

Mobile Application Development

Higher Diploma in Science in Computer Science

Produced
by

Eamonn de Leastar (edelestar@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

<http://www.wit.ie>

<http://elearning.wit.ie>

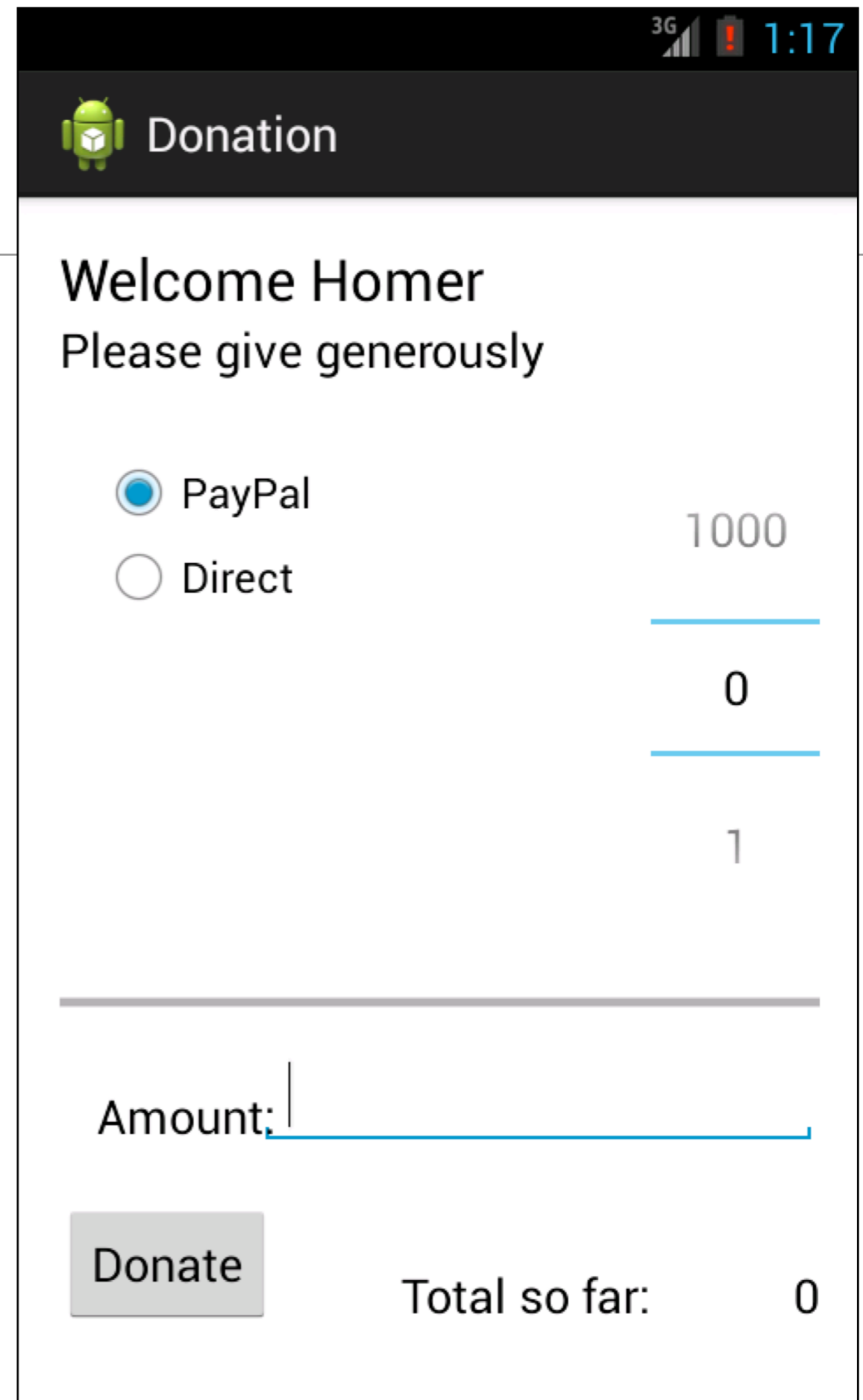
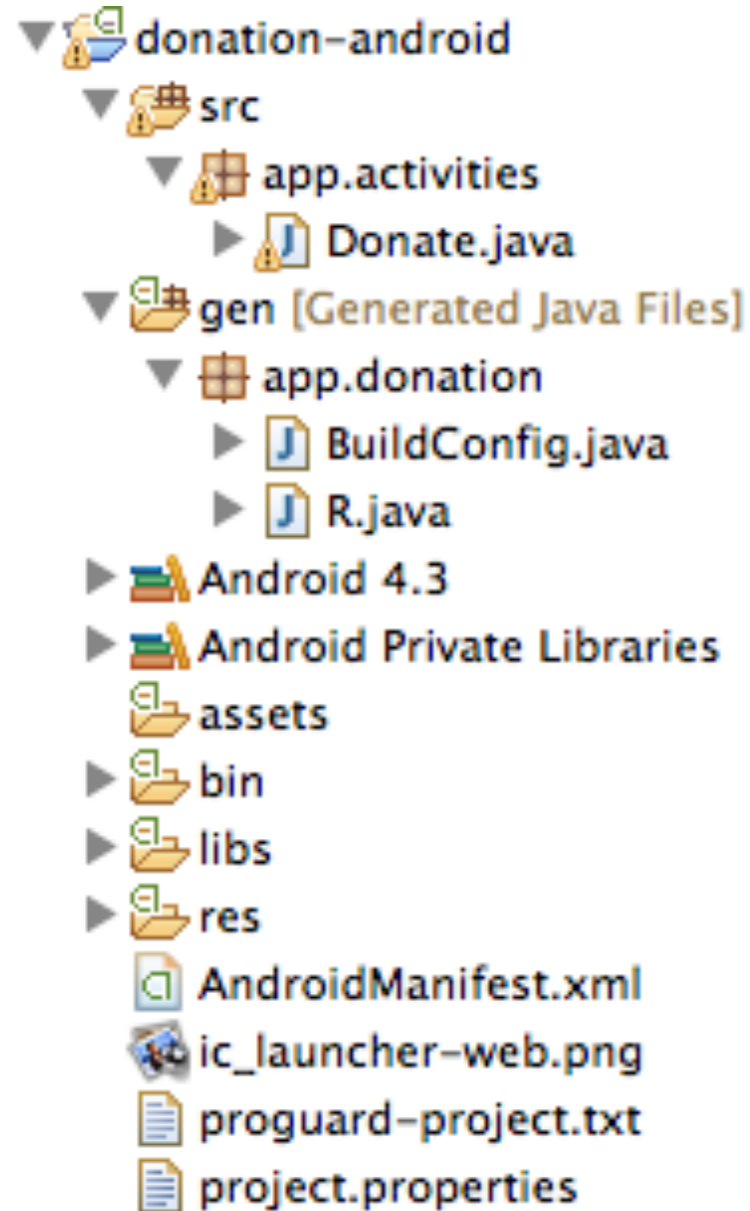


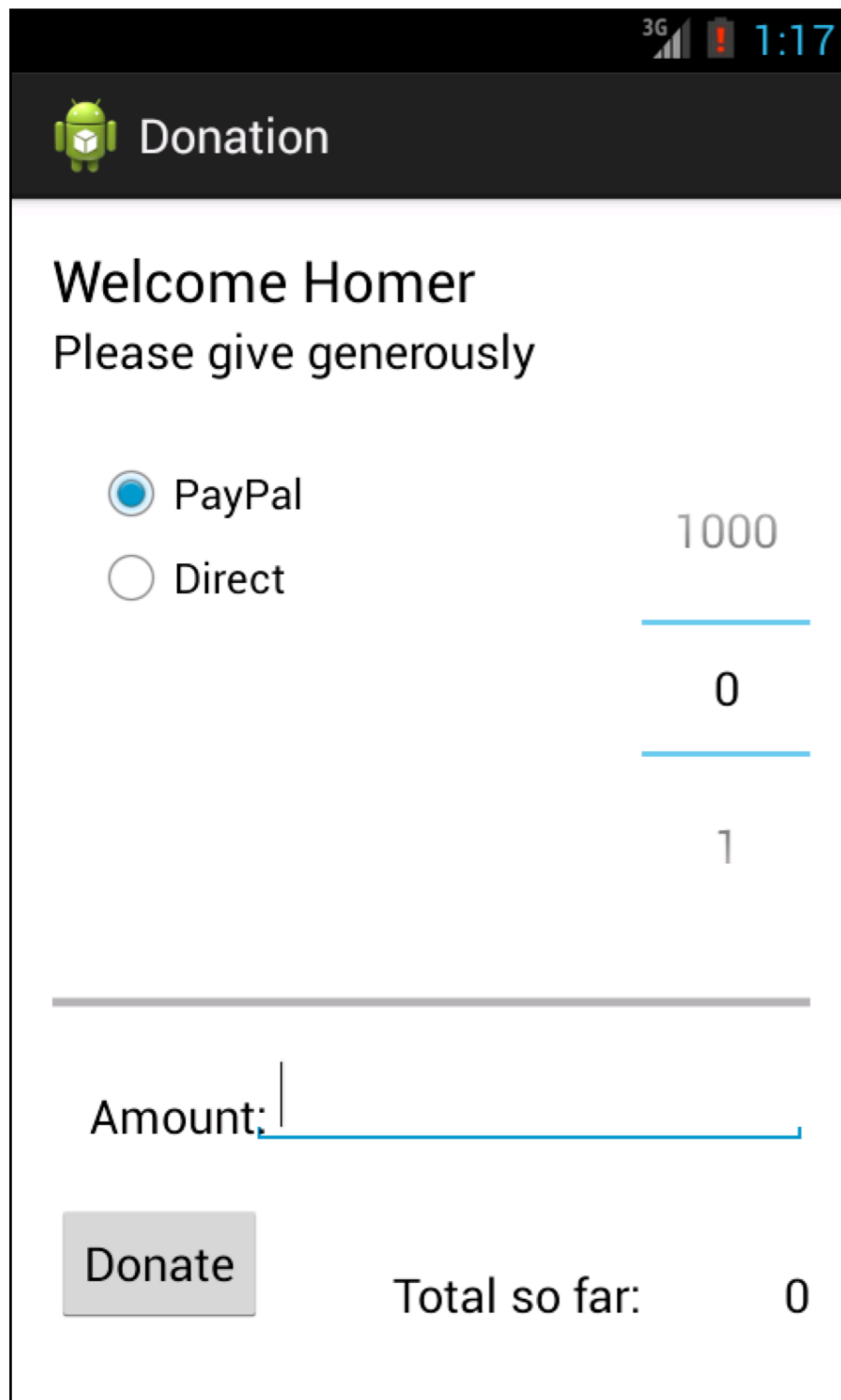
Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRCE



donation-android case study - v2

donation v1





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

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

        amountPicker.setMinValue(0);
        amountPicker.setMaxValue(10000);
        progressBar.setMax(target);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu)
    {
        getMenuInflater().inflate(R.menu.donate, menu);
        return true;
    }
}
```

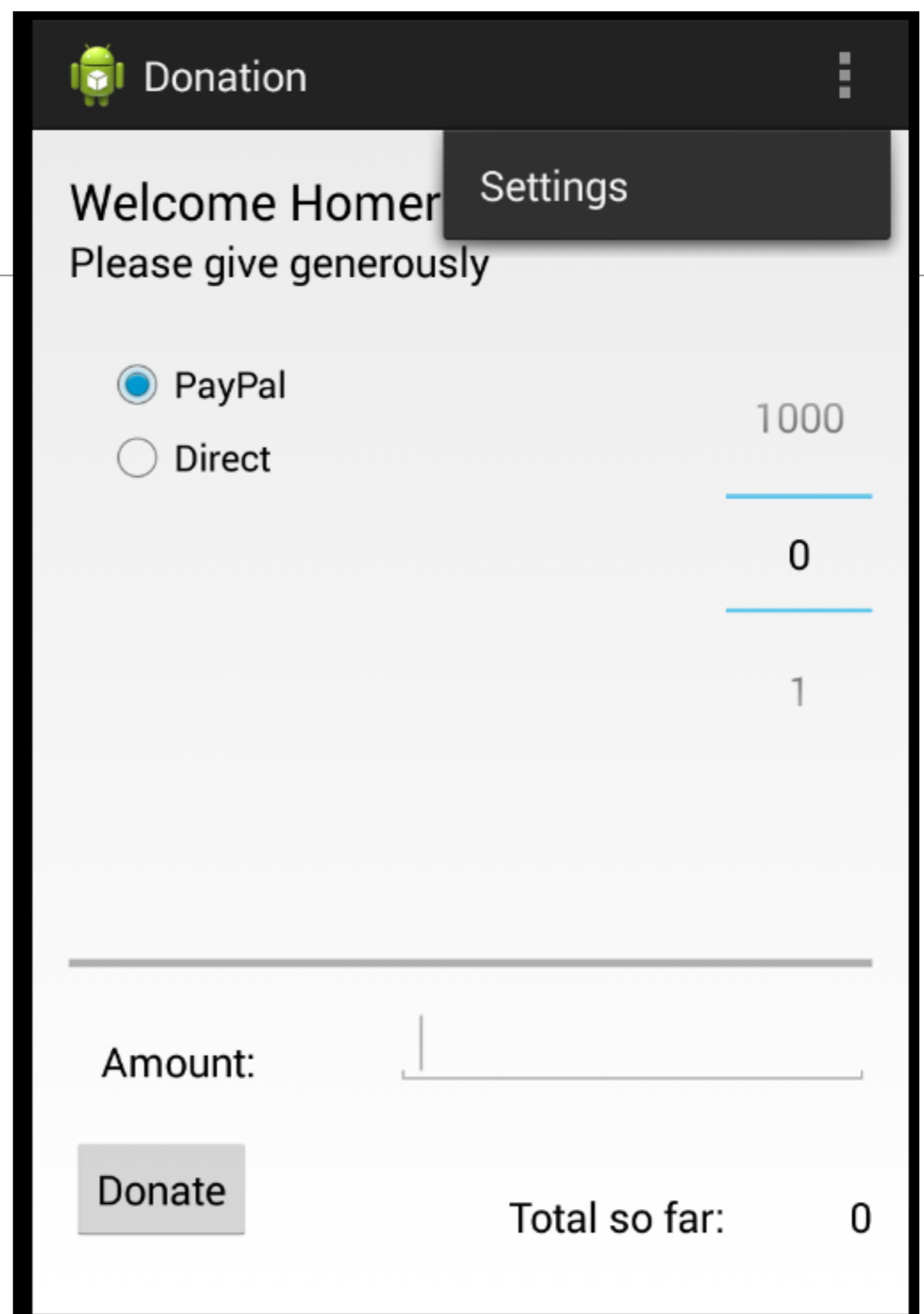
donate button event handler

```
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);
}
```

Menus

- Pressing the 'overflow' icon on the action bar brings up a menu with single entry:



Menu Load

```
public class Donate extends Activity
{
    //...

    @Override
    public boolean onCreateOptionsMenu(Menu menu)
    {
        getMenuInflater().inflate(R.menu.donate, menu);
        return true;
    }

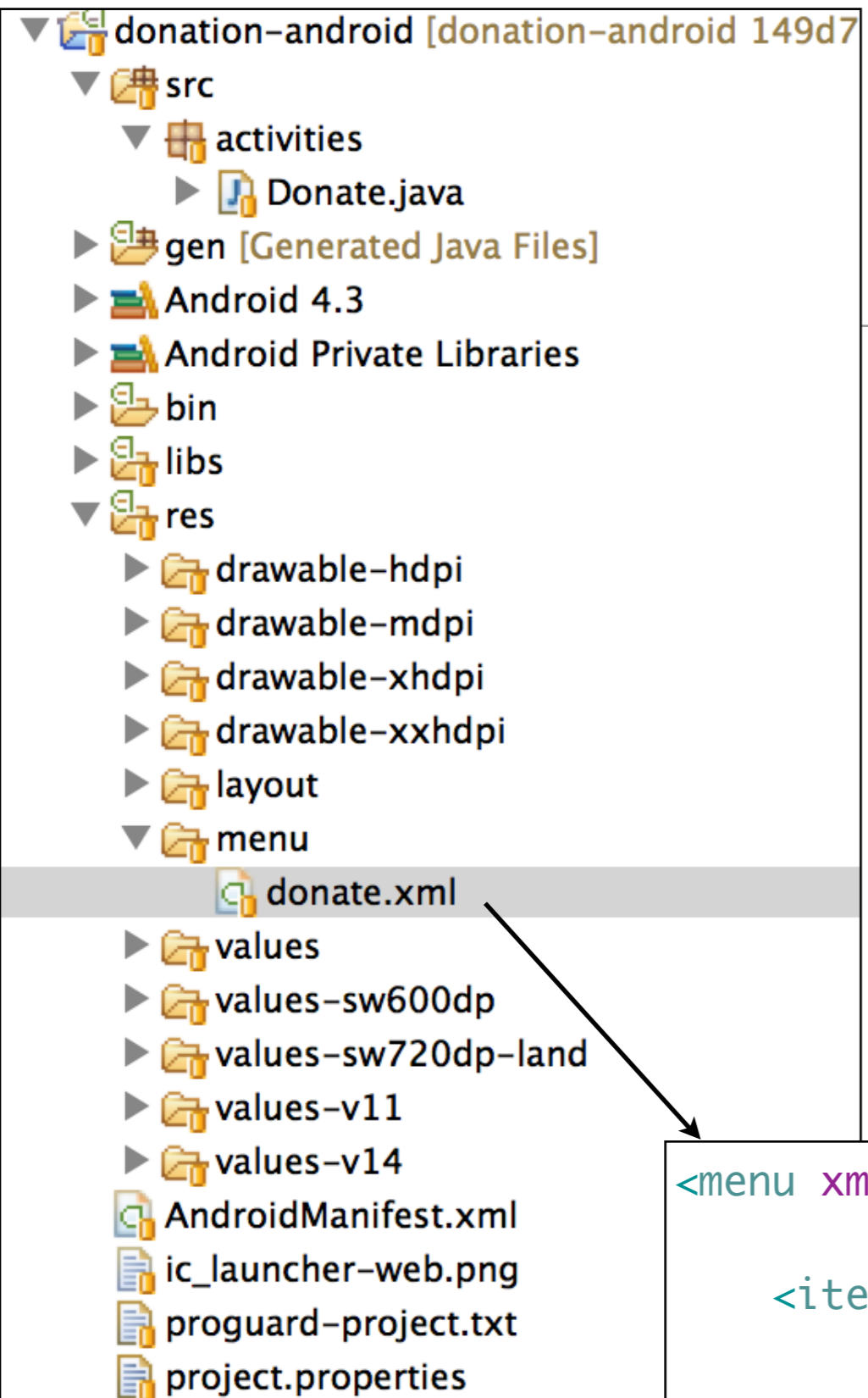
    //...
}
```

Menu Specification

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

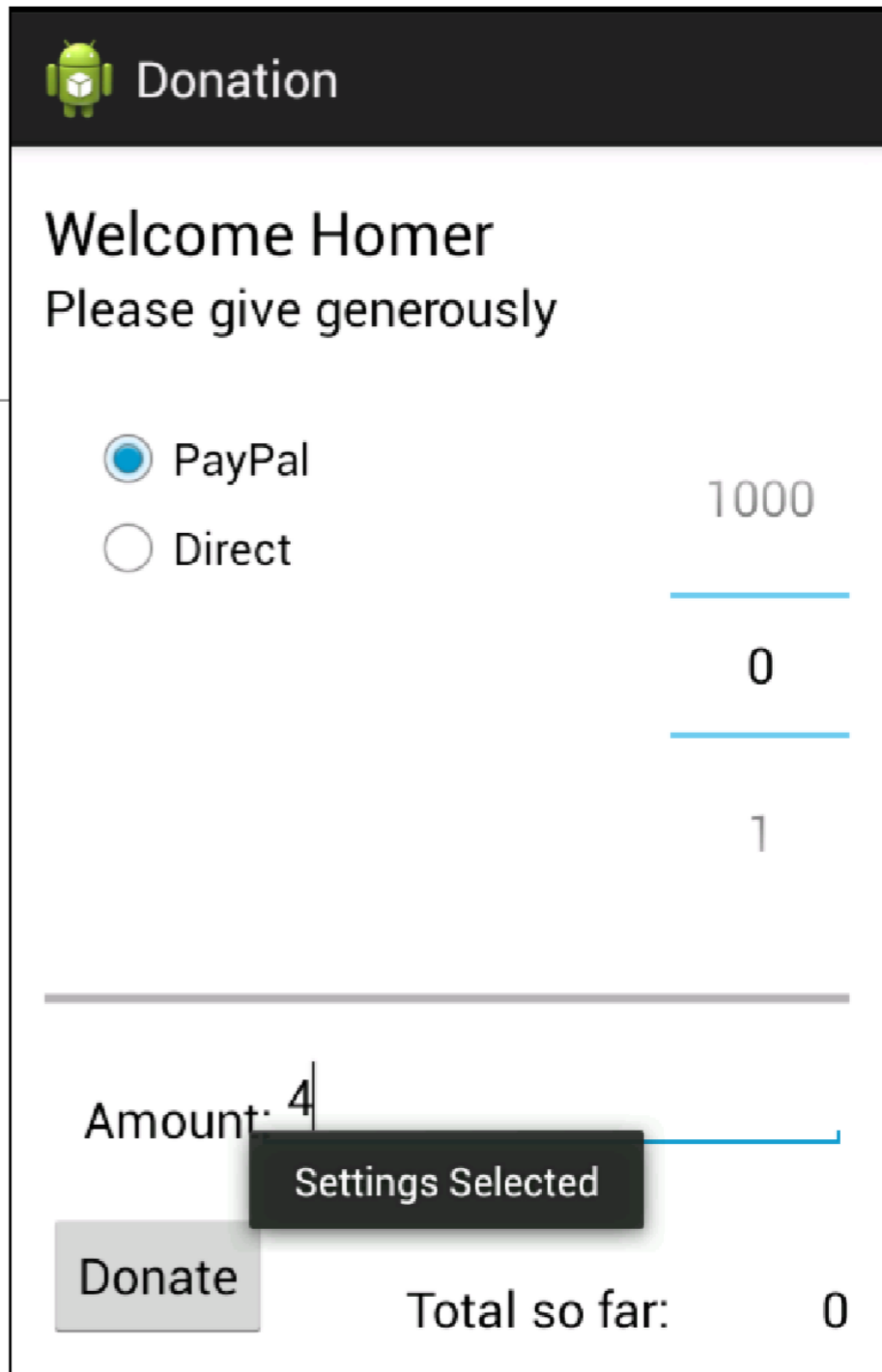
    <item
        android:id="@+id/action_settings"
        android:orderInCategory="100"
        android:showAsAction="never"
        android:title="@string/action_settings"/>

</menu>
```



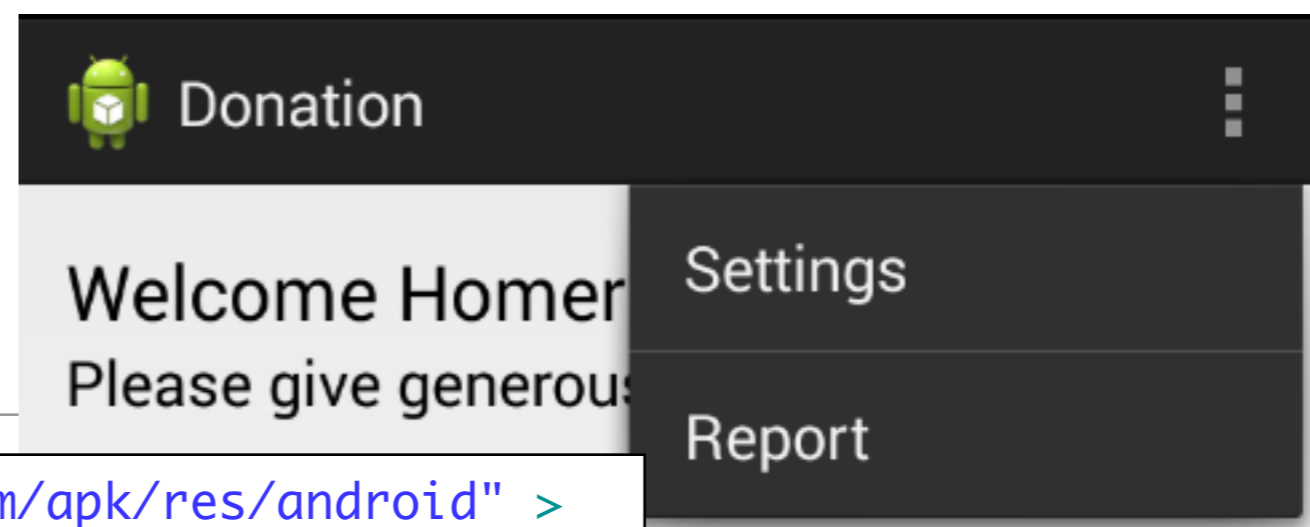
Menu Event Handler

- Display 'Toast' for a few seconds



```
@Override
public boolean onOptionsItemSelected(MenuItem item)
{
    switch (item.getItemId())
    {
        case R.id.action_settings: Toast toast = Toast.makeText(this, "Settings Selected", Toast.LENGTH_SHORT);
                                   toast.show();
                                   break;
    }
    return true;
}
```


New Menu Item



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

    <item
        android:id="@+id/action_settings"
        android:orderInCategory="100"
        android:showAsAction="never"
        android:title="@string/menuSettings"/>

    <item
        android:id="@+id/menuReport"
        android:orderInCategory="100"
        android:showAsAction="never"
        android:title="@string/menuReport"/>

</menu>
```

```
<string name="menuReport">Report</string>
```

Design new Report activity

The screenshot shows the Android Studio IDE with the following components:

- Top Bar:** Nexus One, AppTheme, Test, Android 18.
- Main Canvas:** A mobile app layout titled "Donation" with a list of items: "Report", "Item 1", "Item 2", "Item 3", "Item 4", "Item 5", and "Item 6". Each item has a "Sub Item" label below it. The "Report" activity is highlighted with a blue border and green dimension lines.
- Outline Panel:** Shows the hierarchy of the layout:
 - RelativeLayout
 - Ab reportTitle (TextView) - "Report"
 - reportList (ListView)
- Properties Panel:** Shows the properties of the selected view:

Id	Content Descri...

activity_report.xml

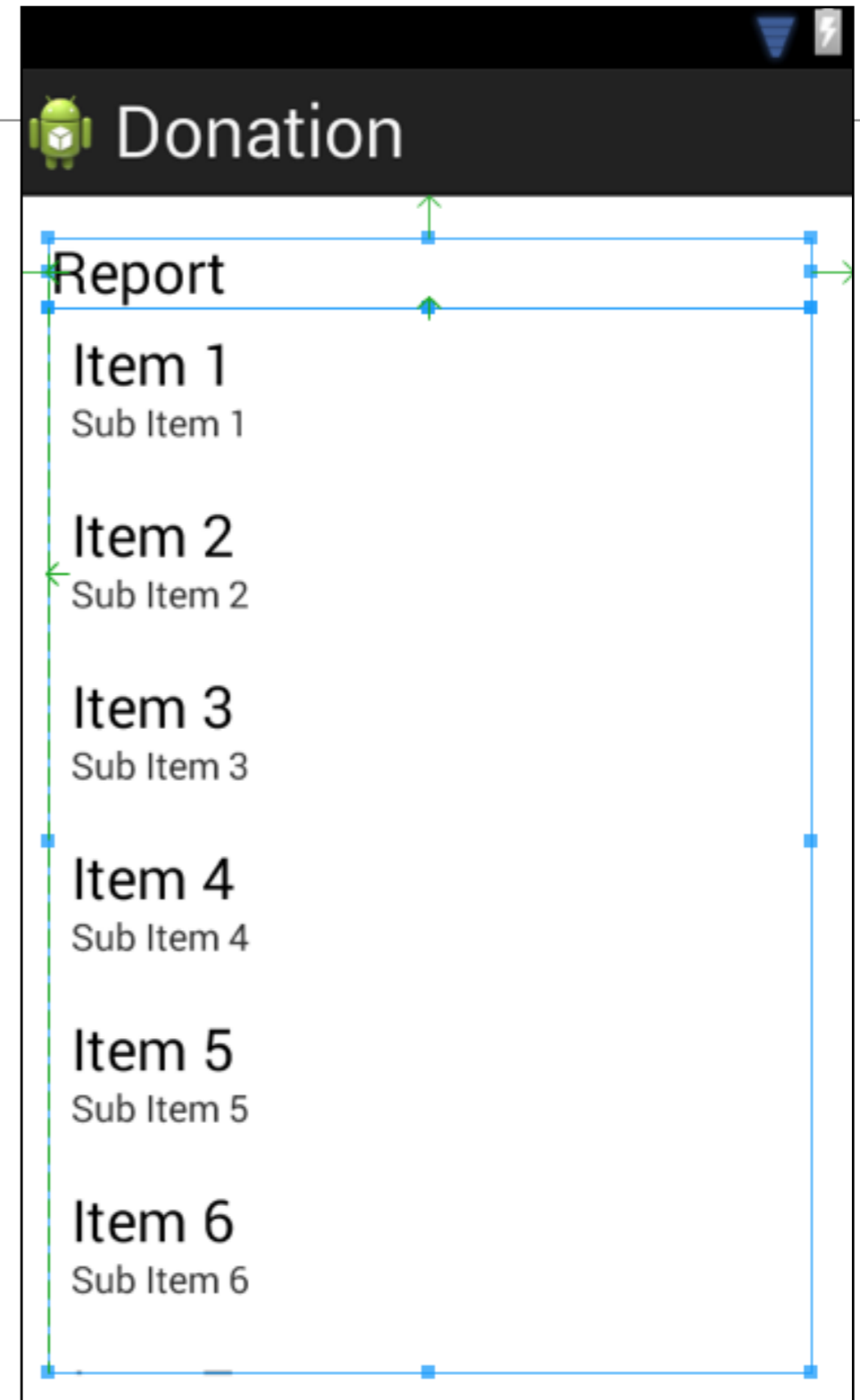
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".Test" >

    <TextView
        android:id="@+id/reportTitle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="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_alignLeft="@+id/reportTitle"
        android:layout_below="@+id/reportTitle" >

    </ListView>

</RelativeLayout>
```



ActivityReport

```
public class Report extends Activity
{
    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);
    }
}
```



Welcome Homer

Report

Amount	Pay method
10	Direct
100	PayPal
1000	Direct
10	PayPal
5000	PayPal

ActivityReport

```
public class Report extends Activity  
{  
    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);
```

```
}
```

```
}
```



Welcome Homer

Report

Amount	Pay method
10	Direct
100	PayPal
1000	Direct
10	PayPal
5000	PayPal

Application Object

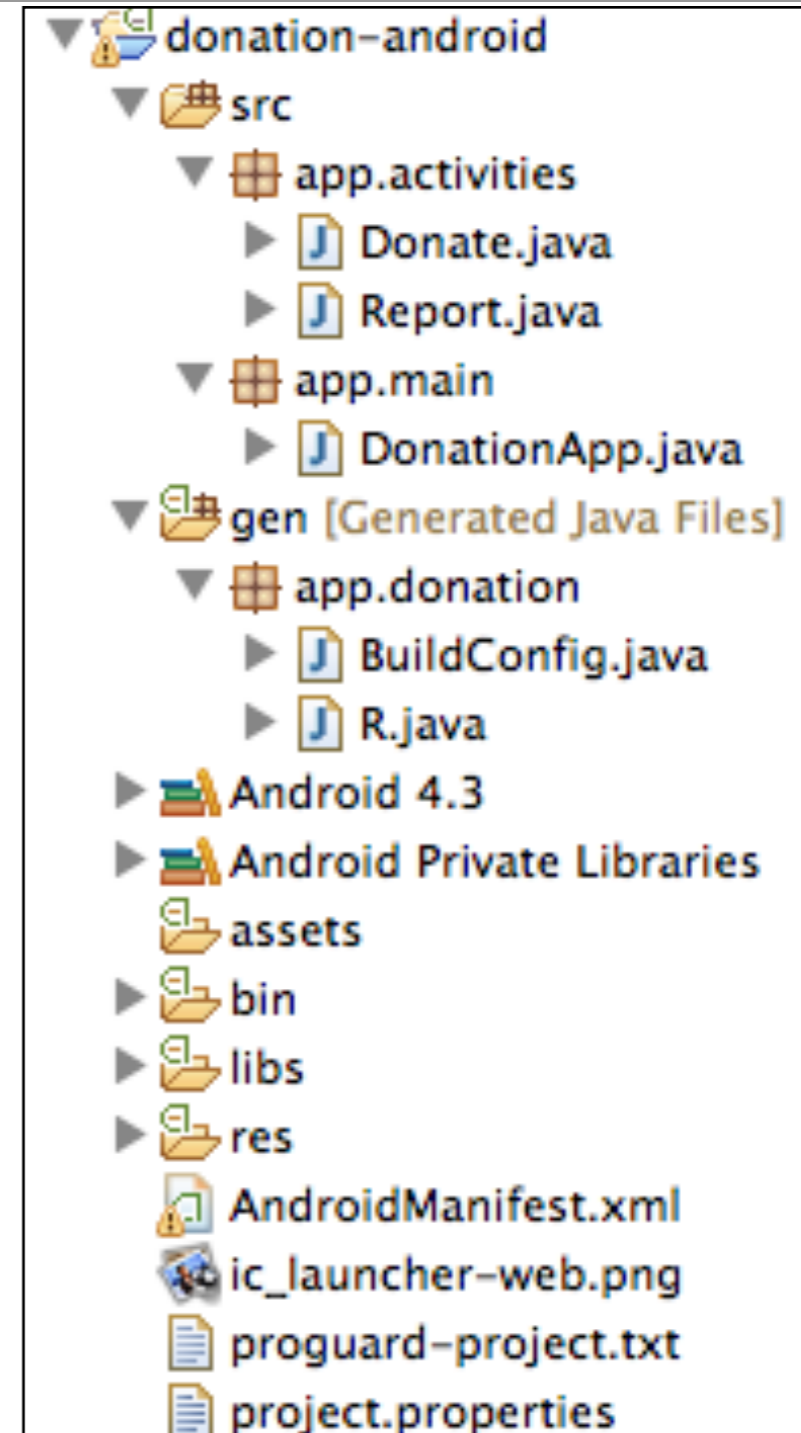
```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme"
    android:name="app.main.DonationApp">
```

```
package app.main;

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

public class DonationApp extends Application
{
    @Override
    public void onCreate()
    {
        super.onCreate();
        Log.v("Donation", "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

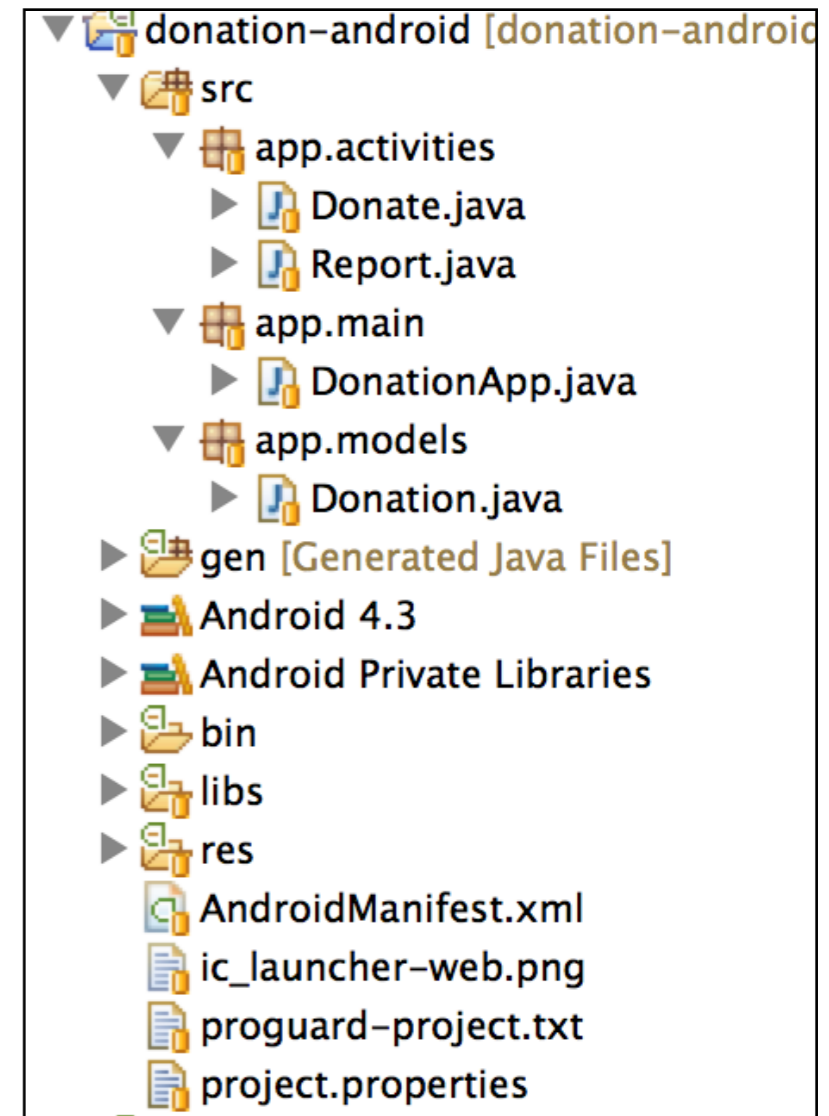


Model Package

- Introduce a 'models' package - similar to play framework models package
- Model key application domain - candidates for objects to be stored in a database:
 - locally (sql_lite)
 - remove (via API)

```
public class Donation
{
    public int    amount;
    public String method;

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



Revised DonationApp

```
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("Donation", "Donation App Started");
    }
}
```

- Maintain list of donations
- Main current total
- Allow donations to be made (via 'newDonation')
- Track if total exceeded or not

Donate Activity

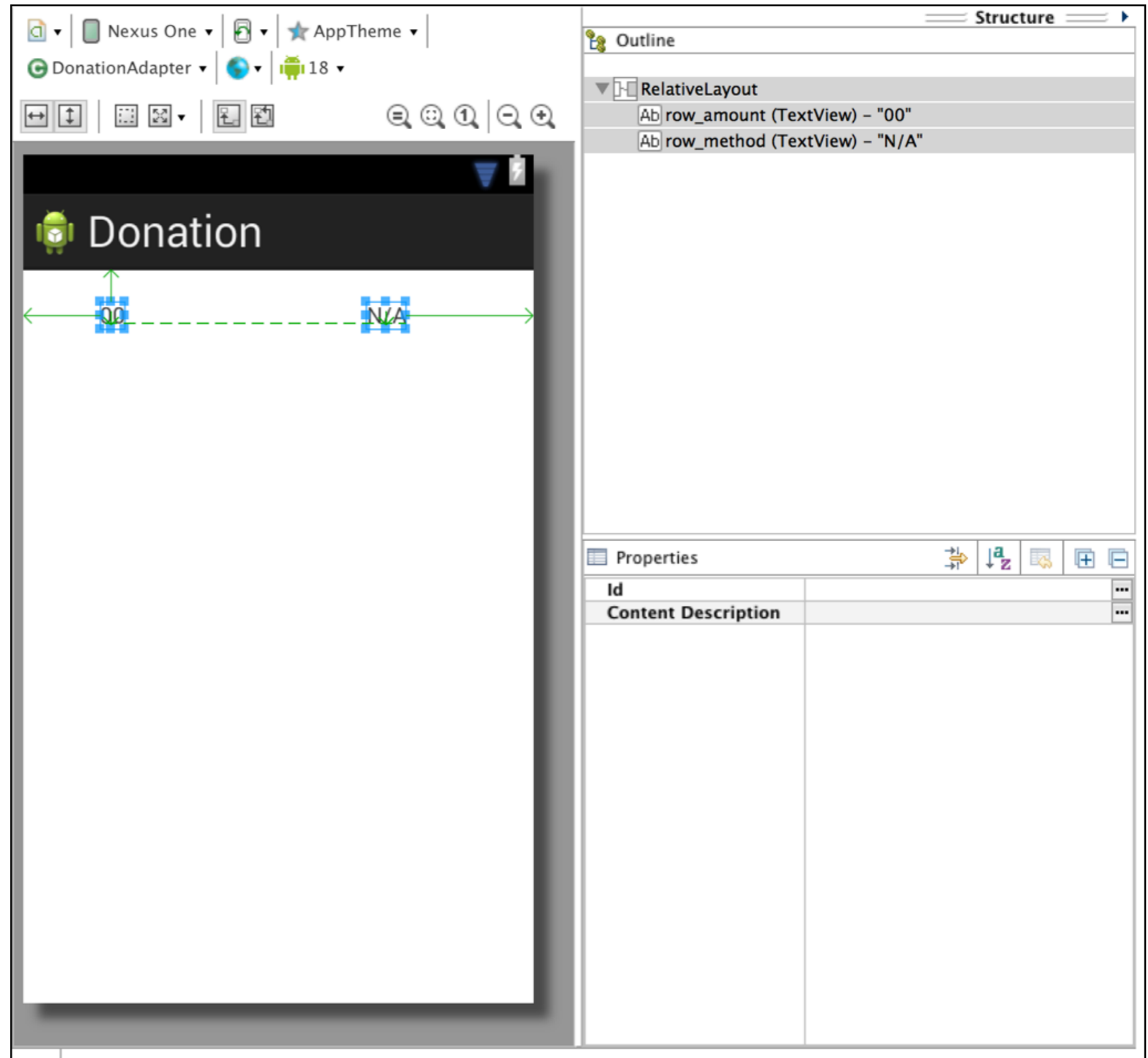
```
public class Donate extends Activity
{
    //...

    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);
        }
        amountText.setText("");
        amountPicker.setValue(0);
    }
    //..
}
```

- Use the Application Object to store donations

row_donate.xml

- 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



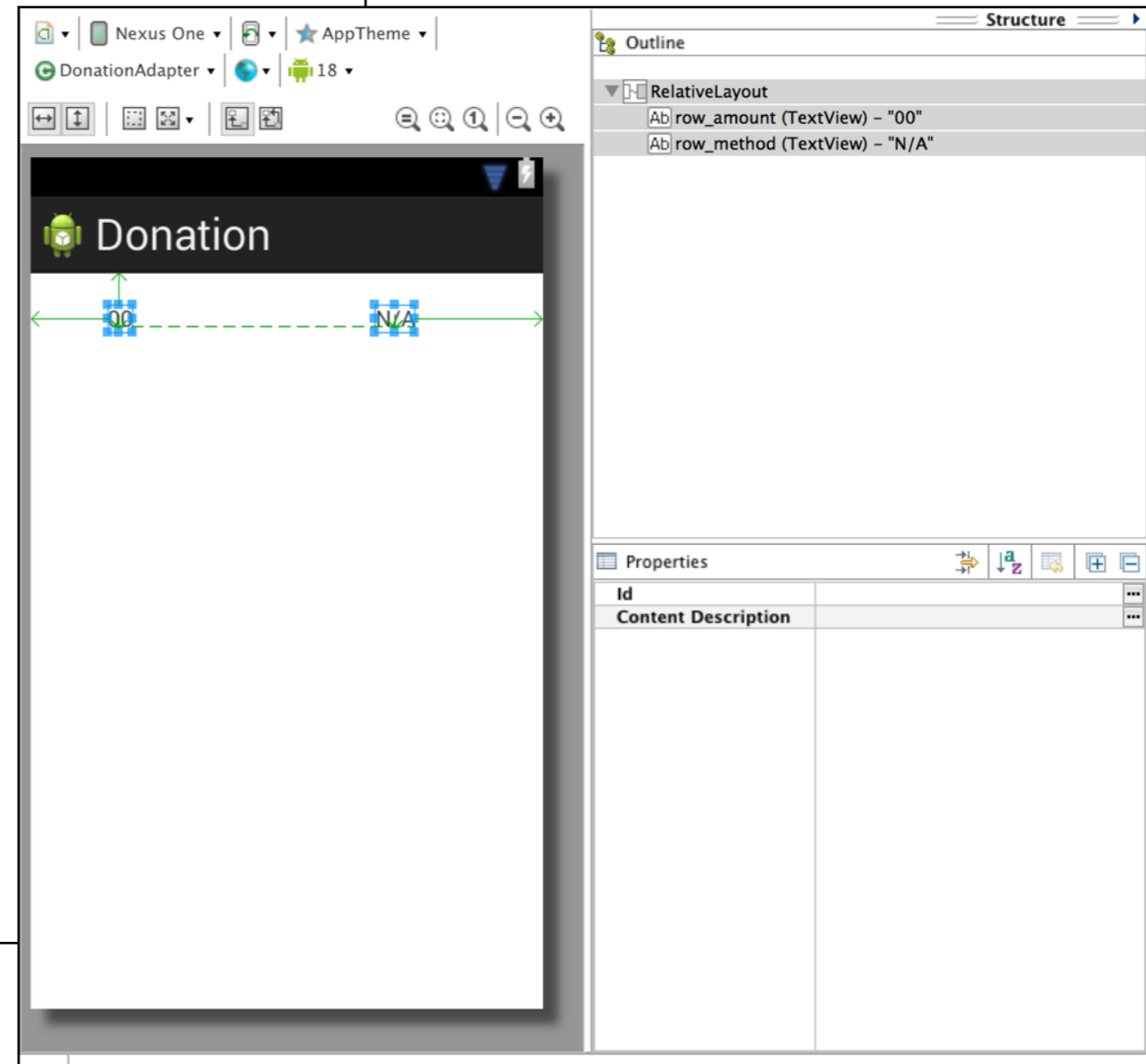
row_donate.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    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_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="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_alignParentRight="true"
        android:layout_marginRight="79dp"
        android:text="@string/defaultMethod" />

</RelativeLayout>
```



Revised Report Activity

```
public class Report extends Activity
{
    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.

DonationAdapter

- ‘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.

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

    public DonationAdapter(Context context, List<Donation> donations)
    {
        super(context, R.layout.row_donate, 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);

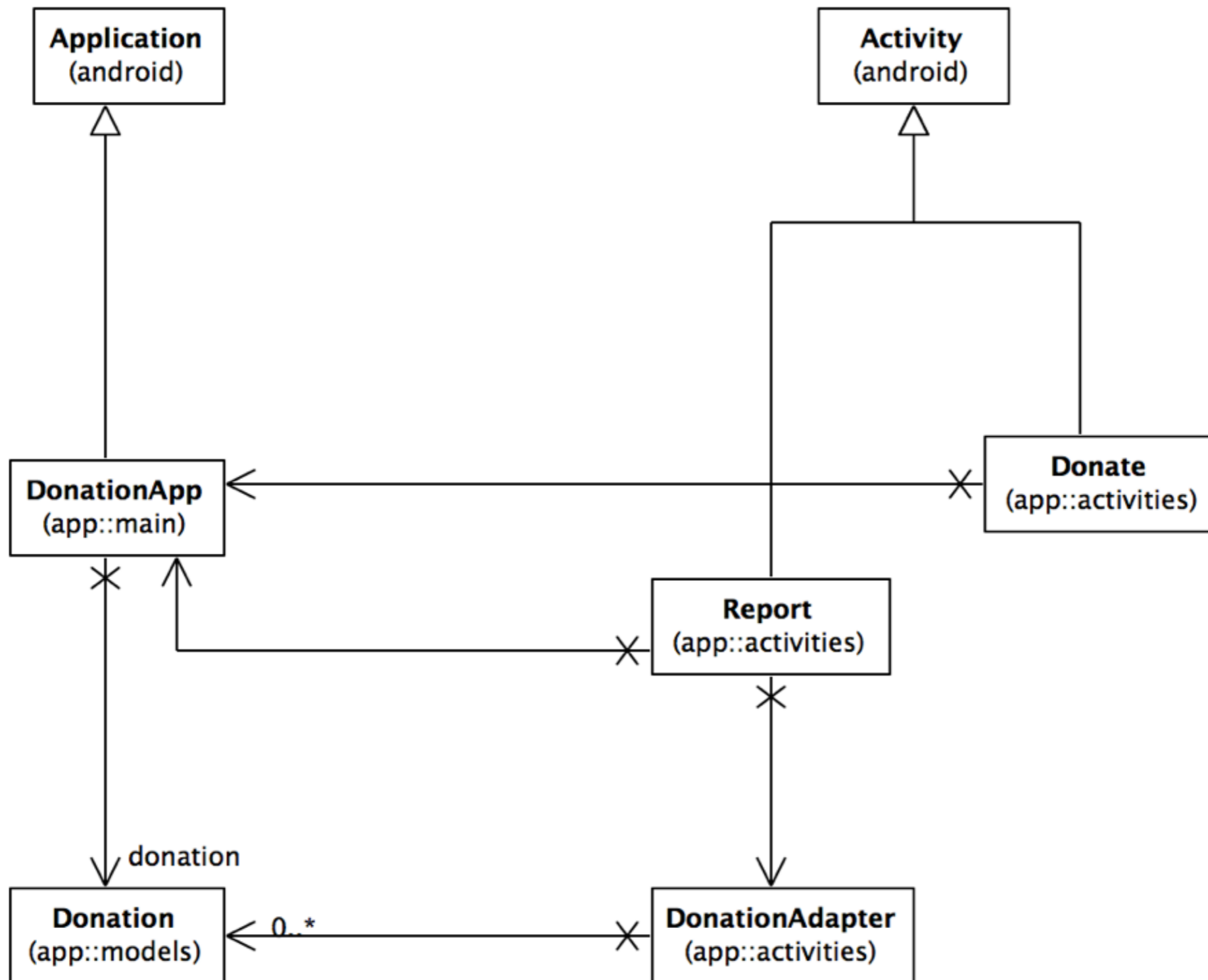
        View view = inflater.inflate(R.layout.row_donate, parent, false);
        Donation donation = donations.get(position);
        TextView amountView = (TextView) view.findViewById(R.id.row_amount);
        TextView methodView = (TextView) view.findViewById(R.id.row_method);

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

        return view;
    }

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

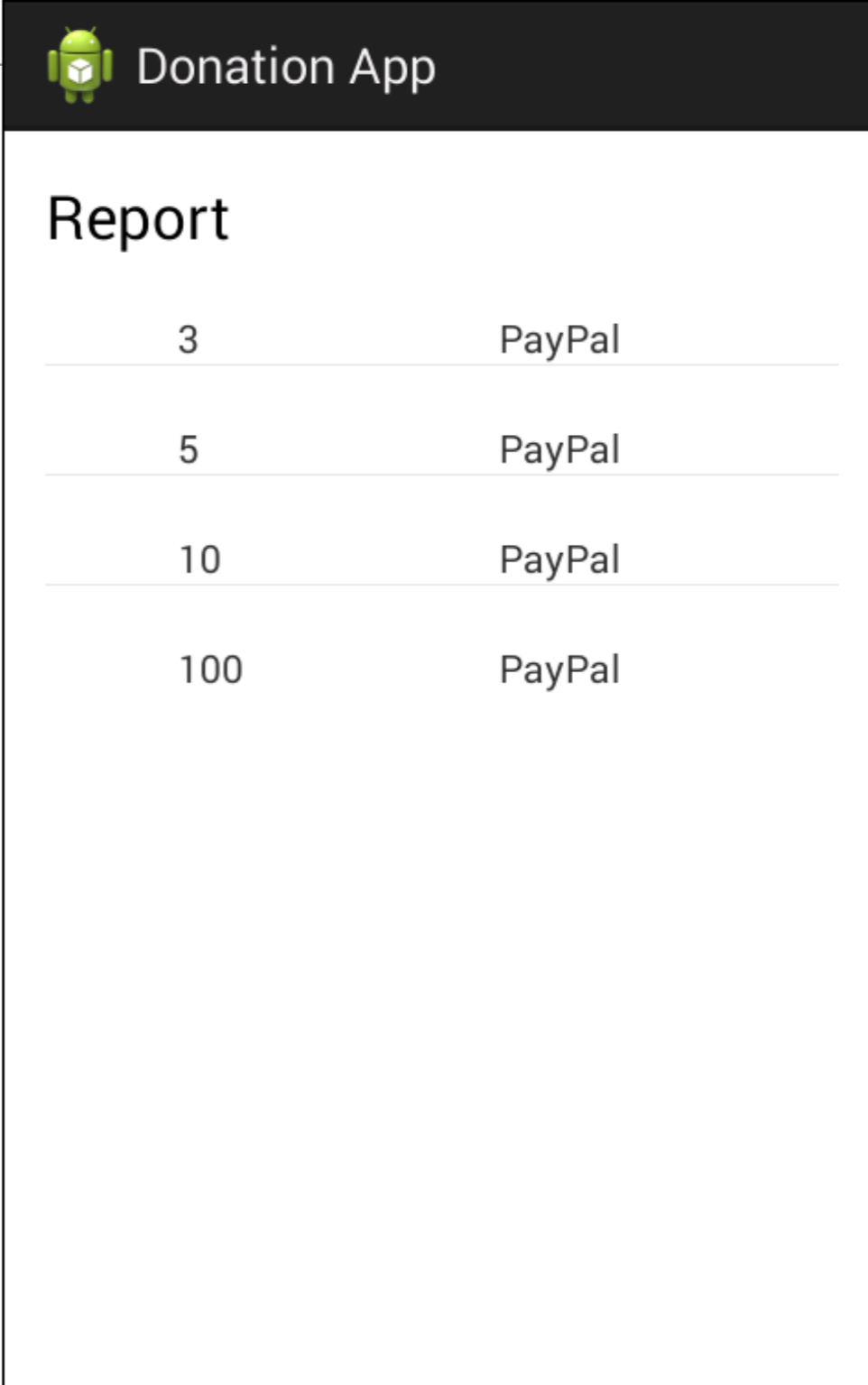
Donation v2 UML Model



Challenges

Exercises 1

- Run the app and insert amounts of varying lengths (1, 222, 23, 2323). Note that the second column - payment method - may be displayed at different positions. If this happens, fix it.
- Hint: each row is laid out by a `row_donate.xml` layout. The easiest way to fix this would be to experiment with they layout, and have the text fields aligned with the edges and not with eachother.



The screenshot shows the 'Donation App' interface. At the top, there is a dark header with the Android logo and the text 'Donation App'. Below the header, the word 'Report' is displayed. The report consists of a table with four rows, each representing a donation. The first column contains the donation amount, and the second column contains the payment method. The rows are: 3, PayPal; 5, PayPal; 10, PayPal; and 100, PayPal. The text in the second column is not aligned with the left edge of the table, but is instead aligned with the text of the first column.

Report	
3	PayPal
5	PayPal
10	PayPal
100	PayPal

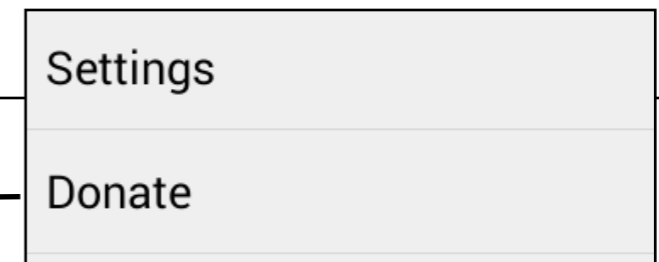
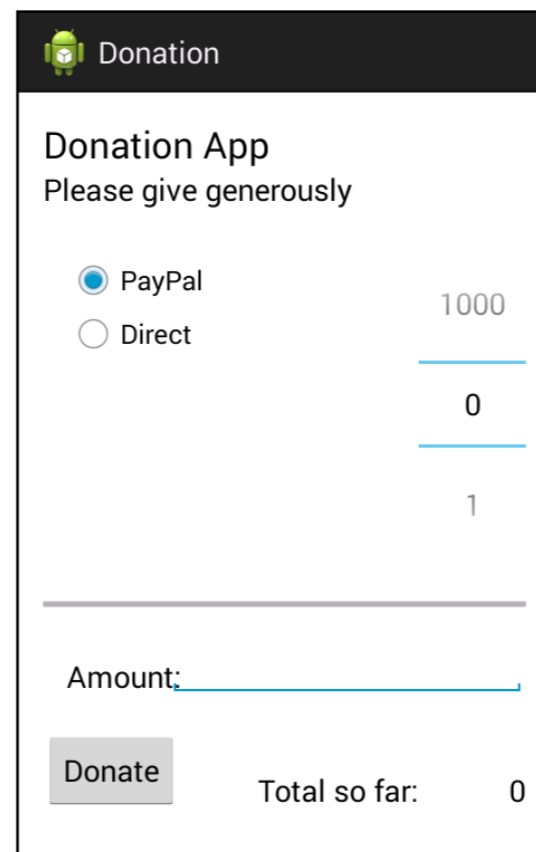
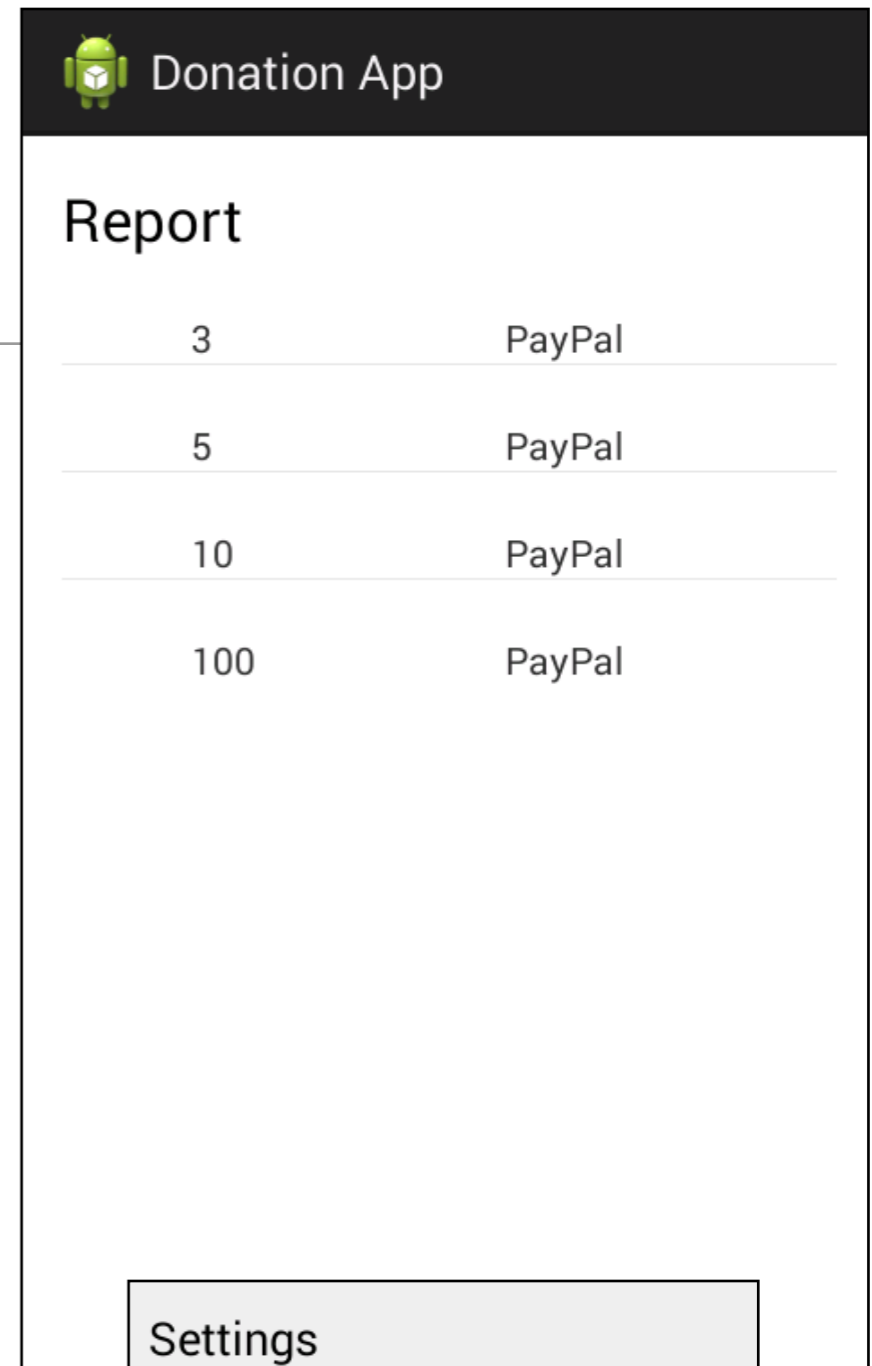
Exercise 2

- When a donation is accepted, set the amount on screen to 0 (in both picker and text field).

The screenshot shows an Android application window titled "Donation" with a green Android robot icon. The app content includes the title "Donation App" and the instruction "Please give generously". There are two radio button options: "PayPal" (selected) and "Direct". To the right of these options is a numeric picker with a value of 0, with "1000" shown above it and "1" below it. Below the picker is a horizontal line. Underneath is a text input field labeled "Amount:". At the bottom left is a grey "Donate" button. At the bottom right, it says "Total so far: 0".

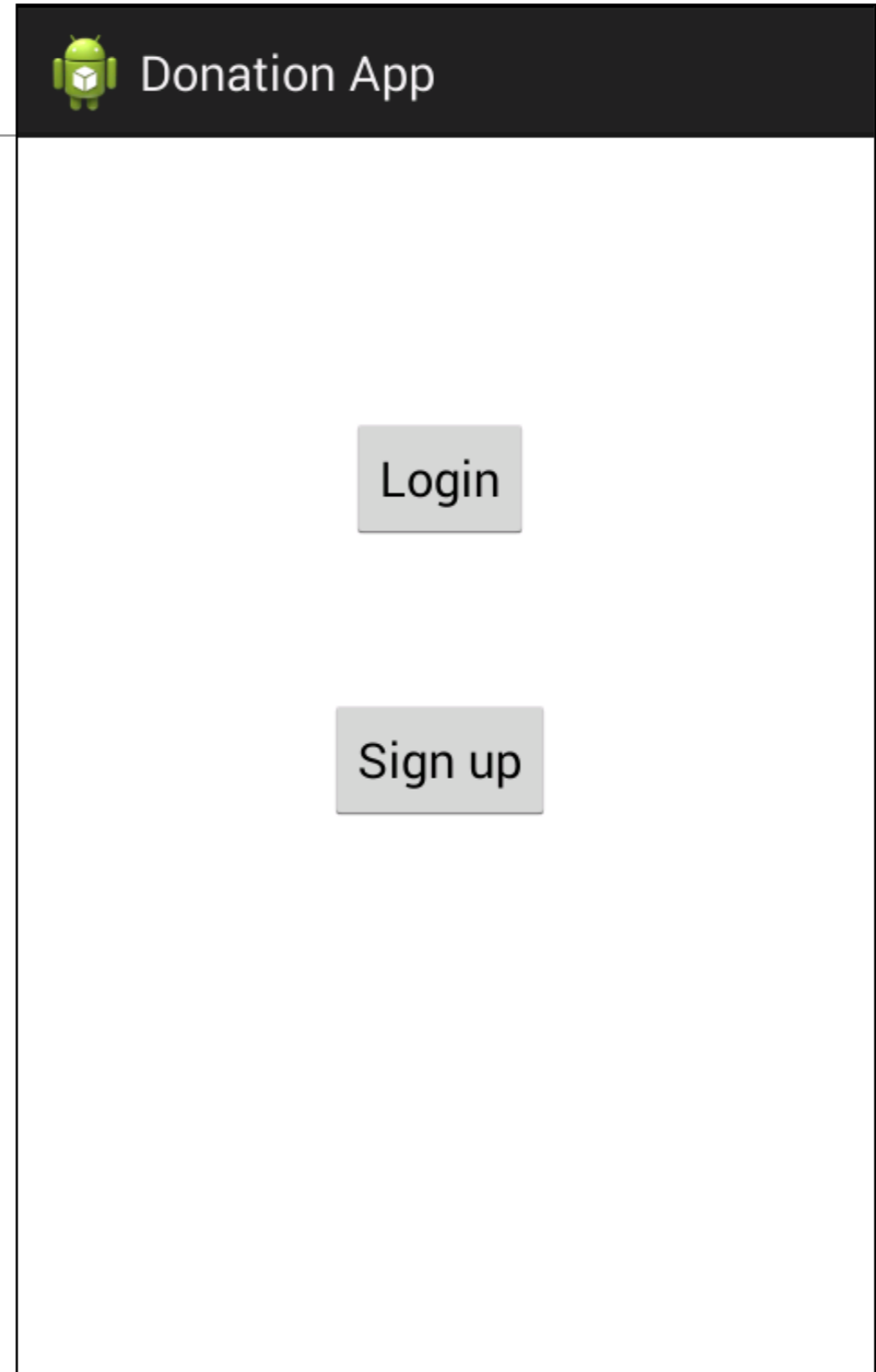
Exercise 3

- When you navigate from the Donate activity to reports, there will be no menu available. Bring in a menu, with two options 'Settings' and 'Donate' - Donate should bring you back to the donate screen.



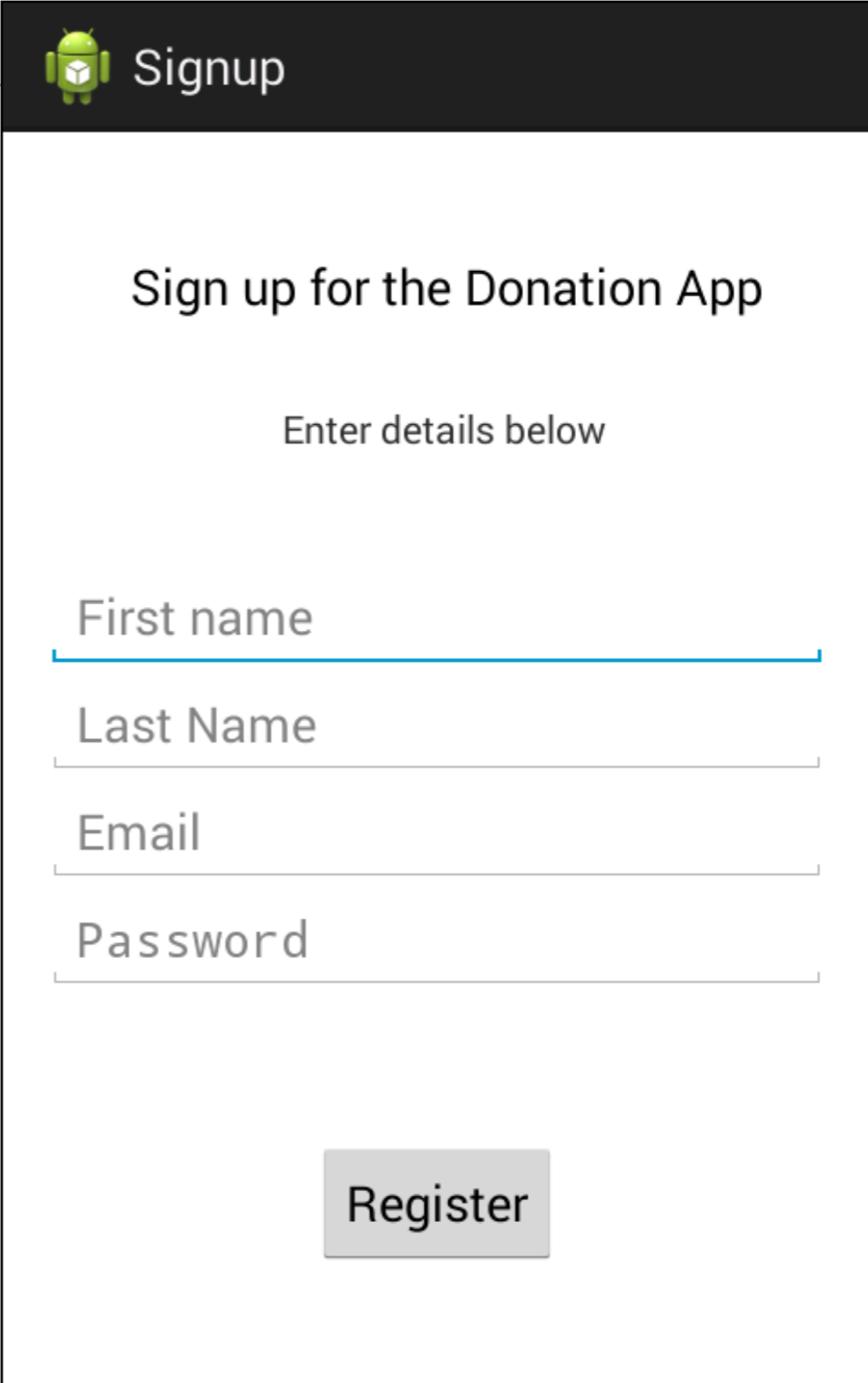
Exercise 4

- Introduce a new welcome screen - which should display a greeting + give the user 2 options (as simple buttons)
 - Signup
 - Login
- When Login is pressed, the app should take you directly to the Donate activity (for the moment).



Exercise 5

- Introduce a Signup Activity, which should present the user with:
 - First Name
 - Last Name
 - Email
 - Password
 - + 'Register' button.
- Pressing Register should take you directly to "Donate" activity



Signup

Sign up for the Donation App

Enter details below

First name

Last Name

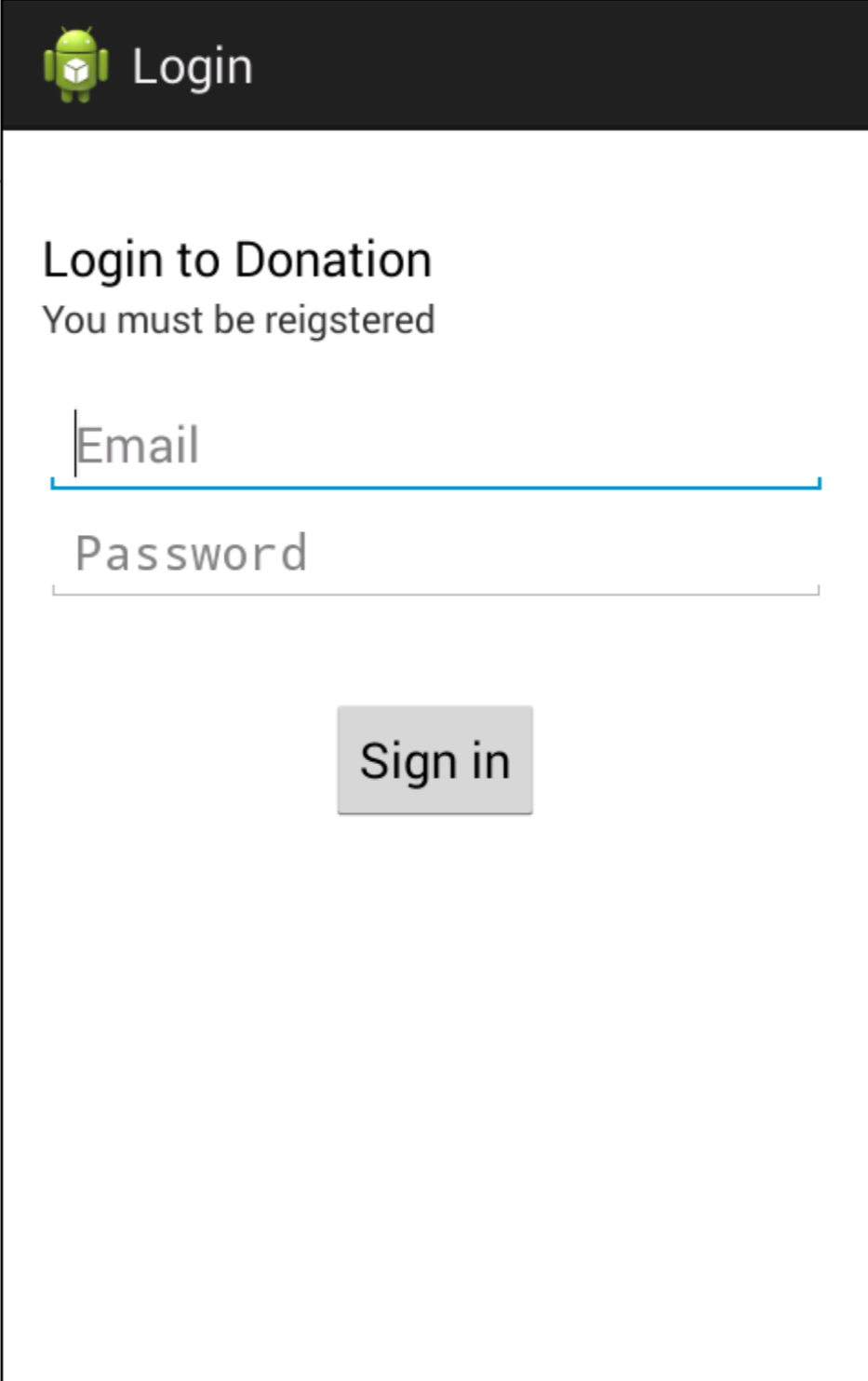
Email

Password

Register

Exercise 6

- Introduce a Login activity, which should just look for
 - email
 - password
 - + a 'Sign in' button
- Pressing Login should take you directly to "Donate" activity.



Login

Login to Donation
You must be registered

Email

Password

Sign in

Donation App

Please give generously

PayPal

Direct

1000

0

1

Amount:

Donate

Total so far:

0

Settings

Report

Logout

Exercise 7

- Bring in a new menu option - 'logout'. It should take you to the welcome screen.

Report

3

PayPal

5

PayPal

10

PayPal

100

PayPal

Login

Sign up

Settings

Donate

Logout


Exercise 8

- Introduce a 'User' into the models package to represent the user in the usual way. Maintain a list of Users in the DonationApp object. Whenever anyone registers, then create a new User object in this list.

Exercise 9


- Implement the Login activity, to now only let users in to Donate if they are registered (i.e. a matching email + password in the list of users maintained by DonationApp)

Navigation Structure

 Donation App

Login

Sign up


 Login

Login to Donation
You must be reigstered

Email

Password

Sign in

 Signup

Sign up for the Donation App

Enter details below

First name

Last Name

Email

Password

Register

Login

Login to Donation
You must be registered

Email

Password

Donation

Donation App
Please give generously

PayPal 1000

Direct 0

1

Amount:

Total so far: 0

Donation

Donation App
Please give generously

PayPal
 Direct

1000

0

1

Amount:

Donate

Total so far: 0

Donation App

Report

3	PayPal
5	PayPal
10	PayPal
100	PayPal

Settings

Report

Logout

Donation App

Login

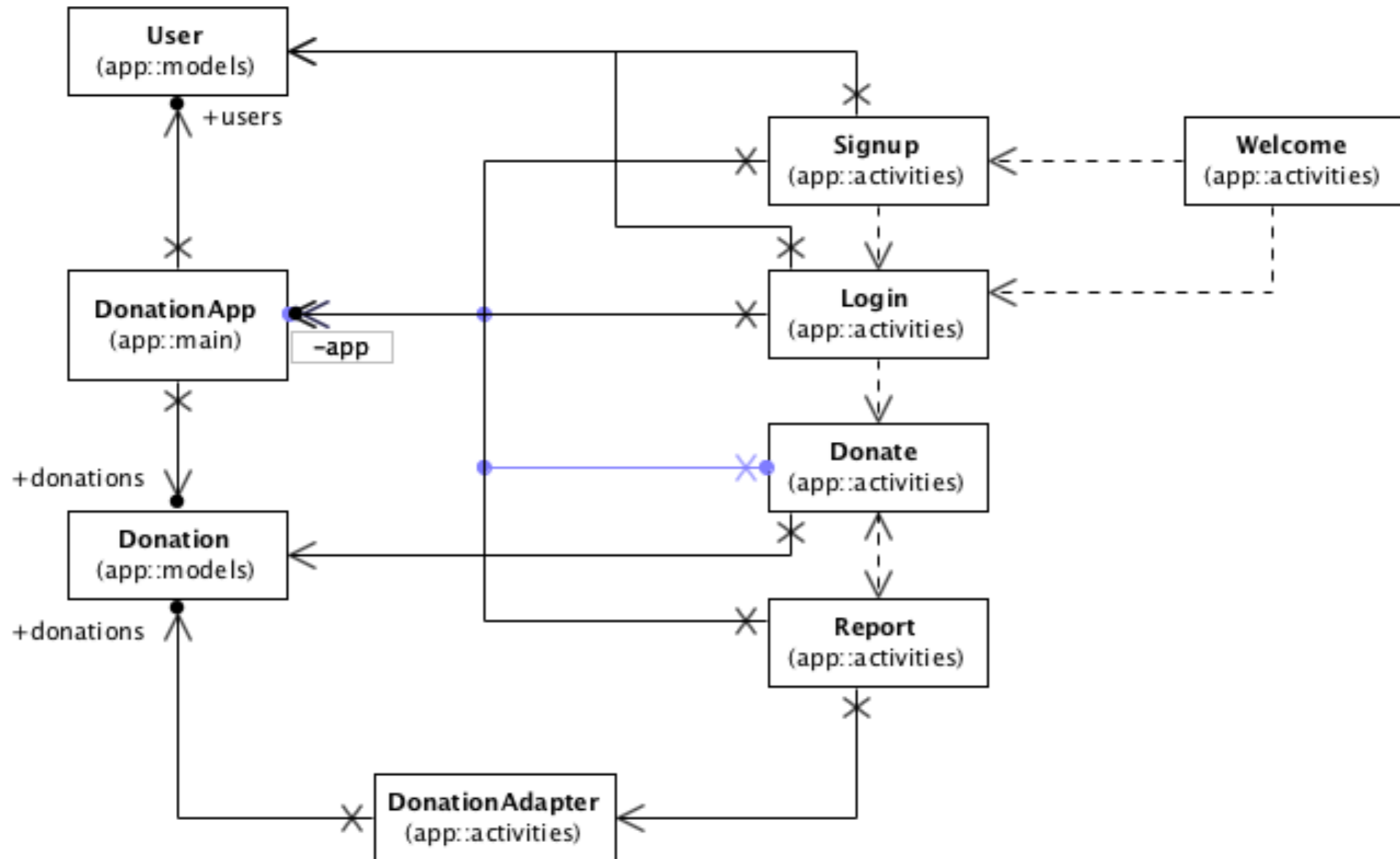
Sign up

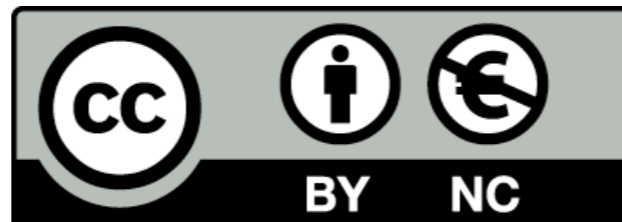
Settings

Donate

Logout

UML Model of donation-android-v3





Except where otherwise noted, this content is licensed under a Creative Commons Attribution-NonCommercial 3.0 License.

For more information, please see <http://creativecommons.org/licenses/by-nc/3.0/>

