# Android

INTRODUCTION TO LAYOUTS & USER INTERFACE


# What is a view?

View subclasses are 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)
- Group views (ConstraintLayout and LinearLayout)

# ViewGroup and View hierarchy

# ViewGroup contains "child" views

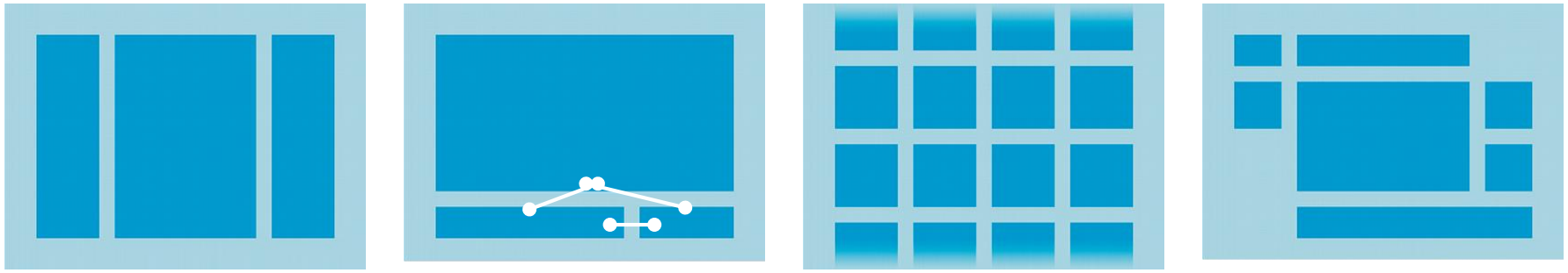
- **ConstraintLayout**: Positions UI elements using constraint connections to other elements and to the layout edges
- **ScrollView**: Contains one element and enables scrolling
- **RecyclerView**: Contains a list of elements and enables scrolling by adding and removing elements dynamically

# ViewGroups for layouts

Layouts

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

# Common Layout Classes

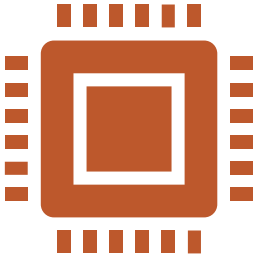


`LinearLayout` `ConstraintLayout` `GridLayout` `TableLayout`

# Common Layout Classes

- `ConstraintLayout`: Connect views with constraints
- `LinearLayout`: Horizontal or vertical row
- `RelativeLayout`: Child views relative to each other
- `TableLayout`: Rows and columns

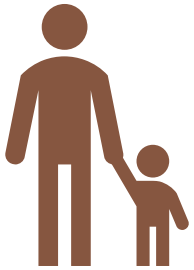
# Class hierarchy vs. layout hierarchy



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

For example, Button is-a TextView is-a View is-an Object

Superclass-subclass relationship

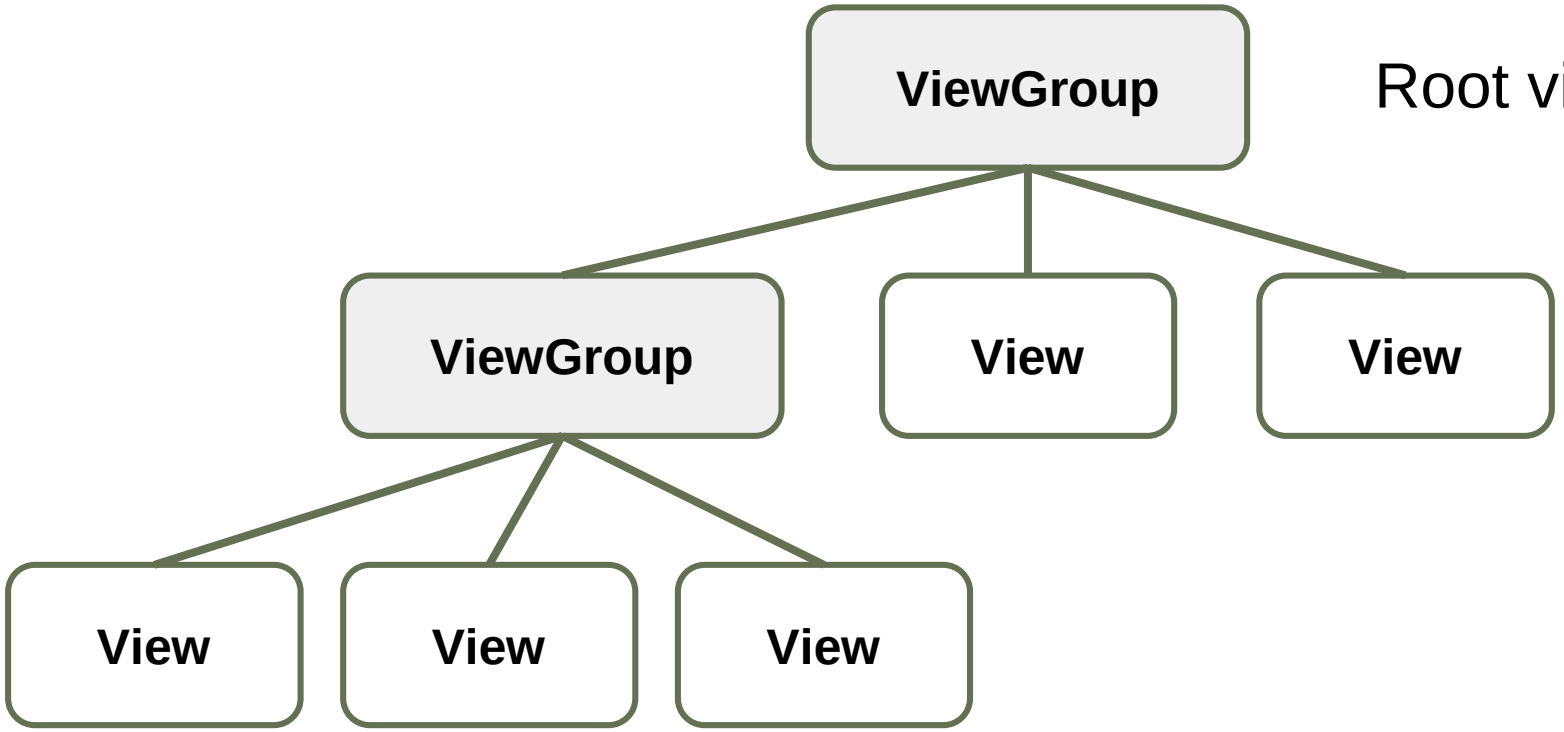


**Layout hierarchy is how views are visually arranged**

For example, LinearLayout can contain Buttons arranged in a row

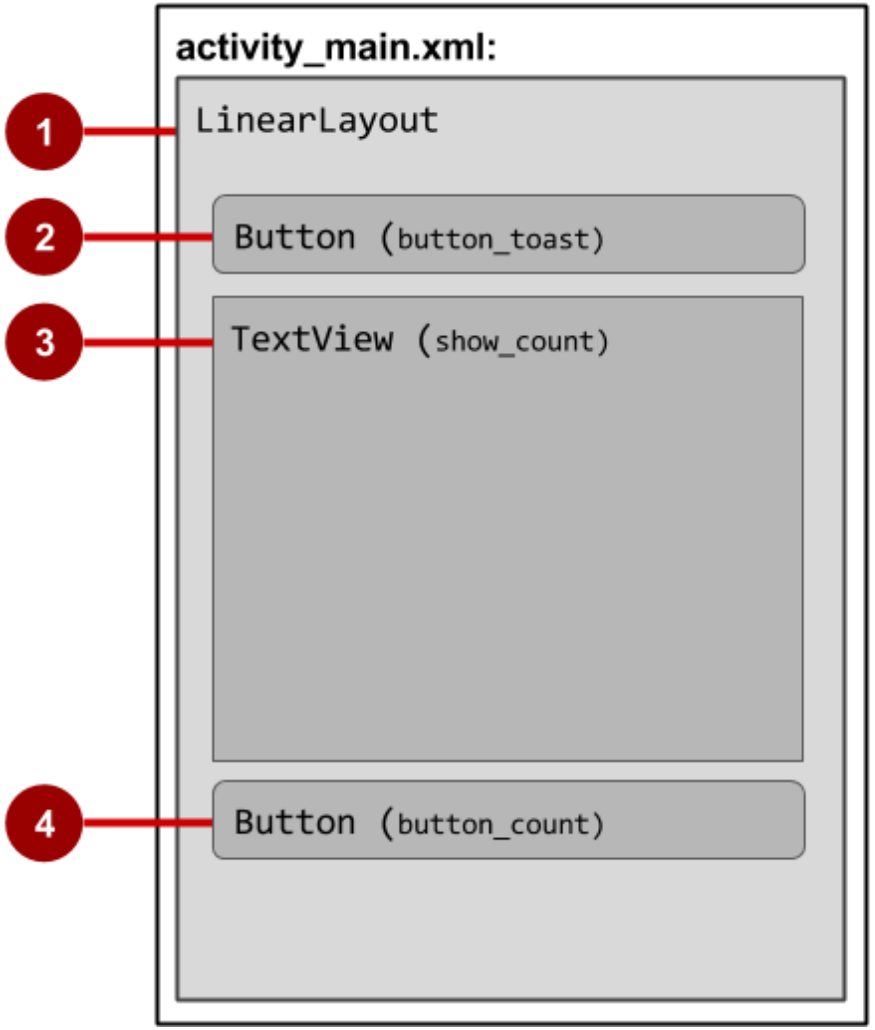
Parent-child relationship

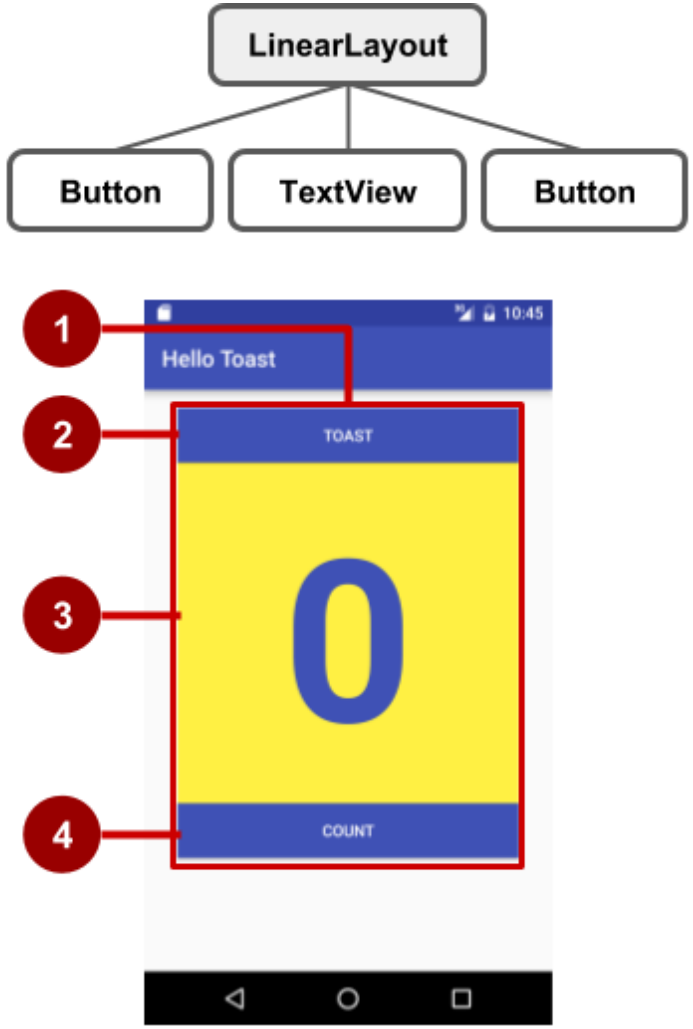
# Hierarchy of viewgroups and views



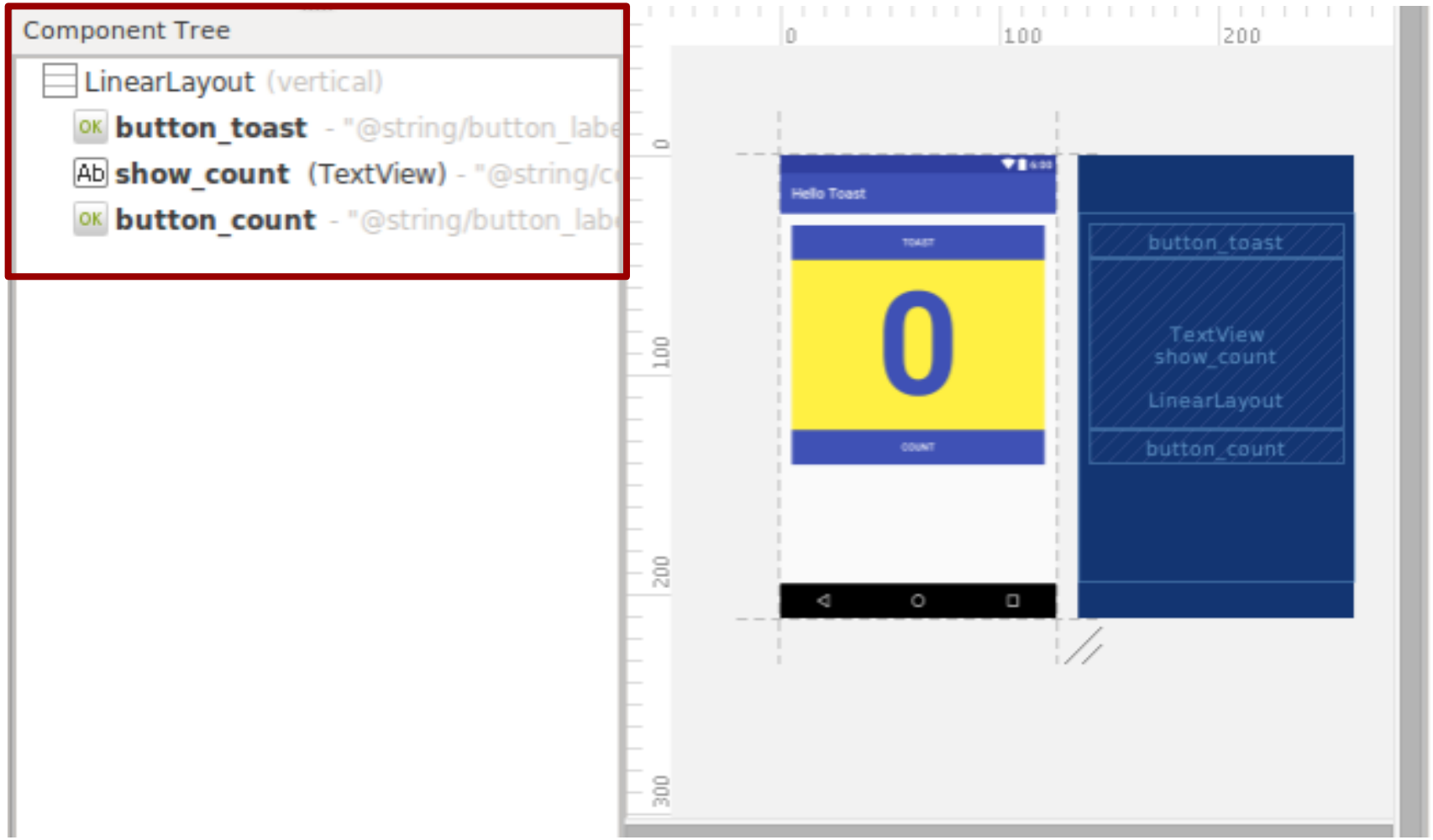

Root view is always a `ViewGroup`

# View hierarchy and screen layout



# View hierarchy in the layout editor



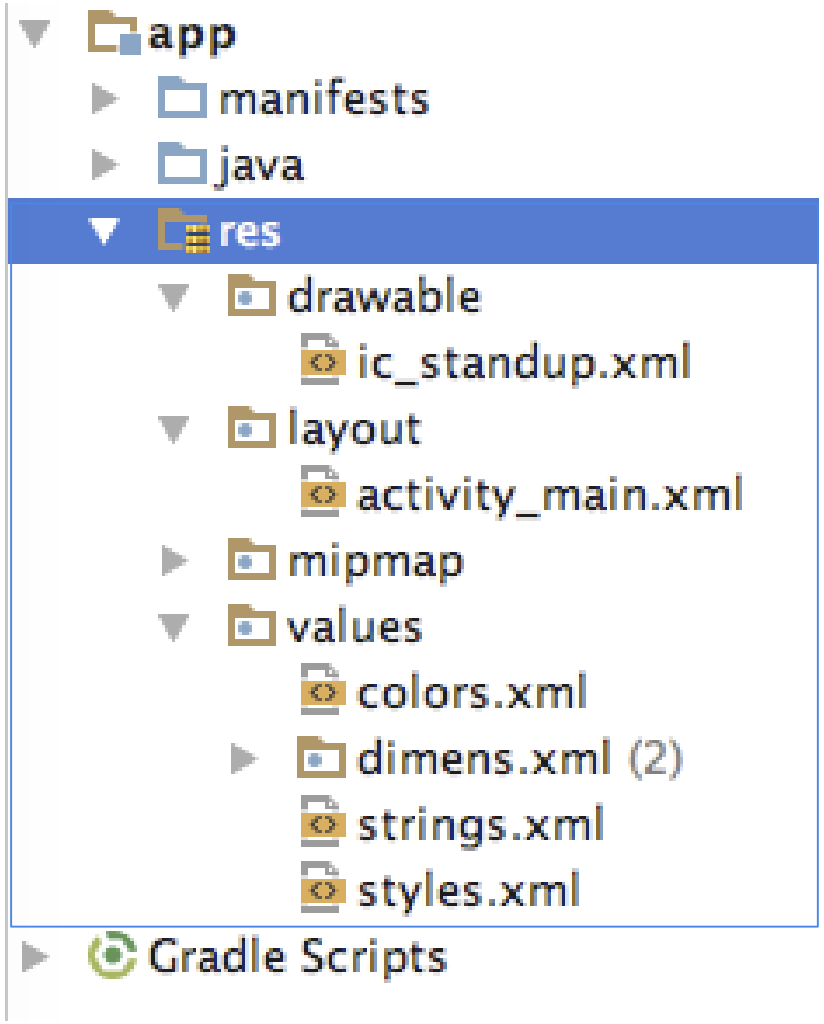
# Layout created in XML

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

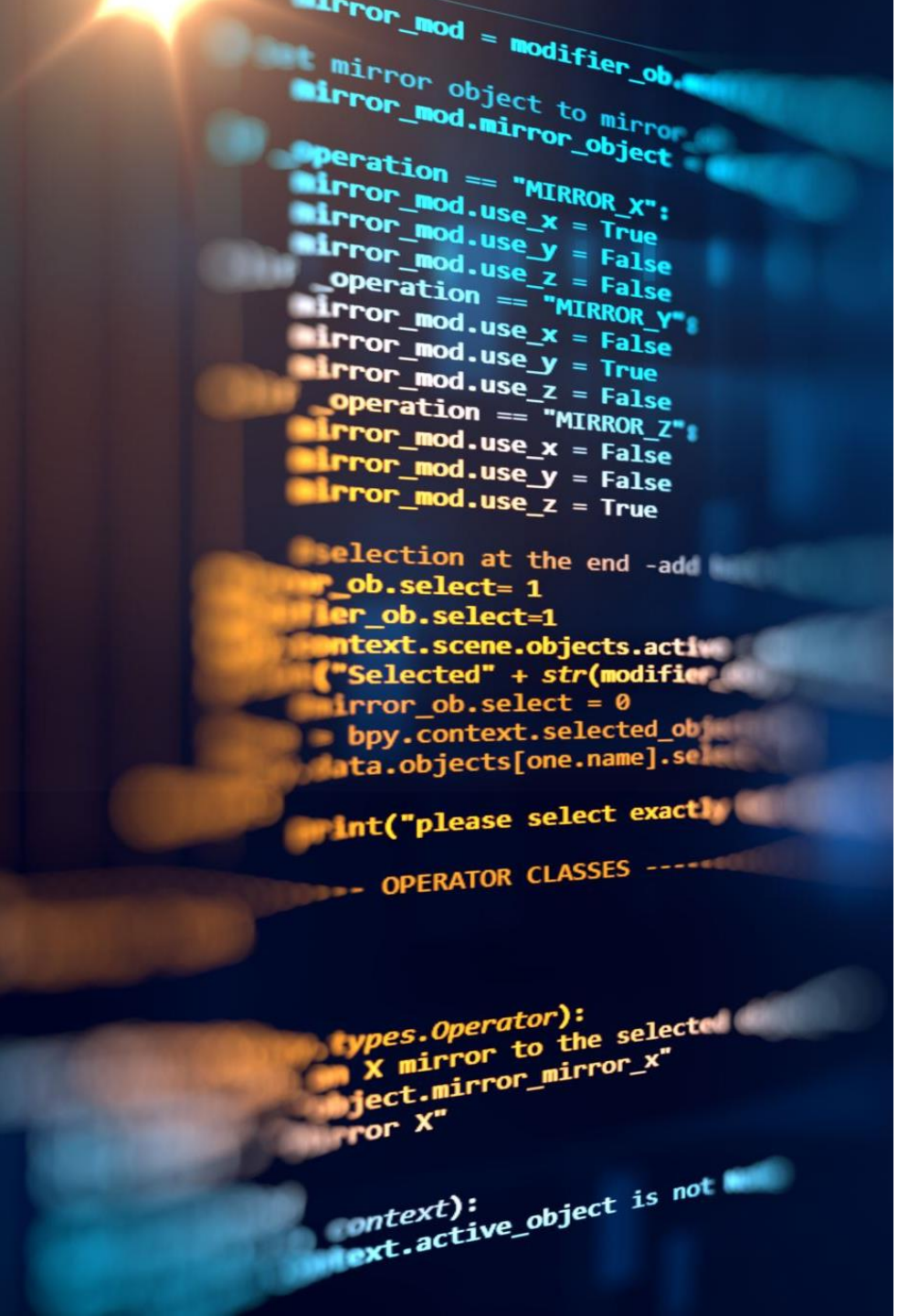
# Project Resources

- 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

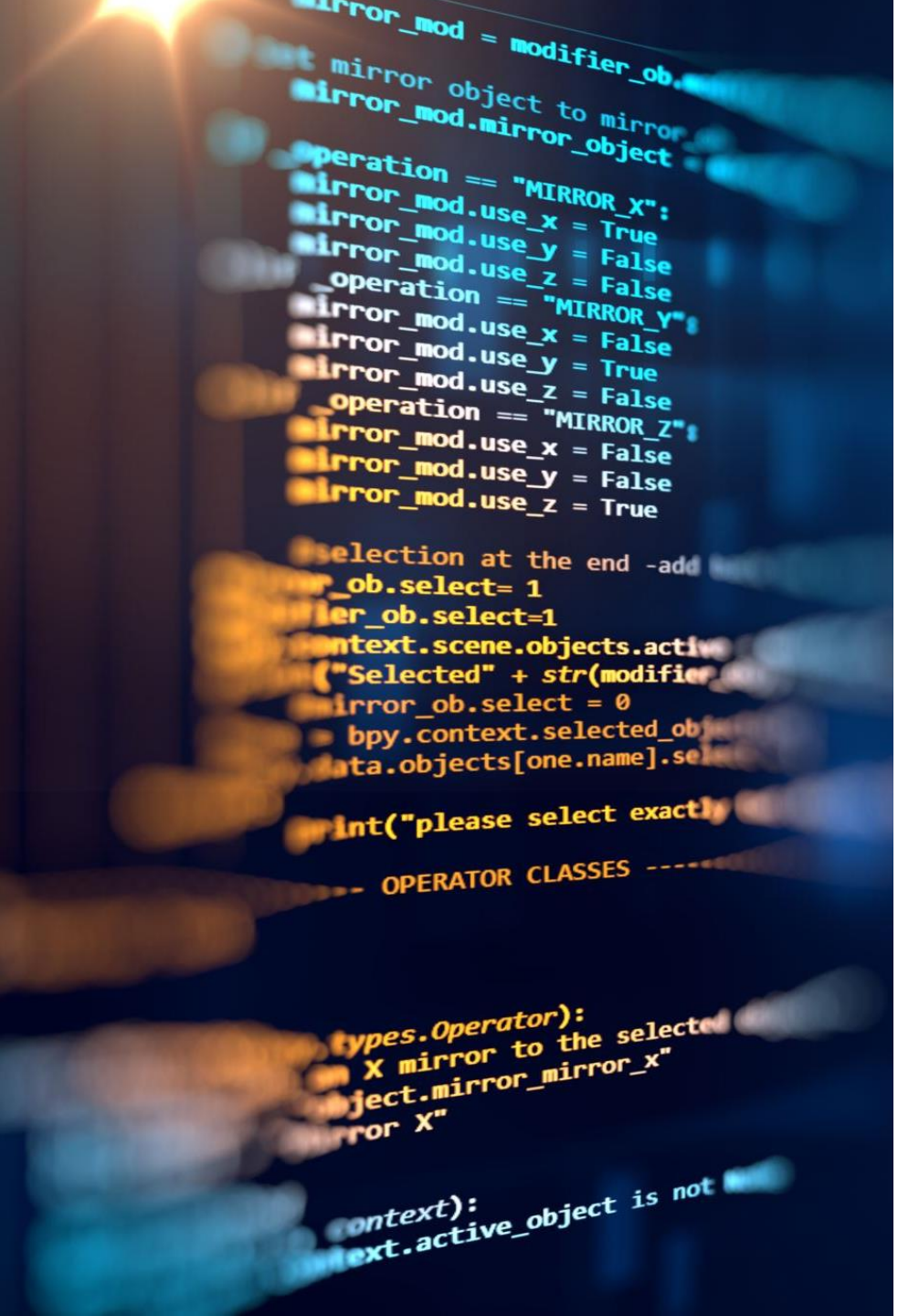
  rv = (RecyclerView) findViewById(R.id.recyclerview);
- String:
  In Java: R.string.title
  In XML: android:text="@string/title"

# Text and scrolling views

# TextView

# TextView for text

- **`TextView`** is `View` subclass for single and multi-line text
- **`EditText`** is `TextView` subclass with editable text
- Controlled with layout attributes
- Set text:
  - Statically from string resource in XML
  - Dynamically from Java code and any source



# Formatting text in string resource

- String resources: one unbroken line = one paragraph
- \n starts a new a line or paragraph
- Escape apostrophes and quotes with backslash (\", \')
- Escape any non-ASCII characters with backslash (\)

# Creating TextView in XML

```
<TextView android:id="@+id/textview"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/my_story"/>
```

# Common TextView attributes

`android:text`—text to display

`android:textColor`—color of text

`android:textAppearance`—predefined style or theme

`android:textSize`—text size in `sp`

`android:textStyle`—`normal`, `bold`, `italic`, or `bold|italic`

`android:typeface`—`normal`, `sans`, `serif`, or `monospace`

`android:lineSpacingExtra`—extra space between lines in `sp`

# Formatting active web links

```
<string name="article_text">... www.rockument.com ...</string>

<TextView
    android:id="@+id/article"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:autoLink="web"
    android:text="@string/article_text"/>
```

autoLink values:"web", "email", "phone", "map", "all"

Don’t use HTML for a web link in free-form text

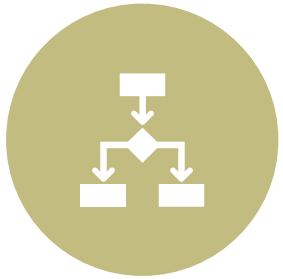
# ScrollView

# What about large amounts of text?

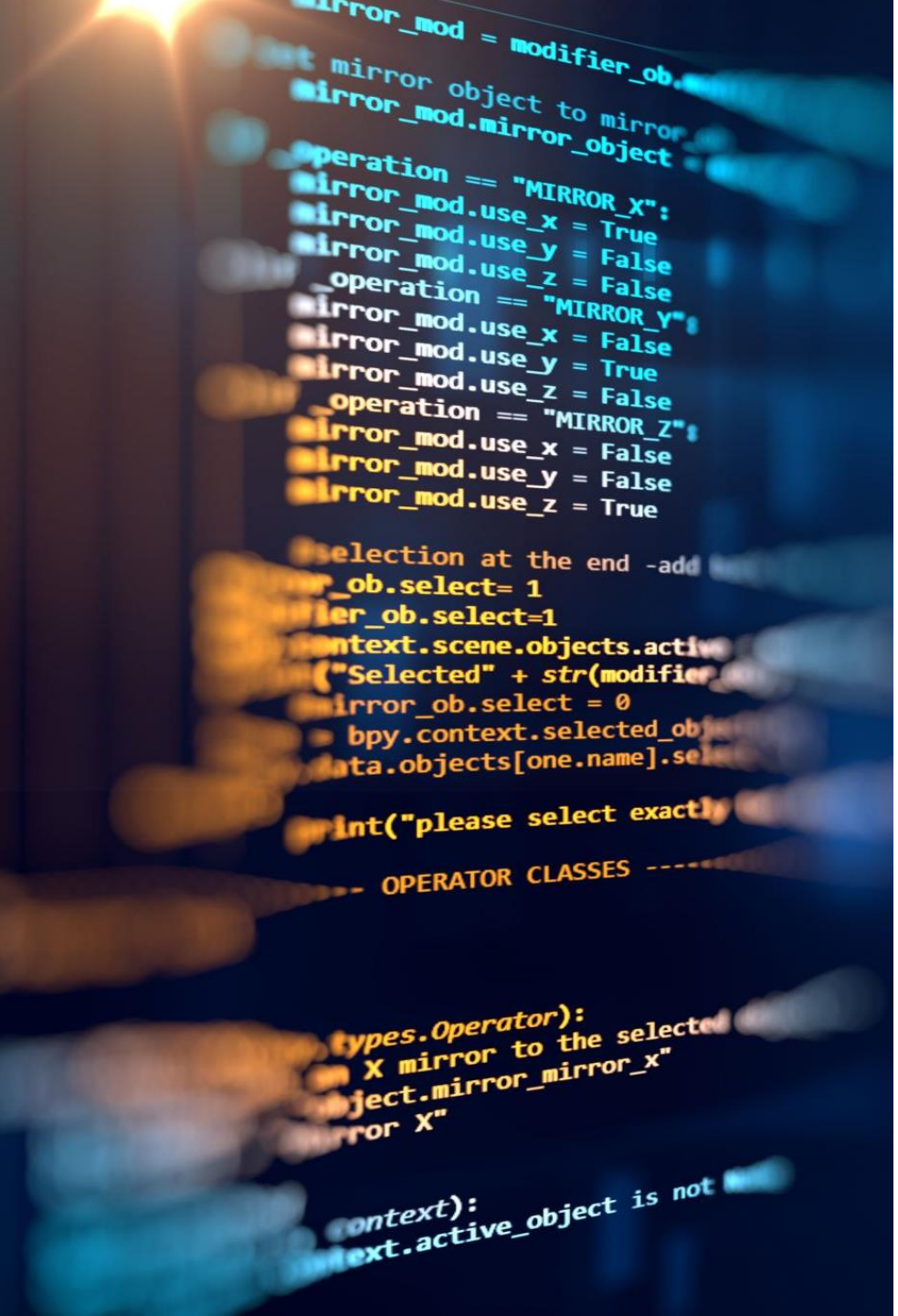
News stories, articles, etc…

To scroll a TextView, embed it in a **ScrollView**

Only *one* View element (usually TextView) allowed in a ScrollView



To scroll multiple elements, use one ViewGroup (such as LinearLayout) within the ScrollView

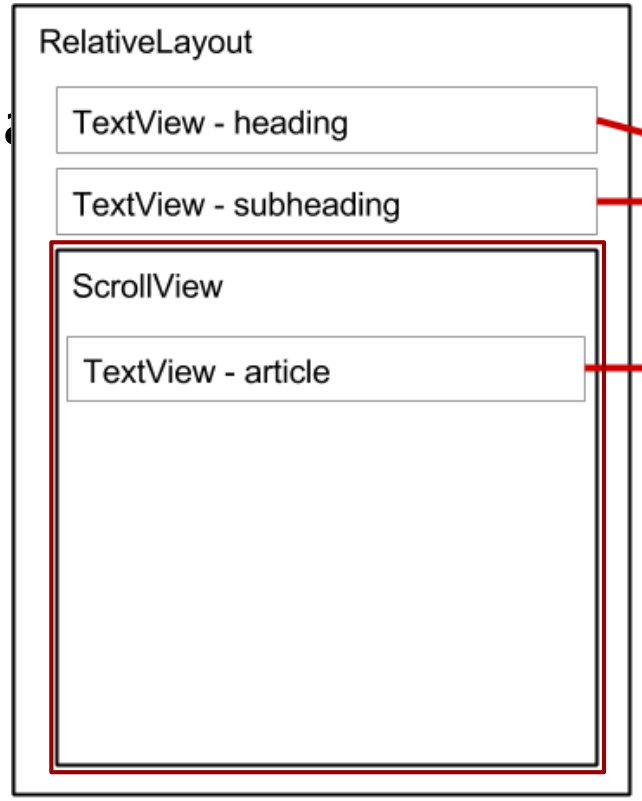


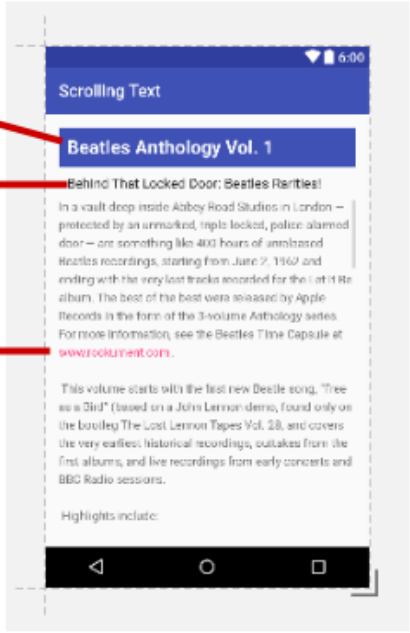
# ScrollView for scrolling content

- Holds all content in memory
- Not good for long texts, complex layouts
- Do not nest multiple scrolling views
- Use **HorizontalScrollView** for horizontal scrolling
- Use a **RecyclerView** for lists

# ScrollView layout with one TextView

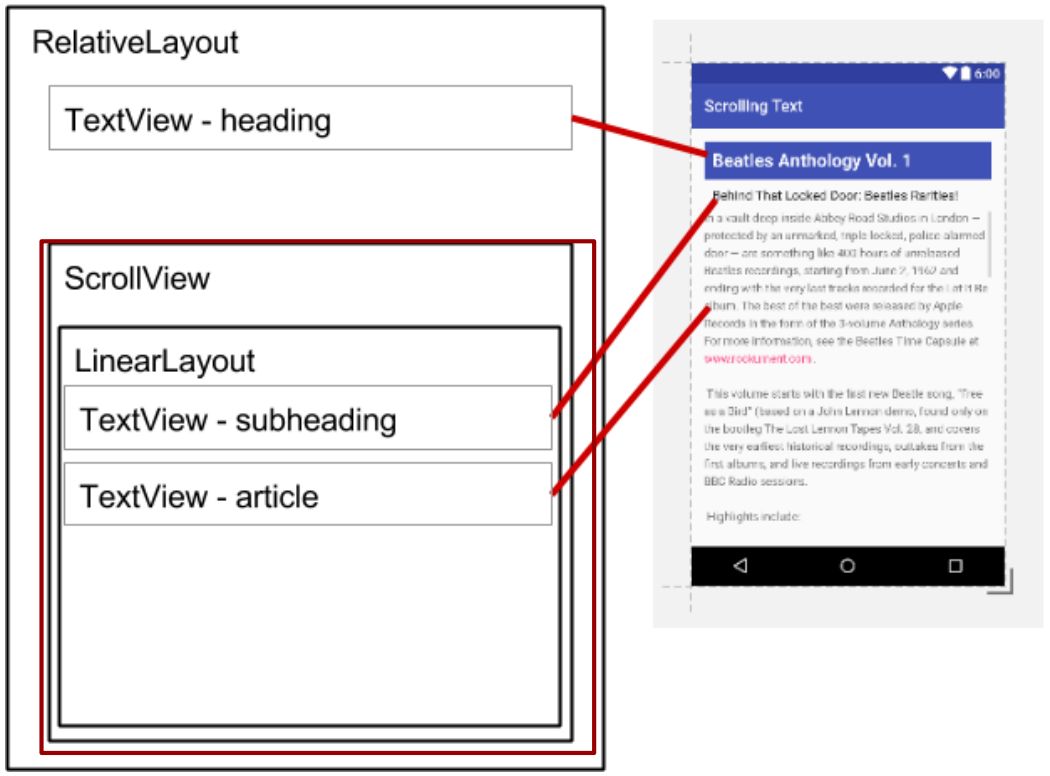
```
<ScrollView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:layout_below="@id/article_subhea

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    .../>

</ScrollView>
```



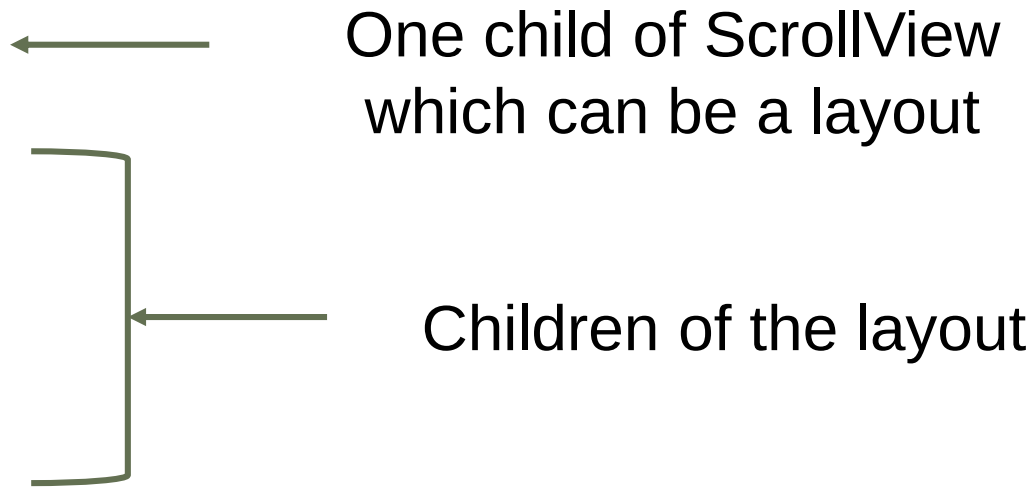
# ScrollView layout with a view group

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

        <TextView
            android:id="@+id/article_subheading"
            .../>

        <TextView
            android:id="@+id/article" ... />
    </LinearLayout>
</ScrollView>
```



# ScrollView with image and button

```
<ScrollView...>
    <LinearLayout...>
        <ImageView.../>
        <Button.../>
        <TextView.../>
    </LinearLayout>
</ScrollView>
```



# References

1. developer.android.com/courses/adf-v2