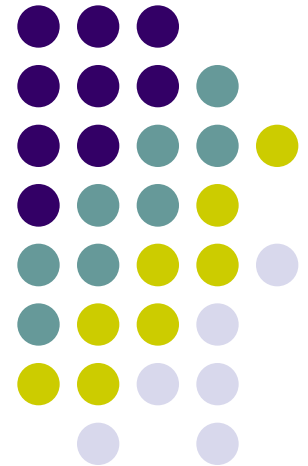


Mobile Application Development

Android Activities



Top-down design

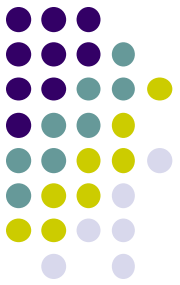


Let's start from a design of an app that we want to create and then learn the necessary skills to build that app.


- "Bigger Number" game
 - user is shown two numbers
 - must choose which one is bigger by clicking on the appropriate button
 - game pops up brief "correct" / "incorrect" message after each guess
 - get points for each correct answer (lose points for incorrect answers)



Creating a new project



Create New Project

 **New Project**
Android Studio

Configure your new project

Application name:

Company Domain:

Package name: [Edit](#)

Project location:

Designing a user interface



open XML file for your layout (e.g. activity_main.xml)

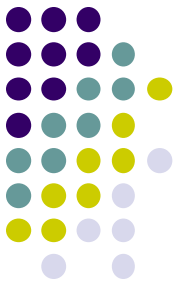
- drag widgets from left **Palette** to the preview image
- set their properties in lower-right **Properties panel**

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

- Palette:** A list of widgets including Layouts (FrameLayout, LinearLayout, etc.) and Widgets (TextView, Button, etc.).
- Preview:** A smartphone displaying the game interface with two buttons labeled '0', a score of 'Points: 0', and a title 'Bigger Number Game!'. A green dashed line indicates a vertical alignment constraint.
- Component Tree:** A tree view showing the hierarchy of UI components: Device Screen > RelativeLayout > number1 (Button) - "0", number2 (Button) - "0", score (TextView) - "Points: 0", textView2 - "Bigger Number Game!", and textView3 - "Press the button of th..".
- Properties:** A table showing the properties of the selected widget (textView2):

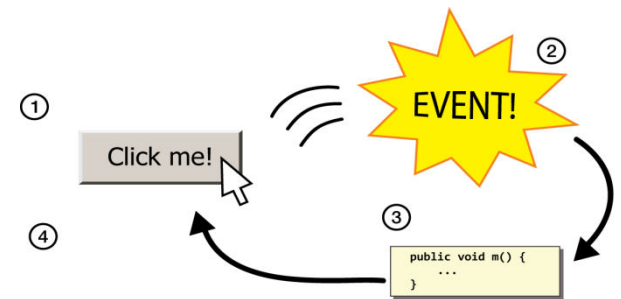
Property	Value
singleLine	<input type="checkbox"/>
stateListAnimator	
text	Bigger Number Game!
textAlignment	
textAppearance	?android:attr/textAppea
textColor	
textColorHighlight	
textColorHint	
textColorLink	

Events



- **event: An external stimulus your program can respond to.**

- Common kinds of events include:
 - Mouse motion / tapping, Keys pressed,
 - Timers expiring, Network data available



- **event-driven programming: Overall**

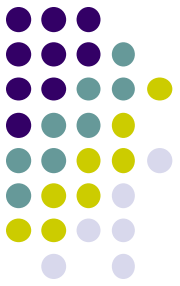
execution of your program is largely dictated by user events.

- Commonly used in graphical programs.

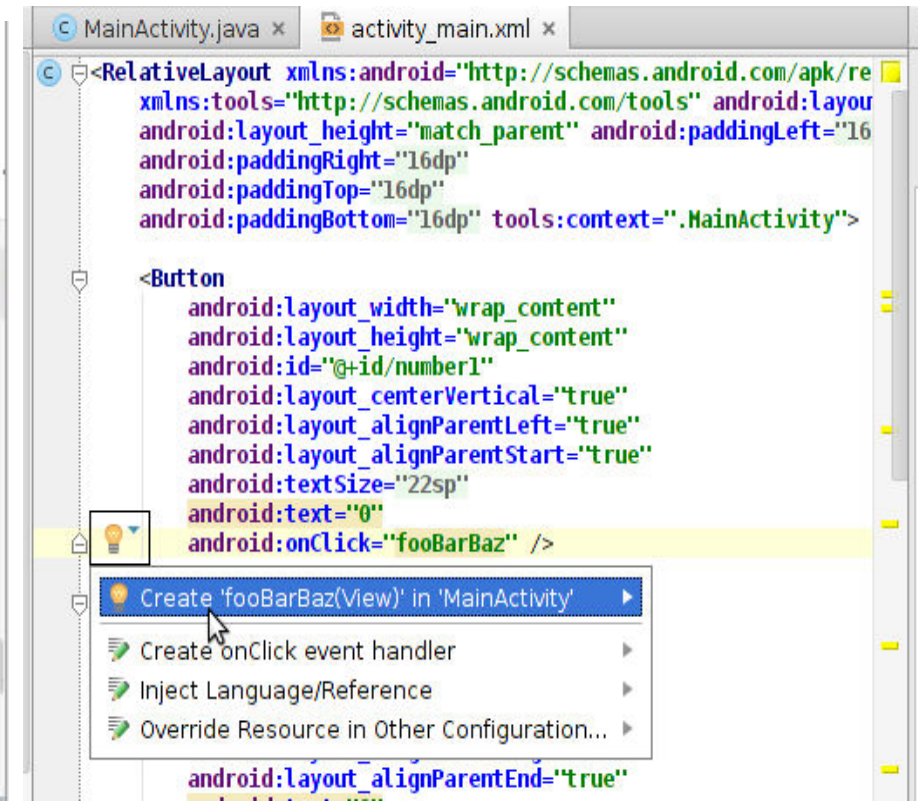
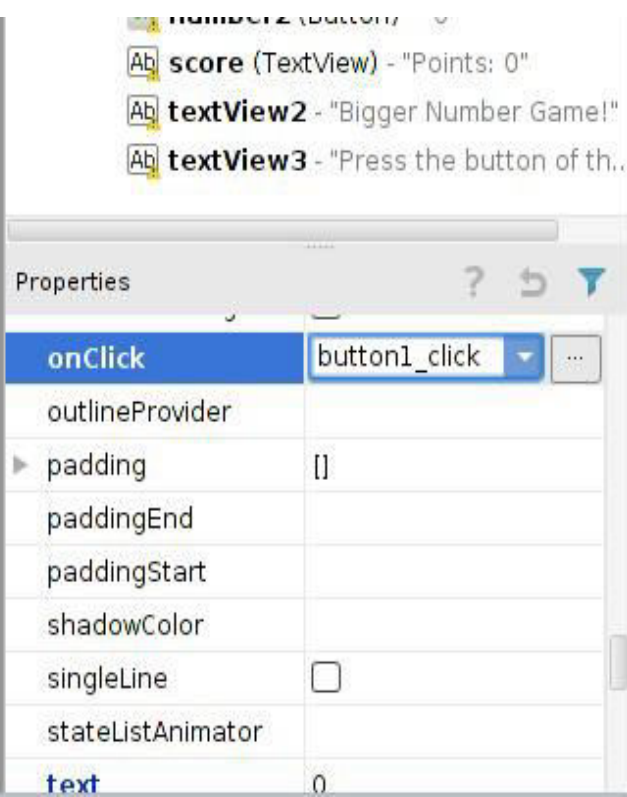
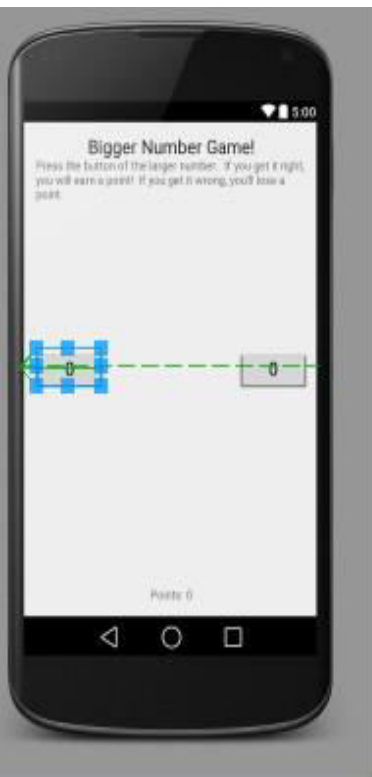
- To respond to events in a program, you must:

- Write methods to handle each kind of event ("listener" methods).
- Attach those methods to particular GUI widgets.

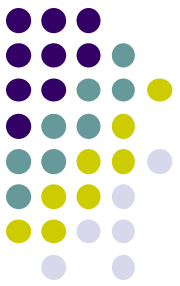
Setting an event listener



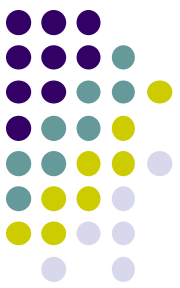
- select the widget in the Design view
- scroll down its Properties until you find onClick
- type the name of a method you'll write to handle the click
- switch to the Text view and find the XML for that button
- click the "Light Bulb" and choose to "Create" the method



Event listener Java code

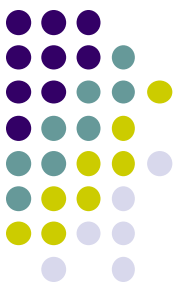


```
MainActivity.java x activity_main.xml x
1 package com.example.stepp.numbergame;
2
3 import ...
8
9 public class MainActivity extends ActionBarActivity {
10     @Override
11     protected void onCreate(Bundle savedInstanceState) {
12         setContentView(R.layout.activity_main);
13         super.onCreate(savedInstanceState);
14     }
15
16     public void button1_click(View view) {
17         // your code goes here
18
19     }
```



AndroidManifest.xml

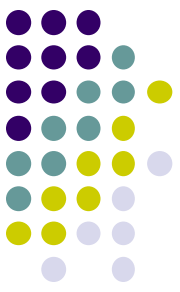
```
<?xml version="1.0" encoding="utf-8" ?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.stepp.numberguessinggame">
    <application android:allowBackup="true" android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" android:theme="@style/AppTheme">
    <activity android:name=".MainActivity" android:label="@string/app_name">
    <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    </application>
</manifest>
```

activity_main.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:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">
```

```
<TextView android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:text="Number Guessing Game!"
    android:id="@+id/textView" android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true" android:textSize="30dp" />
```



activity_main.xml *continues*

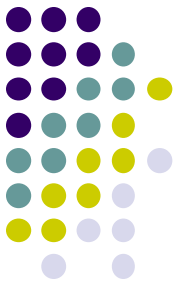
```
<TextView android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:text=" Click the number that is
bigger than the other number. Even you can do this."
    android:id="@+id/textView2" android:layout_below="@+id/textView"
    android:layout_centerHorizontal="true" />

<Button android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:text="0"
    android:id="@+id/buttonLeft" android:layout_centerVertical="true"
    android:layout_alignParentLeft="true" android:layout_alignParentStart="true"
    android:textSize="40dp" android:onClick="clickButton1" />

<Button android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:text="0"
    android:id="@+id/buttonRight" android:layout_centerVertical="true"
    android:layout_alignParentRight="true" android:layout_alignParentEnd="true"
    android:textSize="40dp" android:onClick="clickButton2" />

<TextView android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:text="Points: 0"
    android:id="@+id/pointsTextView" android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true" />

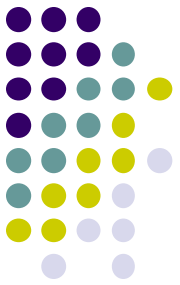
</RelativeLayout>
```



MainActivity.java

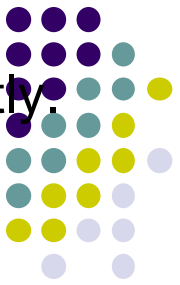
```
import android.app.*;
import android.support.v7.app.*;
import android.os.*;
import android.view.*;
import android.widget.*;
import java.util.*;

public class MainActivity extends Activity {
    private int num1; // the numbers on the left and right buttons
    private int num2;
    private int points; // player's point total; initially 0
```



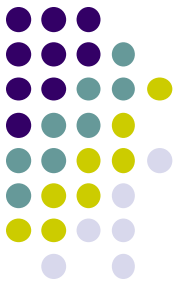
```
/*  
 * Called when the player clicks the left number button.  
 */  
public void clickButton1(View view) {  
    check(num1, num2);  
}
```

```
/*  
 * Called when the player clicks the right number button.  
 */  
public void clickButton2(View view) {  
    check(num2, num1);  
}
```



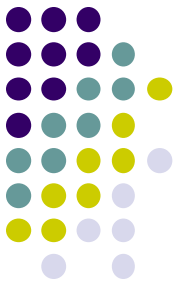
```
/*  
 * Updates the player's score based on whether they guessed correctly.  
 * Also shows a 'toast' which is a brief popup message.  
 */
```

```
private void check(int a, int b) {  
    if (a > b) {  
        points++;  
        Toast.makeText(this, "Correct!",  
                        Toast.LENGTH_SHORT).show();  
    } else {  
        points--;  
        Toast.makeText(this, "You are Wrong.",  
                        Toast.LENGTH_SHORT).show();  
    }  
  
    TextView pointsView = (TextView) findViewById(R.id.pointsTextView);  
    pointsView.setText("Points: " + points);  
    roll();  
}
```



```
/*  
* Chooses new random integers to appear on the two buttons.  
*/
```

```
private void roll() {  
    // pick two random numbers  
    Random r = new Random();  
    num1 = r.nextInt(9);  
    num2 = r.nextInt(9);  
    while (num2 == num1) {  
        num2 = r.nextInt(9);  
    }  
    // set the buttons to display the random numbers  
    Button left = (Button) findViewById(R.id.buttonLeft);  
    left.setText("" + num1);    // "" + int -> converts int to String  
  
    Button right = (Button) findViewById(R.id.buttonRight);  
    right.setText("" + num2);  
}
```



```
/******
```

```
* BELOW THIS POINT IS CODE THAT WAS GENERATED BY  
ANDROID STUDIO THAT WE *
```

```
* DID NOT MODIFY, EXCEPT FOR ONE LINE THAT IS MARKED  
BELOW. *
```

```
*****/
```

```
/*
```

```
* This method is called by Android when our activity is first created.
```

```
*/
```

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState) {
```

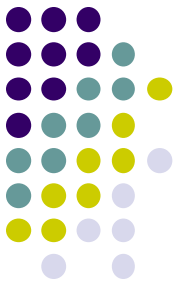
```
    super.onCreate(savedInstanceState);
```

```
    setContentView(R.layout.activity_main);
```

```
    roll(); // <-- we added this line to set initial button random numbers
```

```
}
```

```
}
```



Displaying Toasts

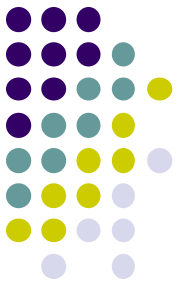
```
Toast.makeText(this, "message", duration).show();
```

– where *duration* is ***Toast.LENGTH_SHORT*** or ***Toast.LENGTH_LONG***

- A "Toast" is a pop-up message that appears for a short time.
- Useful for displaying short updates in response to events.
- Should not be relied upon extensively for important info.

This is the Toast message

References



- Activity class
 - <https://developer.android.com/reference/android/app/Activity.html#Activity>