Lab 2 – Activity & Layout

KUAN-TING LAI

2022/9/19

Concept of Activities

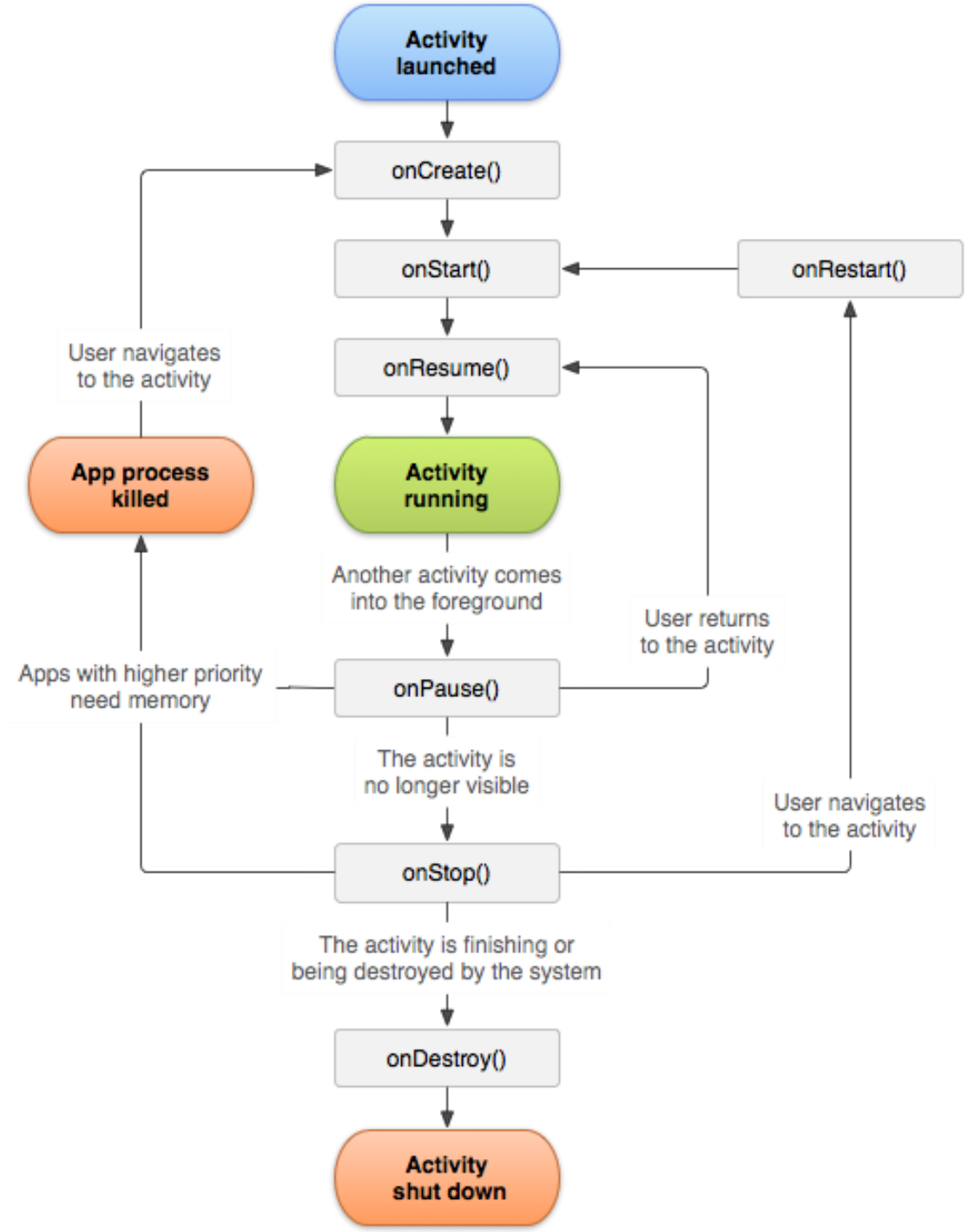
- Activity shows the UI components
- One activity, one window (screen)
- Enables one app to invoke another app
- Use **Intent** to communicate
- An activity can contain multiple fragments

Ref: https://developer.android.com/guide/components/activities/intro-activities#java

Declare Activity

```
<manifest ... >
   <application ... >
       <activity
           android:name=".MainActivity"
           android:exported="true">
           <intent-filter>
               <action android:name="android.intent.action.MAIN" />
               <category android:name="android.intent.category.LAUNCHER" />
           </intent-filter>
       </activity>
       ...
  </application>
    ...
</manifest>
```

Activity Life Cycle



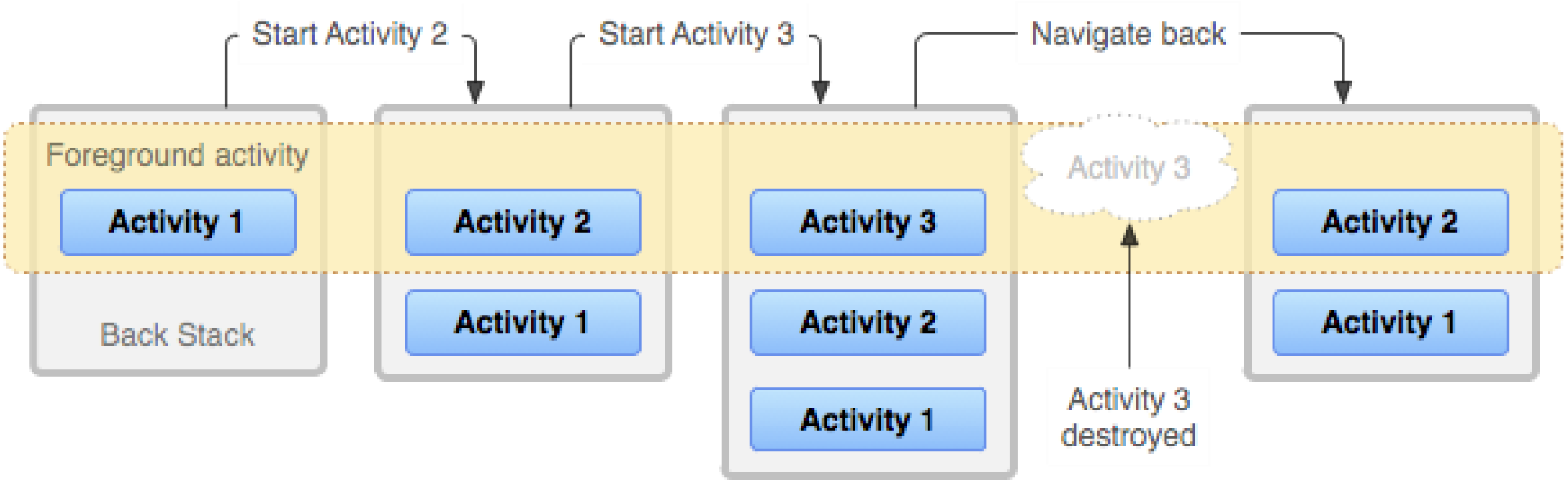
Ref:
https://developer.android.com/guide/components/activities/activity-lifecycle

Managing the Activity Lifecycle

- onCreate()
  - Must implement!
- onStart()
  - Called after onCreate()
- onResume()
  - APP regains focus
- onPause()
  - APP loses focus
- onStop()
  - APP no longer visible to the user
- onDestroy()
  - Activity is finishing or temporarily destroying to save space

Tasks and Back Stack

- Stack -> last in, first out



Note (Back press behavior for root launcher activities):

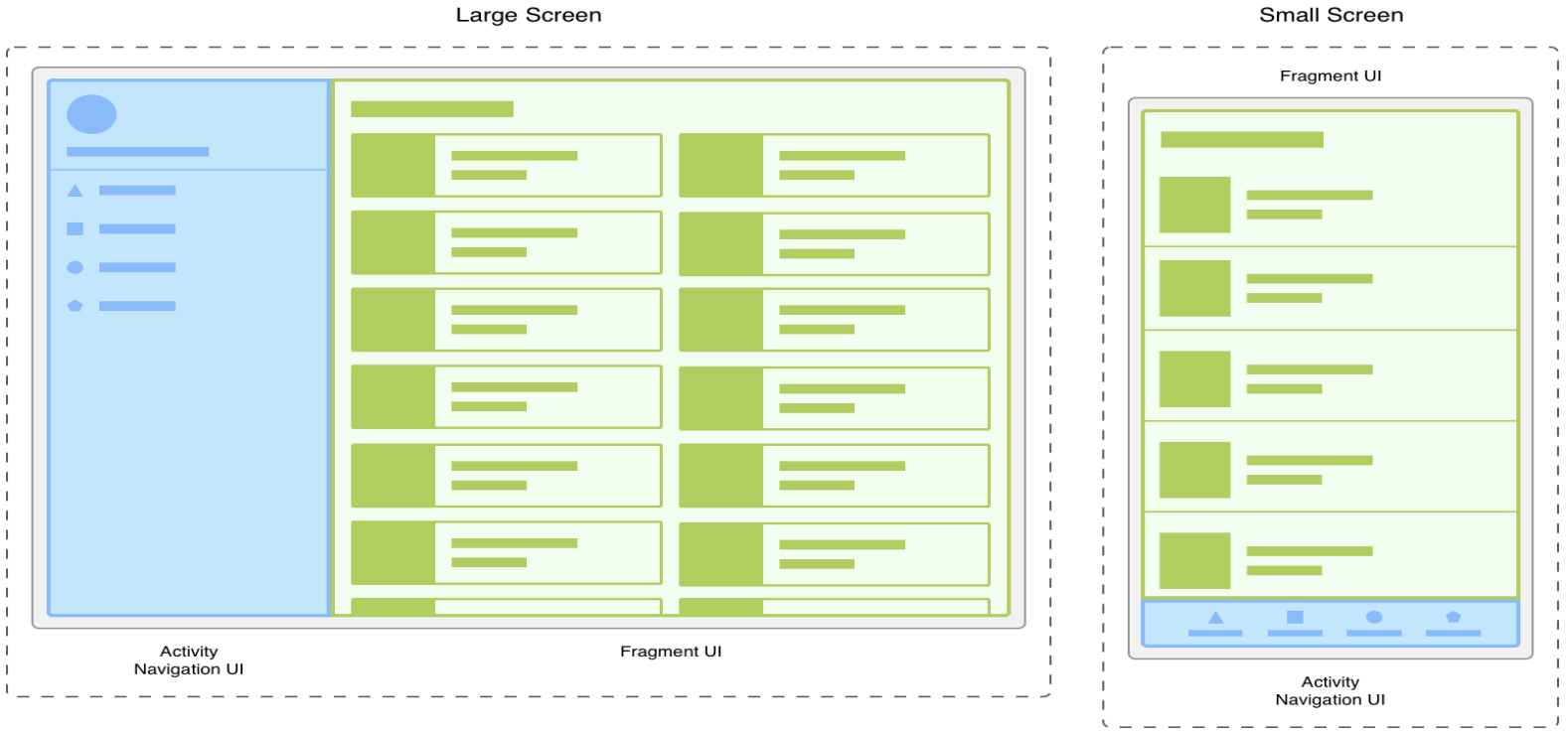
Android 11 and lower : **The system finishes the activity.**

Android 12 and higher : **The system moves the activity and its task to the background instead of finishing the activity.**

Ref: https://developer.android.com/guide/components/activities/tasks-and-back-stack

Fragment

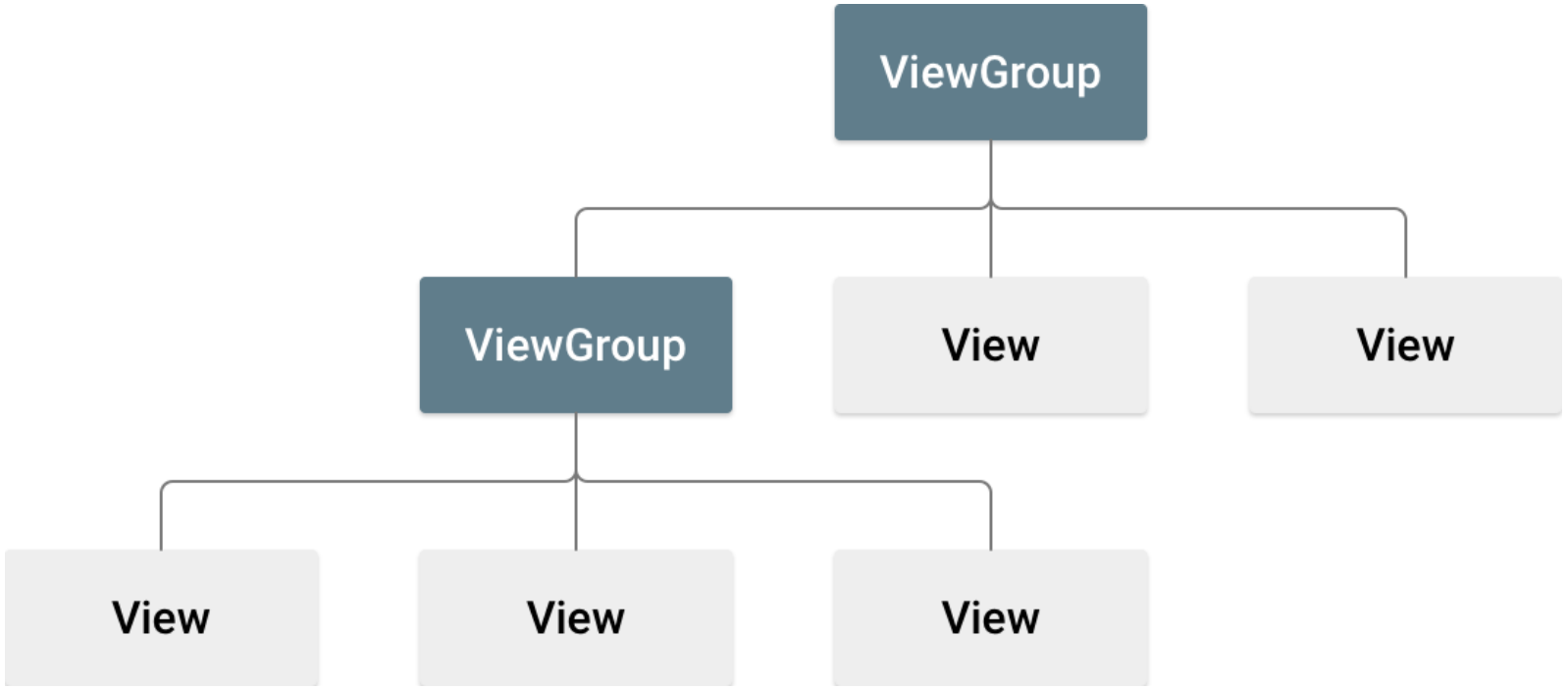
- Require a dependency on the AndroidX Fragment library
- Support dynamic design on large screen
- Fragments introduce modularity and reusability into your activity's UI by allowing you to divide the UI into discrete chunks



Ref: https://developer.android.com/guide/fragments

Layouts

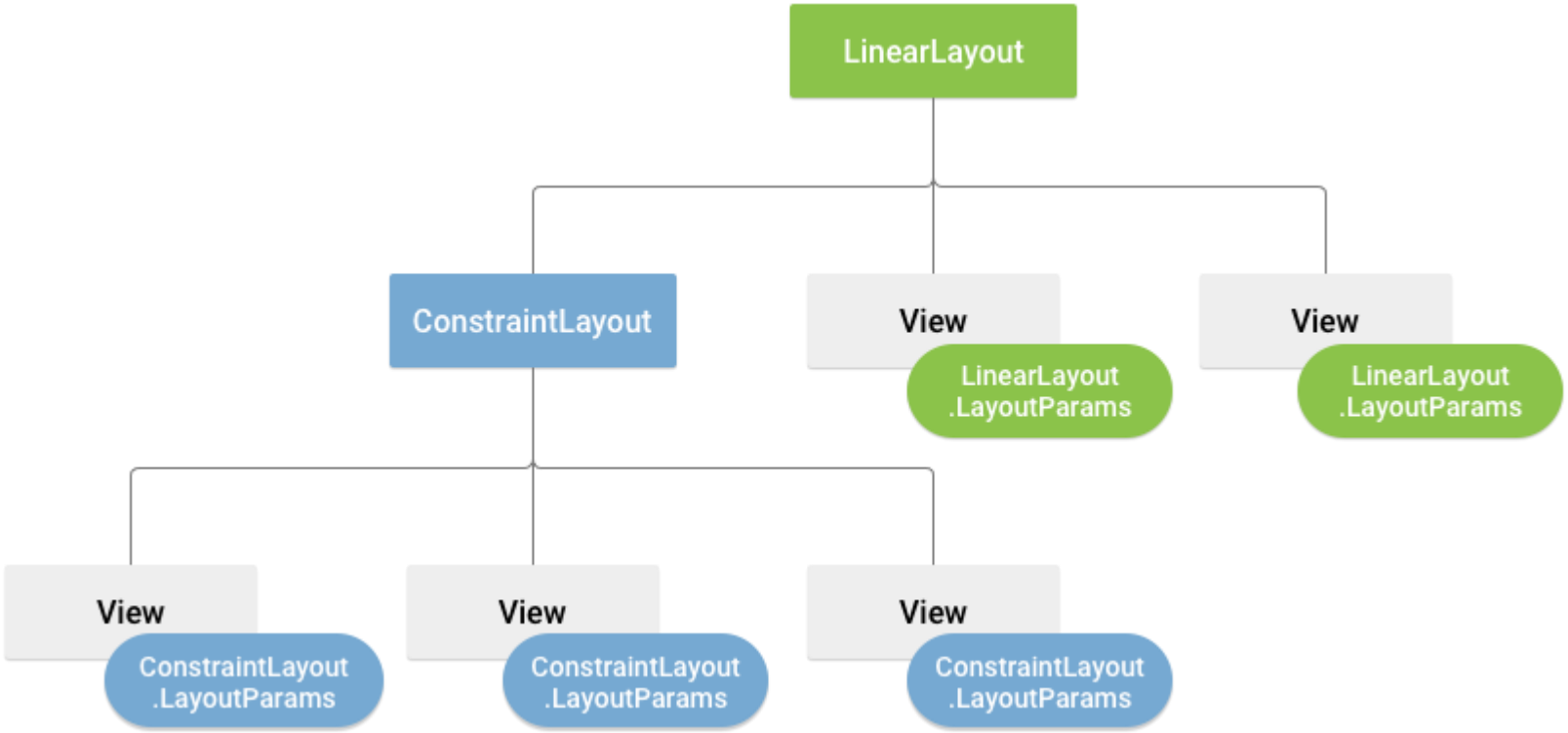
- Define the structure of APP UI
- View – Widgets, like button or text view
- View Group – LinearLayout or ConstraintLayout



Ref: https://developer.android.com/develop/ui/views/layout/declaring-layout

XML Layout Attributes

- Every ViewGroup class implements a nested class that extends ViewGroup.LayoutParams. This subclass contains property types that define the size and position for each child view, as appropriate for the view group.



Layout Types

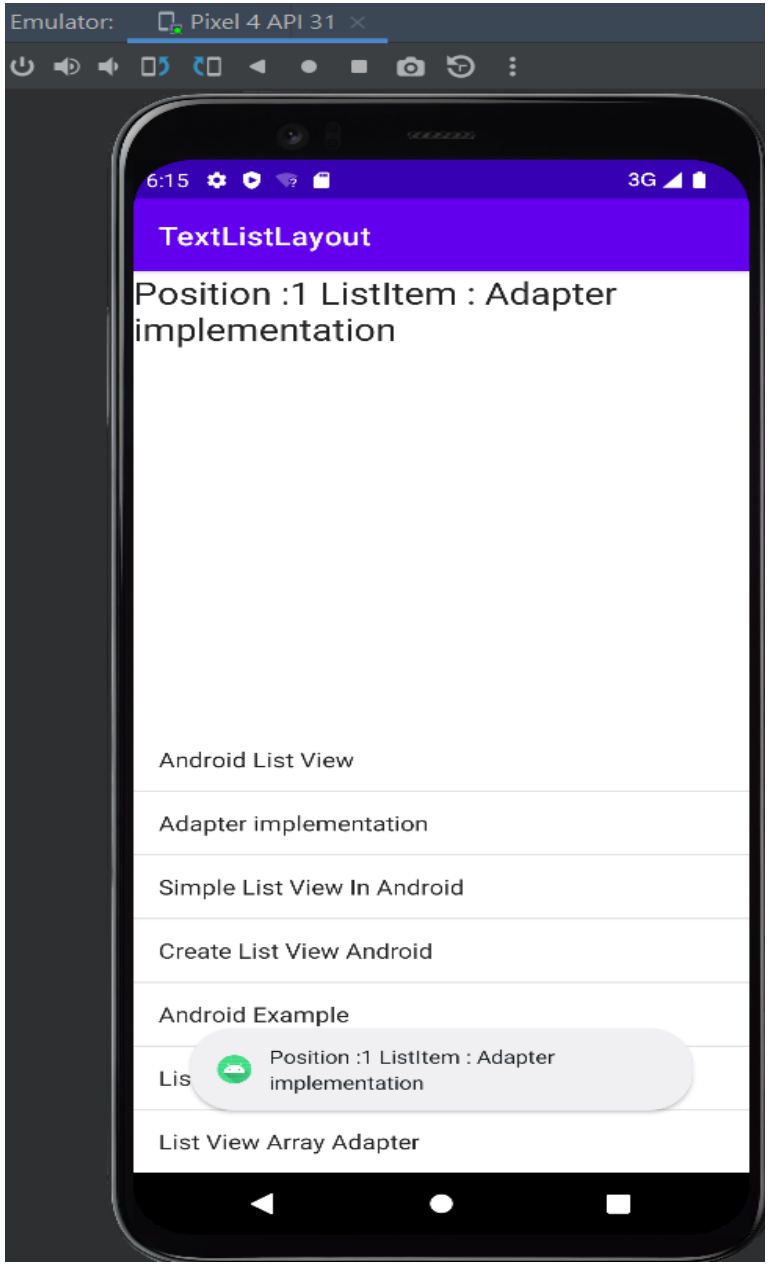
- Linear layout – Vertical or horizontal
- Relative layout
- Web view

Linear Layout

Relative Layout

Web View

Top TextView



FrameLayout + Bottom ListView

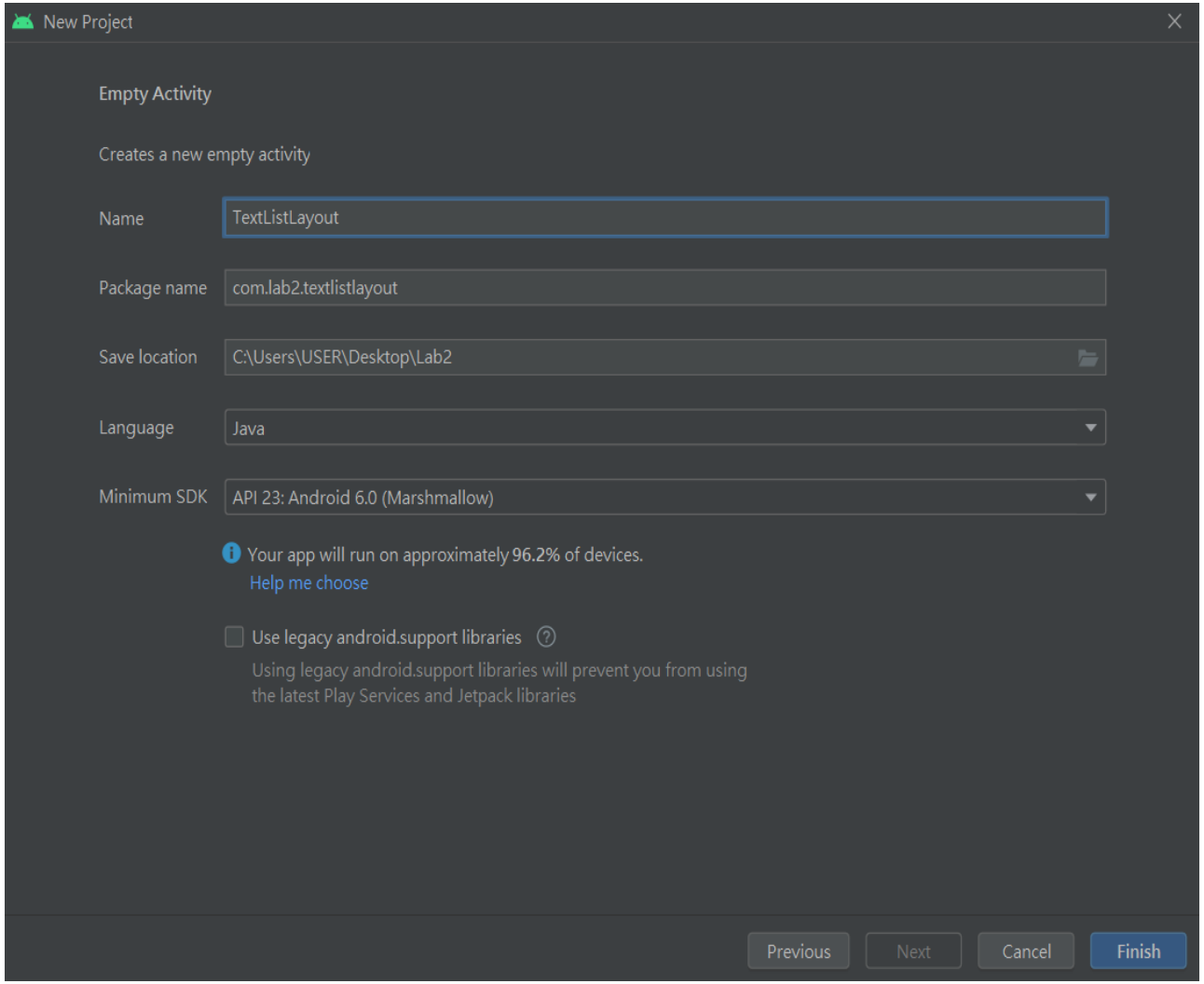
Today's Lab

- TextView + ListView
- Click item to show position and item name on Text View

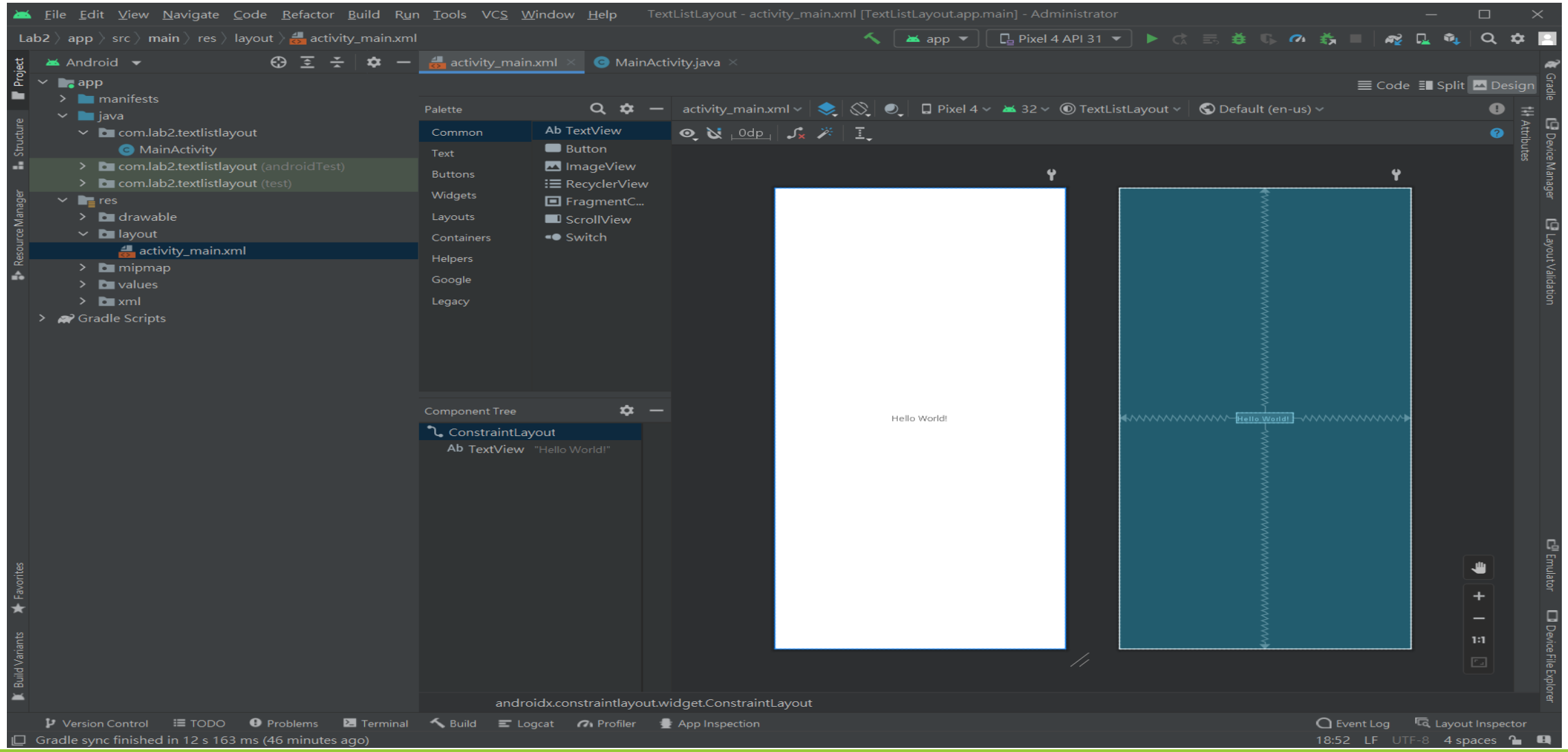
Ref: https://androidexample.com/create-a-simple-listview

Create a New Project

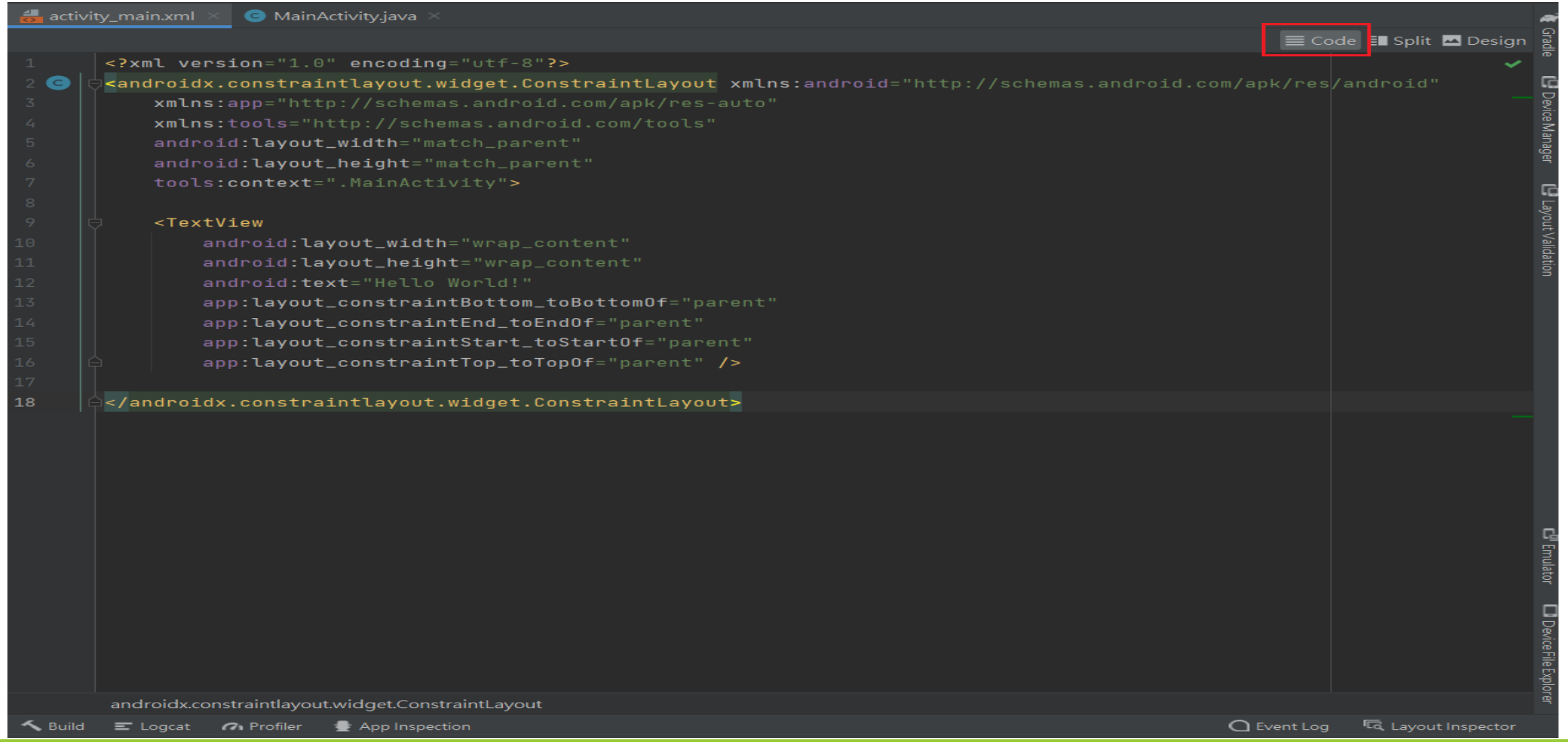
- Create a new project names TextListLayout
- Select **Empty Activity**
- Use default class name “MainActivity”
- Finish



Open “res/activity_main.xml”



Select “Code” to View Code



Modify activity_main.xml

- Change from default ConstraintLayout to LinearLayout

```
<androidx.appcompat.widget.LinearLayoutCompat
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

</androidx.appcompat.widget.LinearLayoutCompat>
```

Create Top Text View

- Add FrameLayout and a TextView

```
<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="0px"
    android:layout_weight="1">
    <TextView
        android:id="@+id/topTextView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAppearance="@style/TextAppearance.AppCompat.Headline" />
</FrameLayout>
```

Create Top Text View (Cont.)

- layout_weight

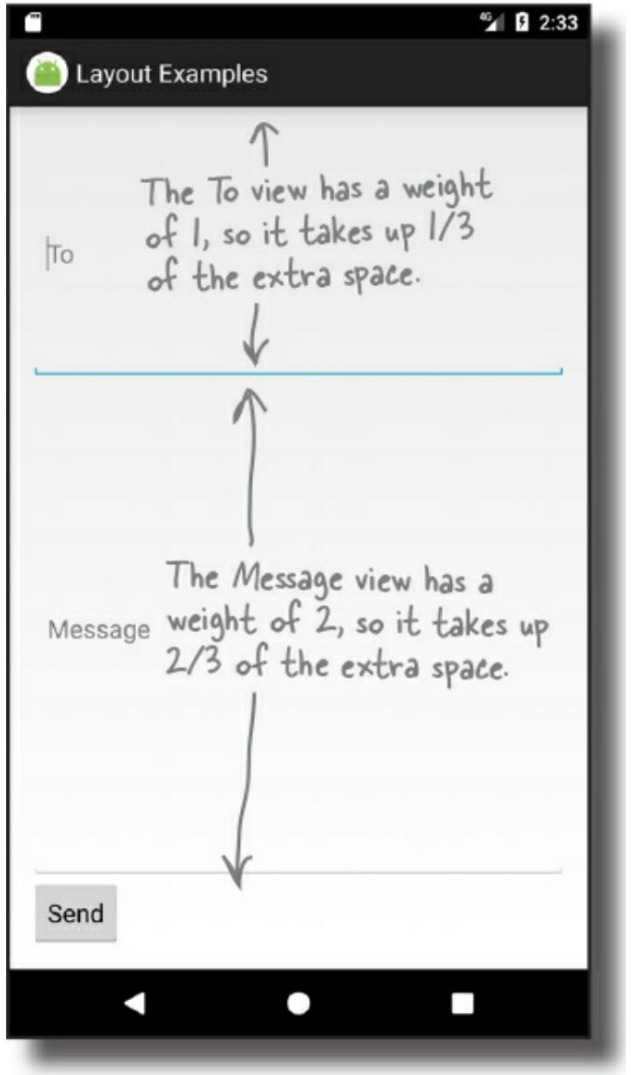
```
<FrameLayout
    android:layout_width="match_parent"
    android:layout_height="0px"
    android:layout_weight="1">
    <TextView
        android:id="@+id/topTextView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textAppearance="@style/TextAppearance.AppCompat.Headline" />
</FrameLayout>
```

0px to let layout_weight decide height

Layout_weight:
Expand and fill all the free space

Adding weights to Multiple Views

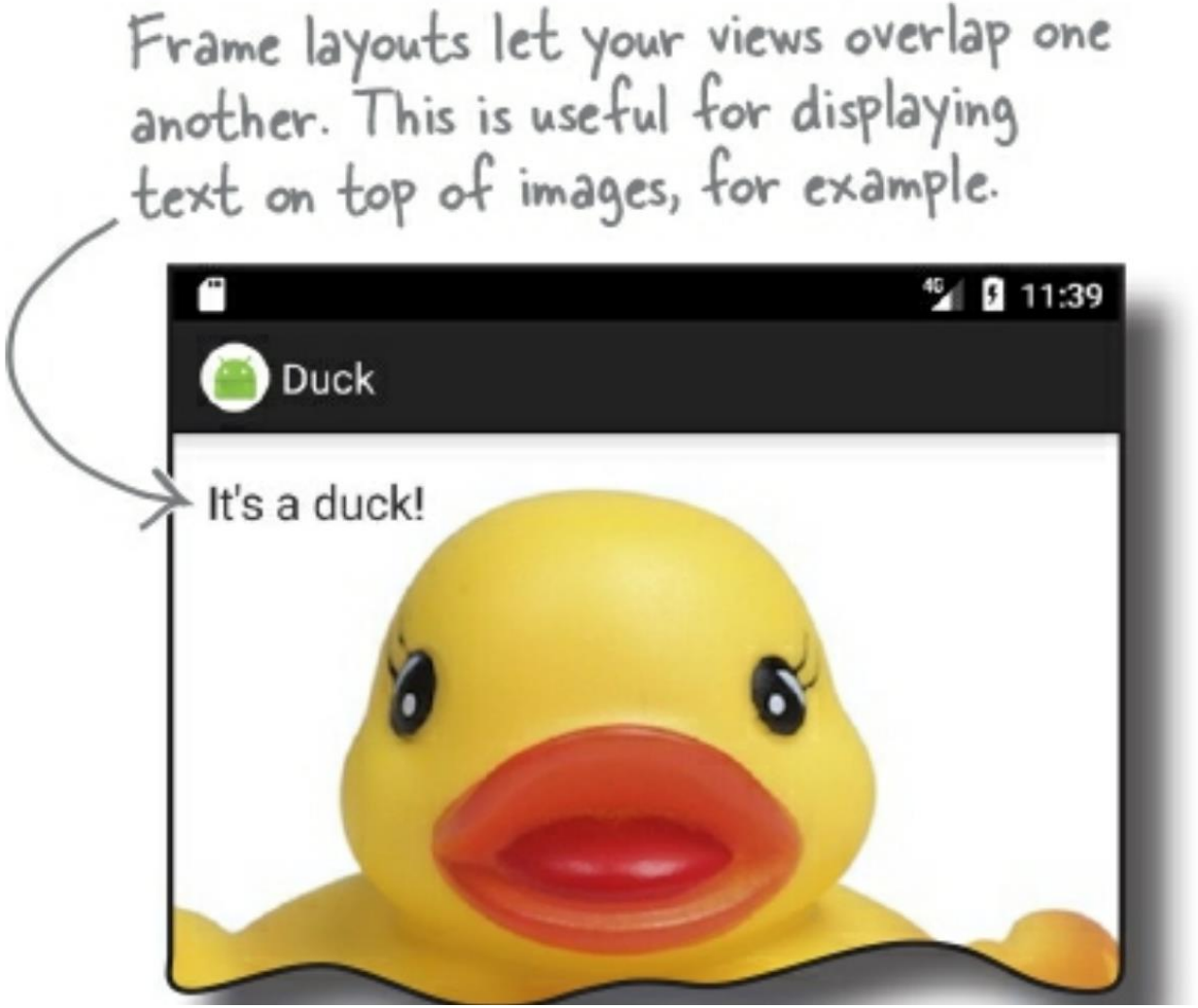
```
<androidx.appcompat.widget.LinearLayoutCompat ... >
   ...
   <EditText
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="1"
      android:hint="@string/to" />
   <EditText
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="2"
      android:hint="@string/message" />
   ...
</androidx.appcompat.widget.LinearLayoutCompat>
```



Gravity

```
< androidx.appcompat.widget.LinearLayoutCompat ... >
   ...
   <EditText
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="1"
      android:gravity="top"
      android:hint="@string/message" />
   ...
</androidx.appcompat.widget.LinearLayoutCompat>
```

FrameLayout



Create Bottom ListView

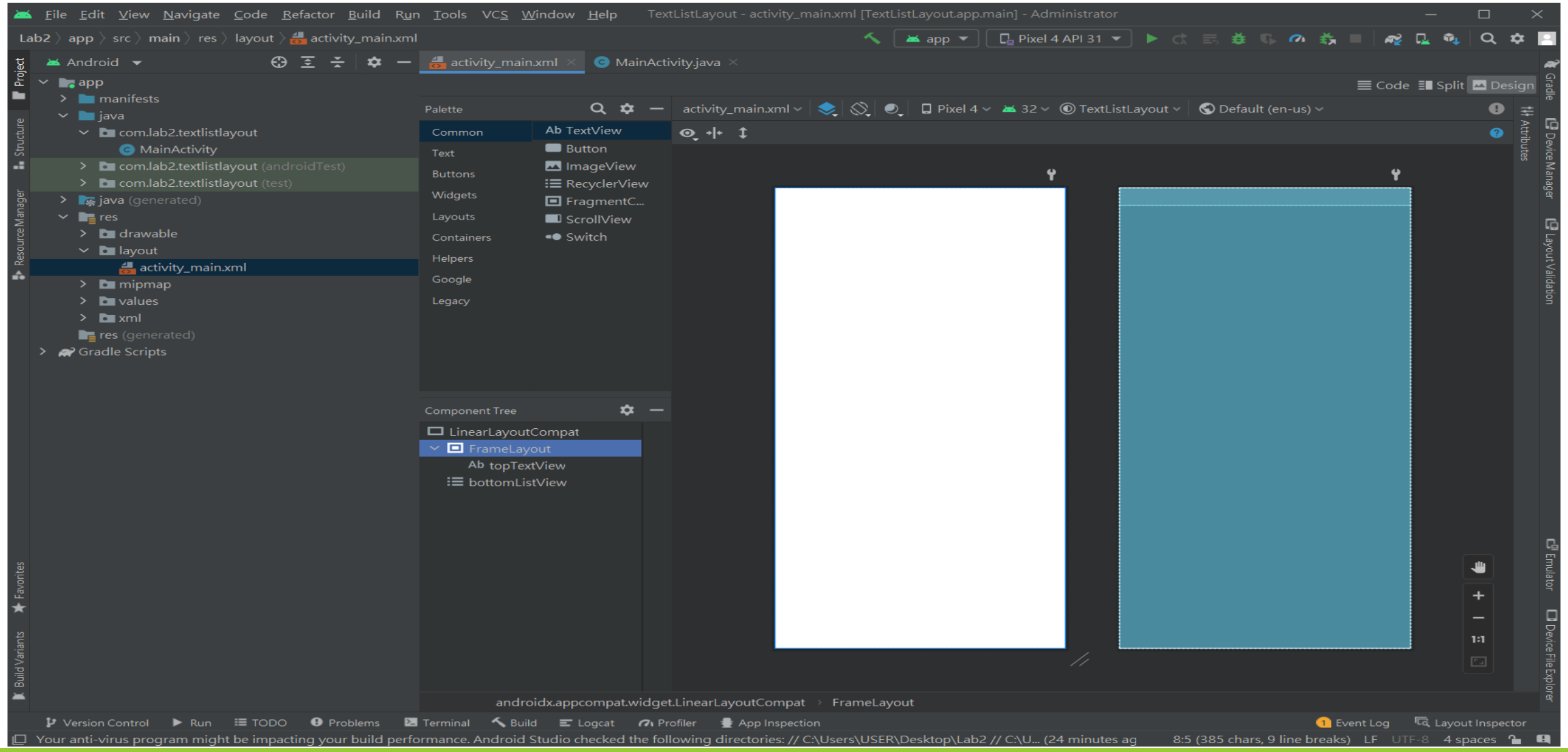
- Insert ListView after </FrameLayout>

```
....
<FrameLayout
....
</FrameLayout>

<ListView
    android:id="@+id/bottomListView"
    android:layout_height="wrap_content"
    android:layout_width="match_parent">
</ListView>
....
```

Final activity_main.xml

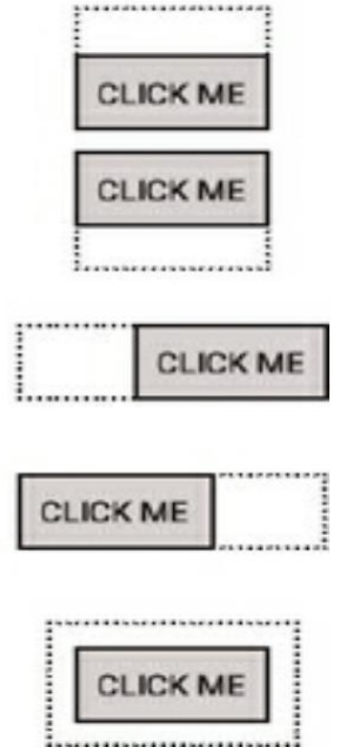
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.appcompat.widget.LinearLayoutCompat
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="0px"
        android:layout_weight="1">
        <TextView
            android:id="@+id/topTextView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:textAppearance="@style/TextAppearance.AppCompat.Headline" />
    </FrameLayout>

    <ListView
        android:id="@+id/bottomListView"
        android:layout_height="wrap_content"
        android:layout_width="match_parent">
    </ListView>
</androidx.appcompat.widget.LinearLayoutCompat>
```

Click “Design” to See the Layout



Margin

android:layout_marginTop	Specifies extra space on the top side of this view.
android:layout_marginBottom	Specifies extra space on the bottom side of this view.
android:layout_marginLeft	Specifies extra space on the left side of this view.
android:layout_marginStart	Specifies extra space on the start side of this view.
android:layout_marginRight	Specifies extra space on the right side of this view.
android:layout_marginEnd	Specifies extra space on the end side of this view.
android:layout_margin	Specifies extra space on the left, top, right and bottom sides of this view.
android:layout_marginHorizontal	Specifies extra space on the left and right sides of this view.
android:layout_marginVertical	Specifies extra space on the top and bottom sides of this view.



Ref: https://developer.android.com/reference/android/view/ViewGroup.MarginLayoutParams

Add ListView Reference in MainActivity.java

```
import …
public class MainActivity extends AppCompatActivity {

    ListView listView;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        listView = findViewById(R.id.bottomListView);
        String[] values = new String[] {
                "Android List View",
                "Adapter implementation",
                "Simple List View In Android",
                "Create List View Android",
                "Android Example",
                "List View Source Code",
                "List View Array Adapter"
        };
        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_1, android.R.id.text1, values);
        listView.setAdapter(adapter);
    }
}
```

Add “setOnItemClickListener”

```
...

listView.setAdapter(adapter);

listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        int itemPosition = position;
        String itemValue = (String) listView.getItemAtPosition(position);
        String posInfo = "Position :"+itemPosition+" ListItem : " +itemValue;
        // Show Alert
        Toast.makeText(getApplicationContext(), posInfo , Toast.LENGTH_LONG).show();
        TextView topView = findViewById(R.id.topTextView);
        topView.setText(posInfo);
    }
});
...
```

Final Result

- Click any text in the listView
- TextView will display your selection
- Toast also show a pop-up message