UNIVERSITY OF
PLYMOUTH

# COMP2000: Software engineering 2

## Android Development – part-III

# Outline

- Menus
- Messages
- Resources (Layouts, drawable and values)

# Menus

- Option menus: The options menu is the primary collection of menu items for an activity.
- Contextual Menus: A context menu is a floating menu that appears when the user performs a long-click on an element
- Popup Menu: A popup menu displays a list of items in a vertical list that's anchored to the view that invoked the menu Creating a Popup Menu..

# Create a menu

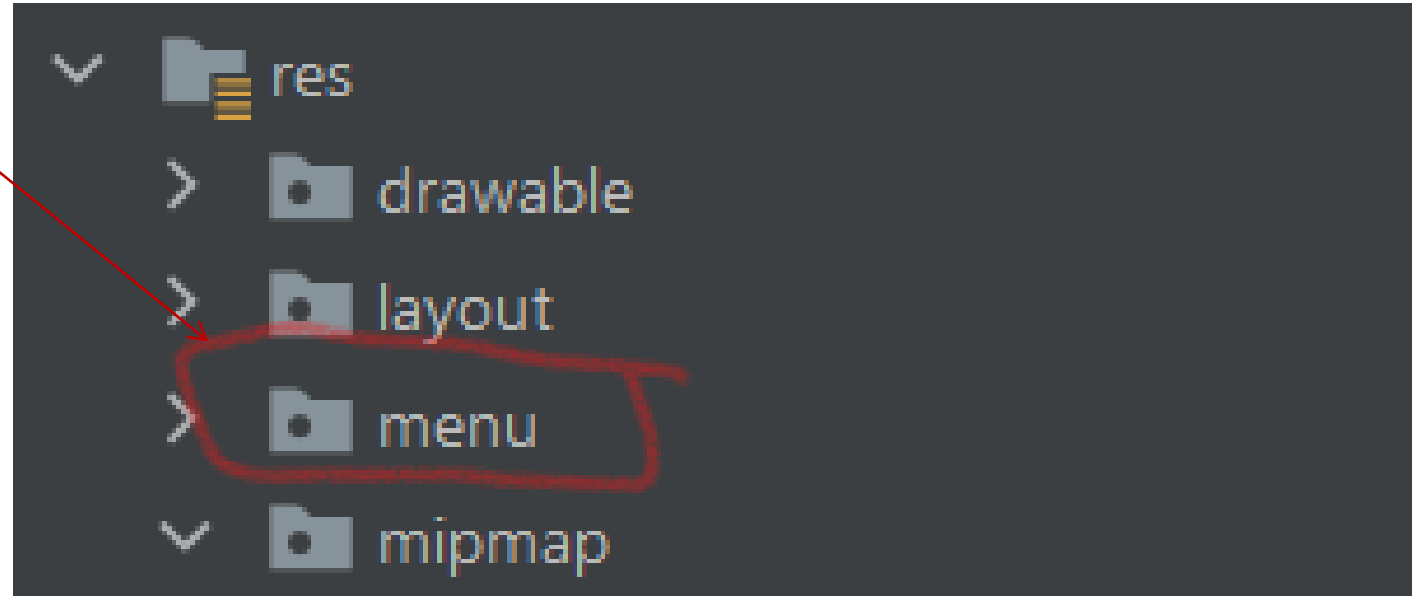
By default, every Activity supports an options menu of actions or options. You can add items to this menu and handle clicks on your additions.

The easiest way of adding menu items is inflating an XML file into the Menu via MenuInflater.

The easiest way of attaching code to clicks is via Activity#onOptionsItemSelected(MenuItem) and Activity#onContextItemSelected(MenuItem).

# Defining a Menu in XML

- To define the menu, create an XML file inside your project's res/menu/ directory and build the menu.

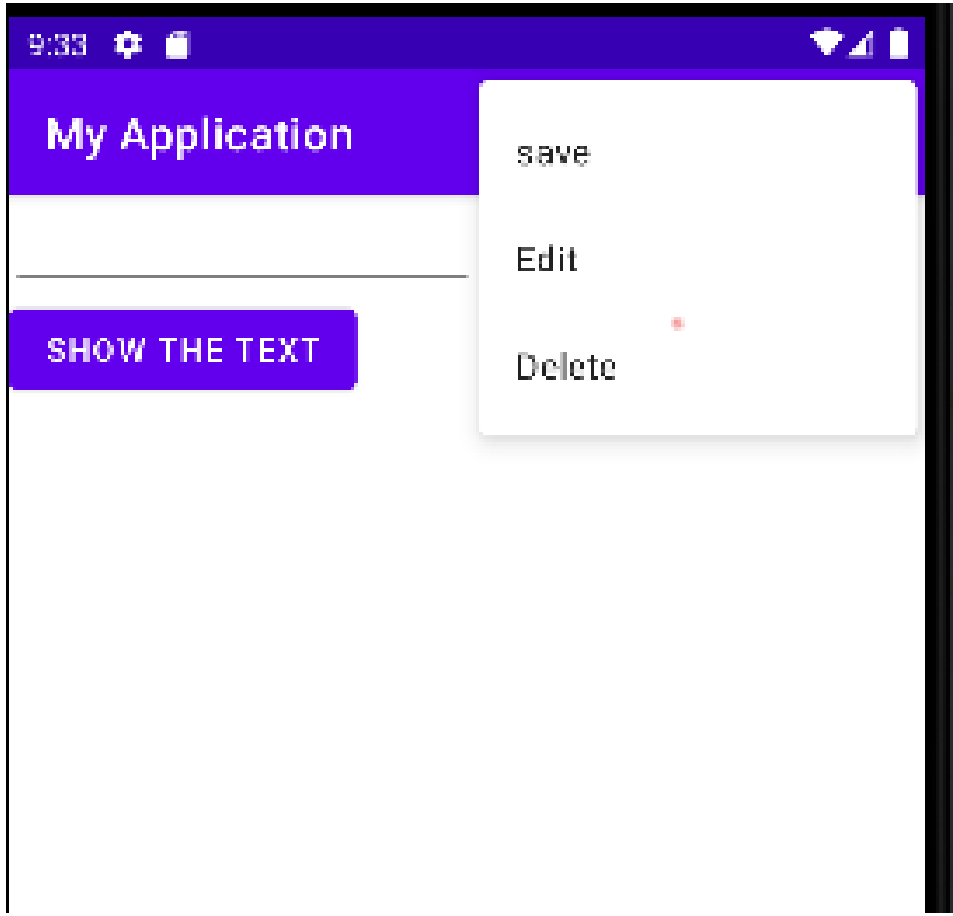


# Example:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:title="save"
      android:id="@+id/savech"/>
   <item android:title="Edit"
      android:id="@+id/editch"/>
   <item android:title="Delete"
      android:id="@+id/deletech"/>
</menu>
```

# Option Menu


Option menu



# Java file

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

# Add Items

```
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    // Handle item selection
    switch (item.getItemId()) {
        case R.id.savech:
            saveItem();
            return true;
        case R.id.editch:
            editItem();
            return true;
        case R.id.deletech:
            deletItem();
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

# Messages

- **Toast message:**
- A toast provides simple feedback about an operation in a small popup.
- It only fills the amount of space required for the message and the current activity remains visible and interactive.
- Toasts automatically disappear after a timeout.

```
Context context = getApplicationContext();
CharSequence text = “This login Button”
int duration = Toast.LENGTH_SHORT;

Toast toast = Toast.makeText(context, text, duration);
toast.show();
```

# Snackbar

You can use a Snackbar to display a brief message to the user.

Snackbars include user-actionable options, which can provide a better app experience.
For example, an email app could use a Snackbar to tell the user that the app successfully sent an email.

```
Snackbar.make(findViewById(R.id.main_layout), "Email sent successfully",
        Snackbar.LENGTH_SHORT)
        .show();
```

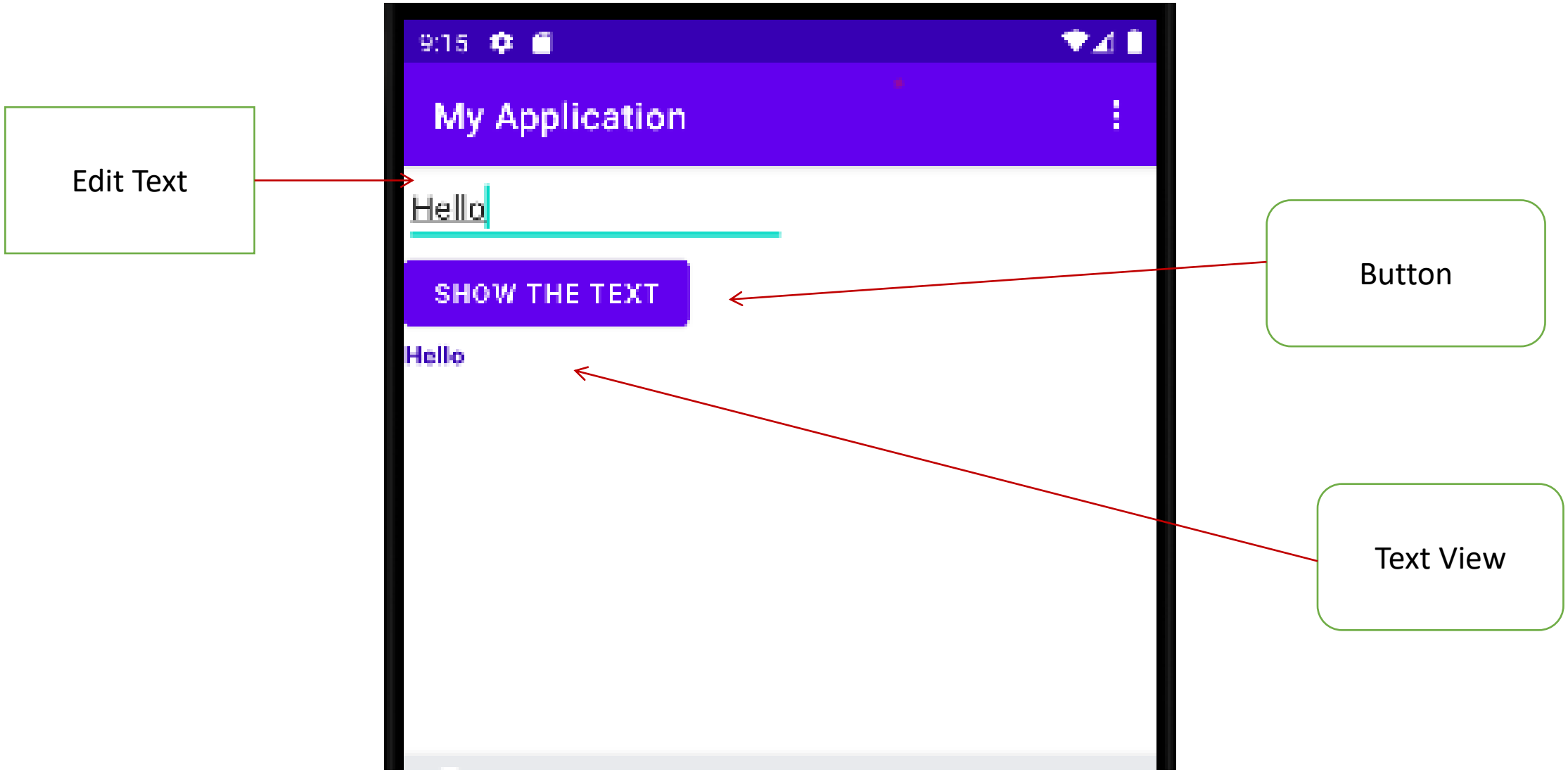
# How to Deal with inputs from UIs

## Examples

```
TextView text= (TextView) findViewById(R.id.typeName);

EditText editText= (EditText) findViewById(R.id.eName);

text.setText(editText.getText().toString());
```


Edit Text
9:15
My Application
Hello
SHOW THE TEXT
Hello
Button
Text View

- **Layout_constraint** Attribute:
- Bottom, Top, Left, etc.

app:layout_constraintBottom_toTopOf="@+id/notifyBtn“

app:layout_constraintTop_toBottomOf="@id/eName“

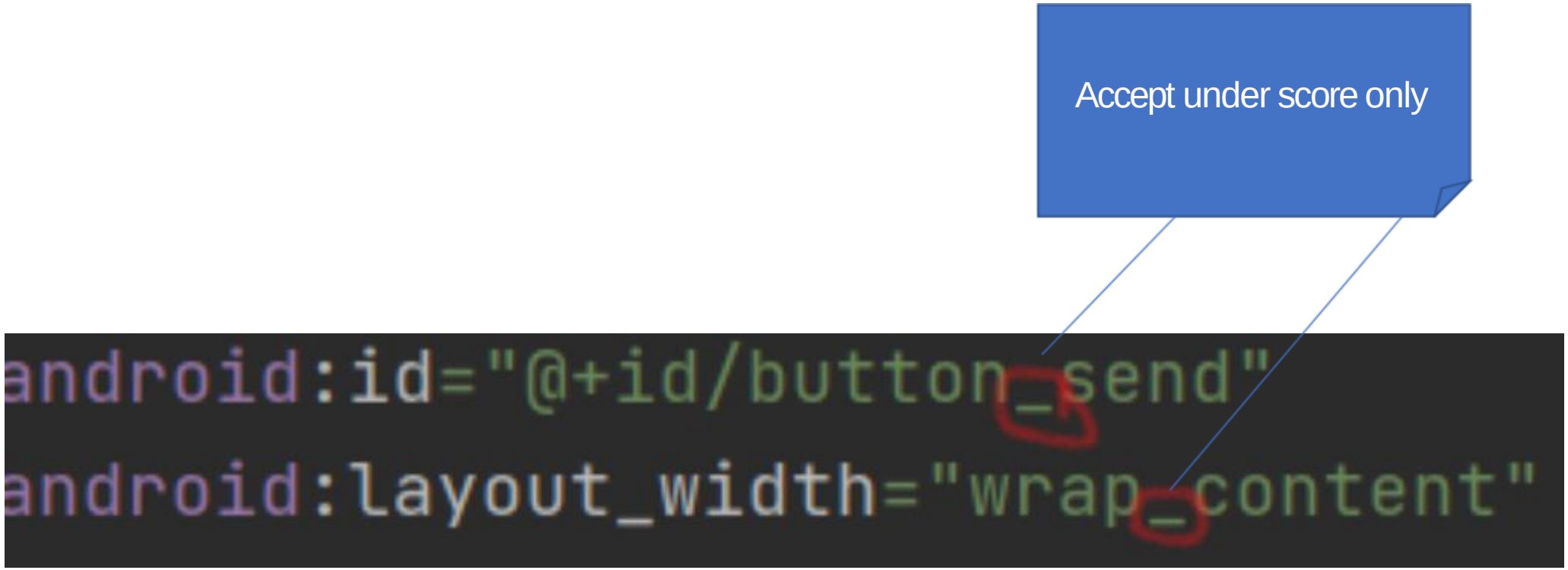
app:layout_constraintTop_toBottomOf="@id/notifyBtn"

# Layouts’ attributes

- Id
- Layouts_width
- Layouts_Height
- layout_marginBottom
- layout_marginTop
- Gravity
- textStyle
- textColor

-

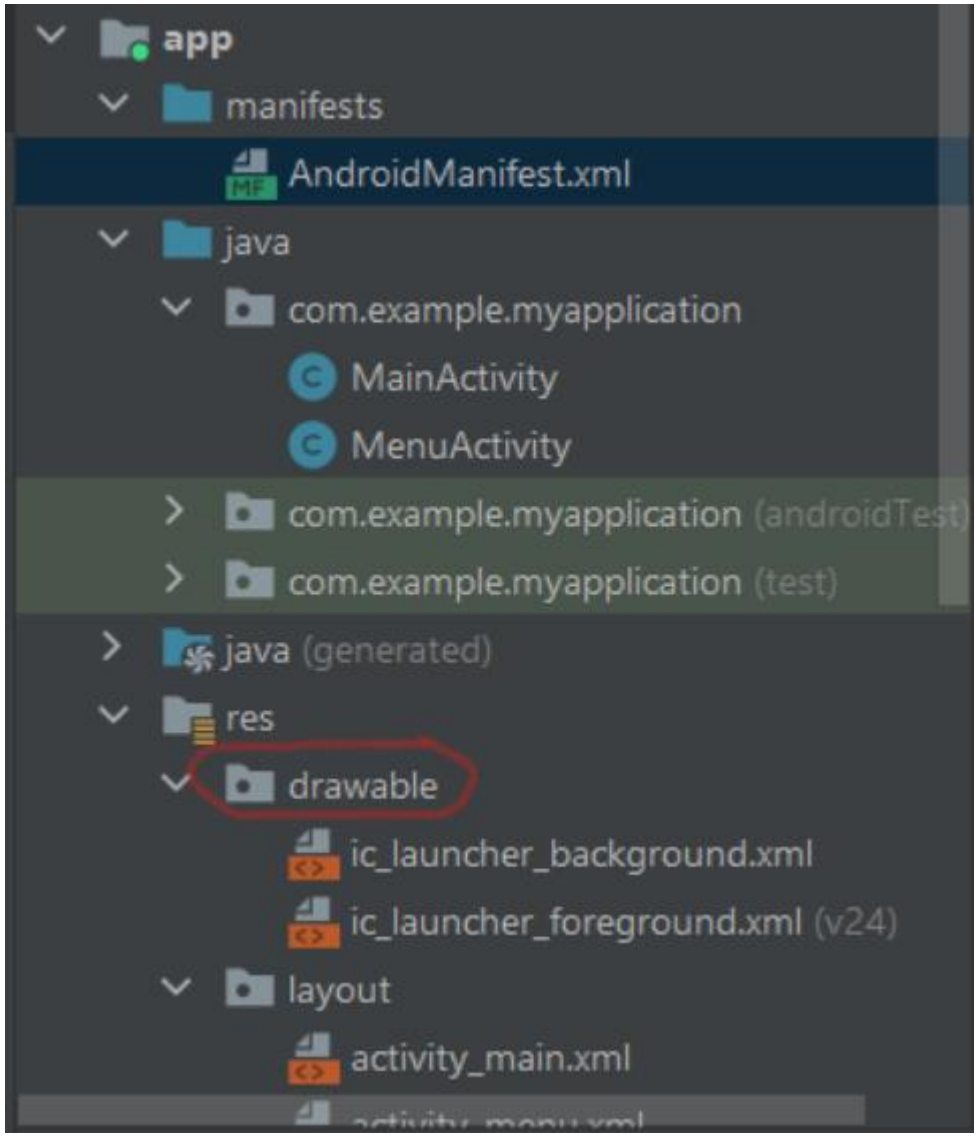
```
<Button xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/button_send"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="0.dp"
    android:layout_marginTop="1pt"
    android:text="button_send"
    android:onClick="implementMenu"
    android:gravity="center"
    android:textStyle="italic"
    android:textColor="@color/yellow"/>
```


Accept under score only
android:id="@+id/button_send"
android:layout_width="wrap_content"

# Drawable resources

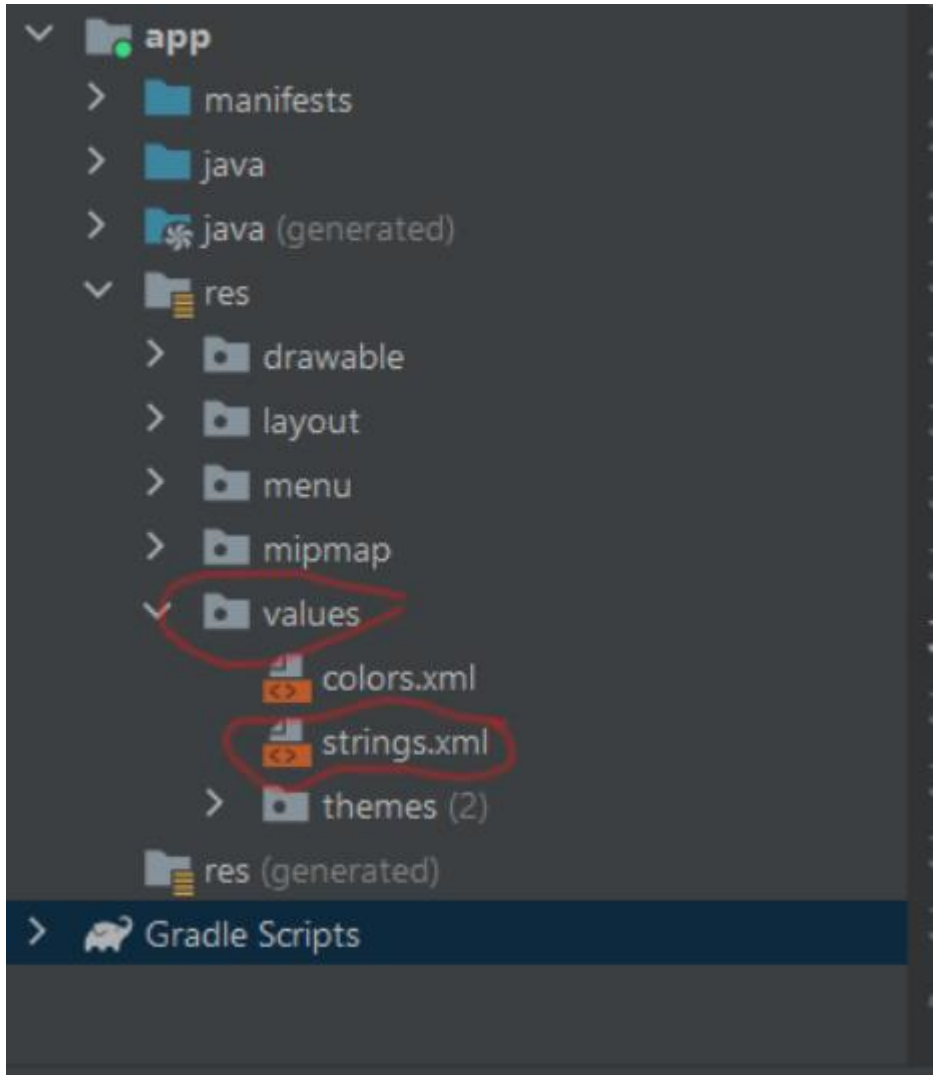
A drawable resource is a general concept for a graphic that can be drawn to the screen and which you can retrieve with APIs such as getDrawable(int) or apply to another XML resource with attributes such as android:drawable and android:icon.

```
<ImageView
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:src="@drawable/myimage" />
```



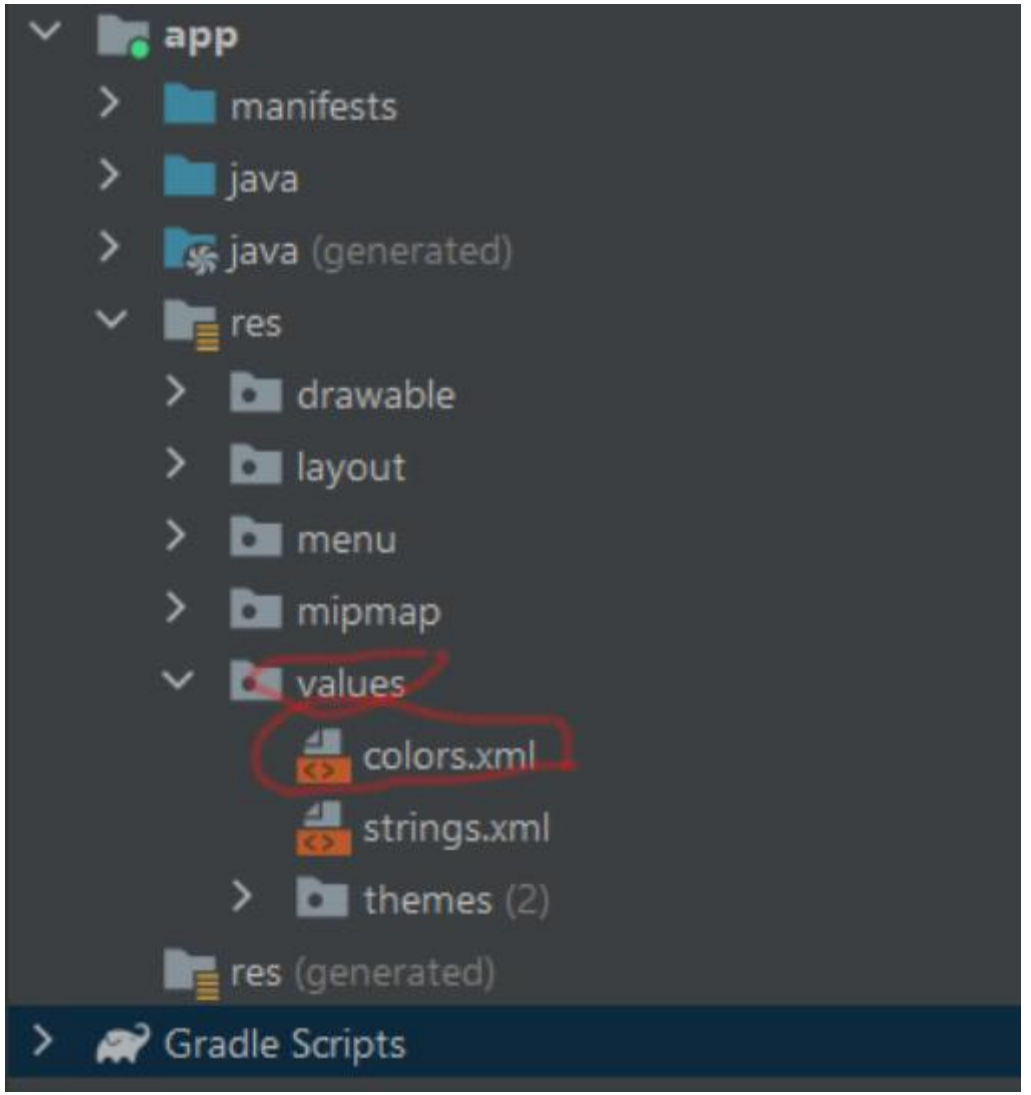
# String Resources

```
<resources>
    <string name="app_name">My Application</string>
</resources>
```



# Colour resources

```
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="yellow">#FFFF00</color>
</resources>
```




- https://www.codexpedia.com/android/list-of-color-names-and-color-code-for-android/

# onClick attribute

```
<Button
    android:id="@+id/butn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Show a message"
    android:onClick= "showAmessage"
     />
```

# Java code

Should pass a View object

```
public void showAmessage(View view){
    Context context = getApplicationContext();
    String text = "This is a show message Button";
    int duration = Toast.LENGTH_SHORT;

    Toast toast = Toast.makeText(context, text, duration);
    toast.show();


}
```

# Messages

- Toast message

```
Context context = getApplicationContext();
CharSequence text = “This login Button”
int duration = Toast.LENGTH_SHORT;

Toast toast = Toast.makeText(context, text, duration);
toast.show();
```

# Snackbar

You can use a Snackbar to display a brief message to the user. For example, an email app could use a Snackbar to tell the user that the app successfully sent an email.

```
Snackbar.make(findViewById(R.id.main_layout), "Email sent successfully",
        Snackbar.LENGTH_SHORT)
        .show();
```

# Canvas class

Canvas is a class in Android that performs 2D drawing of different objects onto the screen.

It is basically, an empty space to draw onto.

To draw a circle onto the view,
give it a center point x,y , its size and a paint object:

```
canvas.drawCircle(x, y, size, paint)
```

```
canvas.drawRect(rect, paint)
```

# Canvas

```
public class CircleView extends View{

    public CircleView(Context context) {
        super(context);
    }
```

```
protected void onDraw(Canvas canvas){
    super.onDraw(canvas);
    Paint paint = new Paint();
    paint.setColor(150);
    canvas.drawCircle(50,50,20,paint);


}
```

# ListView

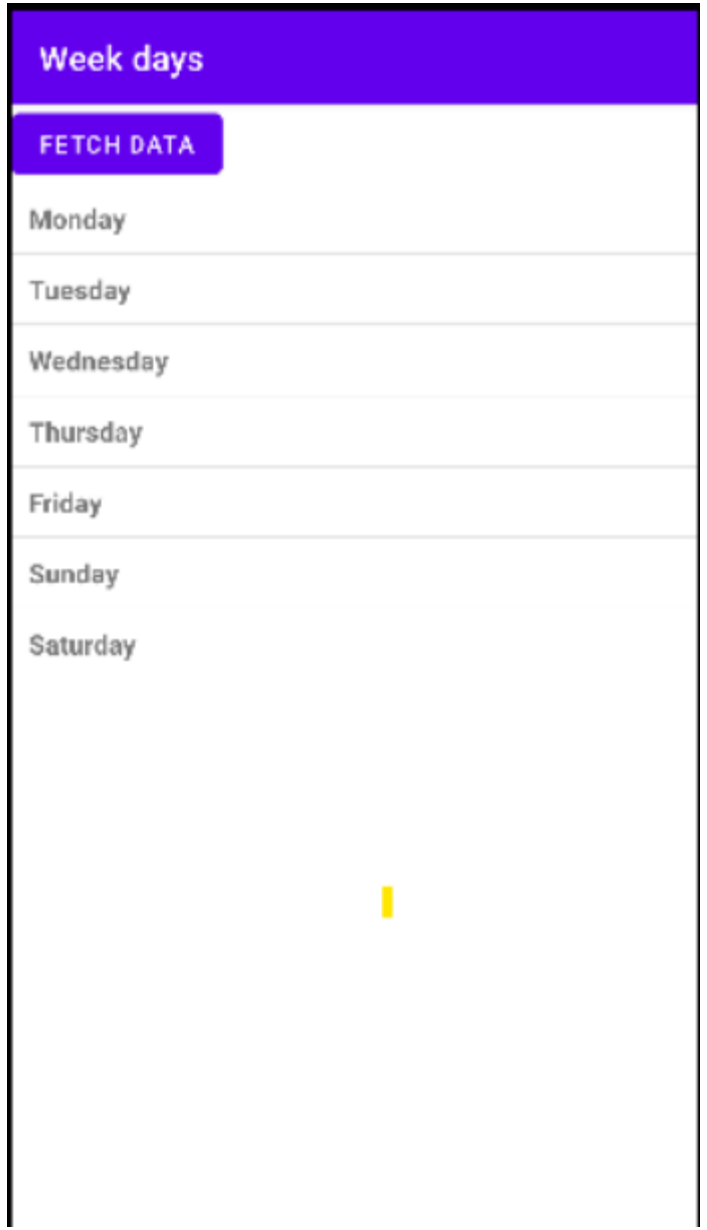
- Android **ListView** is a view which groups several items and display them in vertical scrollable list.
- The list items are automatically inserted to the list using an **Adapter** that pulls content from a source such as an array or database.

## activity_main.xml

```
<ListView
    android:id="@+id/mobile_list"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:layout_constraintTop_toBottomOf="@+id/button"
    app:layout_constraintBaseline_toBottomOf="parent">
</ListView>
```

## List_view.xml

```
<!-- Single List Item Design -->
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/label"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:padding="10dip"
    android:textSize="16dip"
    android:textStyle="bold" >
</TextView>
```



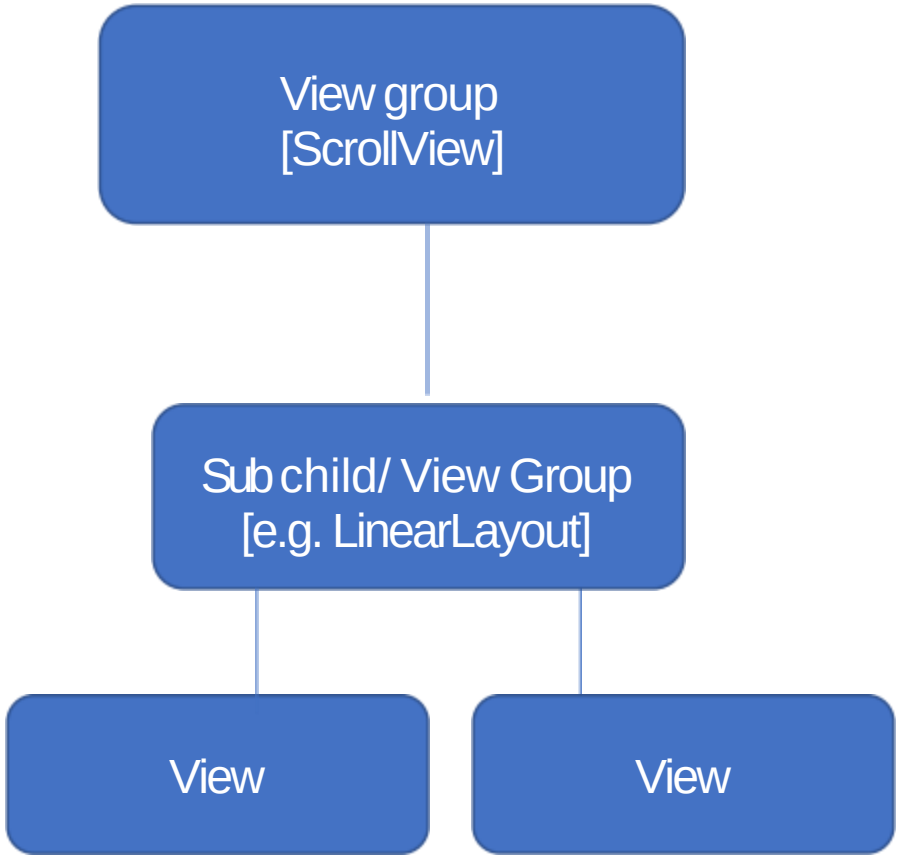
MainActivity.java

```
ArrayAdapter adapter = new ArrayAdapter<String>(this,
        R.layout.list_view, weekArray);

ListView listView = (ListView) findViewById(R.id.mobile_list);
listView.setAdapter(adapter);
```

# ScrollView

A ScrollView is a view group that is used to make vertically scrollable views.



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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <!-- things to scroll -->

    </LinearLayout>
</ScrollView>
```

- Thank you