

# Pemrograman Mobile

3 SKS | Semester 7 | S1 Sistem Informasi

*Nizar Rabbi Radliya*  
*nizar@email.unikom.ac.id*

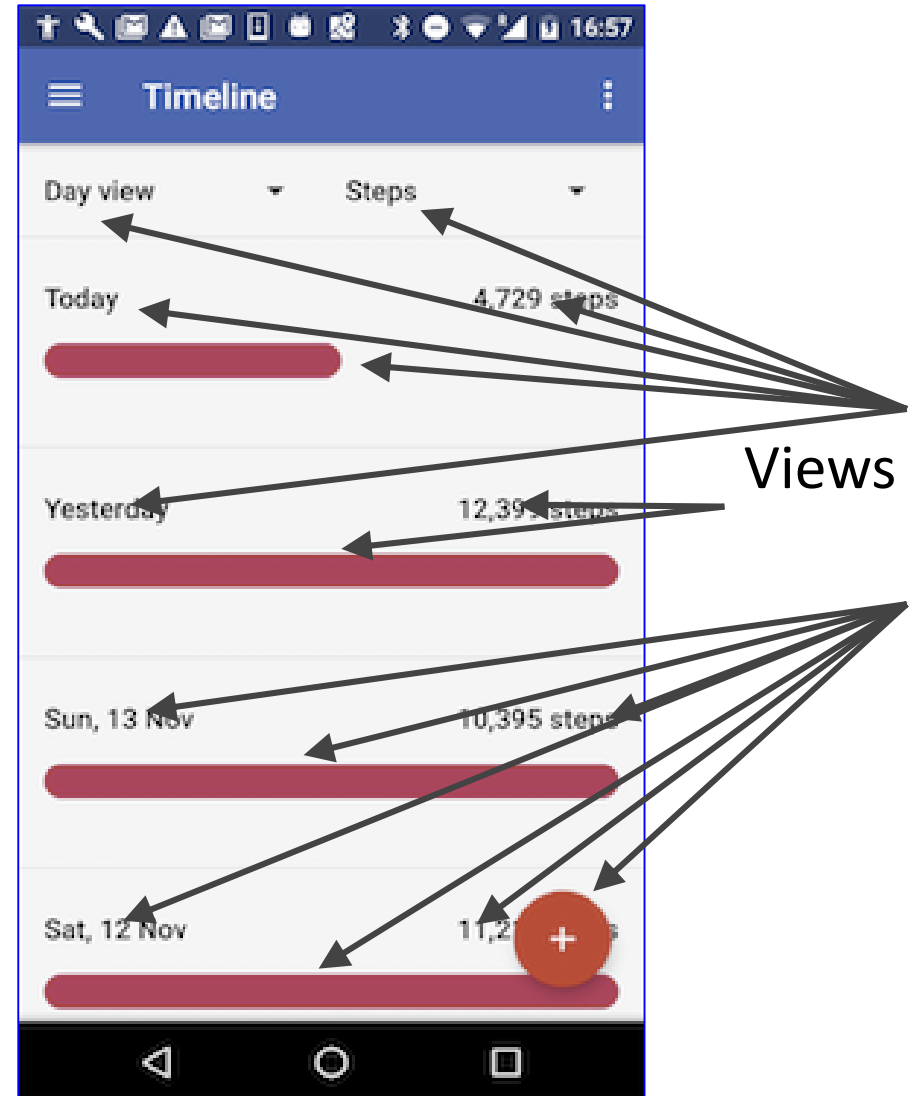
## Lesson 2

### Views, Layouts, and Resources



# Everything you see is a view

If you look at your mobile device,  
every user interface element that you see is a View.



# What is a view?

Views are Android's basic user interface building blocks.

- display text (TextView class), edit text (EditText class)
- buttons (Button class), menus, other controls
- scrollable (ScrollView, RecyclerView)
- show images (ImageView)
- subclass of View class



# Views have properties

- Have properties (e.g., color, dimensions, positioning)
- May have focus (e.g., selected to receive user input)
- May be interactive (respond to user clicks)
- May be visible or not
- Have relationships to other views

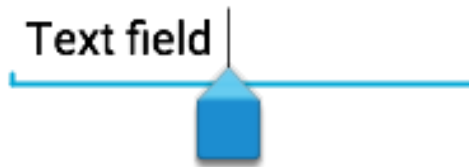


# Examples of views

Button



EditText



SeekBar



CheckBox



RadioButton



Switch

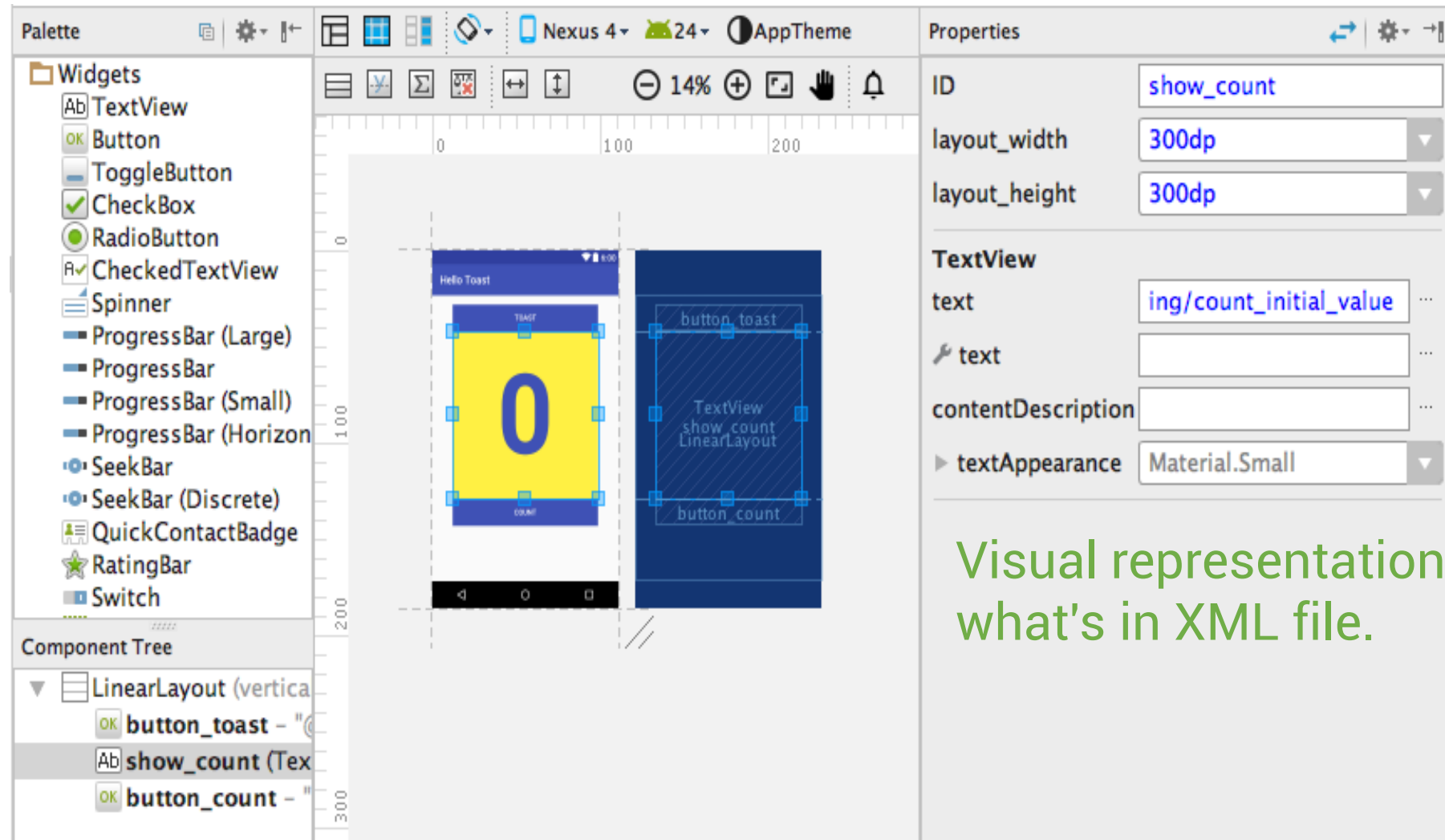


# Creating and laying out views

- Graphically within Android Studio
- XML Files
- Programmatically

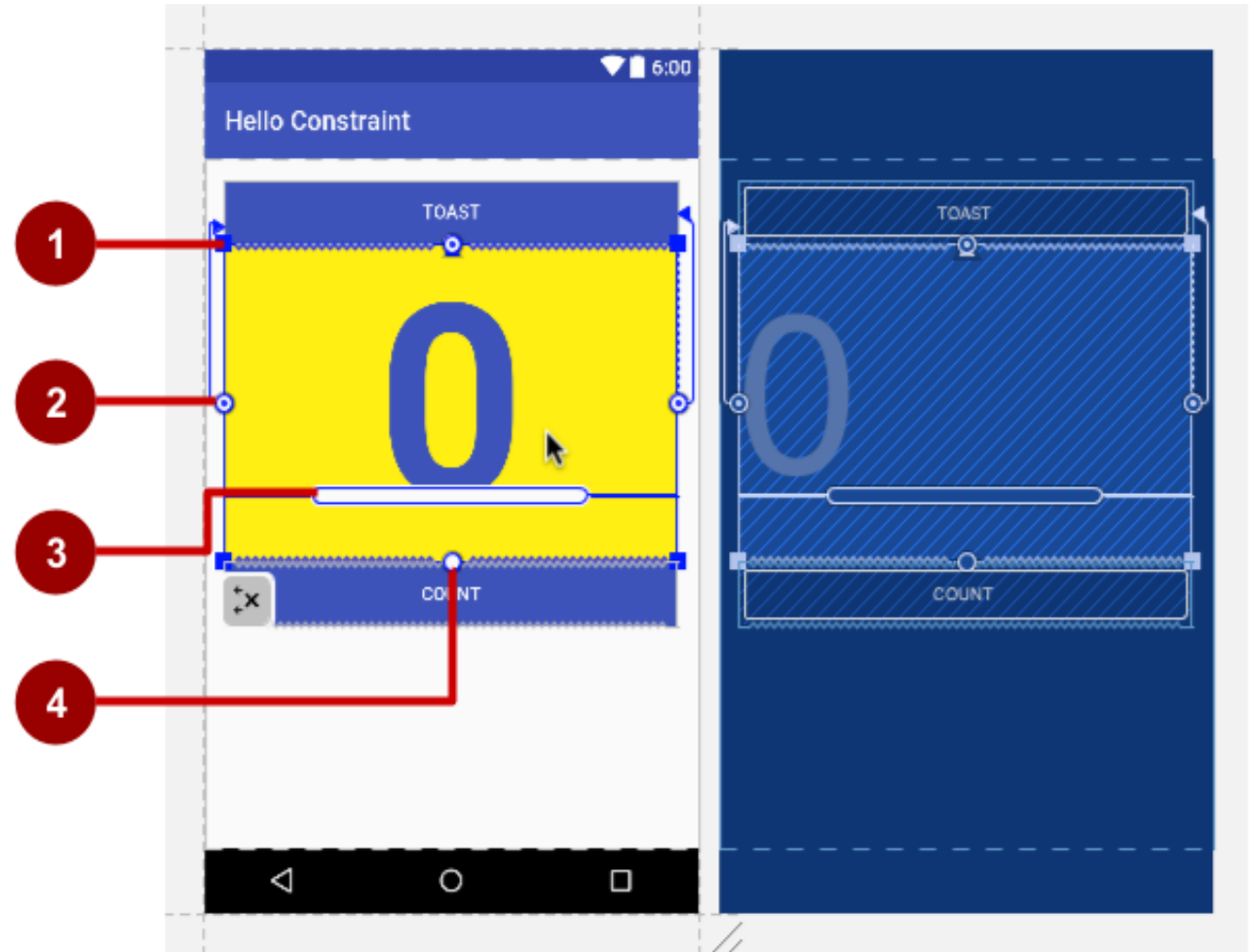


# Views defined in Layout Editor



# Using the Layout Editor

1. Resizing handle
2. Constraint line and handle
3. Baseline handle
4. Constraint handle





## Views defined in XML

### <TextView

```
    android:id="@+id/show_count"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:background="@color/myBackgroundColor"  
    android:text="@string/count_initial_value"  
    android:textColor="@color/colorPrimary"  
    android:textSize="@dimen/count_text_size"  
    android:textStyle="bold"
```

```
/>
```



## View properties in XML

**android:<property\_name>=<property\_value>**

**Example:** android:layout\_width="match\_parent"

**android:<property\_name>="@<resource\_type>/resource\_id"**

**Example:** android:text="@string/button\_label\_next"

**android:<property\_name>="@+id/view\_id"**

**Example:** android:id="@+id/show\_count"



## Create View in Java code

In an Activity:

```
TextView myText = new TextView(this);  
myText.setText("Display this text!");
```



# ViewGroup views

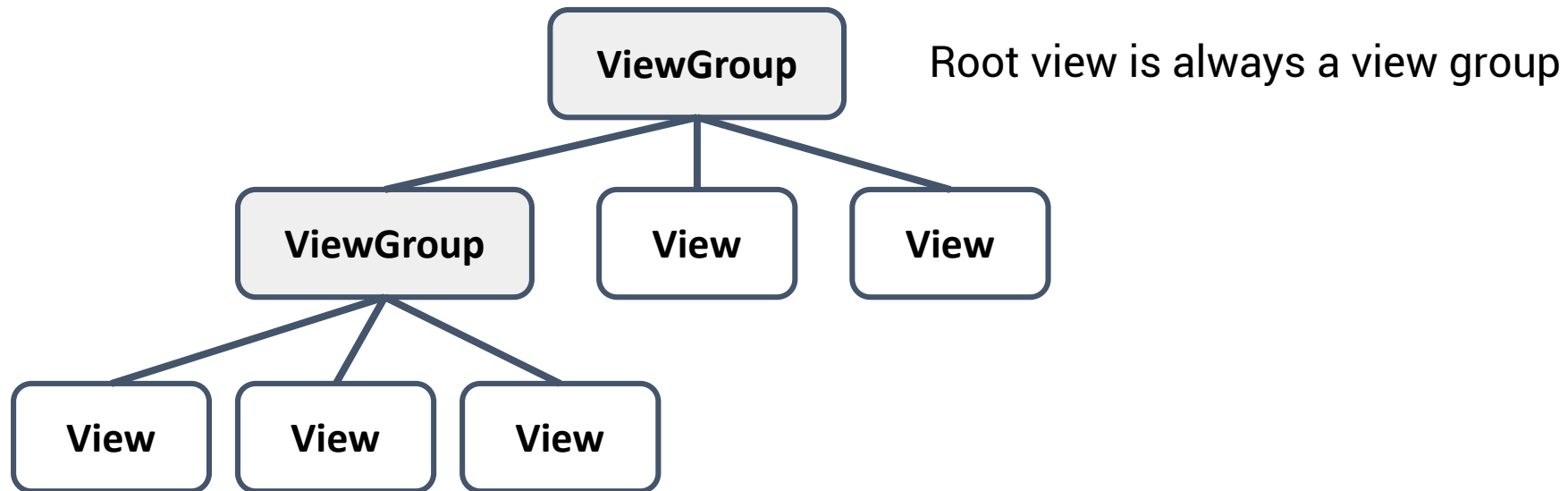
A ViewGroup (parent) is a type of view that can contain other views (children)

ViewGroup is the base class for layouts and view containers

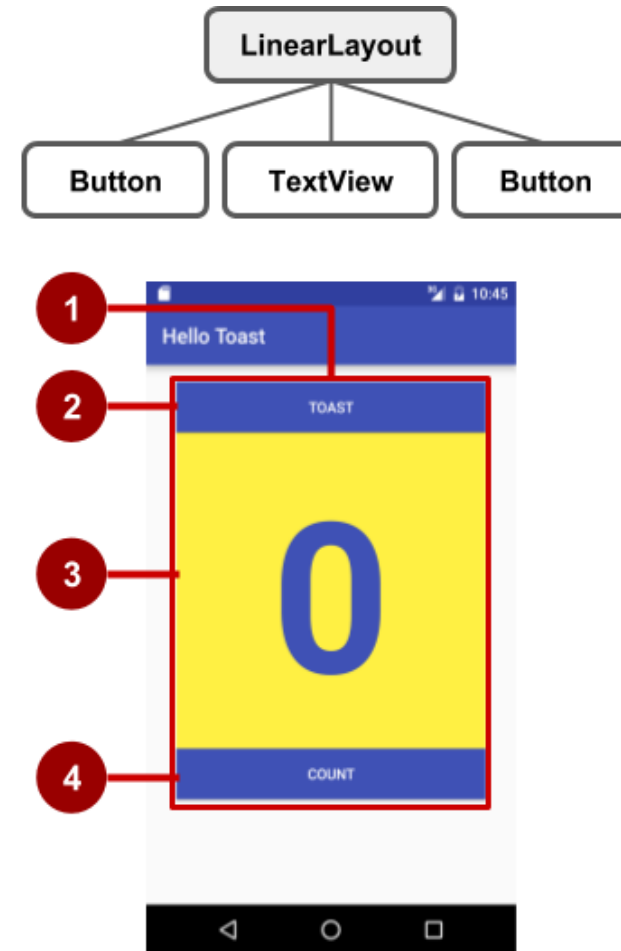
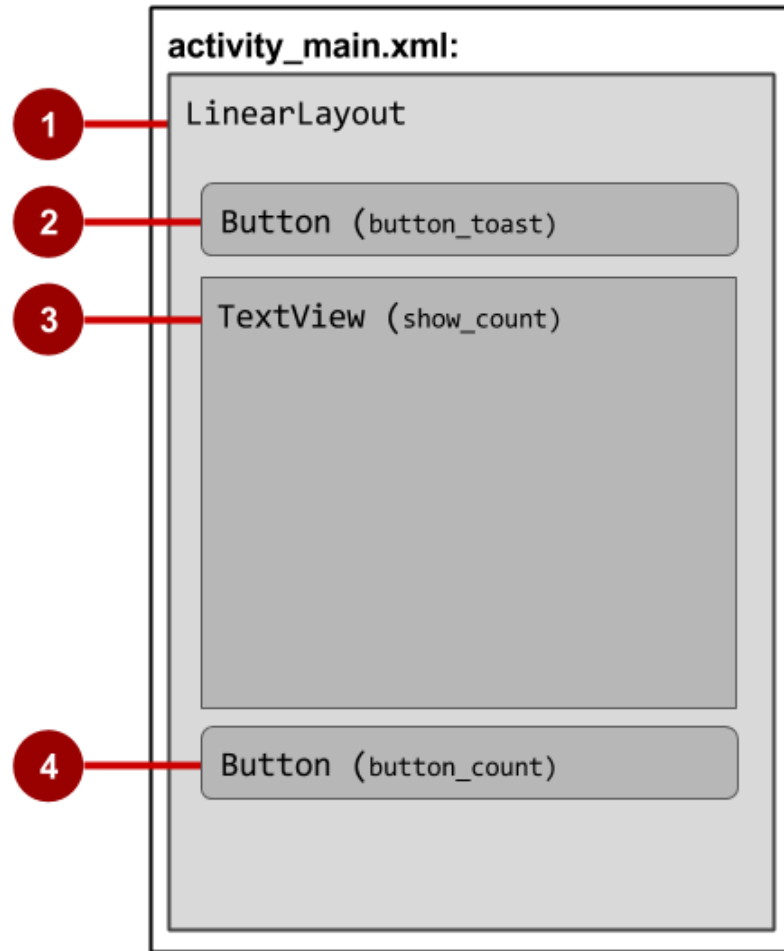
- ScrollView—scrollable view that contains one child view
- LinearLayout—arrange views in horizontal/vertical row
- RecyclerView—scrollable "list" of views or view groups



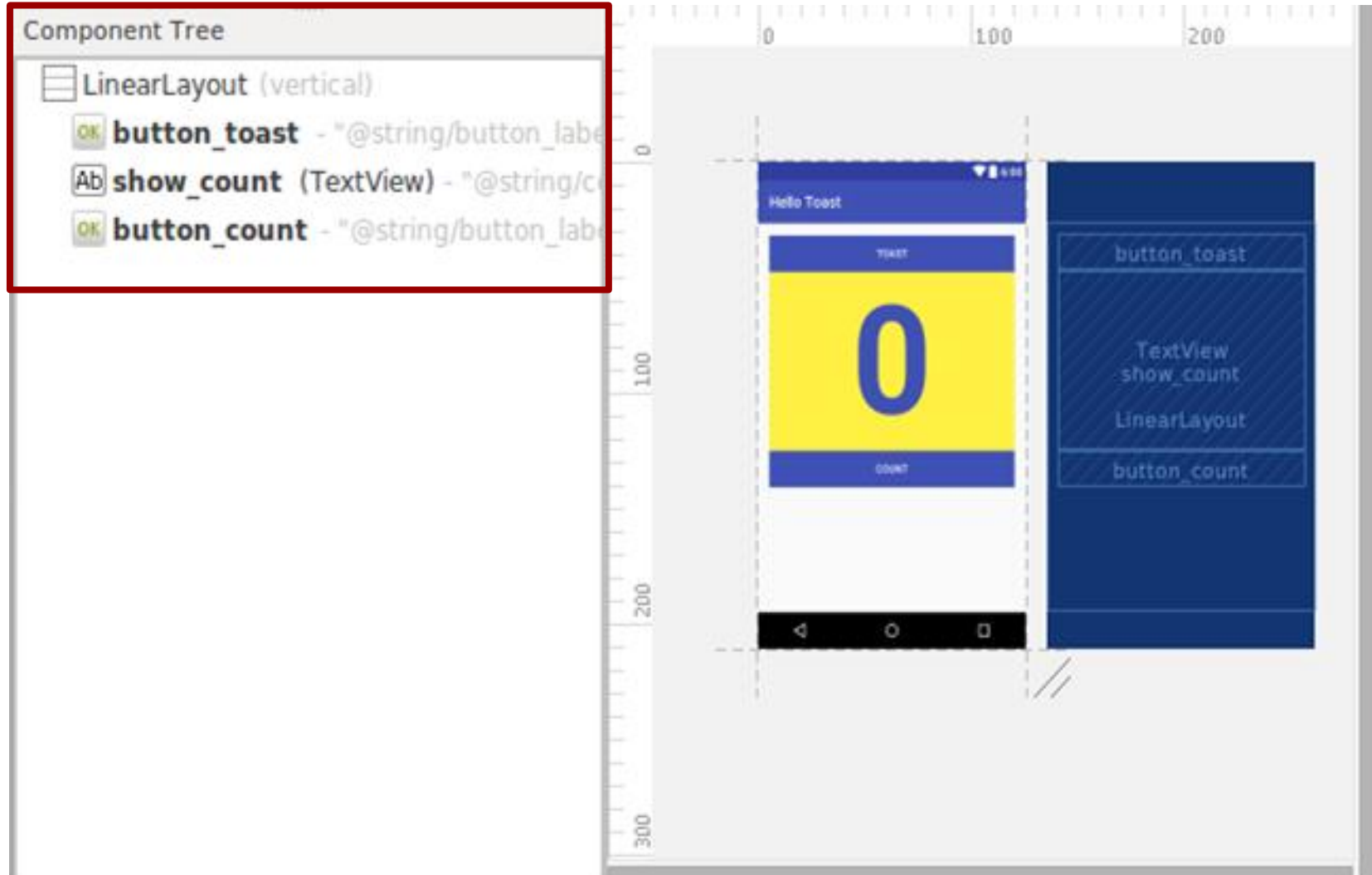
# Hierarchy of view groups and views



# View hierarchy and screen layout



# View hierarchy in the component tree



# Best practices for view hierarchies

- Arrangement of view hierarchy affects app performance
- Use smallest number of simplest views possible
- Keep the hierarchy flat—limit nesting of views and view groups





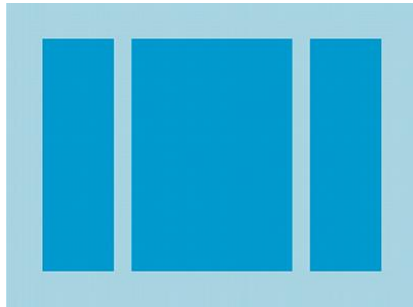
# Layout Views

## Layouts

- are specific types of view groups
- are subclasses of ViewGroup
- contain child views
- can be in a row, column, grid, table, absolute



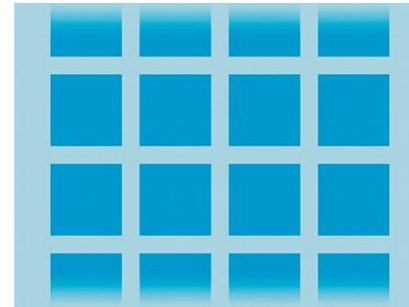
# Common Layout Classes



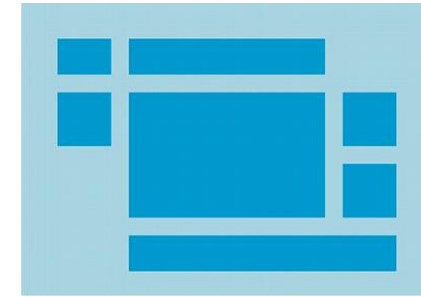
LinearLayout



RelativeLayout



GridLayout



TableLayout

# Common Layout Classes

- ConstraintLayout - connect views with constraints
- LinearLayout - horizontal or vertical row
- RelativeLayout - child views relative to each other
- TableLayout - rows and columns
- FrameLayout - shows one child of a stack of children
- GridView - 2D scrollable grid



# Class Hierarchy vs. Layout Hierarchy

## **View class-hierarchy is standard object-oriented class inheritance**

- For example, Button is-a TextView is-a View is-a Object
- Superclass-subclass relationship

## **Layout hierarchy is how Views are visually arranged**

- For example, LinearLayout can contain Buttons arranged in a row
- Parent-child relationship



## Layout created in XML

```
<LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <EditText
        ... />
    <Button
        ... />
</LinearLayout>
```



## Layout created in Java Activity code

```
LinearLayout linearL = new LinearLayout(this);  
linearL.setOrientation(LinearLayout.VERTICAL);  
  
TextView myText = new TextView(this);  
myText.setText("Display this text!");  
  
linearL.addView(myText);  
setContentView(linearL);
```



## Setting width and height in Java code

Set the width and height of a view:

```
LinearLayout.LayoutParams layoutParams =  
    new LinearLayout.LayoutParams(  
        LinearLayout.LayoutParams.MATCH_PARENT,  
        LinearLayout.LayoutParams.WRAP_CONTENT);  
myView.setLayoutParams(layoutParams);
```



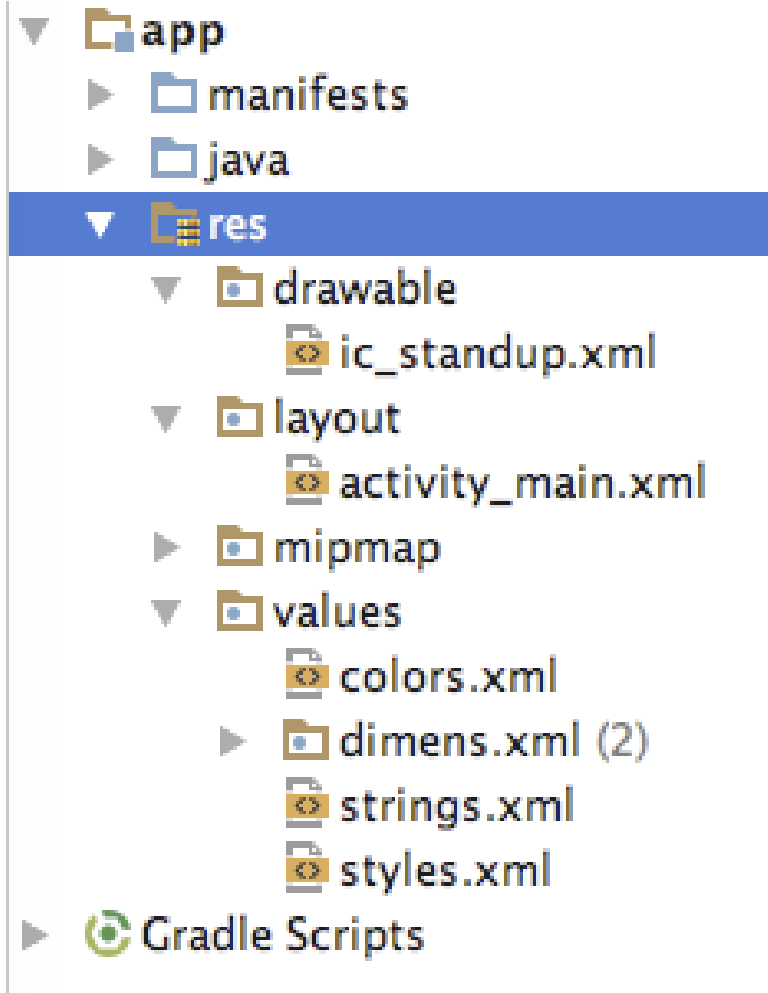
# Resources

- Separate static data from code in your layouts.
- Strings, dimensions, images, menu text, colors, styles
- Useful for localization





## Where are the resources in your project?



← resources and resource files  
stored in **res** folder

## Refer to resources in code

### ●Layout:

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

### ●View:

```
R.id.recyclerview  
  
rv = (RecyclerView) findViewById(R.id.recyclerview);
```

### ●String:

In Java: `R.string.title`

In XML: `android:text="@string/title"`



# Measurements

- Device Independent Pixels (dp) - for Views
- Scale Independent Pixels (sp) - for text
- Don't use device-dependent units:
  - ❑ Actual Pixels (px)
  - ❑ Actual Measurement (in, mm)
  - ❑ Points - typography 1/72 inch (pt)



## Learn more

### Views:

- [View class documentation](#)
- [device independent pixels](#)
- [Button class documentation](#)
- [TextView class documentation](#)
- [Hierarchy Viewer](#) for visualizing the view hierarchy

### Layouts:

- [developer.android.com Layouts](#)
- [Common Layout Objects](#)



## Learn more

### Resources:

- [Android resources](#)
- [Color](#) class definition
- [R.color resources](#)
- [Supporting Different Densities](#)
- [Color Hex Color Codes](#)

### Other:

- [Android Studio documentation](#)
- [Image Asset Studio](#)
- [UI Overview](#)
- [Vocabulary words and concepts glossary](#)
- [Model-View-Presenter](#) (MVP) architecture pattern
- [Architectural patterns](#)



## Learn more

### Developer Documentation:

- [TextView](#)
- [ScrollView](#) and [HorizontalScrollView](#)
- [String Resources](#)

### Other:

- Android Developers Blog: [Linkify your Text!](#)
- Codepath: [Working with a TextView](#)



# What's Next?

## Activities and Intents

