

# Mobile Application Development

---

Produced  
by

Eamonn de Leastar ([edelestar@wit.ie](mailto:edelestar@wit.ie))

David Drohan ([ddrohan@wit.ie](mailto:ddrohan@wit.ie))

Dr. Siobhán Drohan ([sdrohan@wit.ie](mailto:sdrohan@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

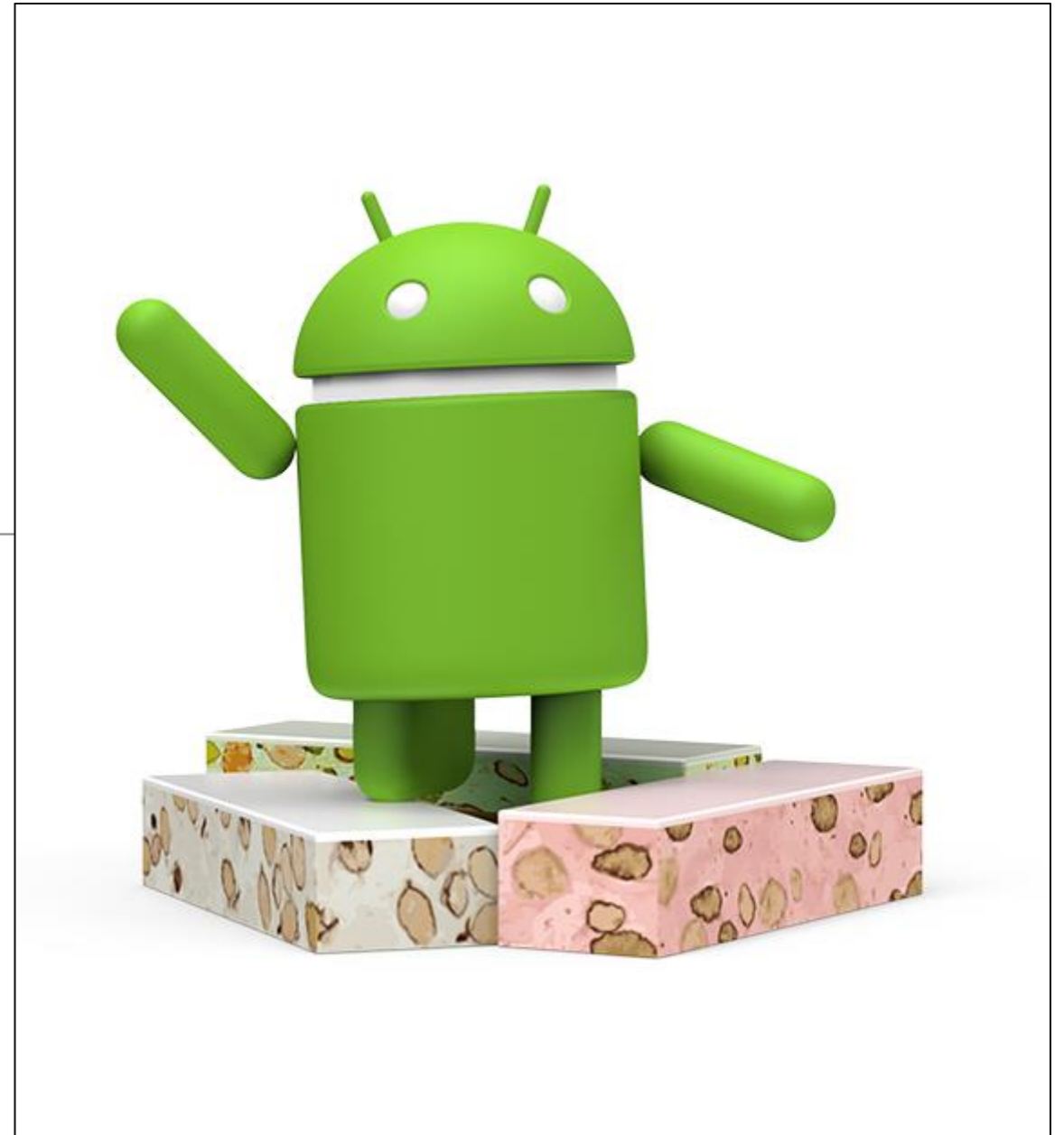


# A First Android Application

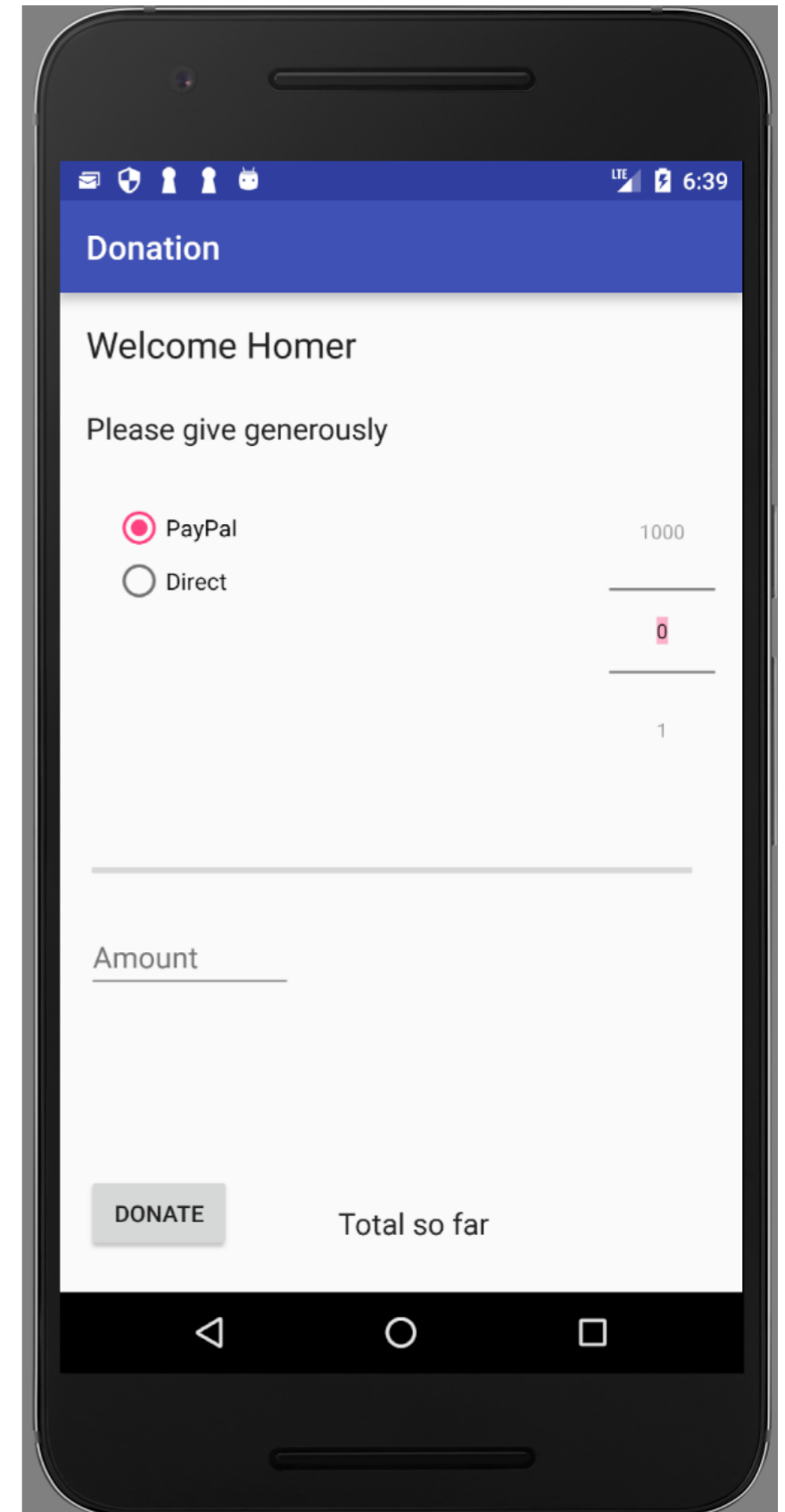
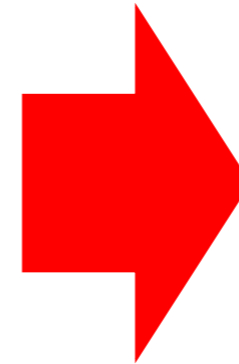
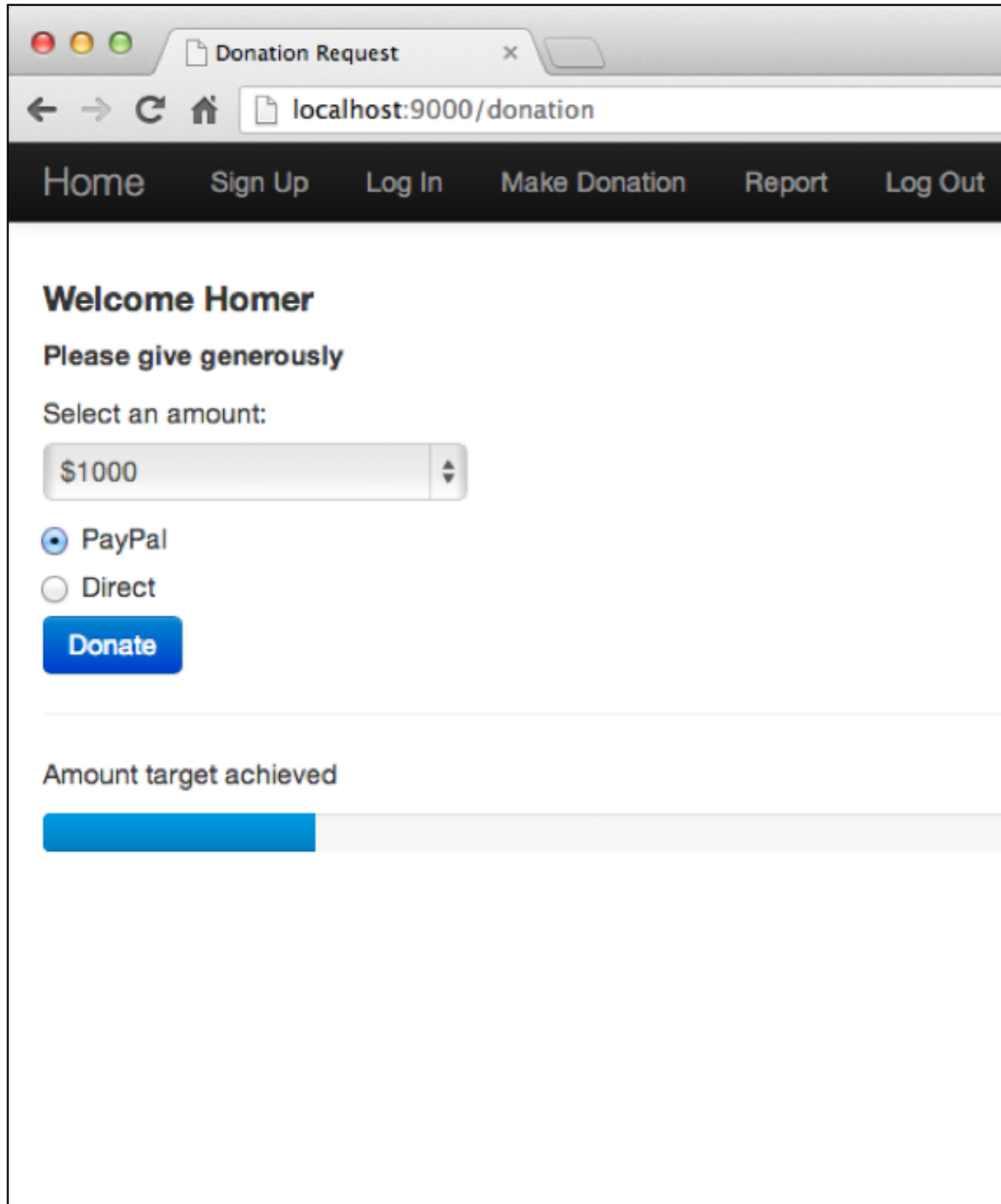
---

Donation 1.0

A single activity and layout app.



Objective is to reproduce this web app feature in Android

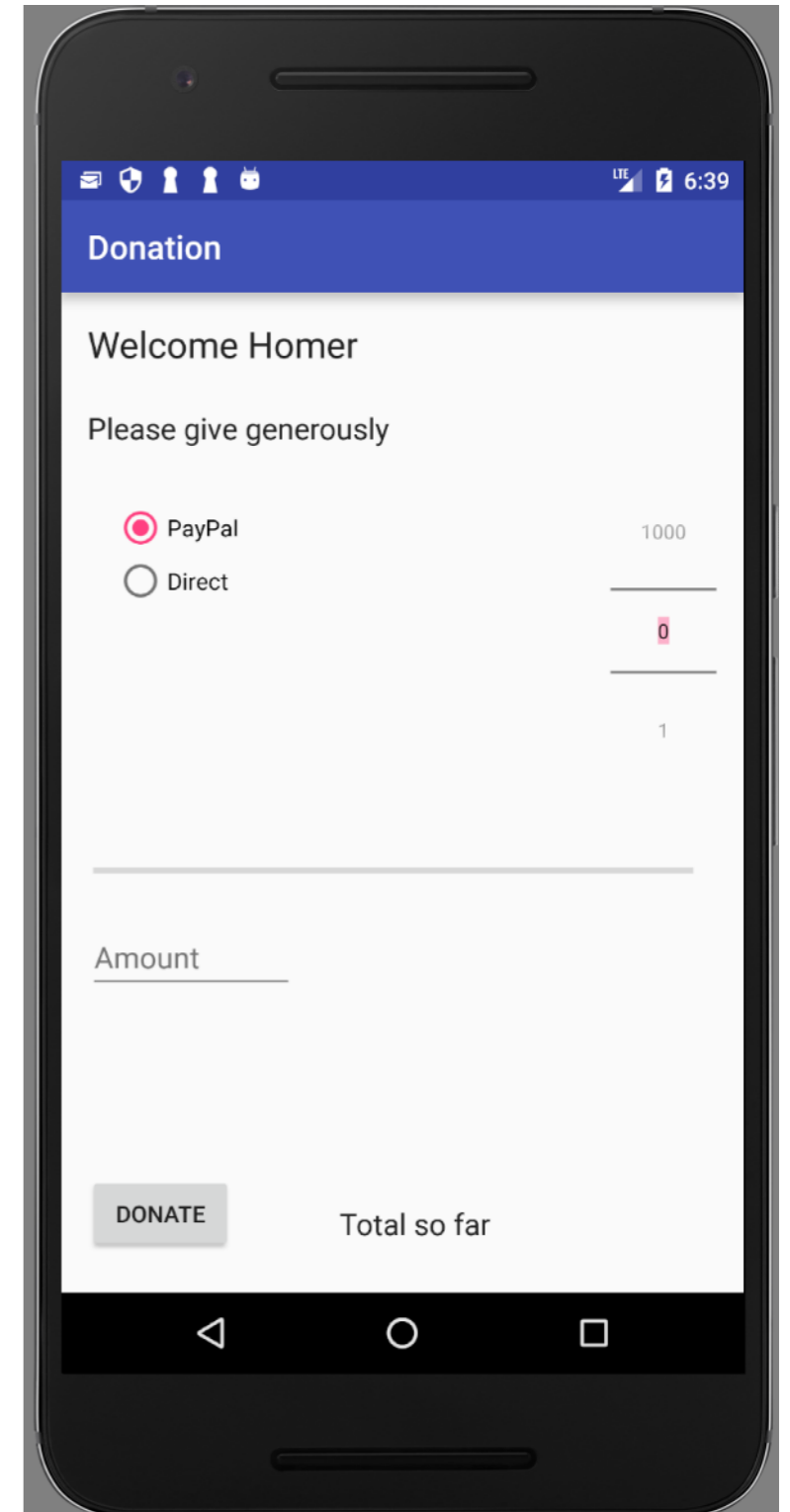


# App Basics

---

Donation is a simple app, so it will have a single Activity subclass named Donate. More complex apps will typically have more activities.

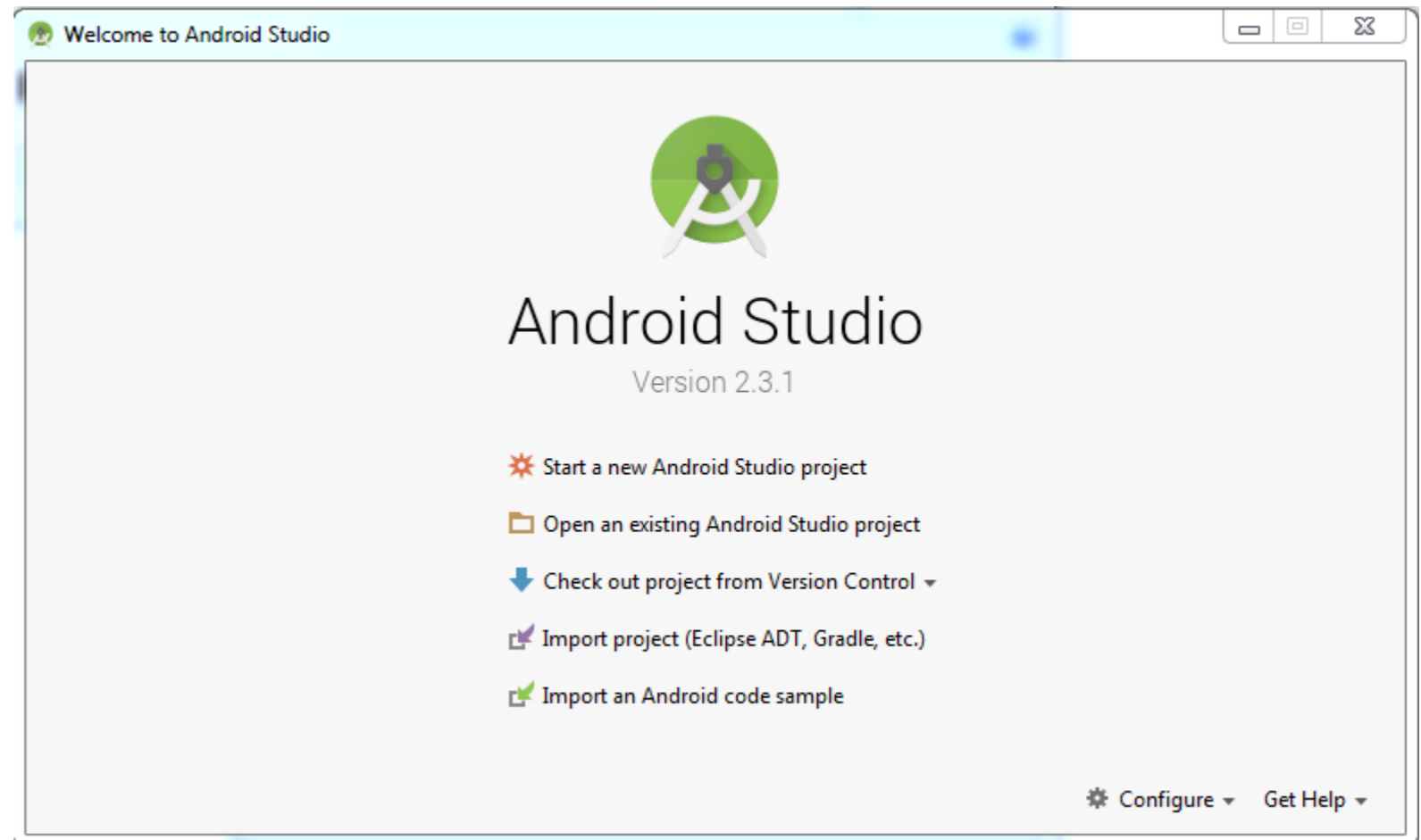
1. Create a new app called “Donation”.
2. Create an activity called “Donate”, which will manage the UI shown.
3. The XML layout defines a set of UI objects and their position on the screen.
4. Run the app.



# 1. Creating a new **App** called “Donation”

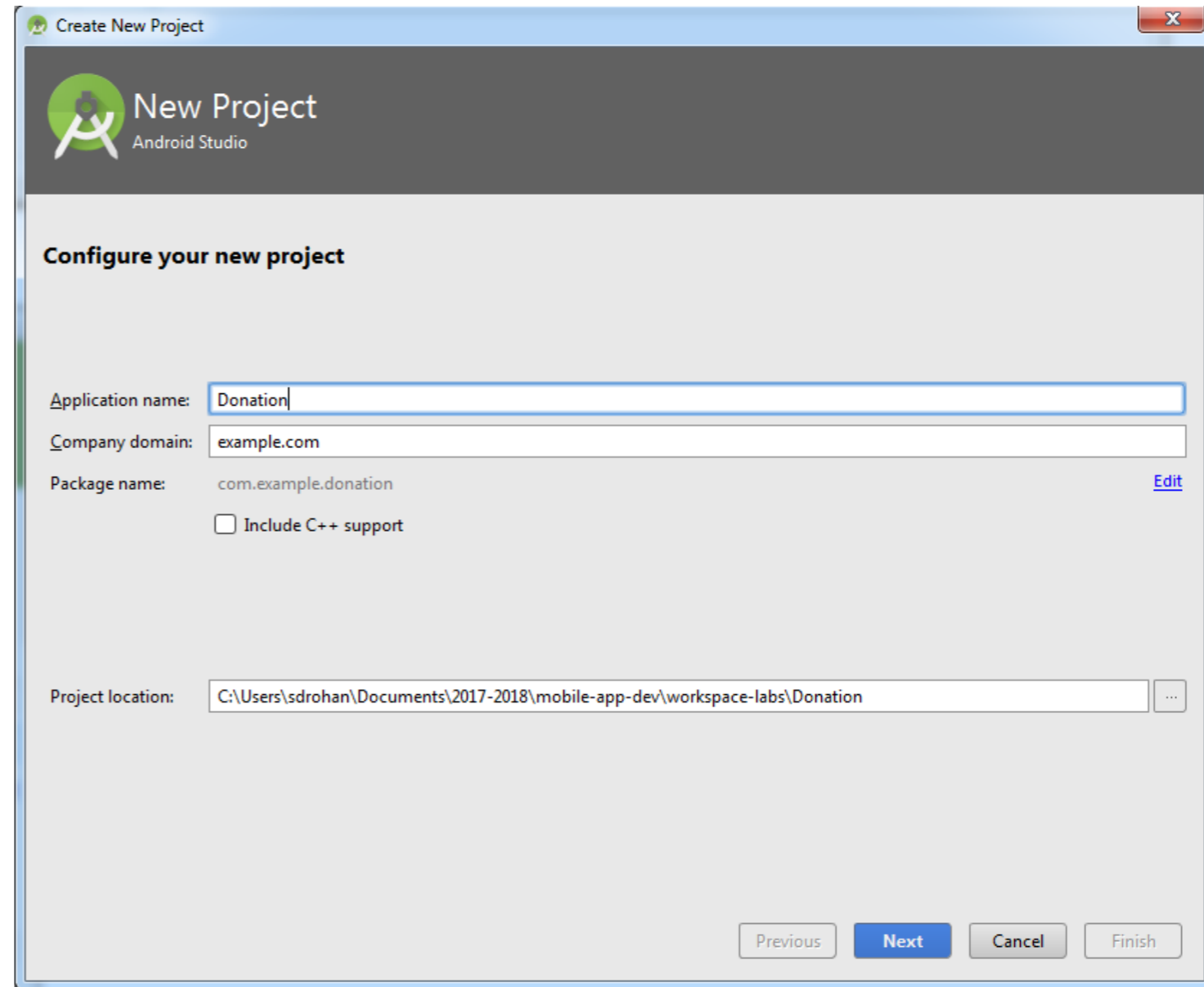
---

- To create a new project, open Android Studio and choose:  
*Start a new Android Studio project*



# 1. Creating a new **App** called “Donation”

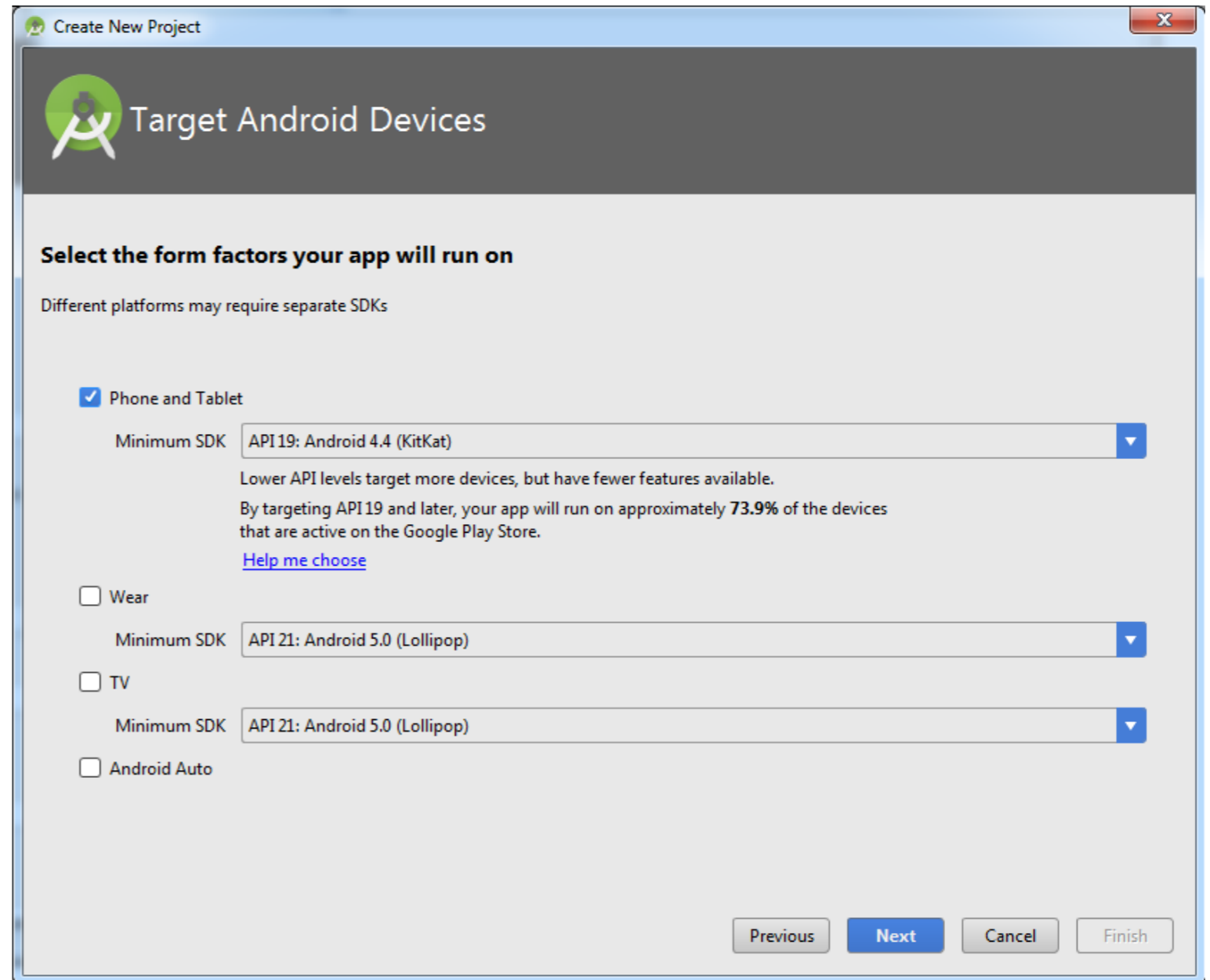
- Enter “Donation” as the application name.
- For the Company Domain, enter *example.com*. Notice that the package name is automatically generated using a “reverse DNS” convention in which the domain name of your organization is reversed and suffixed with further identifiers: *com.example.donation*
- This convention keeps package names unique and distinguishes applications from each other on a device and on Google Play.
- A default Project location is presented. You may change this if you wish.



# 1. Creating a new **App** called “Donation”

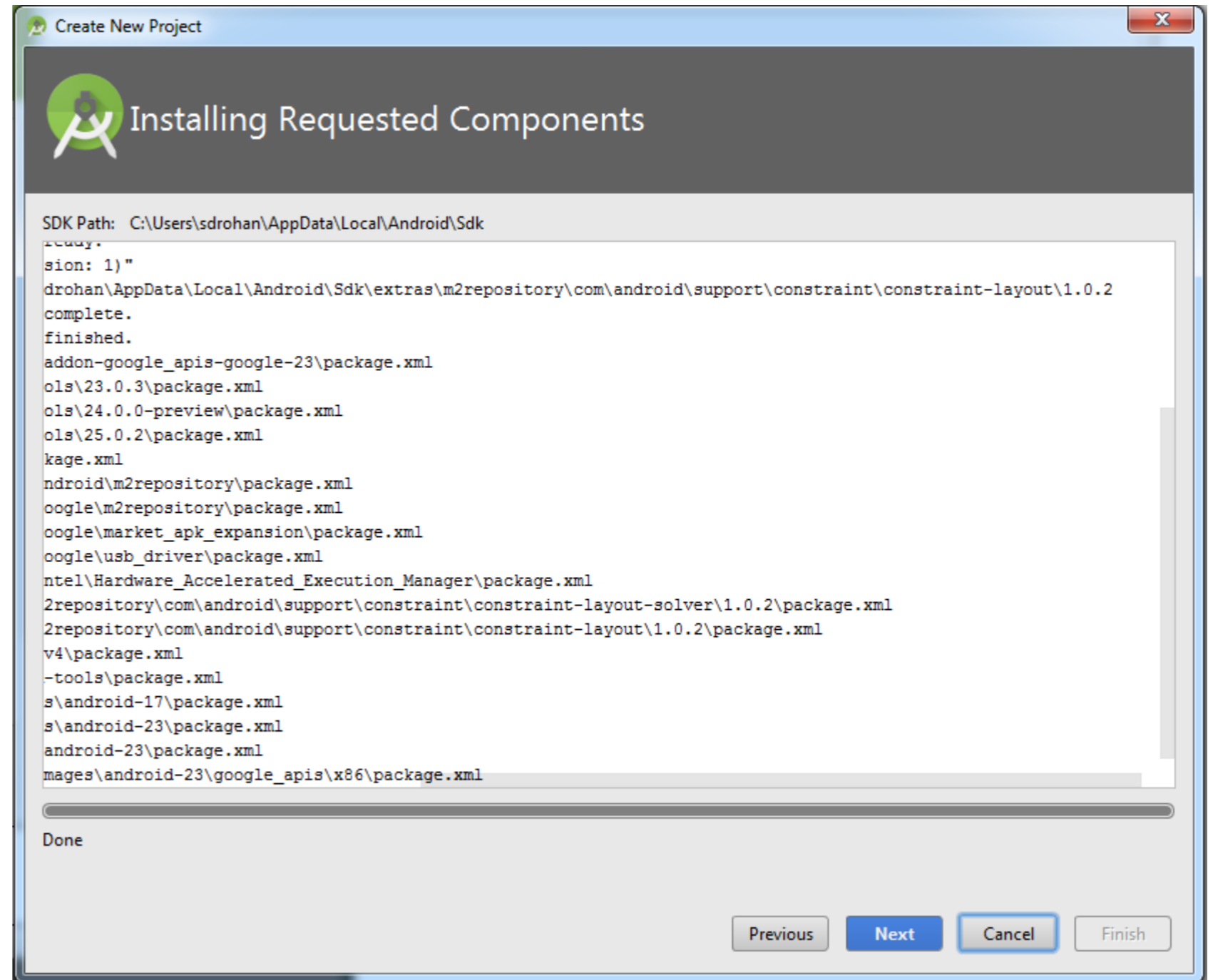
---

- Target the android devices that we intend developing for:
  - We chose Phone & Tablet and a minimum SDK API 19.
  - Our app will run only on devices specified with an API level 19 or greater.



# 1. Creating a new **App** called “Donation”

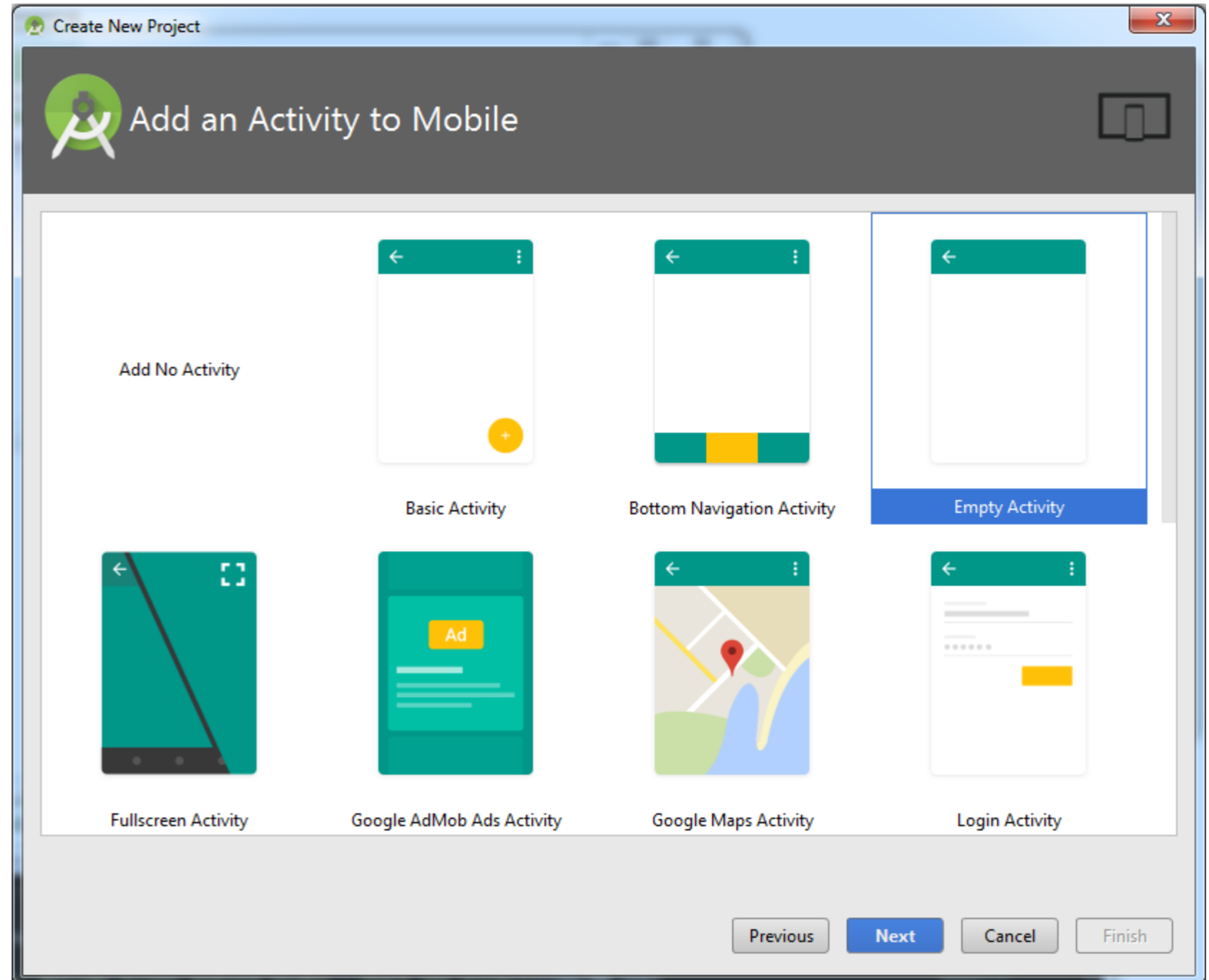
If this is your first time running Android Studio, you may get this screen:





## 2. Create an **Activity** called “DonationActivity”

Choose an Empty Activity as the template for your DonationActivity.



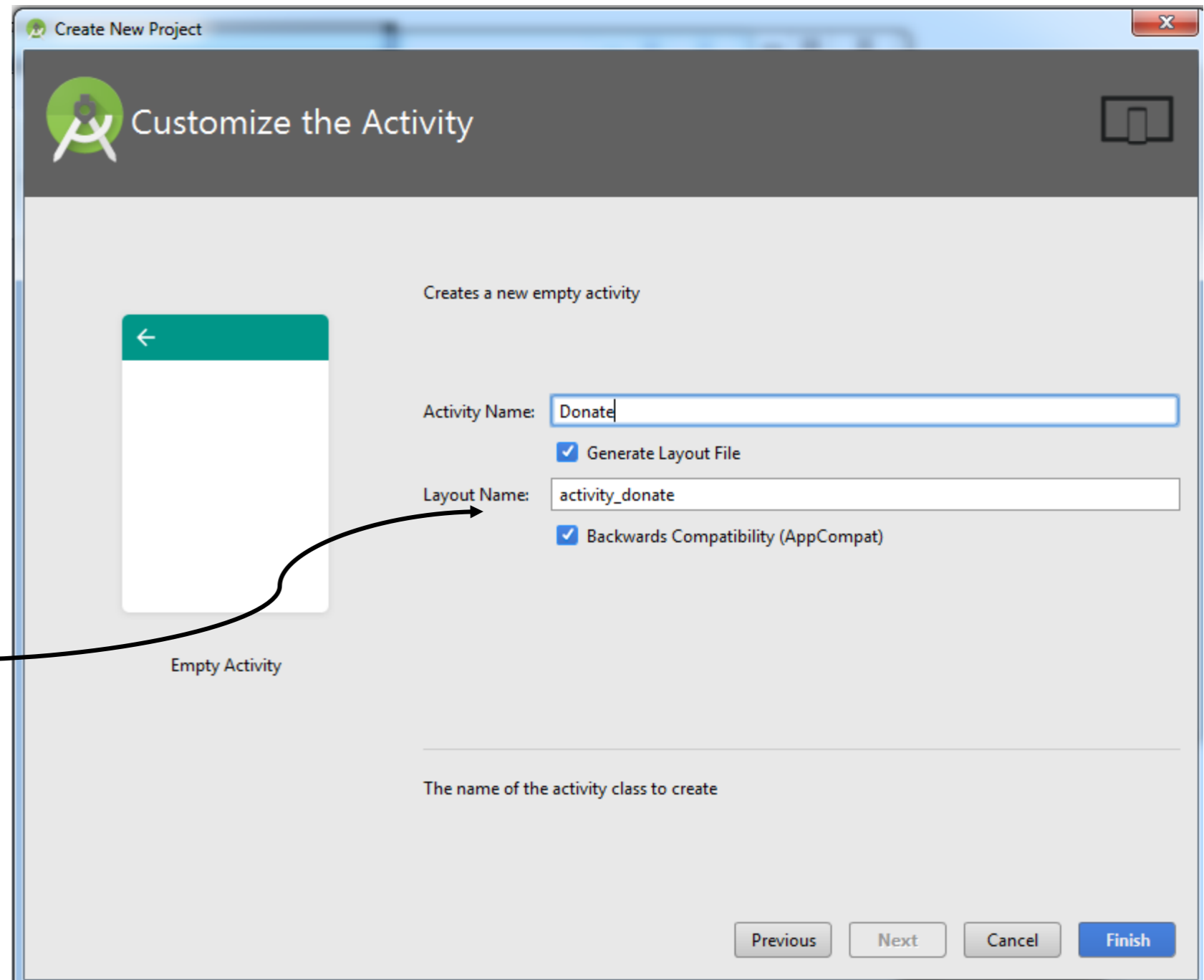
## 2. Create an **Activity** called “DonationActivity”

Call the Activity  
“Donate”.

Notice that the layout name  
“activity\_donate” is  
automatically populated.

The layout name reverses  
the order of the activity  
name, is all lowercase, and  
has underscores between  
words.

This naming style is  
recommended for layouts as  
well as other resources that  
you will learn.

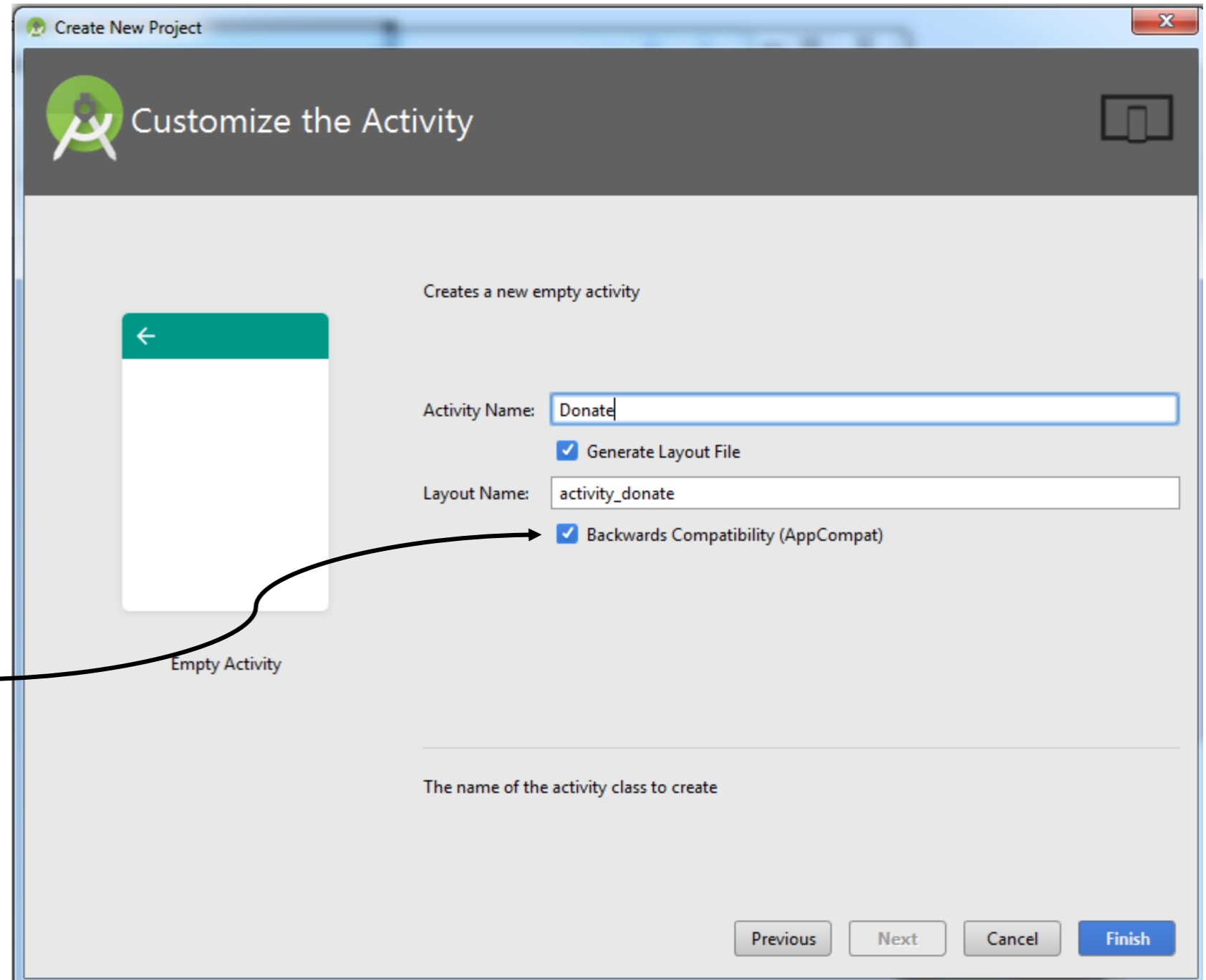


## 2. Create an **Activity** called “DonationActivity”

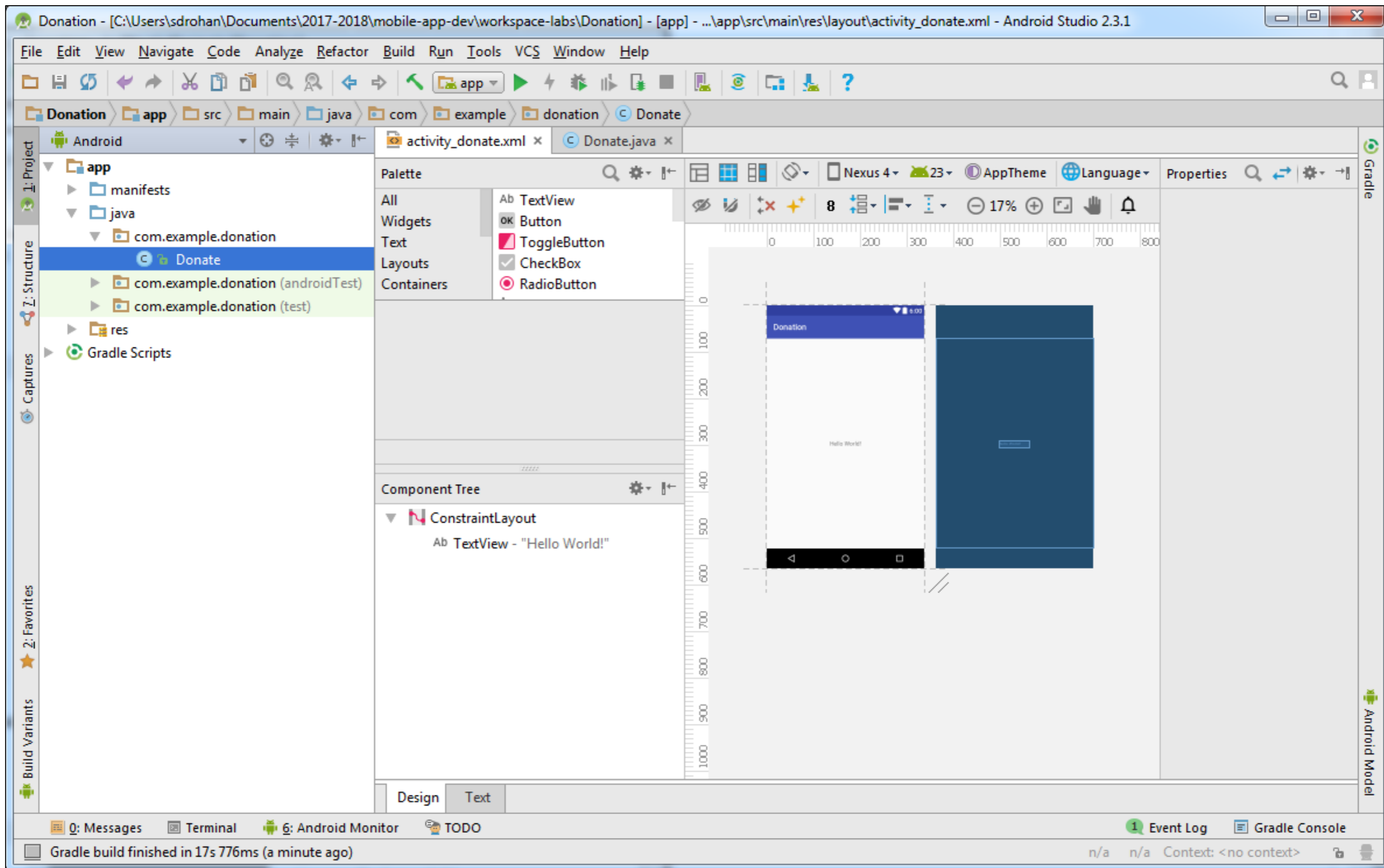
### *Backwards Compatibility*

Ticking this check box ensures that the class extends AppCompatActivity instead of Activity.

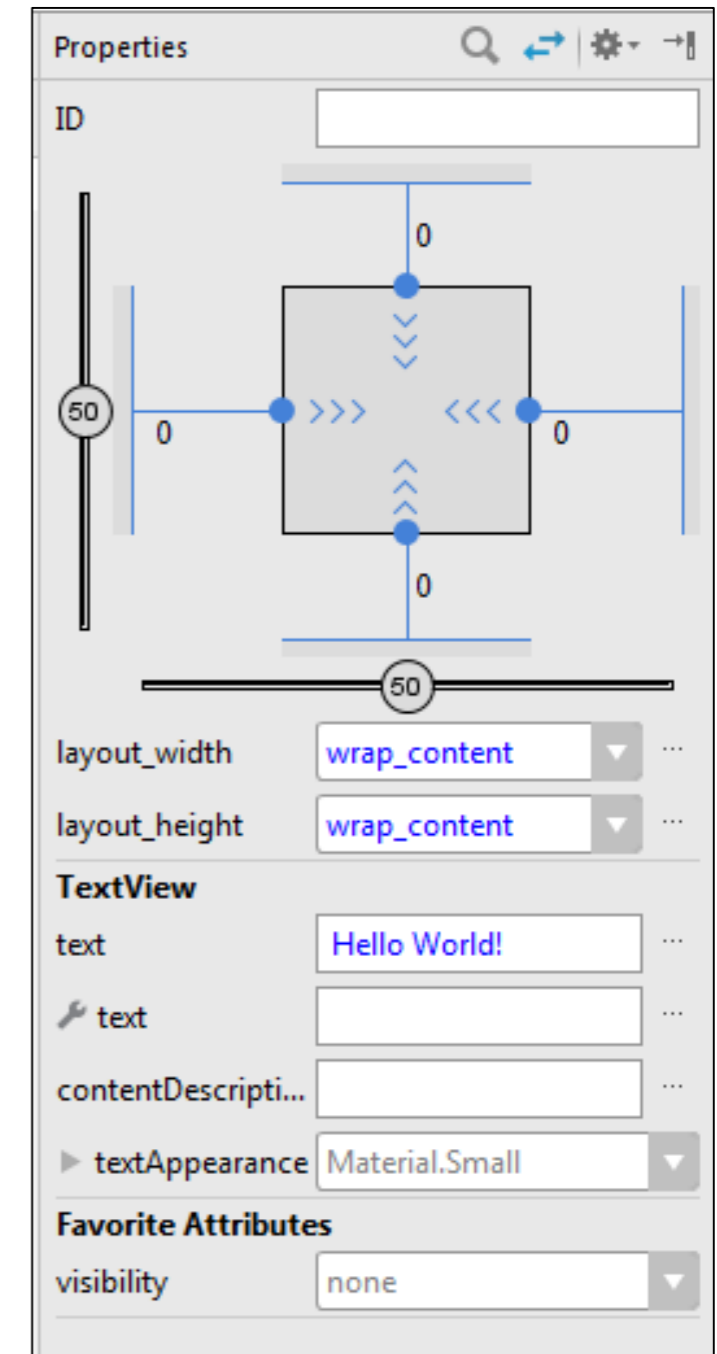
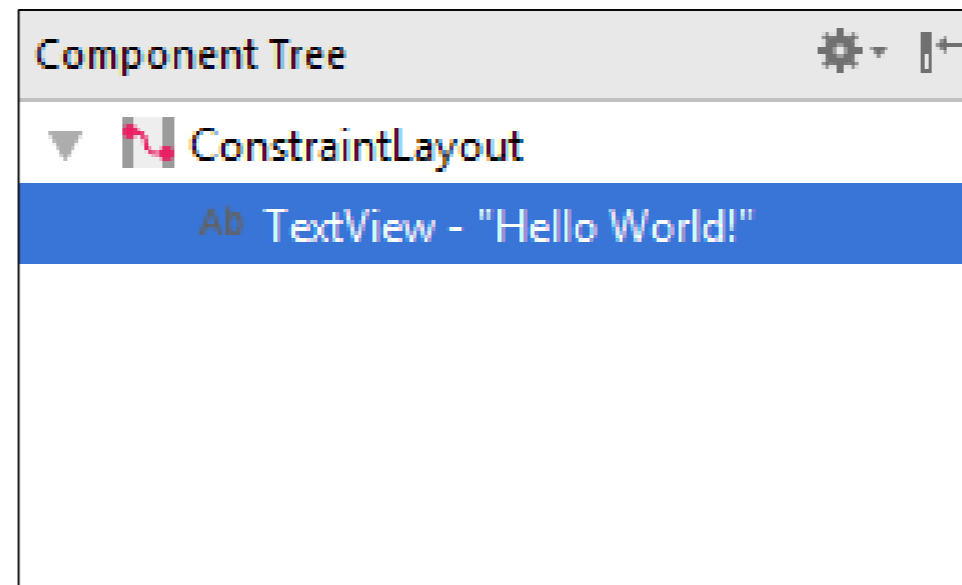
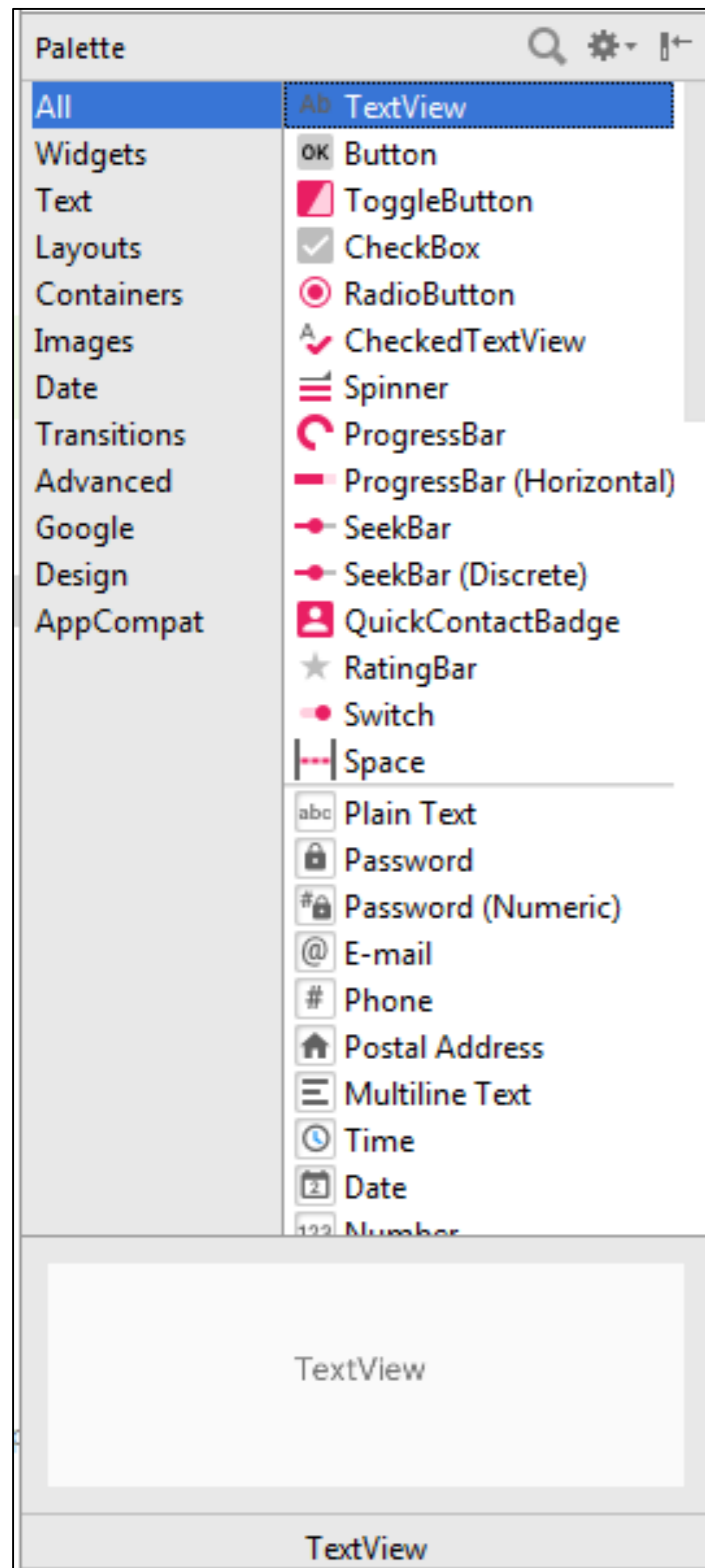
Activity is the base class; AppCompatActivity extends this base class...it is recommended to use the specialised class as opposed to the base class.



# The Donation Project Perspective



# Palette, Component Tree, Properties



# So what was created by the wizard?

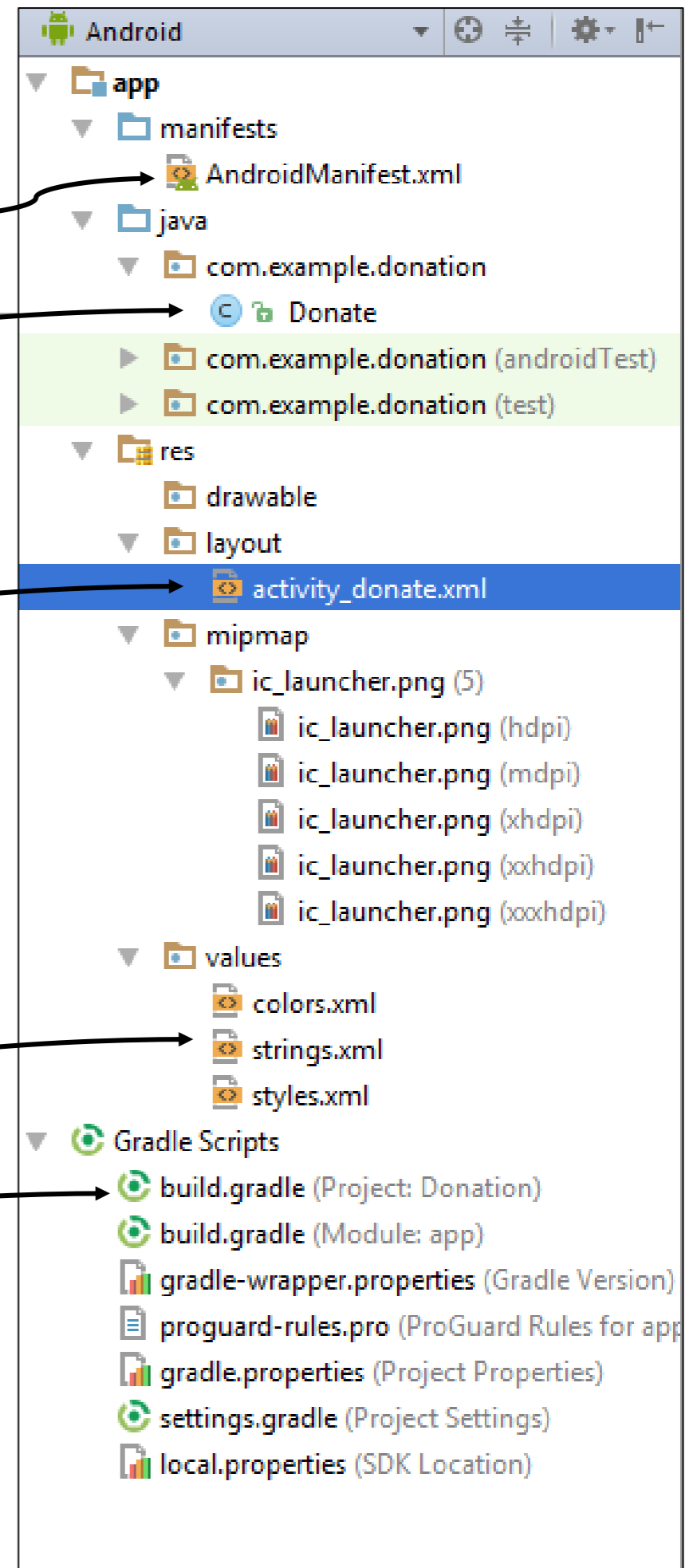
Manifest file

Donate.java (activity class)

activity\_donate.xml (layout)

strings.xml resource file

Gradle build system files



# When finished the labs this week...

---

...your project will look like this

# So what was created by the wizard?

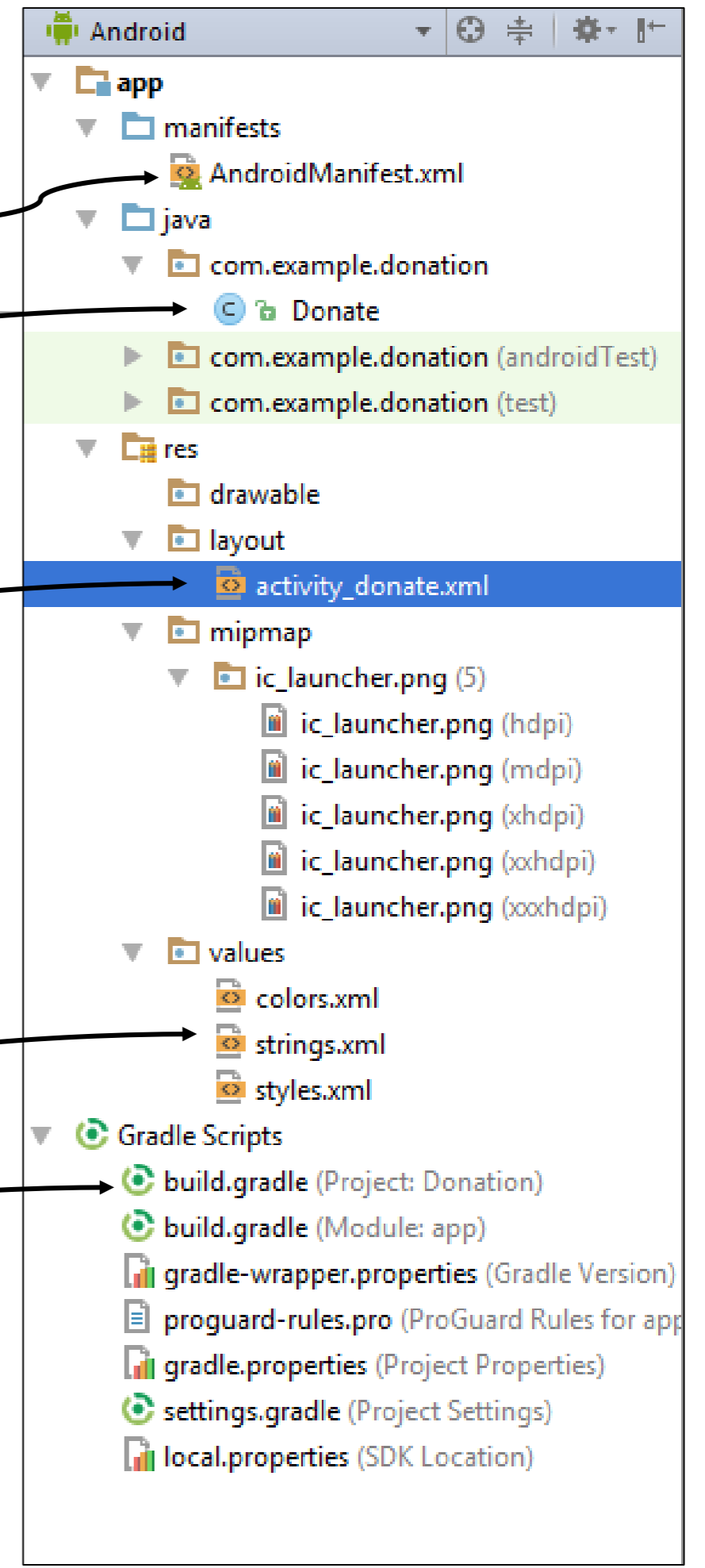
Manifest file

Donate.java (activity class)

activity\_donate.xml (layout)

strings.xml resource file

Gradle build system files





# AndroidManifest.xml

Every application must have an `AndroidManifest.xml` file (with precisely that name) in its root directory. The manifest file provides essential information about your app to the Android system, which the system must have before it can run any of the app's code (<https://developer.android.com/guide/topics/manifest/manifest-intro.html>)

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".Donate">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

# So what was created by the wizard?

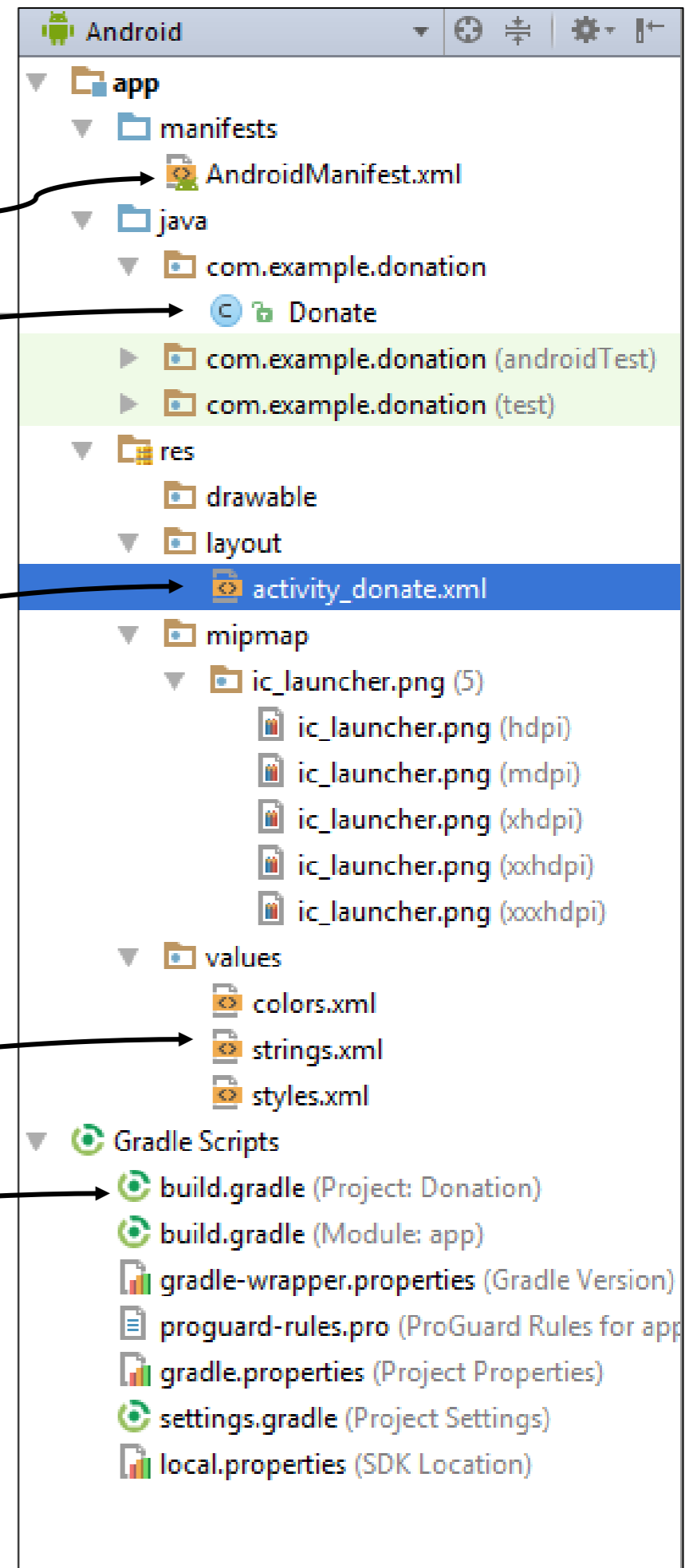
Manifest file

Donate.java (activity class)

activity\_donate.xml  
(layout)

strings.xml resource file

Gradle build system files



# activity\_donate.xml

The image displays the design view of an Android activity named `activity_donate.xml`. The interface is split into two main sections: a top blue header and a main white content area.

**Header:** A blue bar with the title "Donation" in white text. The system status bar at the top shows a Wi-Fi icon, a battery icon, and the time "7:00".

**Main Content Area:**




- Welcome Homer:** A text label.
- Please give generously:** A text label.
- Payment Options:** Two radio buttons labeled "PayPal" (with a red target icon) and "Direct" (with a black circle icon).
- Amount Input:** A text input field with the value "0".
- Amount Label:** A text label "Amount" positioned above the input field.
- DONATE Button:** A grey button with the text "DONATE".
- Total so far:** A text label.

The right side of the image shows the Android Studio design view, which is a dark blue background with a grid. It illustrates the layout of the UI elements with various constraints and dimensions:

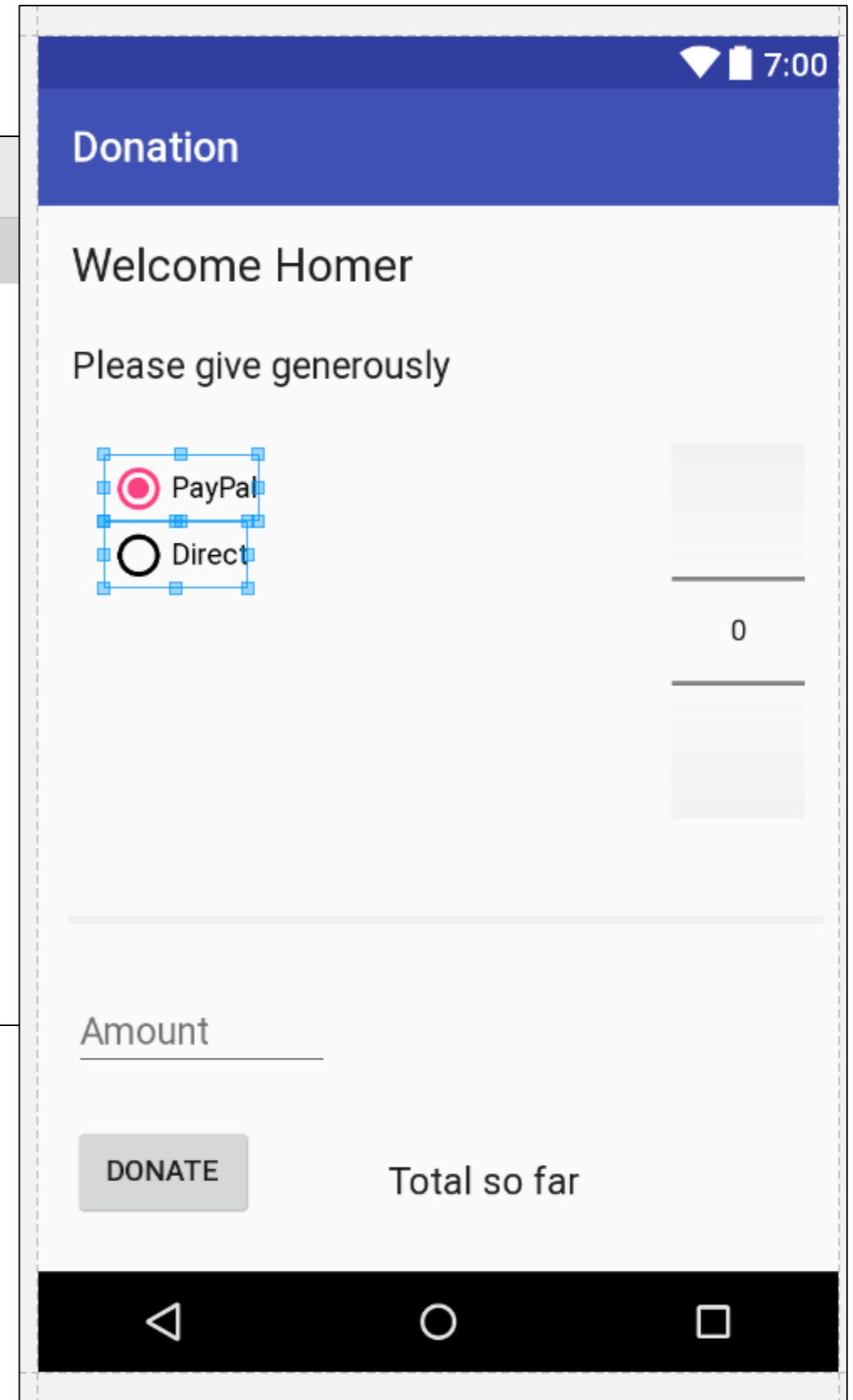
- The "Welcome Homer" and "Please give generously" text views are positioned at the top with a height of 16 pixels.
- The "PayPal" and "Direct" radio buttons are positioned below the text, with a height of 24 pixels.
- The "Amount" text label and the "Amount" input field are positioned below the radio buttons, with a height of 150 pixels.
- The "DONATE" button and the "Total so far" text label are positioned at the bottom, with a height of 24 pixels.
- Dimensions of 16, 24, 150, and 44 are used to define the vertical spacing and heights of the UI elements.

# activity\_donate.xml

## Component Tree

- ▼  ConstraintLayout
  - Ab **donateTitle** (TextView) - "@string/donateTitle"
  - Ab **donateSubtitle** (TextView) - "@string/donateSubtitle"
  - donateButton** (Button) - "@string/donateButton"
  - ▼  **paymentMethod** (RadioGroup) (horizontal)
    - payPal** (RadioButton) - "@string/PayPal"
    - Direct** (RadioButton) - "@string/Direct"
  -  **progressBar** (Horizontal)
  - amountPicker** (NumberPicker)
  - amountText** (EditText)
  - Ab **amountTotalLabel** (TextView) - "@string/TotalSoFar"
  - Ab **amountTotal** (TextView)

## View Hierarchy



# activity\_donate.xml

Component Tree

- ▼ ConstraintLayout
  - Ab **donateTitle** (TextView) - "@string/donateTitle"
  - Ab **donateSubtitle** (TextView) - "@string/donateSubtitle"
  - OK **donateButton** (Button) - "@string/donateButton"
  - ▼ **paymentMethod** (RadioGroup) (horizontal)
    - 📍 **payPal** (RadioButton) - "@string/PayPal"
    - 📍 **Direct** (RadioButton) - "@string/Direct"
  - 🔄 **progressBar** (Horizontal)
  - 10 **amountPicker** (NumberPicker)
  - abc **amountText** (EditText)
  - Ab **amountTotalLabel** (TextView) - "@string/TotalSoFar"
  - Ab **amountTotal** (TextView)

- ConstraintLayout is the root
- It has 9 child nodes
  - 4 TextViews
  - 1 Push Button
  - 1 Radio Group
    - which has 2 child node RadioButtons
  - 1 Progress Bar
  - 1 Number Picker
  - 1 EditText

## View Hierarchy

# activity\_donate.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context="com.example.donation.Donate"
tools:layout_editor_absoluteY="81dp"
tools:layout_editor_absoluteX="0dp">

<TextView
    android:id="@+id/donateTitle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="16dp"
    android:text="@string/donateTitle"
    android:textAppearance="@android:style/TextAppearance"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<TextView
    android:id="@+id/donateSubtitle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/donateSubtitle"
    android:textAppearance="@android:style/TextAppearance"
    android:layout_marginTop="24dp"
    app:layout_constraintTop_toBottomOf="@+id/donateTit"
    android:layout_marginStart="16dp"
    app:layout_constraintLeft_toLeftOf="parent" />

<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" />
```

```
<RadioGroup
    android:id="@+id/paymentMethod"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="32dp"
    android:layout_marginTop="32dp"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/donateSubtitle">

    <RadioButton
        android:id="@+id/payPal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:checked="true"
        android:text="@string/PayPal" />

    <RadioButton
        android:id="@+id/Direct"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/Direct" />
</RadioGroup>

<ProgressBar
    android:id="@+id/progressBar"
    style="?android:attr/progressBarStyleHorizontal"
    android:layout_width="362dp"
    android:layout_height="18dp"
    android:layout_marginLeft="16dp"
    app:layout_constraintLeft_toLeftOf="parent"
    android:layout_marginRight="16dp"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintHorizontal_bias="0.2"
    android:layout_marginTop="150dp"
    app:layout_constraintTop_toBottomOf="@+id/paym"
    android:layout_marginStart="16dp"
    android:layout_marginEnd="16dp" />

<NumberPicker
    android:id="@+id/amountPicker"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="16dp"
    android:layout_marginRight="44dp"
    android:layout_marginTop="114dp"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    tools:layout_editor_absoluteX="273dp" />
```

```
<EditText
    android:id="@+id/amountText"
    android:layout_width="125dp"
    android:layout_height="42dp"
    android:ems="10"
    android:hint="@string/Amount"
    android:inputType="number"
    android:layout_marginStart="16dp"
    app:layout_constraintLeft_toLeftOf="parent"
    android:layout_marginTop="24dp"
    app:layout_constraintTop_toBottomOf="parent"
    android:layout_marginLeft="16dp" />
```

```
<TextView
    android:id="@+id/amountTotalLabel"
    android:layout_width="94dp"
    android:layout_height="22dp"
    android:text="@string/TotalSoFar"
    android:textAlignment="viewStart"
    android:textAppearance="@android:style/TextAppearance"
    app:layout_constraintLeft_toRightOf="parent"
    android:layout_marginStart="64dp"
    android:layout_marginLeft="64dp"
    app:layout_constraintBottom_toBottomOf="parent"
    android:layout_marginBottom="32dp" />
```

```
<TextView
    android:id="@+id/amountTotal"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="32dp"
    android:layout_marginLeft="32dp"
    android:layout_marginStart="32dp"
    android:contentDescription="@string/AmountTotal"
    android:textAppearance="@android:style/TextAppearance"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toRightOf="parent" />
```

```
</android.support.constraint.ConstraintLayout>
```

Raw XML

# Widget attributes

The screenshot displays the Android Studio IDE with a 'Donation' app. The interface is split into two views: a design view on the left and a code view on the right. The design view shows a blue header with the title 'Donation', a welcome message 'Welcome Homer', and a prompt 'Please give generously'. A 'DONATE' button is visible at the bottom. The code view shows the corresponding XML layout with a `TextView` widget for the button text. The properties panel on the right is open, showing the widget's ID as 'donateButton'. The 'TextView' section is expanded, showing the 'text' attribute set to '@string/donateButton'. Red arrows point to the ID field, the 'text' attribute, and the 'layout\_width' and 'layout\_height' attributes.

Properties

ID `donateButton`

layout\_width `wrap_content`

layout\_height `wrap_content`

**Button**

style `buttonStyle`

background

backgroundTint

stateListAnimator

elevation

visibility `none`

onClick `none`

**TextView**

text `@string/donateButton`

contentDescription

textAppearance `appCompat.Widget.Button`

**Favorite Attributes**

visibility `none`



# Widget attributes

The screenshot displays the Android Studio interface for a 'Donation' app. The design view shows a blue header with the word 'Donation', a welcome message 'Welcome Homer', and a prompt 'Please give generously'. A 'DONATE' button is located at the bottom. The Properties panel on the right shows the 'donateButton' widget with various attributes. A code snippet in a white box shows the XML for the button with attributes like android:id, android:layout\_width, android:layout\_height, android:layout\_marginBottom, android:text, app:layout\_constraintBottom\_toBottomOf, android:onClick, android:layout\_marginLeft, and app:layout\_constraintLeft\_toLeftOf.

```
<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" />
```



# Widget attributes

---

```
<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" />
```

- The **android:layout\_width** and **android:layout\_height** attributes are required for almost every type of widget.
- They are typically set to either **match\_parent** or **wrap\_content**:
  - **match\_parent** view will be as big as its parent
  - **wrap\_content** view will be as big as its contents require
  - Or a specific measurement.

# So what was created by the wizard?

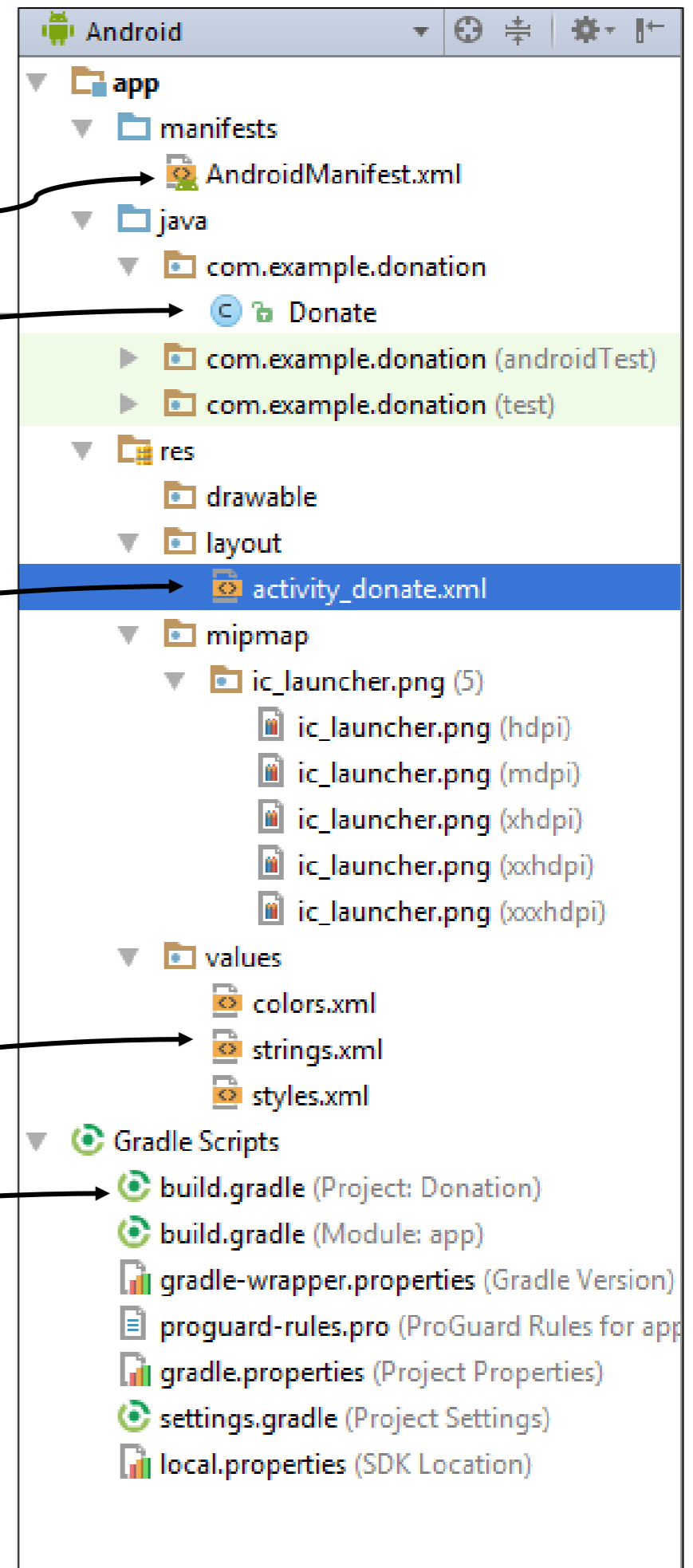
Manifest file

Donate.java (activity class)

activity\_donate.xml (layout)

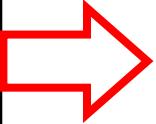
strings.xml resource file

Gradle build system files



# strings.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" />
```

- Did you notice that the values of strings are not literal strings. They are references to string resources
- A string resource is a string that lives in a separate XML file called a strings file.
- You can give a widget a hard-coded string, like  
**android:text="True"**  
but it is usually not a good idea.
- Placing strings into a separate file and then referencing them is better, making localization easy.

# String Resources File

---

```
<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>
</resources>
```

- Every project includes a default strings file named strings.xml (in the res/values/ directory).
- Whenever you refer to, for example, '@string/Direct' in any XML file in the project, you will get the literal string "Direct" at runtime.
- The default strings file is named strings.xml, but you can name a strings file anything you want.
- You can also have multiple strings files in a project. As long as the file is located in res/values/, has a resources root element, and contains child string elements, your strings will be found and used appropriately.

```

@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);
}
}

```

```

package com.example.donation;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.NumberPicker;
import android.widget.ProgressBar;
import android.widget.RadioGroup;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import java.util.Locale;

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;
}

```

Donate.java

```

@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);
}
}

```

```

package com.example.donation;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.NumberPicker;
import android.widget.ProgressBar;
import android.widget.RadioGroup;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import java.util.Locale;

public class Donate extends AppCompatActivity {

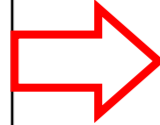
    private int totalDonated = 0;
    private int target = 10000;
    private RadioGroup paymentMethod;
    private ProgressBar progressBar;
    private NumberPicker amountPicker;
    private EditText amountText;
    private TextView amountTotal;
}

```

Donate.java

# R.java

---



```
<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" />
```

- Did you notice that the id is also not a literal string. They are references to resources in R.java.
- This is a file generated by the android build system.
- It bridges the world of resources and Java, allowing resource IDs to be used in pure java code.
- Never edit or modify this file, it is automatically updated as new resources are added/edited.



# R.java

```
<TextView
    android:id="@+id/donateTitle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="16dp"
    android:text="@string/donateTitle"
    android:textAppearance="@android:style/TextAppearance.Large"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<TextView
    android:id="@+id/donateSubtitle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/donateSubtitle"
    android:textAppearance="@android:style/TextAppearance.Medium"
    android:layout_marginTop="24dp"
    app:layout_constraintTop_toBottomOf="@+id/donateTitle"
    android:layout_marginStart="16dp"
    app:layout_constraintLeft_toLeftOf="parent" />

<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" />
```

```
public final class R
{
    //...
    public static final class id
    {
        public static final int Direct = 0x7f080006;
        public static final int PayPal = 0x7f080005;
        public static final int action_settings = 0x7f08000c;
        public static final int amountLabel = 0x7f080009;
        public static final int amountPicker = 0x7f080004;
        public static final int amountText = 0x7f080008;
        public static final int amountTotal = 0x7f08000a;
        public static final int donateButton = 0x7f080007;
        public static final int donateSubtitle = 0x7f080001;
        public static final int donateTitle = 0x7f080000;
        public static final int paymentMethod = 0x7f080002;
        public static final int progressBar = 0x7f080003;
        public static final int totalLabel = 0x7f08000b;
    }
    public static final class layout
    {
        public static final int activity_donate = 0x7f030000;
    }
    public static final class menu
    {
        public static final int donate = 0x7f070000;
    }
    public static final class string
    {
        public static final int Direct = 0x7f050006;
        public static final int PayPal = 0x7f050005;
        public static final int action_settings = 0x7f050001;
        public static final int amount = 0x7f050007;
        public static final int amountSoFarLabel = 0x7f050009;
        public static final int app_name = 0x7f050000;
        public static final int donateButton = 0x7f050004;
        public static final int donateSubtitle = 0x7f050003;
        public static final int donateTitle = 0x7f050002;
        public static final int initialAmount = 0x7f050008;
    }
    //...
}
```



# Donate.java

@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.p

    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.
        toast.show();
        Log.v("Donate", "Target Exceeded: " + totalDonated);
    }
    else {
        totalDonated = totalDonated + donatedAmount;
        progressBar.setProgress(totalDonated);
        Log.v("Donate", amountPicker.getValue() + " donated by " + me

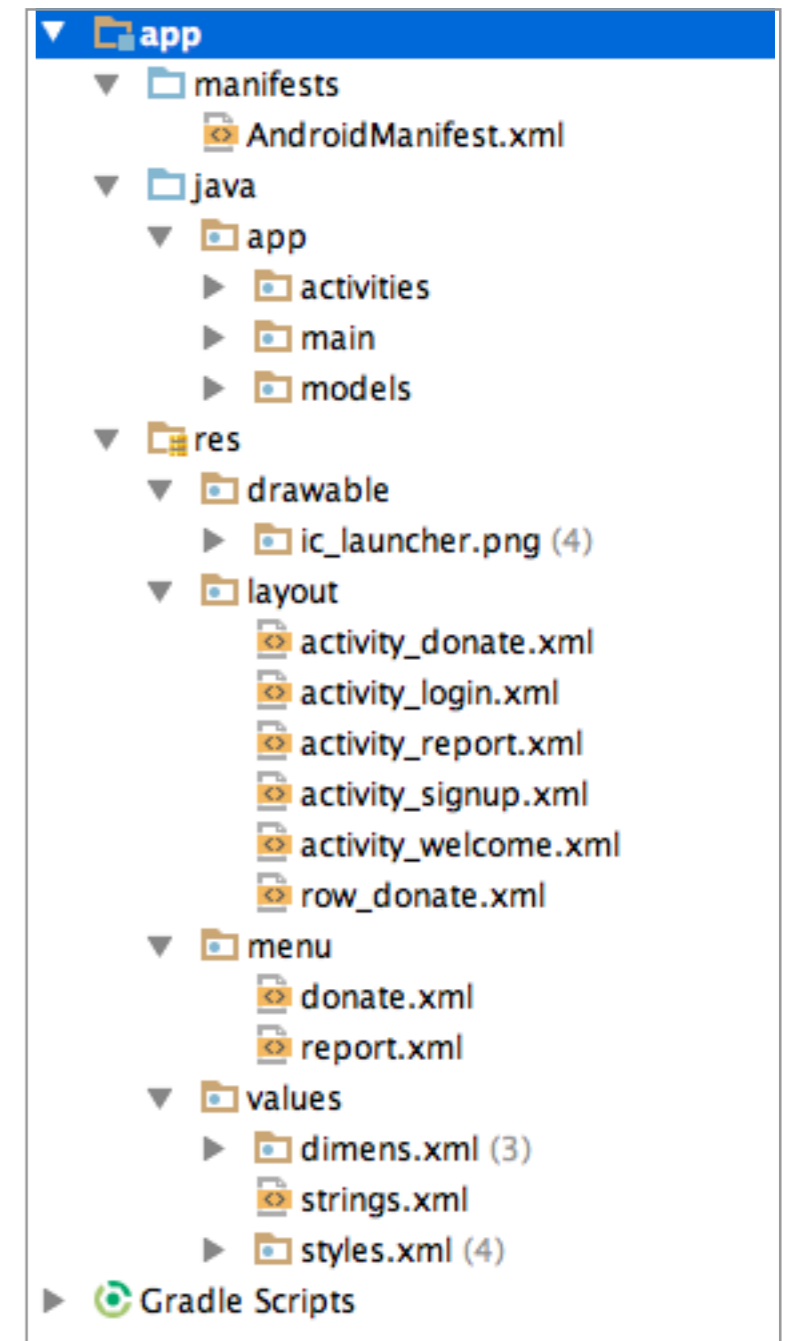
    }

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

```
public final class R
{
    //...
    public static final class id
    {
        public static final int Direct = 0x7f080006;
        public static final int PayPal = 0x7f080005;
        public static final int action_settings = 0x7f08000c;
        public static final int amountLabel = 0x7f080009;
        public static final int amountPicker = 0x7f080004;
        public static final int amountText = 0x7f080008;
        public static final int amountTotal = 0x7f08000a;
        public static final int donateButton = 0x7f080007;
        public static final int donateSubtitle = 0x7f080001;
        public static final int donateTitle = 0x7f080000;
        public static final int paymentMethod = 0x7f080002;
        public static final int progressBar = 0x7f080003;
        public static final int totalLabel = 0x7f08000b;
    }
    public static final class layout
    {
        public static final int activity_donate = 0x7f030000;
    }
    public static final class menu
    {
        public static final int donate = 0x7f070000;
    }
    public static final class string
    {
        public static final int Direct = 0x7f050006;
        public static final int PayPal = 0x7f050005;
        public static final int action_settings = 0x7f050001;
        public static final int amount = 0x7f050007;
        public static final int amountSoFarLabel = 0x7f050009;
        public static final int app_name = 0x7f050000;
        public static final int donateButton = 0x7f050004;
        public static final int donateSubtitle = 0x7f050003;
        public static final int donateTitle = 0x7f050002;
        public static final int initialAmount = 0x7f050008;
    }
    //...
}
```

# Resources and resource IDs

- A layout is a resource. A resource is a piece of your application that is not code - things like image files, audio files, and XML files.
- Resources for your project live in a subdirectory of the res directory.
- To access a resource in code, you use its resource ID.
- To see the current resource IDs for your app, go to the package explorer and reveal the contents of the gen directory. Find and open R.java.
- Because this is generated by the Android build process, you should not change it, as you are subtly warned at the top of the file.



# So what was created by the wizard?

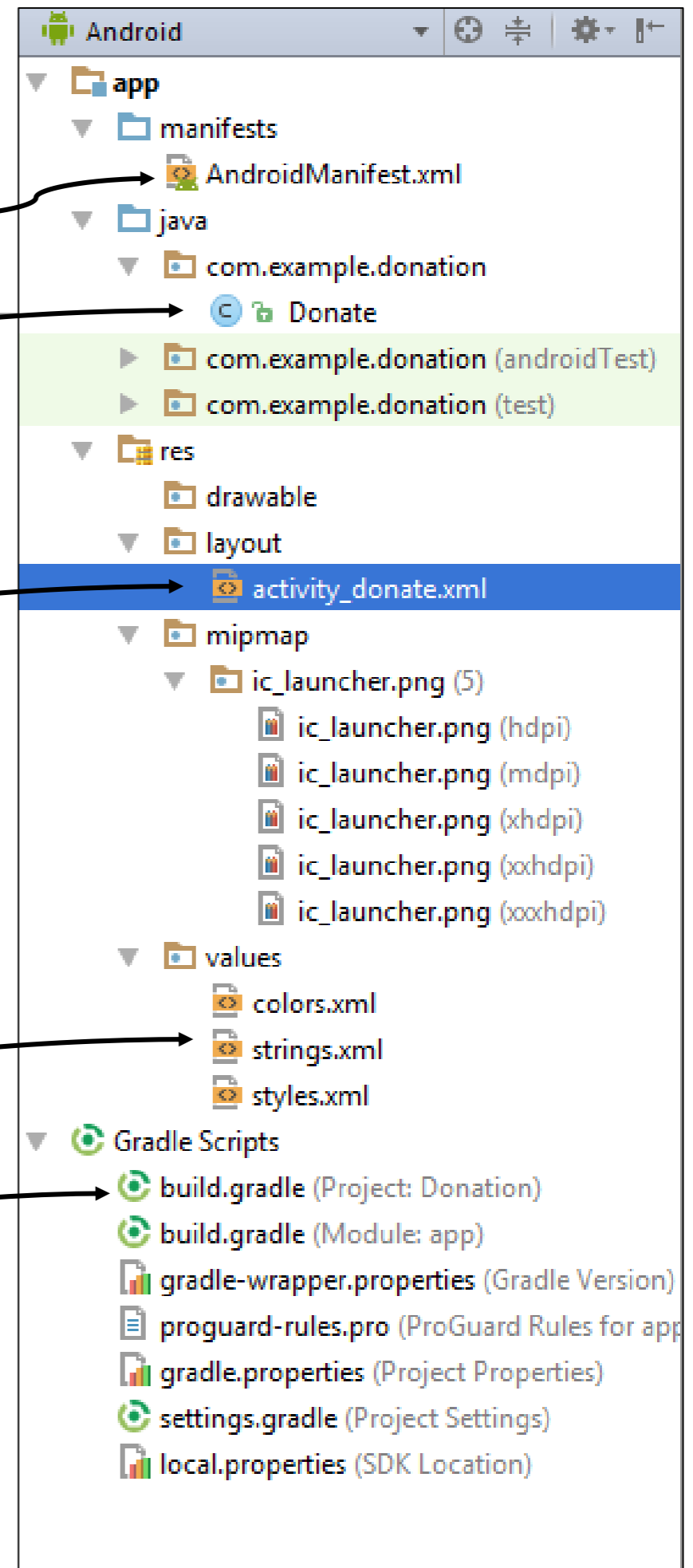
Manifest file

Donate.java (activity class)

activity\_donate.xml (layout)

strings.xml resource file

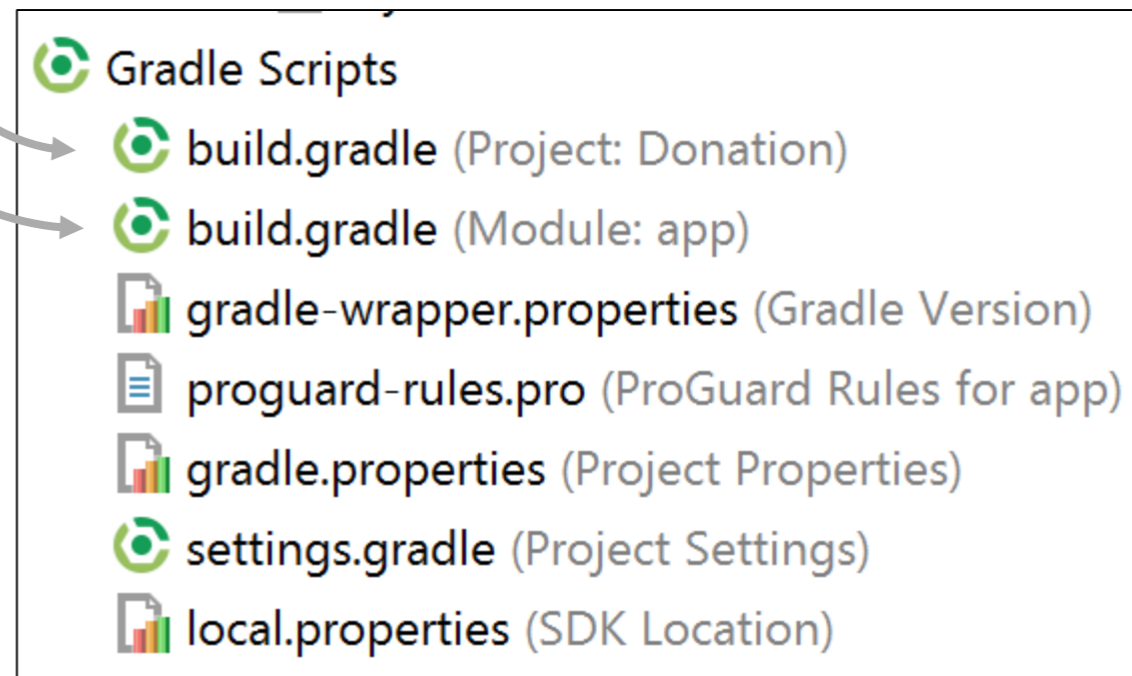
Gradle build system files



# Gradle build system

---

- Gradle is an automated build system that is integrated into Android Studio.
- It allows us to add libraries to our project with just one line of code.
- Our project has two Gradle files (written in Groovy) that we are interested in:



# Gradle build system

---

- *build.gradle(Project:AppName)*
  - contains configuration for all projects and modules in the application.
- *build.gradle(Module:app)* file.
  - contains specific configuration for the module it's included with.
- After any change you make to these files, you will need to *sync* Gradle from the bar that appears, or from the icon on the toolbar.

Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.

[Sync Now](#)



# *build.gradle(Project:Donation)*

---

```
// Top-level build file where you can add configuration options common to all sub-projects/modules.

buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:2.3.1'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        jcenter()
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

# *build.gradle(Module:app)*

---

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 23
    buildToolsVersion "25.0.2"
    defaultConfig {
        applicationId "com.example.donation"
        minSdkVersion 19
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:23.4.0'
    compile 'com.android.support.constraint:constraint-layout:1.0.2'
    testCompile 'junit:junit:4.12'
}
```

# Example: Constraint Layout Install

The screenshot shows the Android Studio interface with the 'Default Settings' dialog open to the 'Android SDK' section. The 'SDK Tools' tab is selected, and the 'Support Repository' section is expanded. The 'ConstraintLayout for Android' and 'Solver for ConstraintLayout' are checked and highlighted in blue. A red arrow points to the 'Apply' button at the bottom right of the dialog.

Name	Version	Status
<input checked="" type="checkbox"/> Android SDK Build-Tools		Installed
<input type="checkbox"/> GPU Debugging tools		Not Installed
<input type="checkbox"/> CMake		Not Installed
<input type="checkbox"/> LLDB		Not Installed
<input type="checkbox"/> Android Auto API Simulators	1	Not installed
<input type="checkbox"/> Android Auto Desktop Head Unit emulator	1.1	Not installed
<input checked="" type="checkbox"/> Android Emulator	26.0.0	Installed
<input checked="" type="checkbox"/> Android SDK Platform-Tools	25.0.5	Installed
<input checked="" type="checkbox"/> Android SDK Tools	26.0.2	Installed
<input checked="" type="checkbox"/> Documentation for Android SDK	1	Installed
<input type="checkbox"/> Google Play APK Expansion library	1	Not installed
<input type="checkbox"/> Google Play Billing Library	5	Not installed
<input type="checkbox"/> Google Play Licensing Library	1	Not installed
<input type="checkbox"/> Google Play services	39	Not installed
<input type="checkbox"/> Google USB Driver	11	Not installed
<input type="checkbox"/> Google Web Driver	2	Not installed
<input checked="" type="checkbox"/> Intel x86 Emulator Accelerator (HAXM installer)	6.0.6	Installed
<input type="checkbox"/> NDK	14.1.3816874	Not installed
<input checked="" type="checkbox"/> Support Repository		
<input checked="" type="checkbox"/> ConstraintLayout for Android		Not Installed
<input checked="" type="checkbox"/> Solver for ConstraintLayout		Not Installed
<input checked="" type="checkbox"/> Android Support Repository	47.0.0	Installed
<input checked="" type="checkbox"/> Google Repository	47	Installed

We need to install this developer tool in Android Studio. To do this, open the SDK manager and select both the **Constraint Layout for Android** and **Solver for ConstraintLayout**. Click the Apply button.



# Example: Constraint Layout Install

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 23
    buildToolsVersion "25.0.2"
    defaultConfig {
        applicationId "com.example.donation"
        minSdkVersion 19
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:23.4.0'
    compile 'com.android.support.constraint:constraint-layout:1.0.2'
    testCompile 'junit:junit:4.12'
}
```

To use ConstraintLayout in the app, we included this dependency.

# So what was created by the wizard?

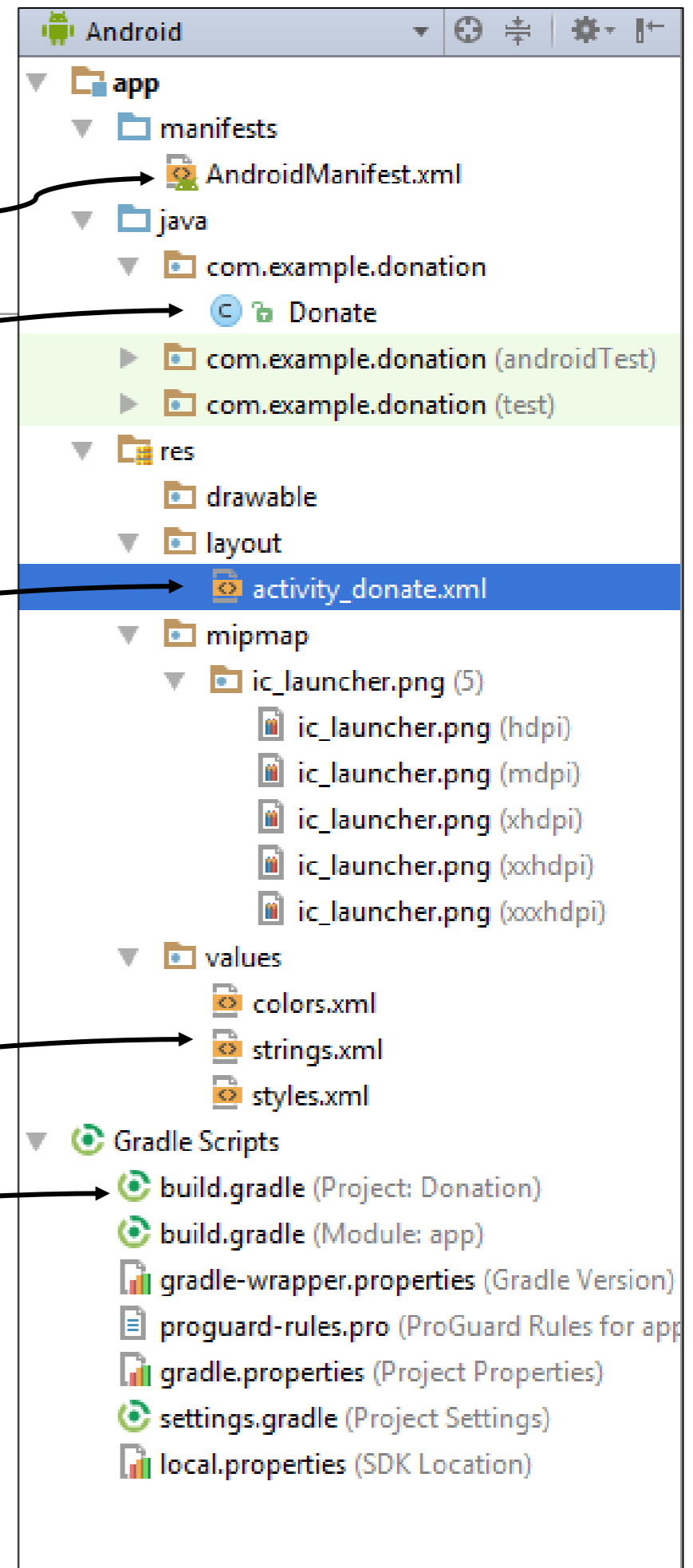
Manifest file

Donate.java (activity class)

activity\_donate.xml (layout)

strings.xml resource file

Gradle build system files



# Activity and Activity Life Cycle

# So what is an Activity?

---

- An activity:
  - represents a single screen with a user interface just like window or frame of Java.
  - is implemented as a class e.g. Java, Kotlin.
  - is responsible for managing user interaction with the layout i.e. XML.
  - subclass the Activity (Fragment Activity, AppCompatActivity, etc.) class based on what role you want the activity to perform.





# onCreate() callback

```
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  
}
```

**onCreate(Bundle)** method is called when an instance of the activity subclass is created.

Donate.java

# onCreate() callback

```
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
}
```

When an activity is created, it needs a user interface to manage. To get the activity its user interface, you call this Activity method.

Donate.java



# onCreate() callback

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

# onCreate() callback

```
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  
}
```

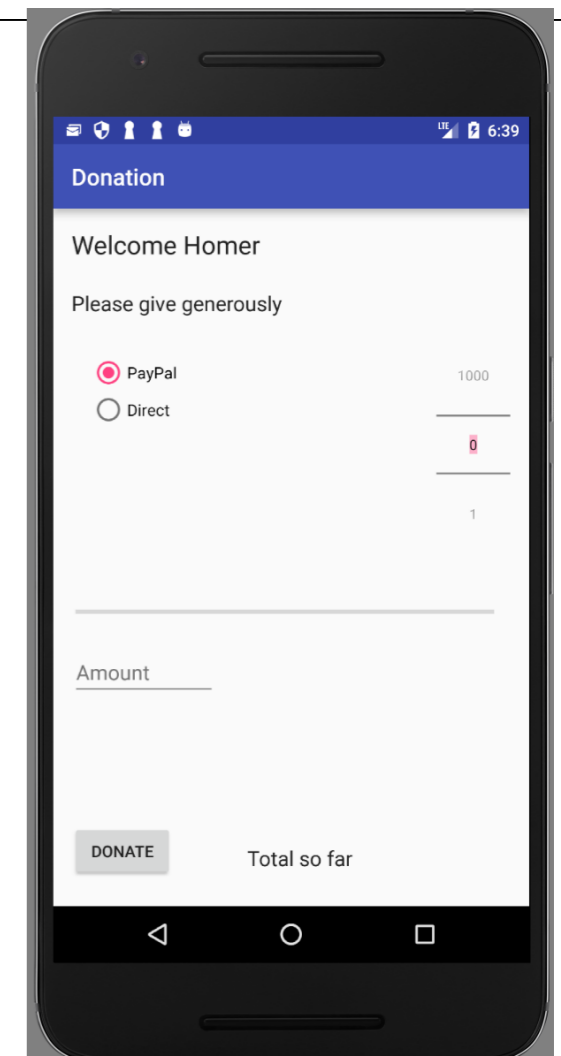
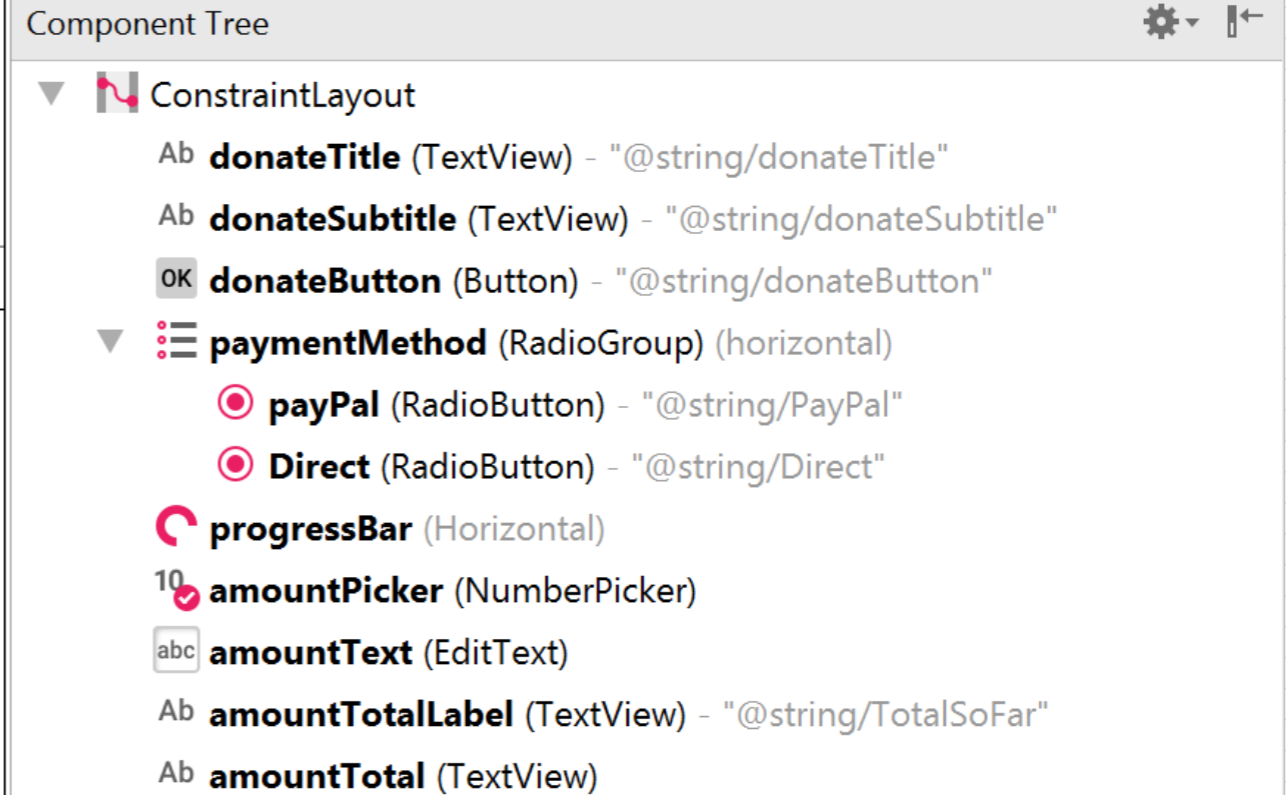
A Bundle is used for passing data between activities. Bundles can hold all types of values and pass them to activities.

Donate.java

# onCreate() callback

```
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



Listeners

# Setting listeners

---

- Android applications are typically event-driven.
- Unlike command-line programs or scripts, event-driven applications start and then wait for an event, such as the user pressing a button.
  - (Events can also be initiated by the OS or another application, but user-initiated events are the most obvious.)
- When your application is waiting for a specific event, we say that it is "listening for" that event.
- The object that you create to respond to an event is called a listener. A listener is an object that implements a listener interface for that event.

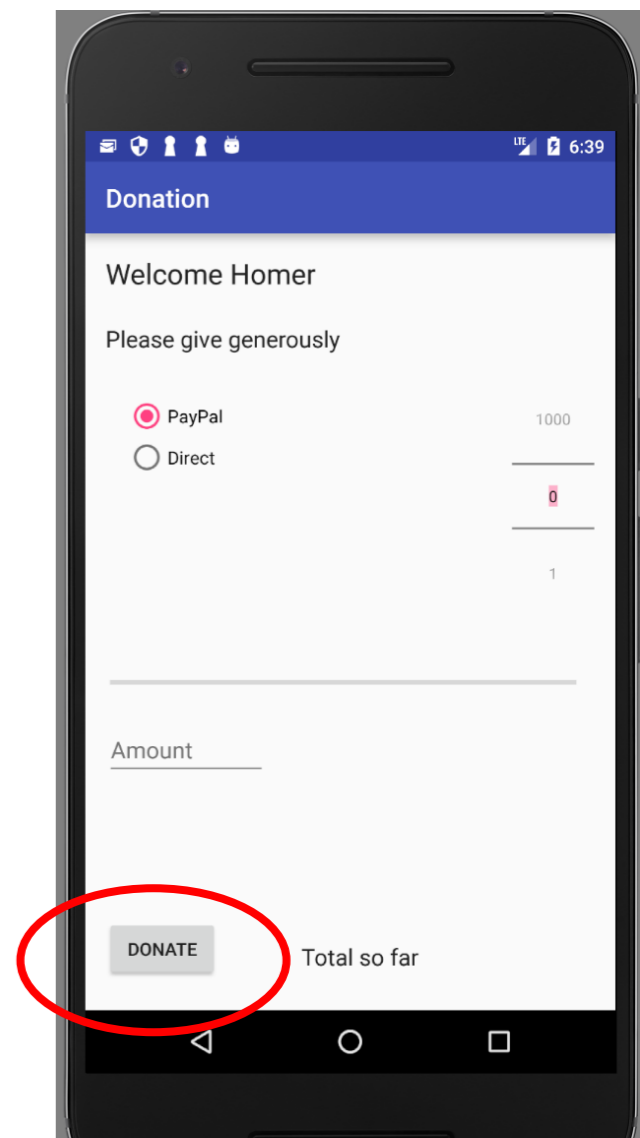
# Setting Listeners - 3 Different Styles

---

- The three styles are:
  1. Explicitly set in Resource File
  2. Using Listener Interface
  3. Using Anonymous Inner Class
- We need to master all three; but we will cover just the first one this week!

# Listeners: explicitly set in Resource File

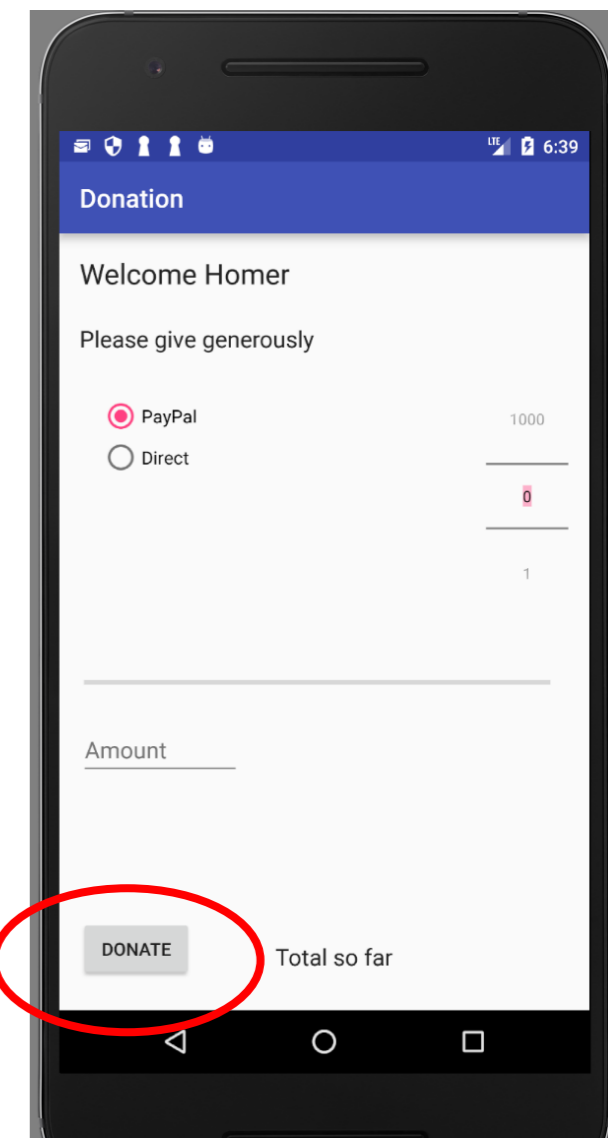
---



# Listeners: explicitly set in Resource File

activity\_donate.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" />
```





# Listeners: explicitly set in Resource File

activity\_donate.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" />
```

```
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
    }

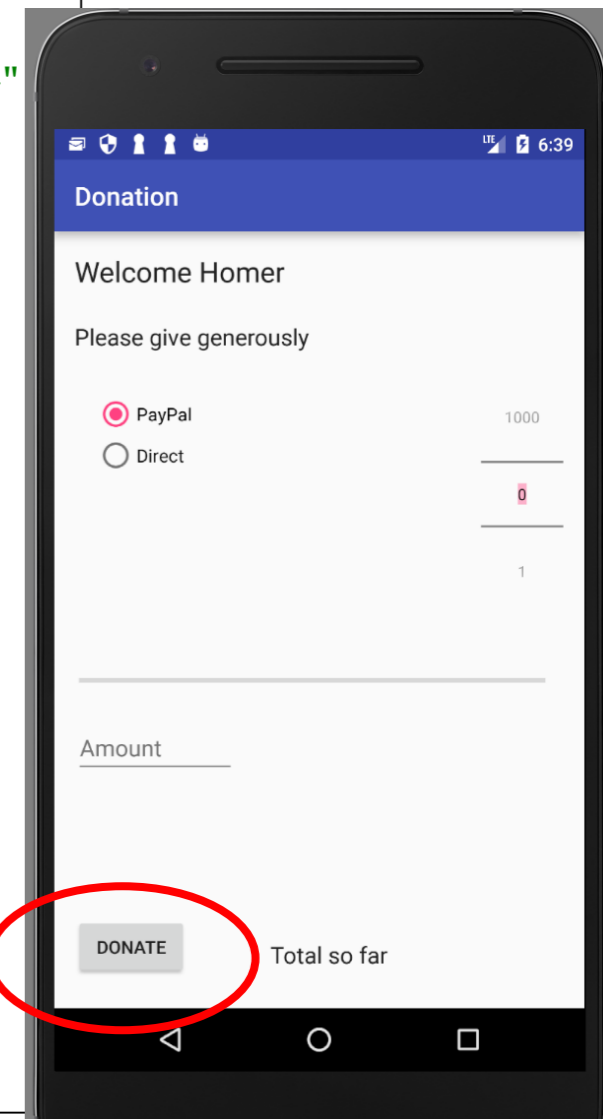
    public void donateButtonPressed (View view){
        String method = paymentMethod.getCheckedRadioButtonId() == R.id.payPal ? "PayPal" : "Direct"

        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



Toast

# Toasts

Donate.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);
    }
}
```

A toast is a short message that informs the user of something...but it does not require any input or action.

# Making Toasts

---

Donate.java

```
if (totalDonated > target) {  
    Toast toast = Toast.makeText(this, "Target Exceeded!", Toast.LENGTH_SHORT);  
    toast.show();  
    Log.v("Donate", "Target Exceeded: " + totalDonated);  
}
```

To create a toast, you call the following method from the Toast class:

```
public static Toast makeText(Context context, int resId, int duration)
```

- context:** typically an instance of Activity (Activity is a subclass of Context).
- resId:** the resource ID of the string that the toast should display. The Context is needed by the Toast class to be able to find and use the string's resource ID.
- duration:** usually one of two Toast constants that specify how long the toast should be visible.

# Displaying Toasts

---

Donate.java

```
if (totalDonated > target) {  
    Toast toast = Toast.makeText(this, "Target Exceeded!", Toast.LENGTH_SHORT);  
    toast.show();  
    Log.v("Donate", "Target Exceeded: " + totalDonated);  
}
```

After you have created a toast, you call:

`Toast.show()`

on it to display it on the screen.

# Log and Logcat

# Log

## Donate.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);
    }
}
```

Log enables messages to be written to the log. Log.v writes a VERBOSE log message.

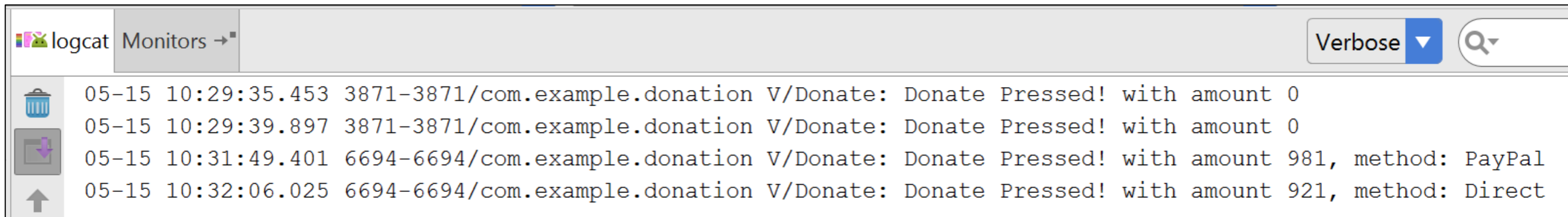
# Logcat

## Donate.java

```
if (totalDonated > target) {  
    Toast toast = Toast.makeText(this, "Target Exceeded!", Toast.LENGTH_SHORT);  
    toast.show();  
    Log.v("Donate", "Target Exceeded: " + totalDonated);  
}
```

“The Logcat Monitor displays system messages as well as messages you can add using the Log class. It displays messages in real time and also keeps a history so you can view older messages”

<https://developer.android.com/studio/debug/am-logcat.html>



The screenshot shows the Logcat window in Android Studio. The window title is "logcat Monitors →". On the right side, there is a "Verbose" filter dropdown and a search icon. The log output shows four messages:

```
05-15 10:29:35.453 3871-3871/com.example.donation V/Donate: Donate Pressed! with amount 0  
05-15 10:29:39.897 3871-3871/com.example.donation V/Donate: Donate Pressed! with amount 0  
05-15 10:31:49.401 6694-6694/com.example.donation V/Donate: Donate Pressed! with amount 981, method: PayPal  
05-15 10:32:06.025 6694-6694/com.example.donation V/Donate: Donate Pressed! with amount 921, method: Direct
```



Models

# Model?

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.NumberPicker;
import android.widget.ProgressBar;
import android.widget.RadioGroup;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import java.util.Locale;

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);

        //code omitted
    }

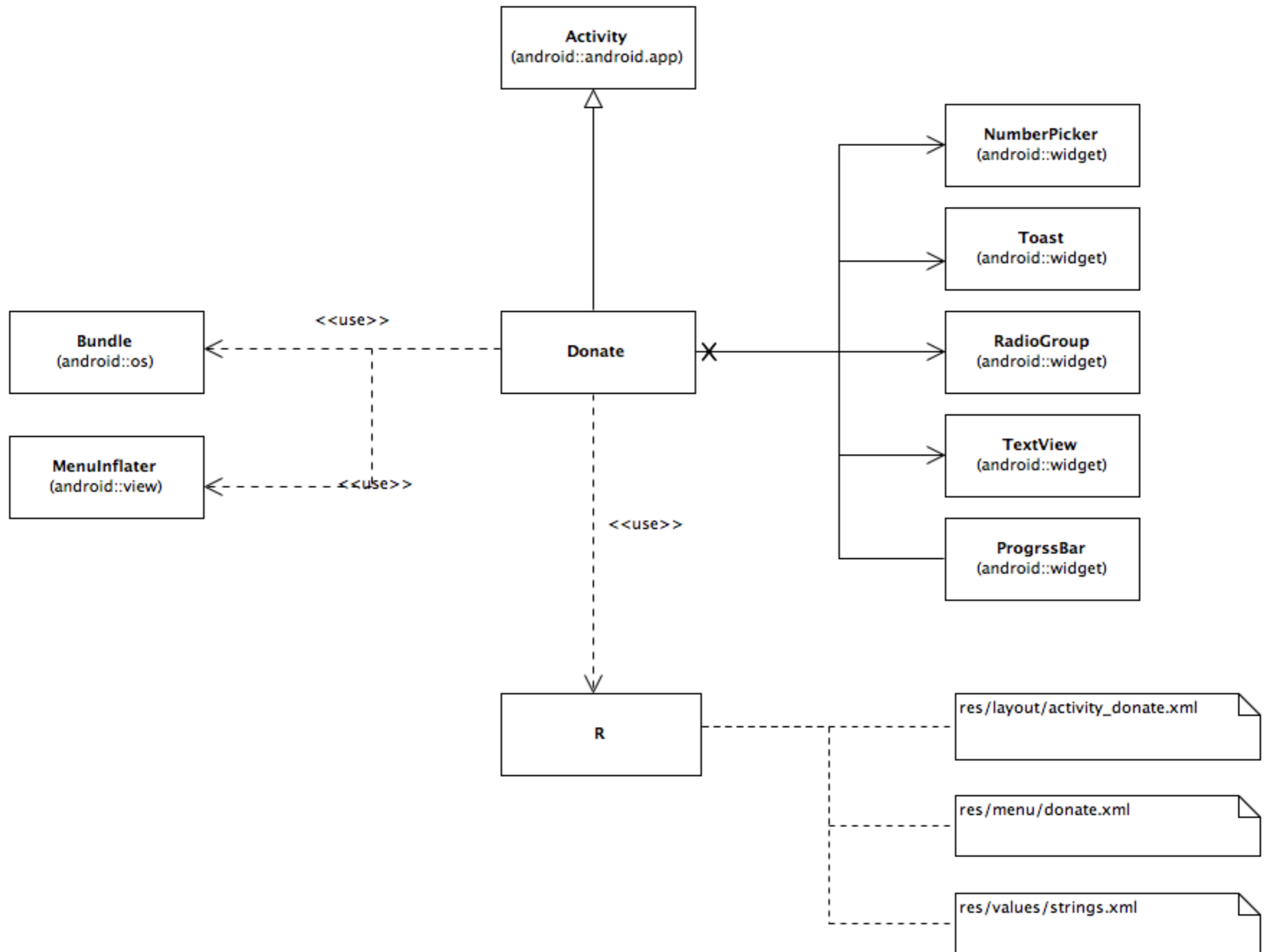
    public void donateButtonPressed (View view) {
        String method = paymentMethod.getCheckedRadioButtonId() == R.id.payPal ? "PayPal" : "Direct";

        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);
    }
}
```

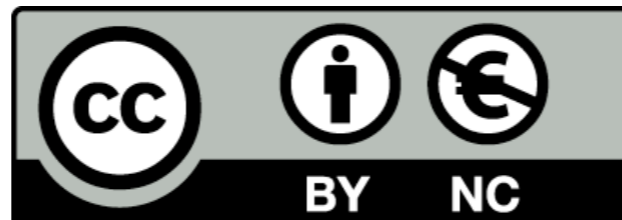
- Only a single class, so model not particularly useful.
- However, the Donate class interacts with at least 8 android framework classes.



# Questions?

---





Except where otherwise noted, this content is licensed under a [Creative Commons Attribution-NonCommercial 3.0 License](http://creativecommons.org/licenses/by-nc/3.0/).

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

