into a view that's placed into the list.

Activity Report

First draft displays hardcoded data

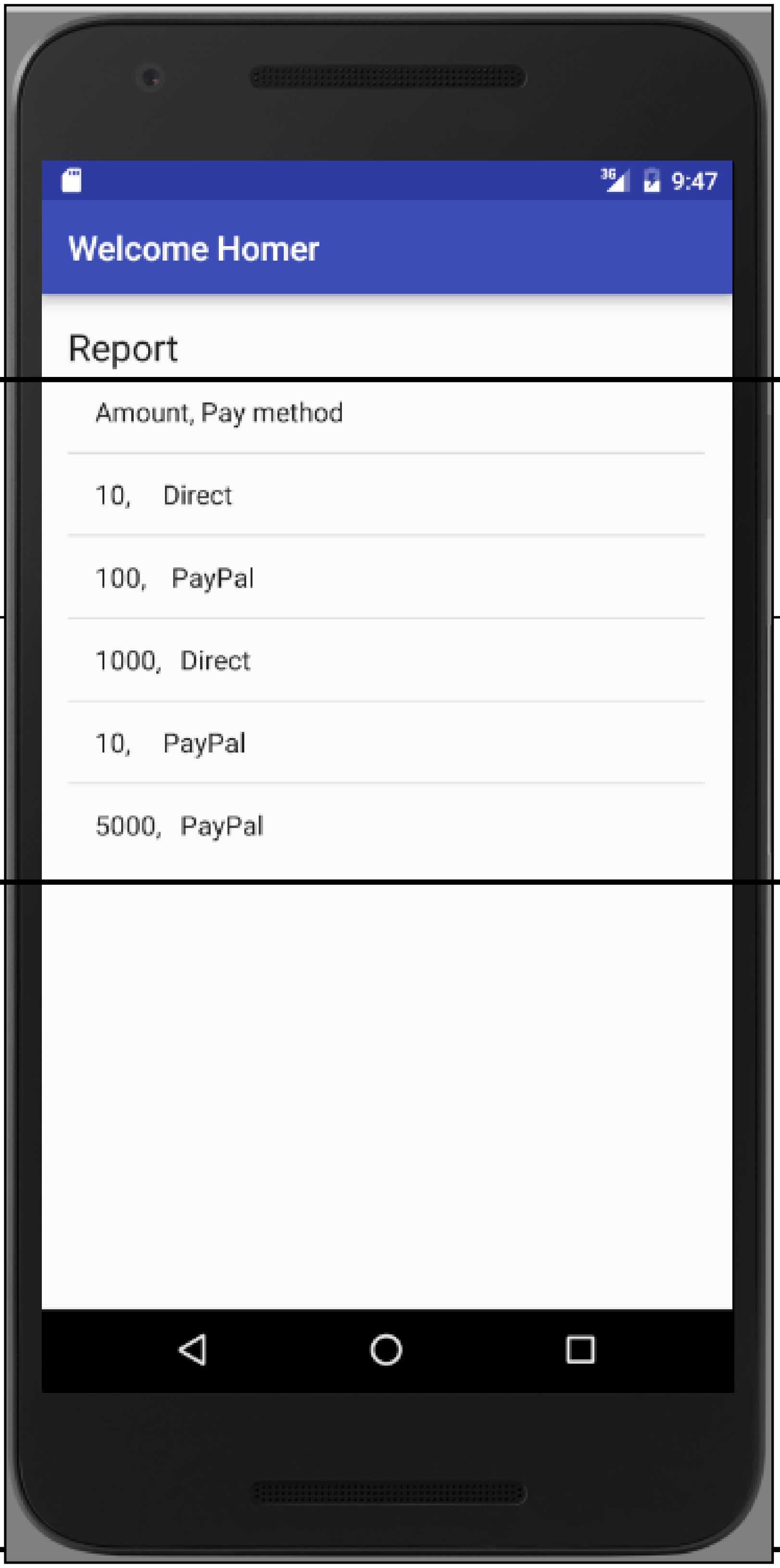



```java
public class Report extends AppCompatActivity
{
  ListView listView;

  static final String[] numbers = new String[] {
     "Amount, Pay method",
     "10,     Direct",
     "100,    PayPal",
     "1000,   Direct",
     "10,     PayPal",
     "5000,   PayPal"};

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

    listView = (ListView) findViewById(R.id.reportList);
    ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,  android.R.layout.simple_list_item_1, numbers);

    listView.setAdapter(adapter);
  }
}
```

Activity Report

First draft displays hardcoded data

```java
public class Report extends AppCompatActivity
{
  ListView listView;

  static final String[] numbers = new String[] {
     "Amount, Pay method",
     "10,     Direct",
     "100,    PayPal",
     "1000,   Direct",
     "10,     PayPal",
     "5000,   PayPal"};

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

    listView = (ListView) findViewById(R.id.reportList);
    ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,  android.R.layout.simple_list_item_1, numbers);

    listView.setAdapter(adapter);
  }
}
```



ArrayAdapter

An **adapter** is the bridge between a UI component and its data source.

An **ArrayAdapter** is commonly used in Android. It returns a view for each object in a collection of data objects you provide, and can be used with list-based user interface widgets such as **ListView** or **Spinner**.

```
static final String[] numbers = new String[] {
    "Amount, Pay method",
    "10,    Direct",
    "100,   PayPal",
    "1000,  Direct",
    "10,    PayPal",
    "5000,  PayPal"};
```

```
listView = (ListView) findViewById(R.id.reportList);

ArrayAdapter<String> adapter
        = new ArrayAdapter<String>(this,  android.R.layout.simple_list_item_1, numbers);

listView.setAdapter(adapter);
```

https://developer.android.com/reference/android/widget/ArrayAdapter.html

ArrayAdapter

- android.R.layout contains publicly available layouts that Android uses to display various items.
- android.R.layout.simple_list_item_1 a simple layout to display a single string; saves you having to write simple layouts when using adapters.

```
listView = (ListView) findViewById(R.id.reportList);

ArrayAdapter<String> adapter
        = new ArrayAdapter<String>(this,  android.R.layout.simple_list_item_1, numbers);

listView.setAdapter(adapter);
```

For the new Report activity to display…

…we need to start a new Intent in Donate.java:

```
@Override
public boolean onOptionsItemSelected(MenuItem item)
{
    switch (item.getItemId())
    {
        case R.id.menuReport:
            startActivity(new Intent(this, Report.class));
            break;
        case R.id.menuSettings:
            Toast.makeText(this, "Settings Selected", Toast.LENGTH_SHORT).show();
            break;
    }
    return true;
}
```

…and add the Report activity to AndroidManifest.xml:

```
<activity
    android:name="app.donation.Report"
    android:label="@string/donateTitle" >
</activity>
```

Intent

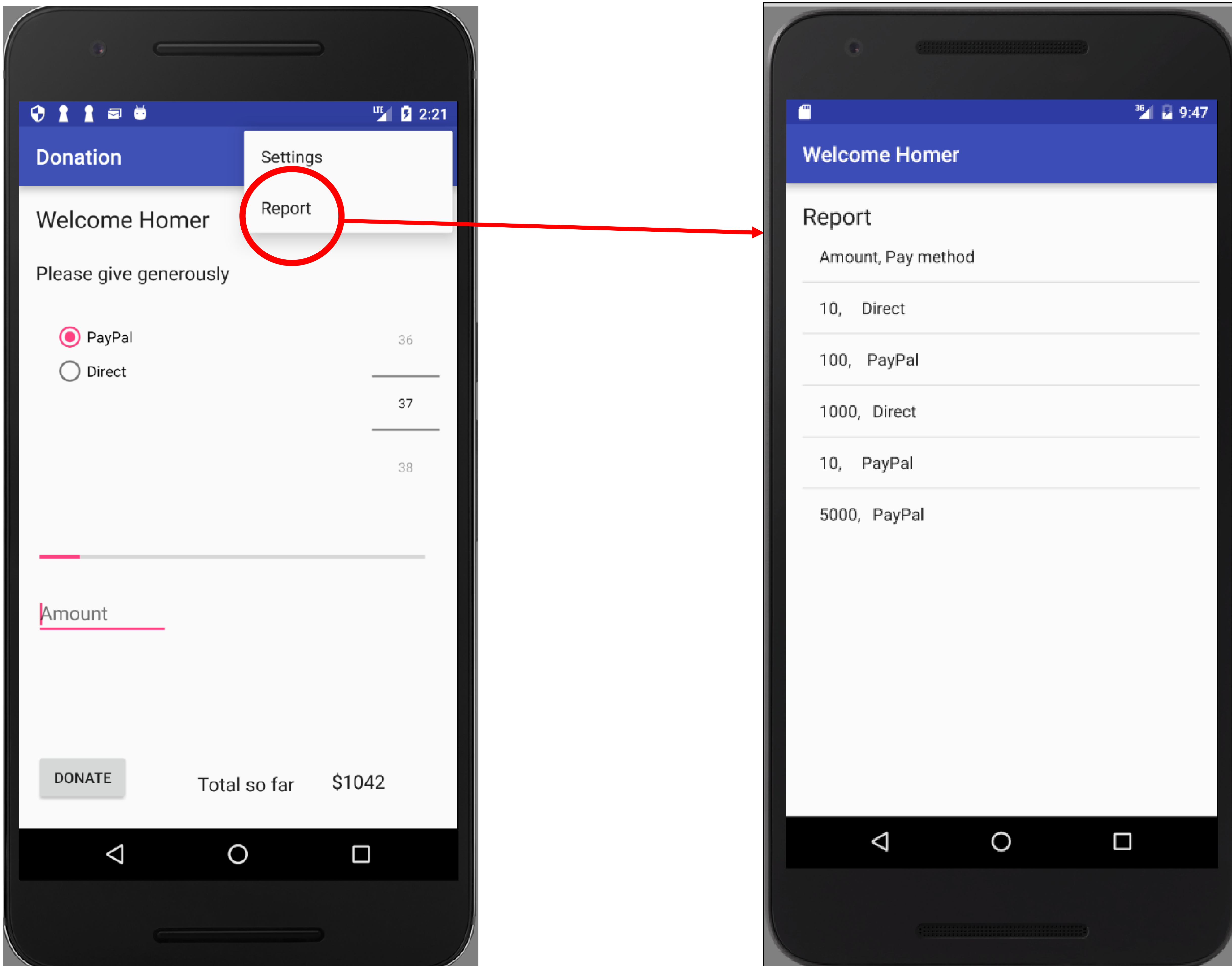
An `Intent` is a messaging object you can use to request an action from another app component. Three fundamental uses:

1. Starting an Activity (i.e. a single screen)
   start a new instance of an `Activity` by passing an `Intent` to `startActivity()`. The `Intent` describes the activity to start and carries any necessary data.

2. Starting a Service – background operations not requiring a UI e.g. Internet downloads, data processing.

3. Delivering a Broadcast – a message that any app can receive e.g. device has started charging.

https://developer.android.com/guide/components/intents-filters.html


2:21
Donation
Settings
Report
Welcome Homer
Please give generously
PayPal
Direct
36
37
38
Amount
DONATE
Total so far
$1042
9:47
Welcome Homer
Report
Amount, Pay method
10, Direct
100, PayPal
1000, Direct
10, PayPal
5000, PayPal

Android's Application Object

A base class for maintaining global application state

Application Object in Android

- Base class for maintaining global application state.

- You can provide your own implementation by creating a subclass and specifying the fully-qualified name of this subclass as the `"android:name"` attribute in your AndroidManifest.xml's `<application>` tag.

- The Application class, or your subclass of the Application class, is instantiated before any other class when the process for your application/package is created.

https://developer.android.com/reference/android/app/Application.html

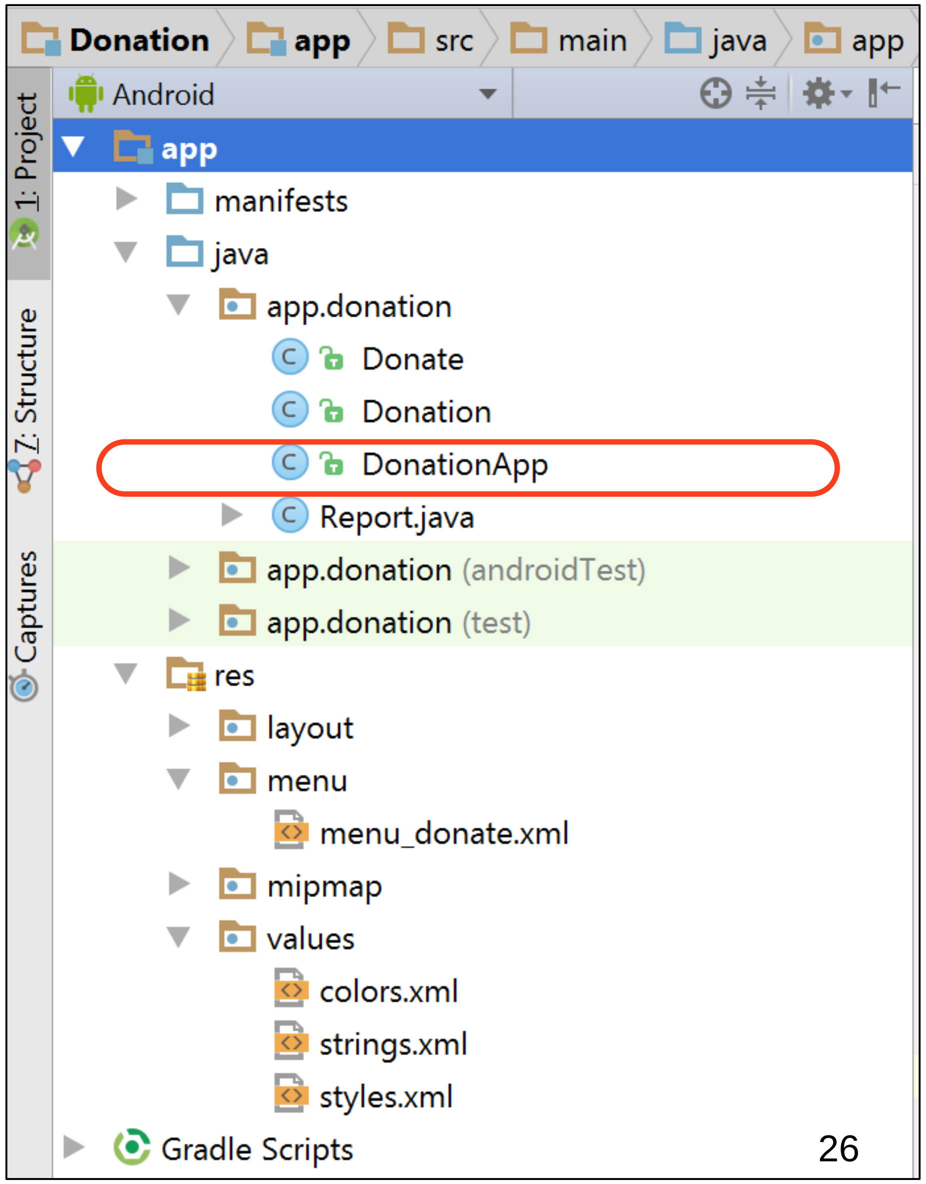
Partial AndroidManifest.xml

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/AppTheme"
    android:name="app.donation.DonationApp">
```

Application Object, V1.0

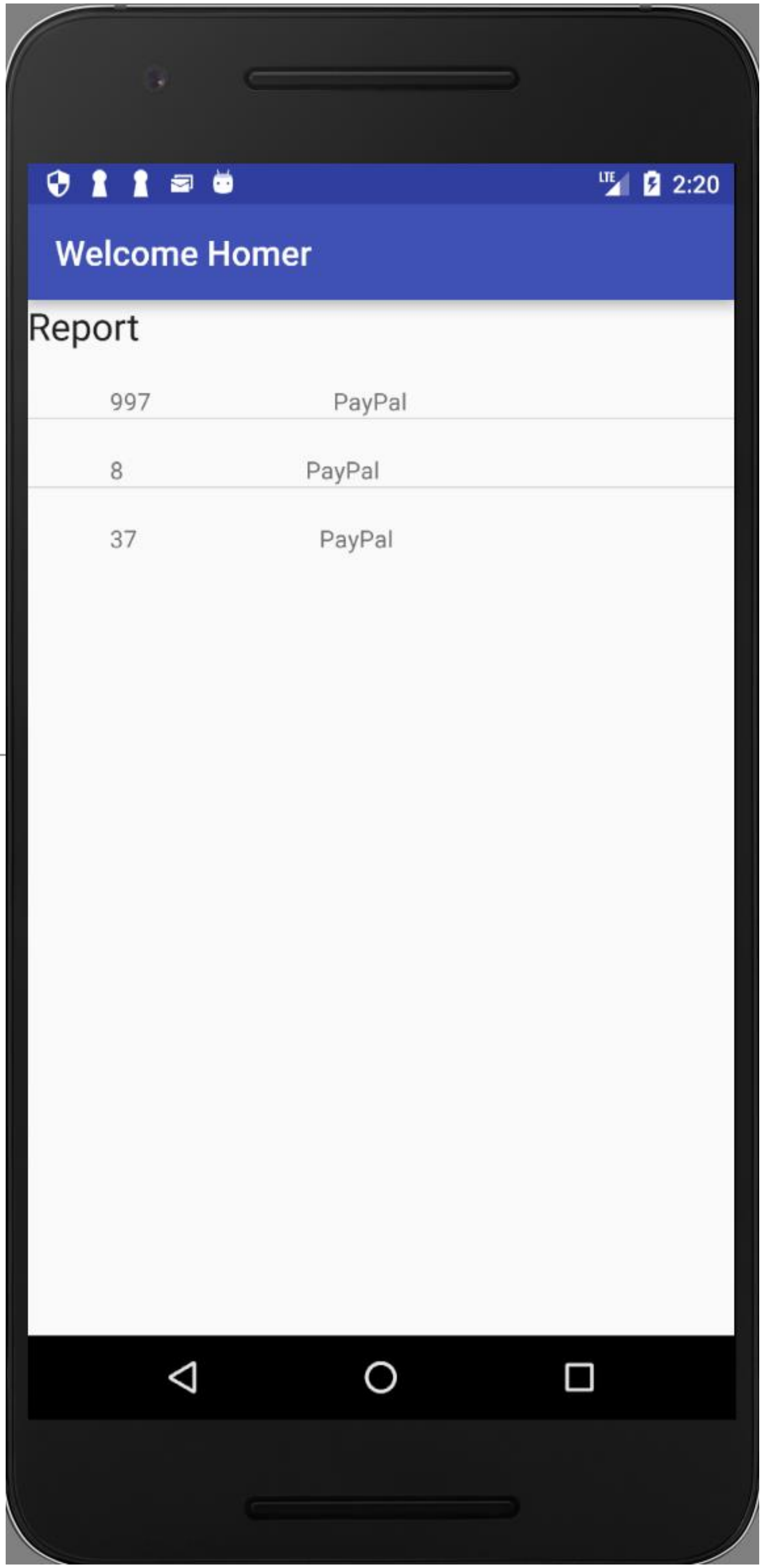
```java
package app.donation;

import android.app.Application;
import android.util.Log;


public class DonationApp extends Application{

    @Override
    public void onCreate()
    {
        super.onCreate();
        Log.v("Donate","Donation App Started");
    }
}
```

- Activities come and go based on user interaction.
- Application objects can be a useful ‘anchor’ for an android app.
- Use it to hold information shared by all activities.



New Report Activity – Second Draft

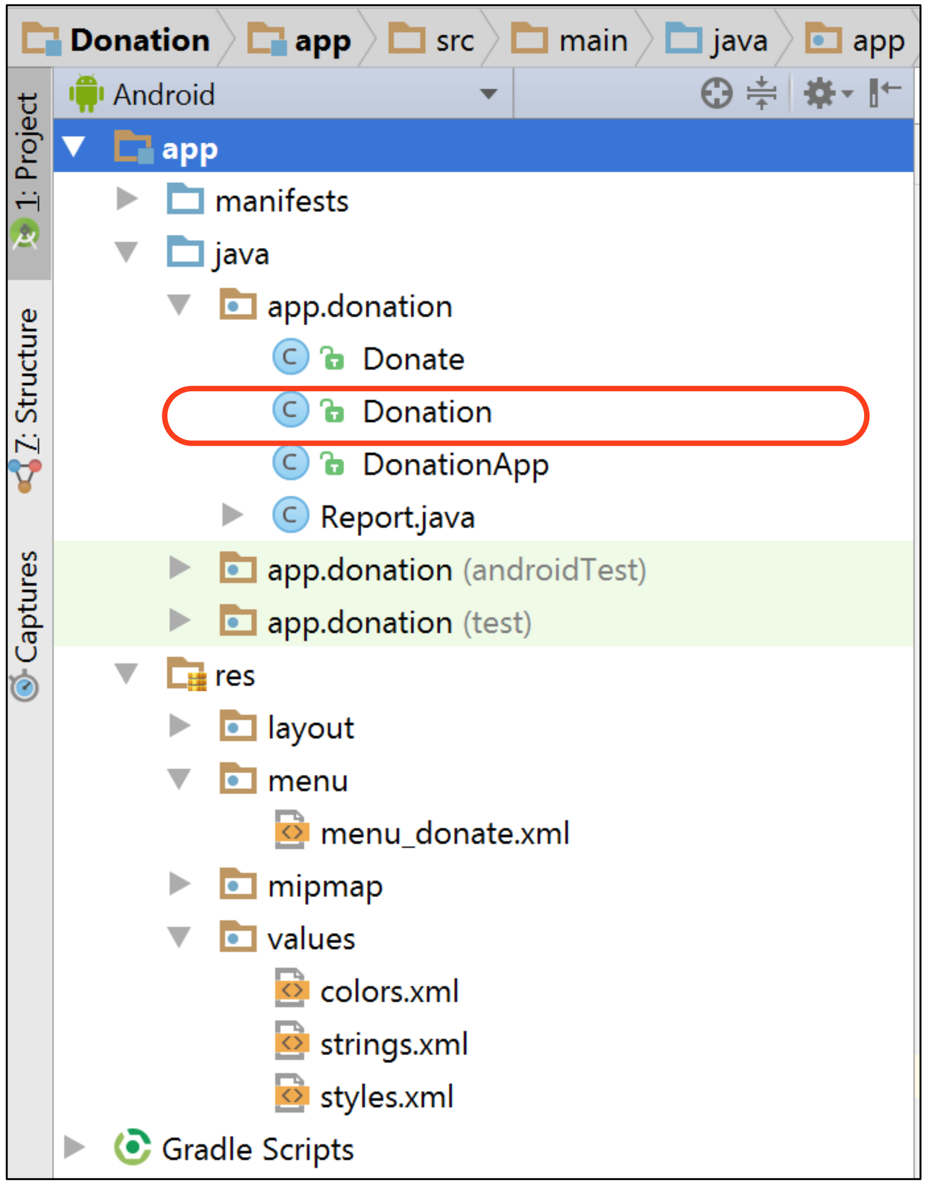
…to display a list of donations made by the user



We need a Model to store the donation data

Create a model class, **Donation**
(similar to play framework models)

```
public class Donation {

    public int amount;
    public String method;

    public Donation (int amount, String method) {
        this.amount = amount;
        this.method = method;
    }
}
```

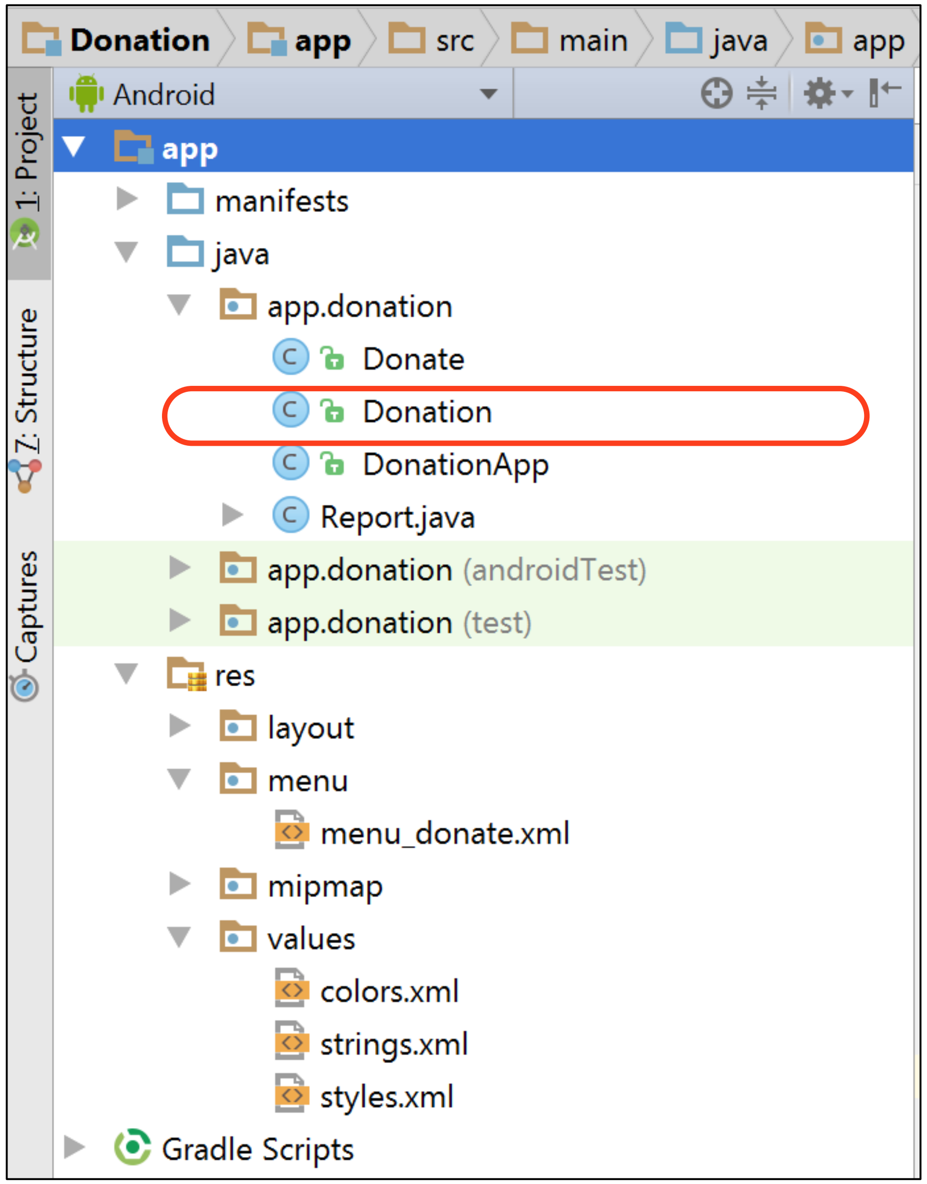


Donation model

Where is the standard boilerplate code?

- Accessors ?
- Mutators ?
- Public access fields ?

```
public class Donation {

    public int amount;
    public String method;

    public Donation (int amount, String method) {
        this.amount = amount;
        this.method = method;
    }
}
```



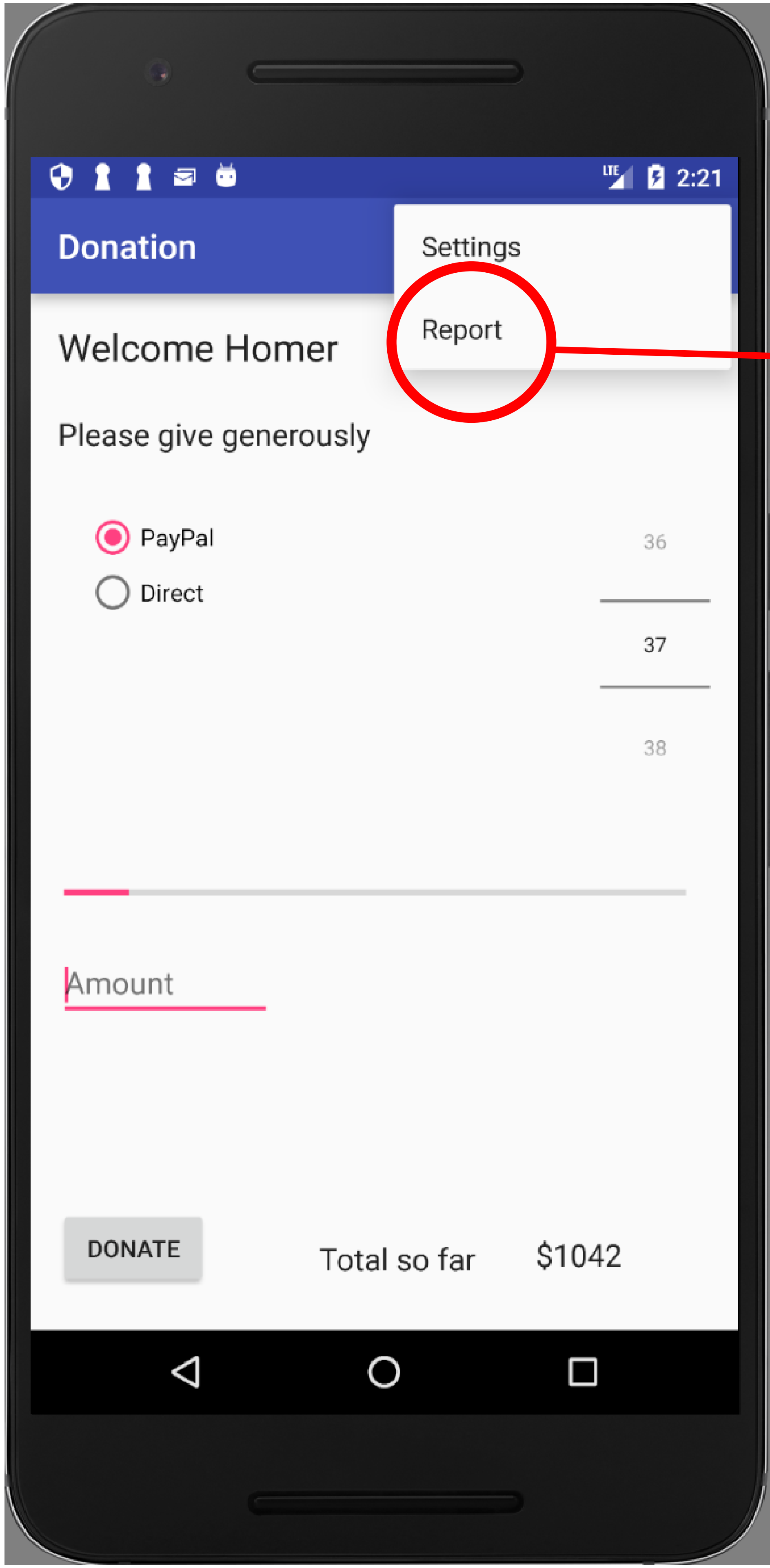
Donation model

- **Performance**! Calling a getter is 3X longer than directly accessing a field.

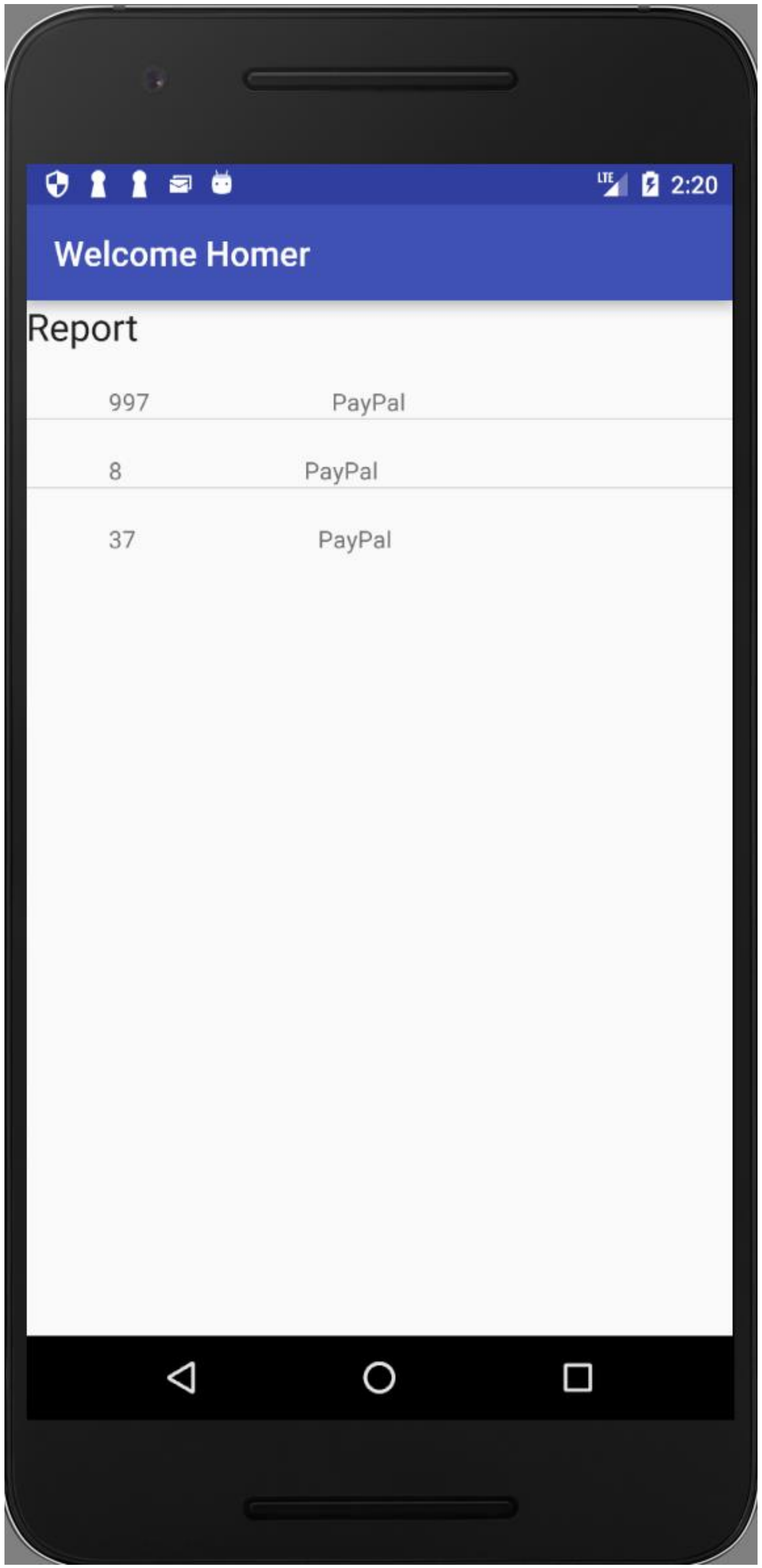
- A trend in Android is the avoidance of unnecessary boilerplate methods where possible.

```
public class Donation {

    public int amount;
    public String method;

    public Donation (int amount, String method) {
        this.amount = amount;
        this.method = method;
    }
}
```

- Also when we use JSON API Serialize/Deserialize (for REST access from Android), then having public fields makes things easier!



This is what we want to see…



Application Object, V2.0

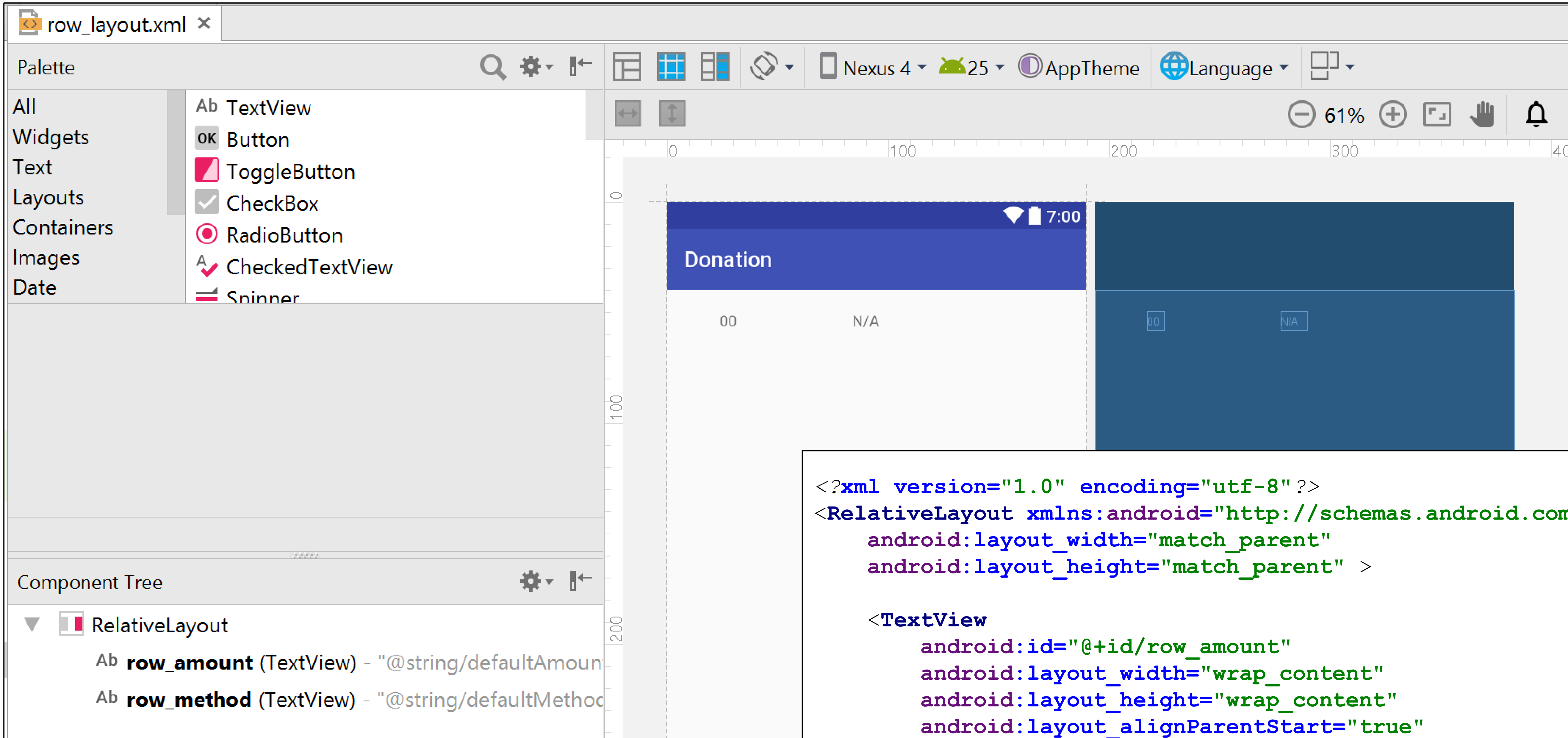
- Maintains list of donations.
- Maintains current total.
- Allow donations to be made (via ‘newDonation’).
- Track if total exceeded or not.

```
public class DonationApp extends Application{

    public final int target = 10000;
    public int totalDonated = 0;
    public List<Donation> donations = new ArrayList<Donation>();

    public boolean newDonation(Donation donation) {
        boolean targetAchieved = totalDonated > target;

        if(!targetAchieved) {
            donations.add(donation);
            totalDonated += donation.amount;
        }
        else {
            Toast toast = Toast.makeText(this, "Target Exceeded!", Toast.LENGTH_SHORT);
            toast.show();
        }
        return targetAchieved;
    }

    @Override
    public void onCreate(){
        super.onCreate();
        Log.v("Donate","Donation App Started");
    }
}
```

Revised Donate.java

Use the Application Object to store donations.

```java
private ProgressBar  progressBar;
private NumberPicker amountPicker;
private EditText     amountText;
private TextView     amountTotal;
private DonationApp  app;


public void donateButtonPressed (View view) {
    String method = paymentMethod.getCheckedRadioButtonId()
               == R.id.payPal ? "PayPal" : "Direct";

    int donatedAmount = amountPicker.getValue();
    if (donatedAmount == 0) {
        String text = amountText.getText().toString();
        if (!text.equals(""))
            donatedAmount = Integer.parseInt(text);
    }

    if (donatedAmount > 0) {
        app.newDonation(new Donation(donatedAmount, method));
        progressBar.setProgress(app.totalDonated);
        String totalDonatedStr = "$" + app.totalDonated;
        amountTotal.setText(totalDonatedStr);
    }
}
```



- Not all layouts need to be full screen activities.
- A layout xml file is just a description of a set of UI elements.
- It can be a full activity, or loaded as a part of some other activity.

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/ap
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:id="@+id/row_amount"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="48dp"
        android:layout_marginTop="20dp"
        android:text="@string/defaultAmount" />

    <TextView
        android:id="@+id/row_method"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/row_amount"
        android:layout_alignBottom="@+id/row_amount"
        android:layout_marginStart="106dp"
        android:layout_toEndOf="@+id/row_amount"
        android:text="@string/defaultMethod" />

</RelativeLayout>
```

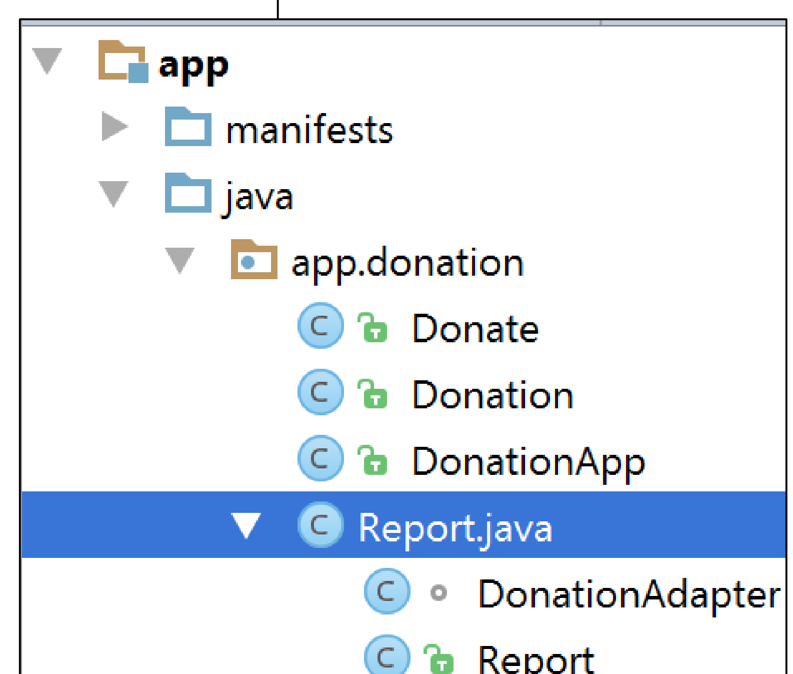
First Draft of Report.java
(using hard coded data and an ArrayAdapter)

```
public class Report extends AppCompatActivity
{
  ListView listView;

  static final String[] numbers = new String[] {
      "Amount, Pay method",
      "10,     Direct",
      "100,    PayPal",
      "1000,   Direct",
      "10,     PayPal",
      "5000,   PayPal"};

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

    listView = (ListView) findViewById(R.id.reportList);
    ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,  android.R.layout.simple_list_item_1, numbers);

    listView.setAdapter(adapter);
  }
}
```

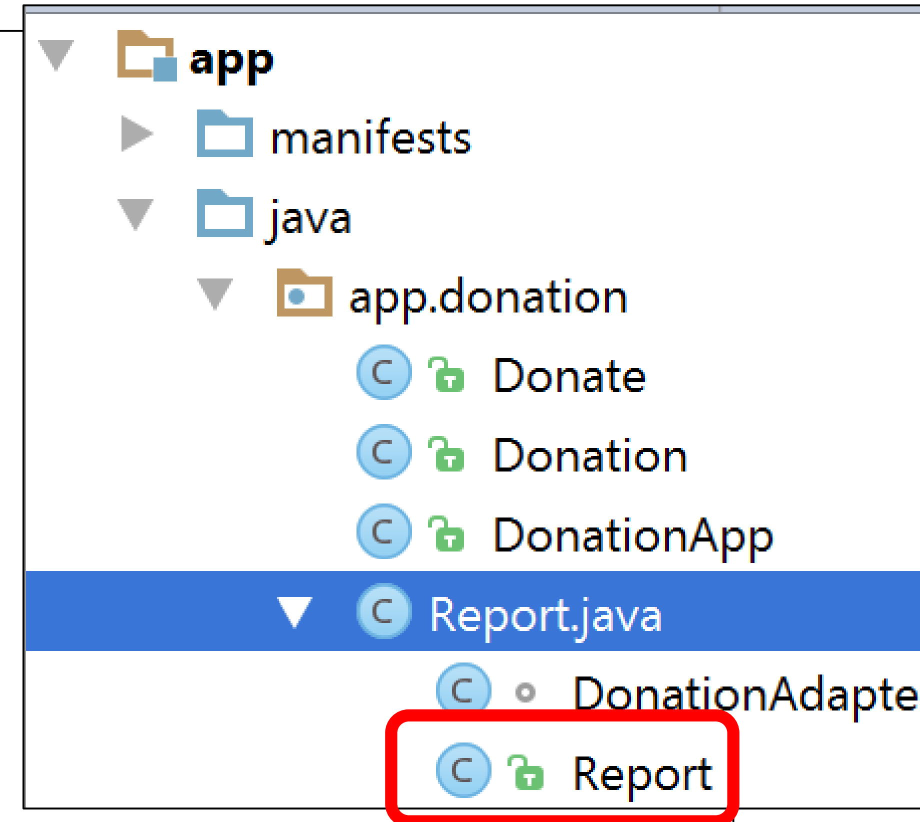
```java
package app.donation;

//import statements

public class Report extends AppCompatActivity {
    private ListView listView;
    private DonationApp app;

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

        app = (DonationApp) getApplication();

        listView = (ListView) findViewById(R.id.reportList);
        DonationAdapter adapter = new DonationAdapter(this, app.donations);
        listView.setAdapter(adapter);
    }
}

class DonationAdapter extends ArrayAdapter<Donation> {
    private Context context;
    public List<Donation> donations;

    public DonationAdapter (Context context, List<Donation> donations) {
        super(context, R.layout.row_layout, donations);
        this.context = context;
        this.donations = donations;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {

            LayoutInflater inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

            View view       = inflater.inflate(R.layout.row_layout, parent, false);
            Donation donation   = donations.get(position);
            TextView amountView = (TextView) view.findViewById(R.id.row_amount);
            TextView methodView = (TextView) view.findViewById(R.id.row_method);

            amountView.setText("" + donation.amount);
            methodView.setText(donation.method);

            return view;
    }

    @Override
    public int getCount()
    {
        return donations.size();
    }

}
```

Report.java

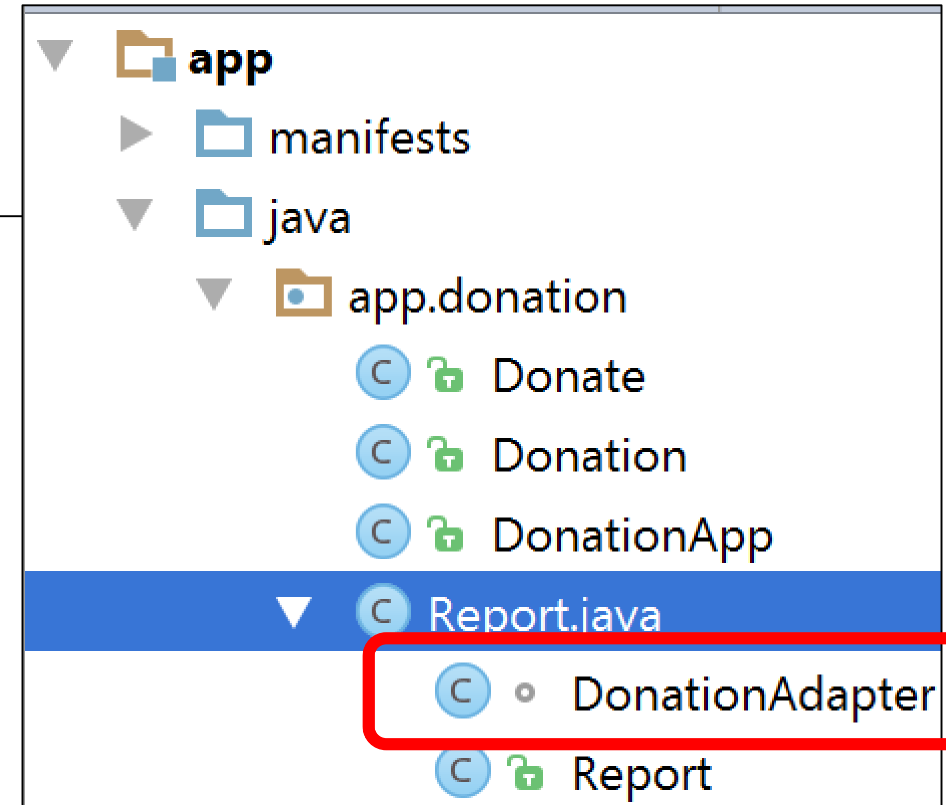
Revised using dynamic data and a bespoke, top-level class called DonationAdapter.



```java
package app.donation;

//import statements

public class Report extends AppCompatActivity {
    private ListView listView;
    private DonationApp app;

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

        app = (DonationApp) getApplication();

        listView = (ListView) findViewById(R.id.reportList);
        DonationAdapter adapter = new DonationAdapter(this, app.donations);
        listView.setAdapter(adapter);
    }
}
```



- Remove hard coded list of donations.
- Fetch current donations list from Application Object.
- Pass this list to a ‘DonationAdapter’ - and give the adapter to the list view.




```java
class DonationAdapter extends ArrayAdapter<Donation> {
    private Context context;
    public List<Donation> donations;

    public DonationAdapter (Context context,
                           List<Donation> donations) {
        super(context, R.layout.row_layout, donations);
        this.context = context;
        this.donations = donations;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {

            LayoutInflater inflater
                      = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

            View view           = inflater.inflate(R.layout.row_layout, parent, false);
            Donation donation   = donations.get(position);
            TextView amountView = (TextView) view.findViewById(R.id.row_amount);
            TextView methodView = (TextView) view.findViewById(R.id.row_method);

            amountView.setText("" + donation.amount);
            methodView.setText(donation.method);

            return view;
    }

    @Override
    public int getCount(){
        return donations.size();
    }
}
```

- ‘Adapt’ a list of Donation objects for display in a ListView
- Report the size of the list when asked (getCount())
- Given a specific position - create a ‘View’ representing a row when asked
- This row is created using the row_donate.xml layout we have just designed.

LayoutInflater

Recall in V1.0...

This method inflates a layout and puts it on screen. When a layout is inflated, each widget in the layout file is instantiated as defined by its attributes. You specify which layout to inflate by passing in the layouts resource ID.

```
public class Donate extends AppCompatActivity {

    private int          totalDonated = 0;
    private int          target = 10000;

    private RadioGroup   paymentMethod;
    private ProgressBar  progressBar;
    private NumberPicker amountPicker;
    private EditText     amountText;
    private TextView     amountTotal;

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

        paymentMethod = (RadioGroup)   findViewById(R.id.paymentMethod);
        progressBar   = (ProgressBar)  findViewById(R.id.progressBar);
        amountPicker  = (NumberPicker) findViewById(R.id.amountPicker);
        amountTotal   = (TextView)     findViewById(R.id.amountTotal);
        amountText    = (EditText)     findViewById(R.id.amountText);

        amountPicker.setMinValue(0);
        amountPicker.setMaxValue(1000);
        progressBar.setMax(target);
    }

// code omitted
```

Donate.java

LayoutInflater

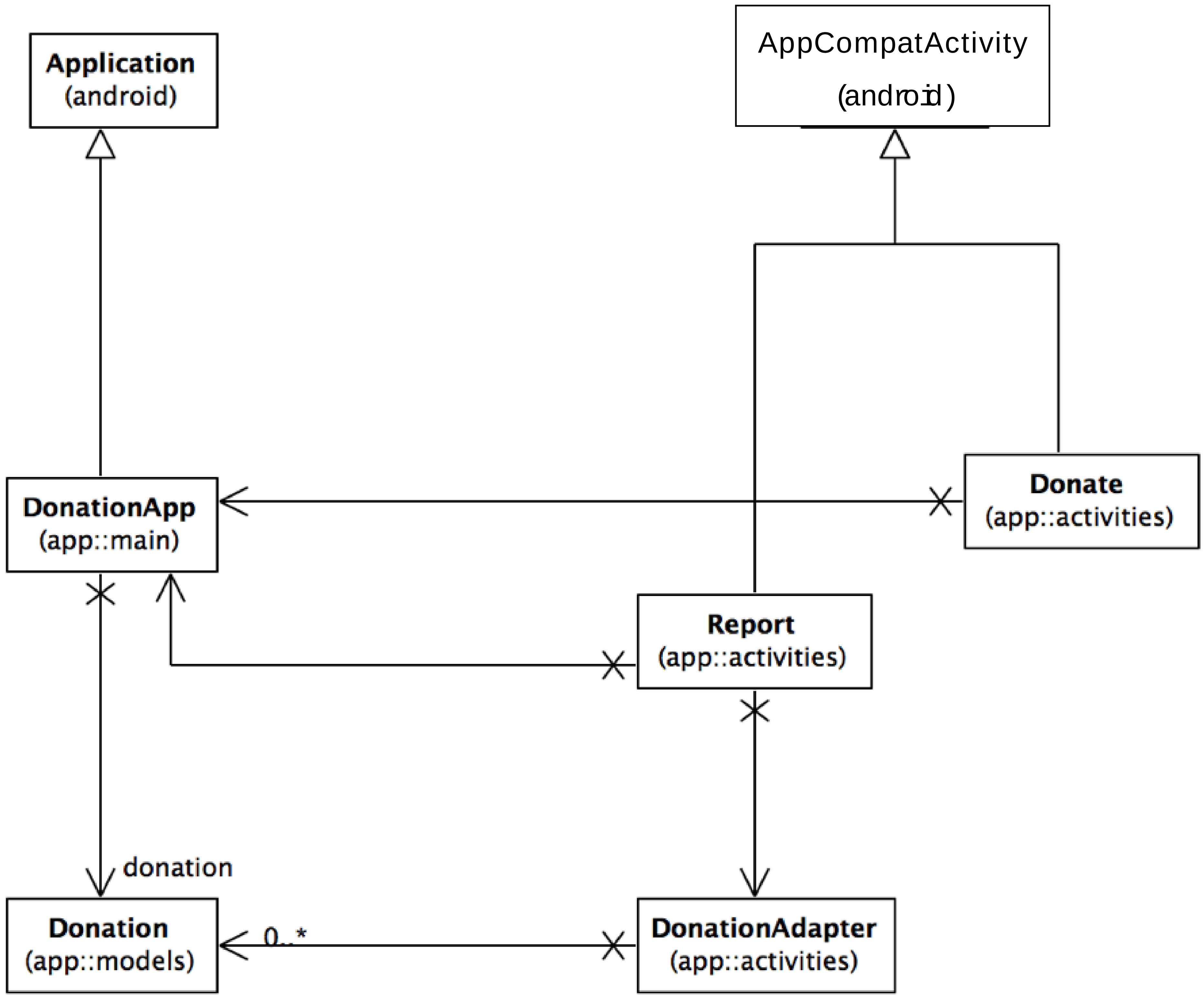
But in ArrayAdapter classes….

```java
class DonationAdapter extends ArrayAdapter<Donation> {
    private Context context;
    public List<Donation> donations;

    public DonationAdapter (Context context,
                            List<Donation> donations) {
        super(context, R.layout.row_layout, donations);
        this.context = context;
        this.donations = donations;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        LayoutInflater inflater
                = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

        View view             = inflater.inflate(R.layout.row_layout, parent, false);
        Donation donation     = donations.get(position);
        TextView amountView = (TextView) view.findViewById(R.id.row_amount);
        TextView methodView = (TextView) view.findViewById(R.id.row_method);

        amountView.setText("" + donation.amount);
        methodView.setText(donation.method);

        return view;
    }

    @Override
    public int getCount(){
        return donations.size();
    }

}
```

- The ArrayAdapter class is out of application context.
- You cannot inflate views out of application context.

→you need to get an application context instance within the ArrayAdapter class to get a LayoutInflater instance. More info: ContextWrapper

Donation V2 UML Model



Questions?