Android Studio Views & Layouts

Asst. Prof. Monlica Wattana, Ph.D
Department of Computer Science,
Khon Kaen University

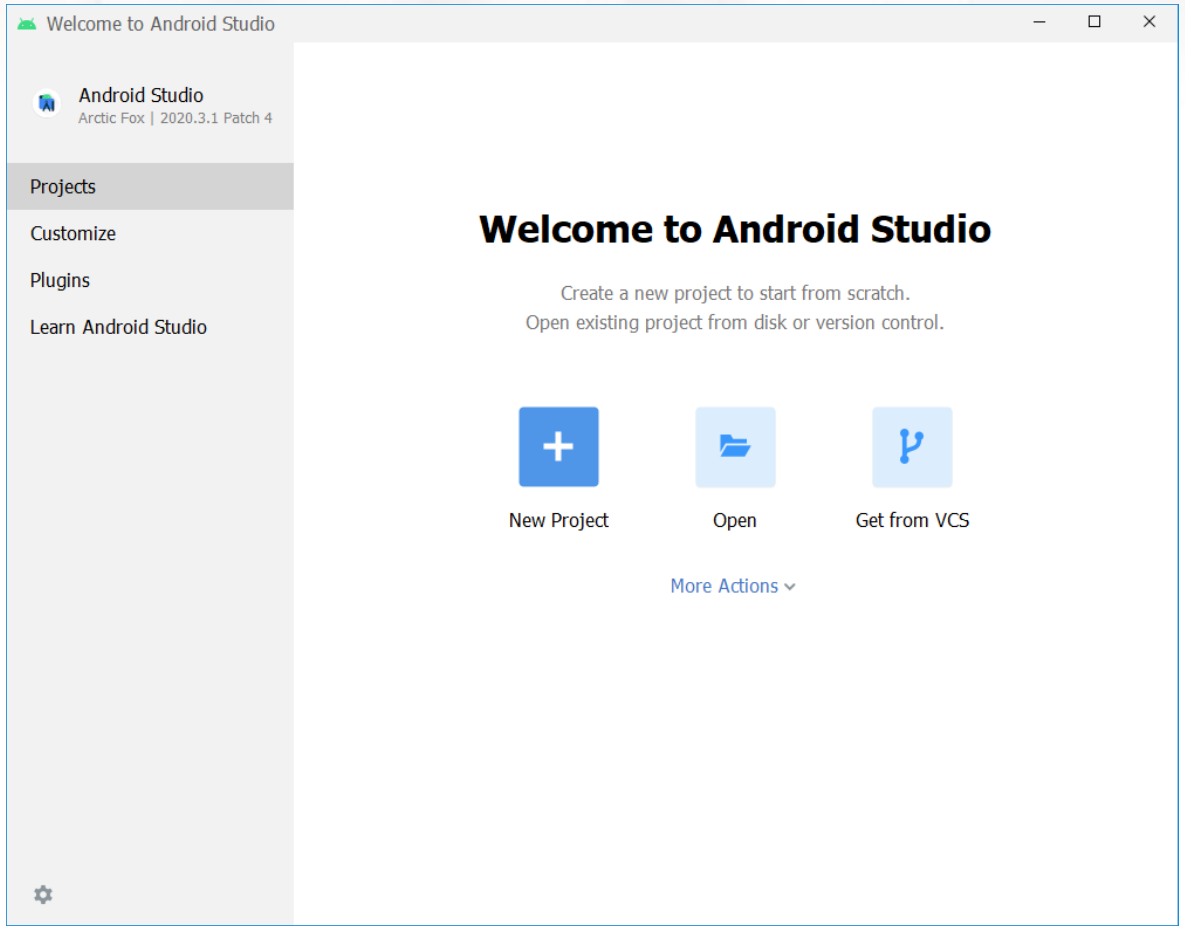
SC 362 007 Mobile Device Programming

Outline

- Android Studio
- Anatomy of an Android App project
- Views & Layout
- XML- eXtensible Markup Language
- XML and Layouts

Android Studio

New Project



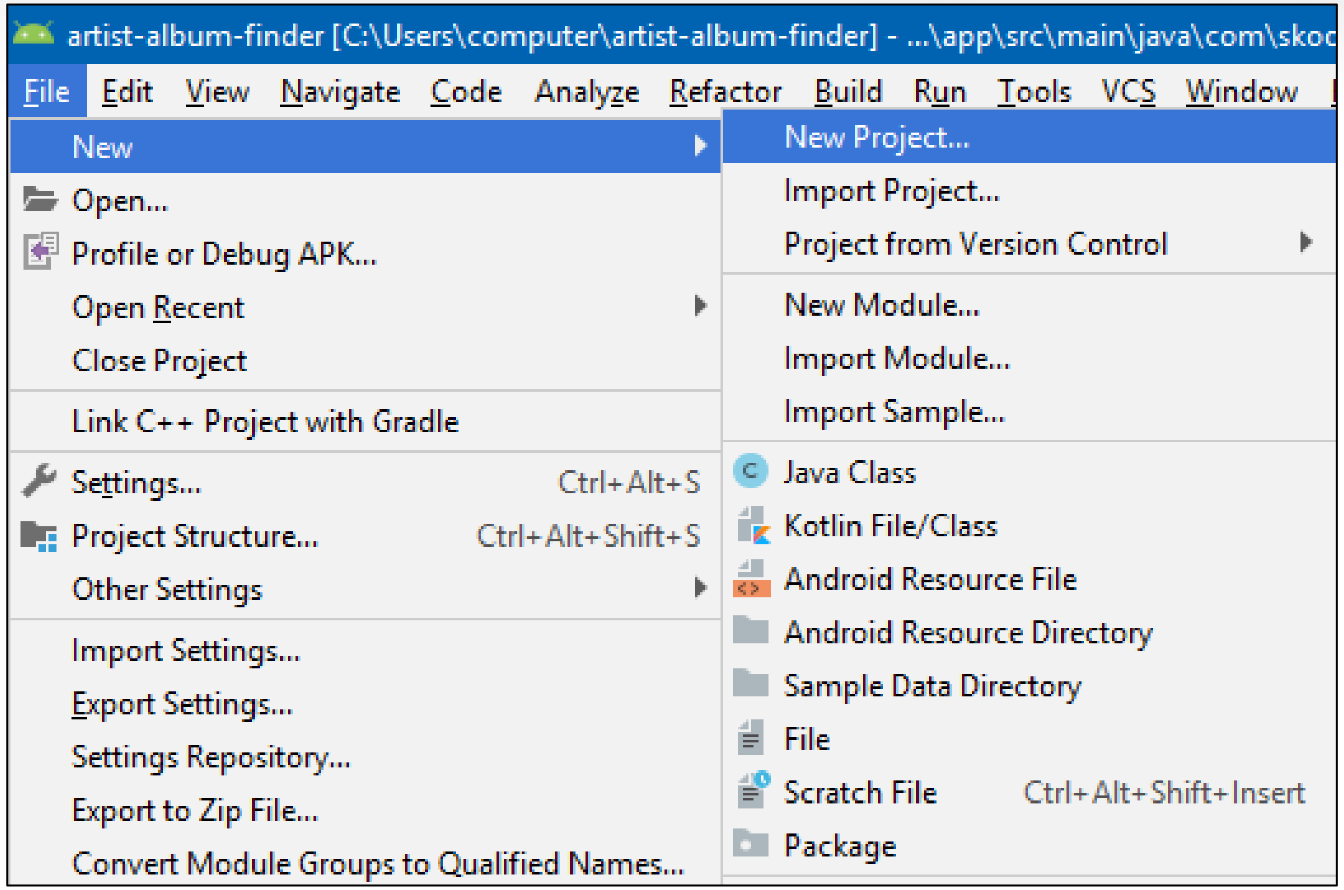
Welcome to Android Studio

Android Studio
Arctic Fox | 2020.3.1 Patch 4

Projects
Customize
Plugins
Learn Android Studio

Welcome to Android Studio

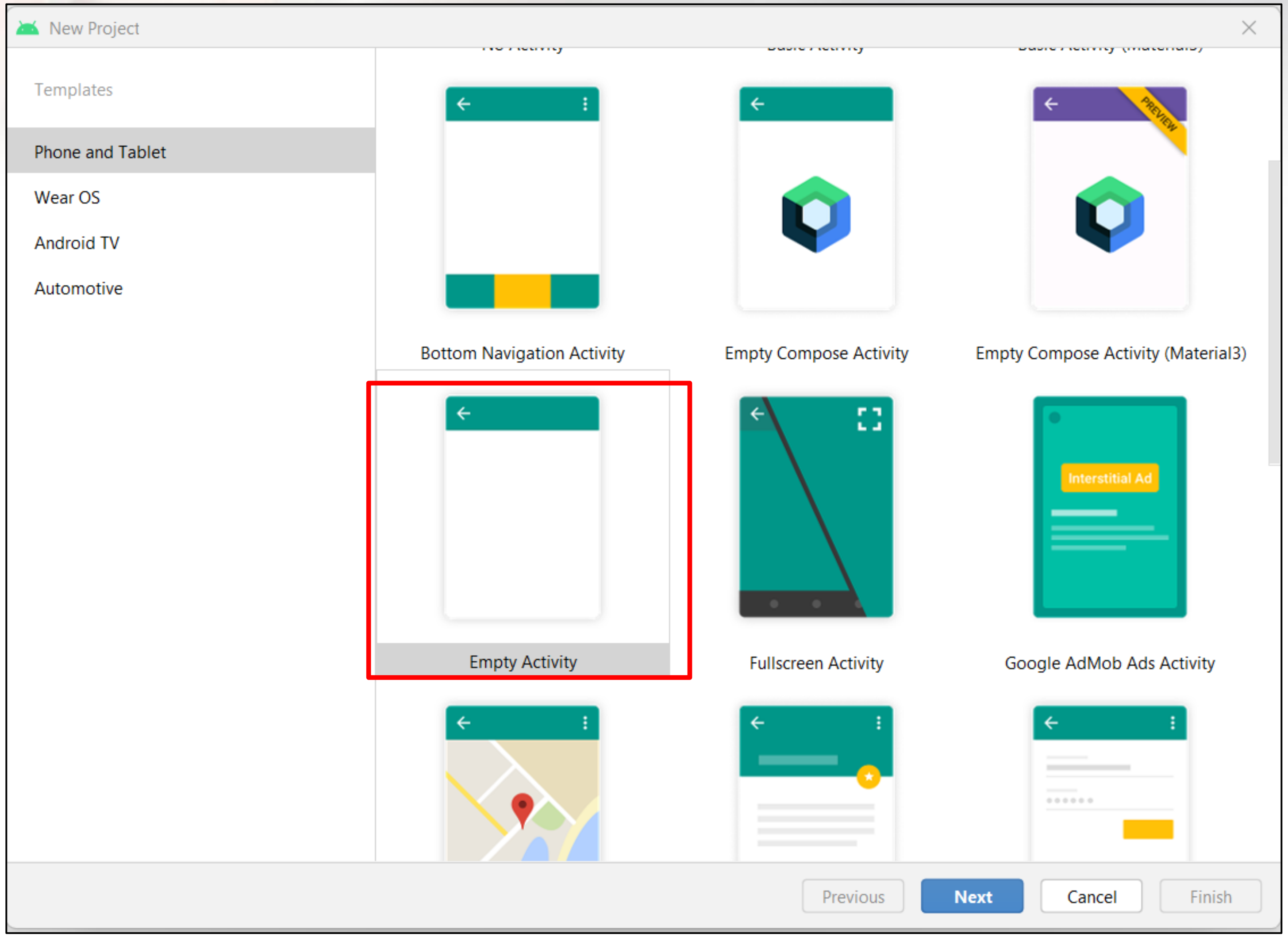
Create a new project to start from scratch.
Open existing project from disk or version control.



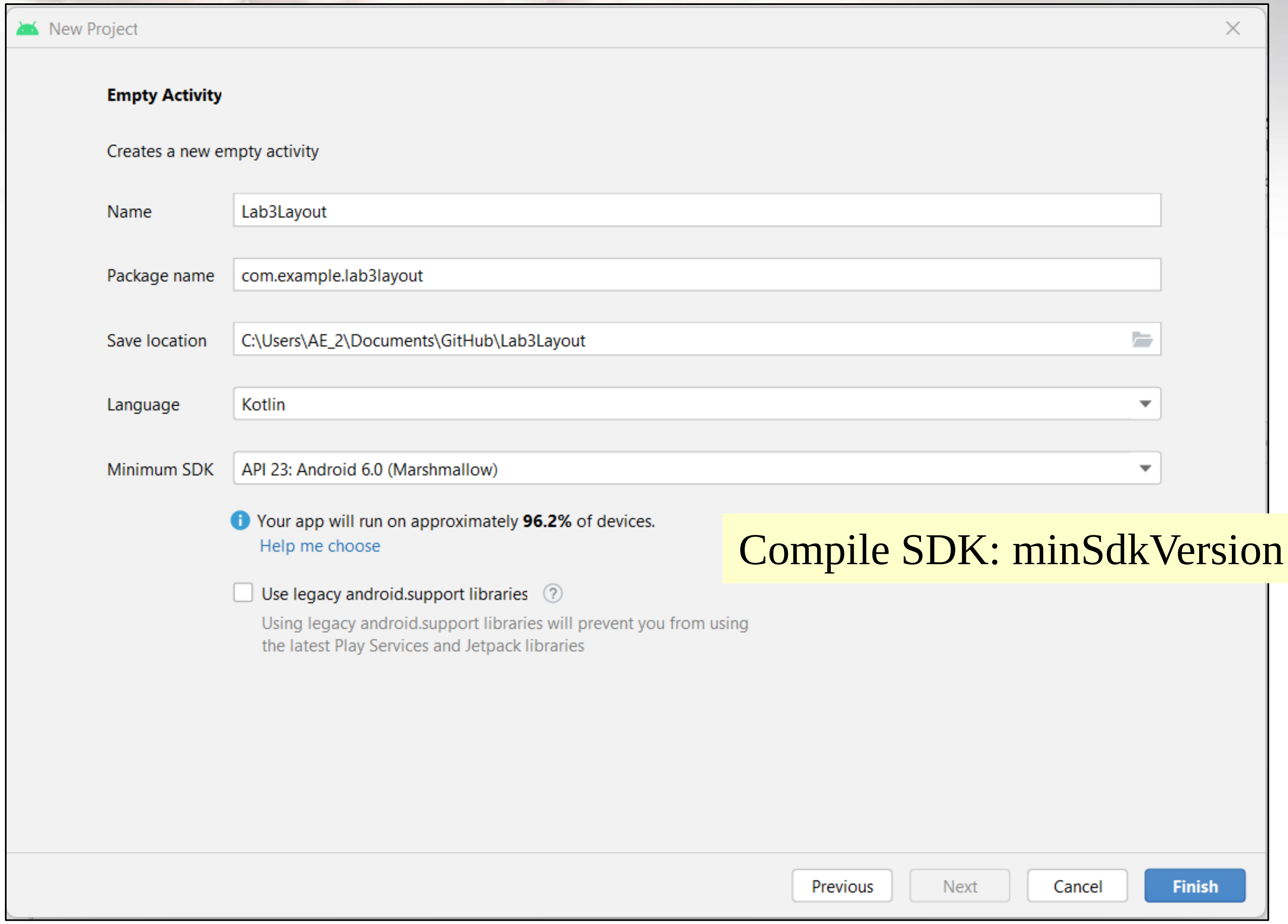
artist-album-finder [C:\Users\computer\artist-album-finder] - ...\app\src\main\java\com\skoc

Android Studio

New Project



Android Studio



Empty Activity

Creates a new empty activity

Name: Lab3Layout

Package name: com.example.lab3layout

Save location: C:\Users\AE_2\Documents\GitHub\Lab3Layout

Language: Kotlin

Minimum SDK: API 23: Android 6.0 (Marshmallow)

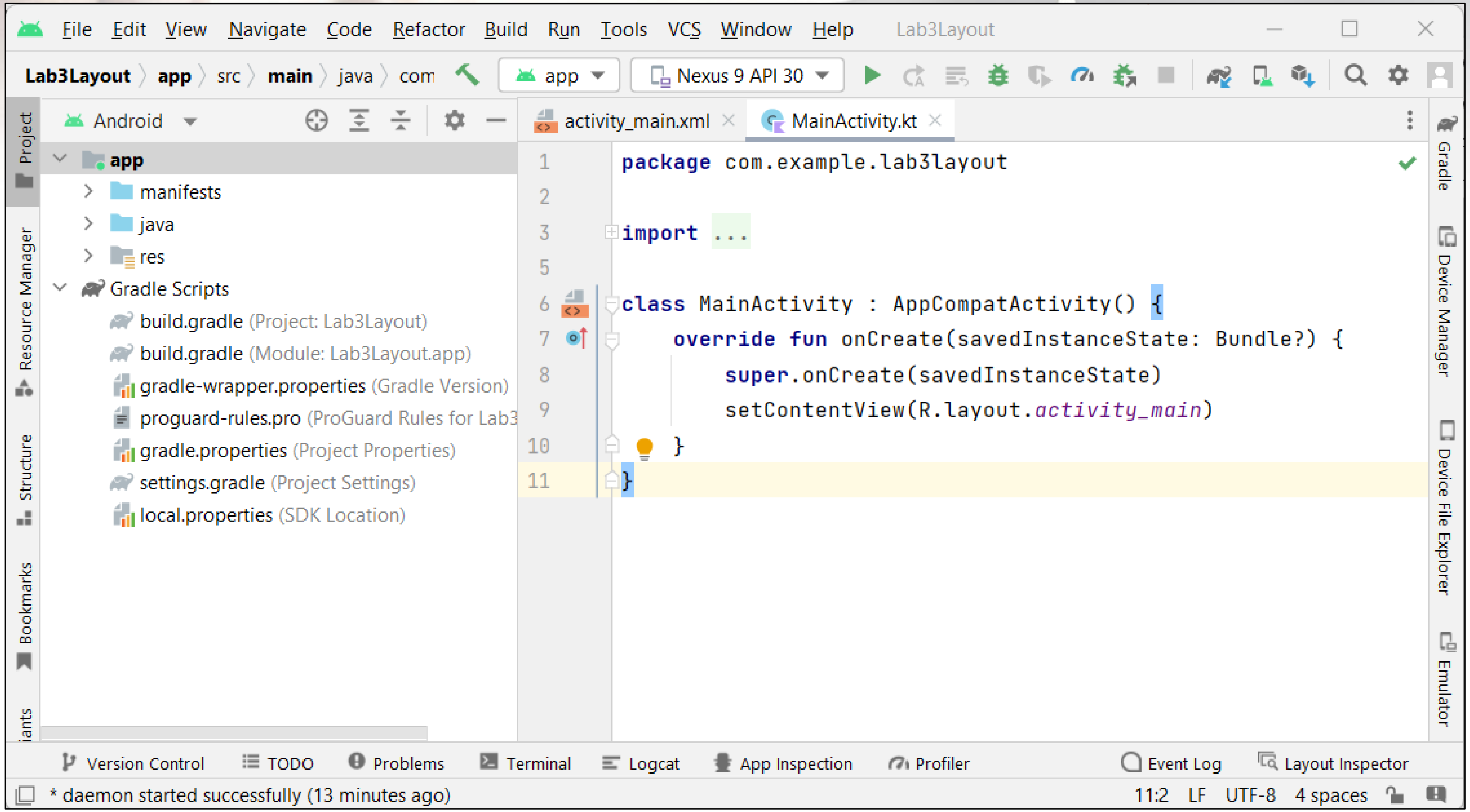
Your app will run on approximately **96.2%** of devices.
Help me choose

[ ] Use legacy android.support libraries

Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

Compile SDK: minSdkVersion <= targetSdkVersion <= compileSdk Version

Android Studio



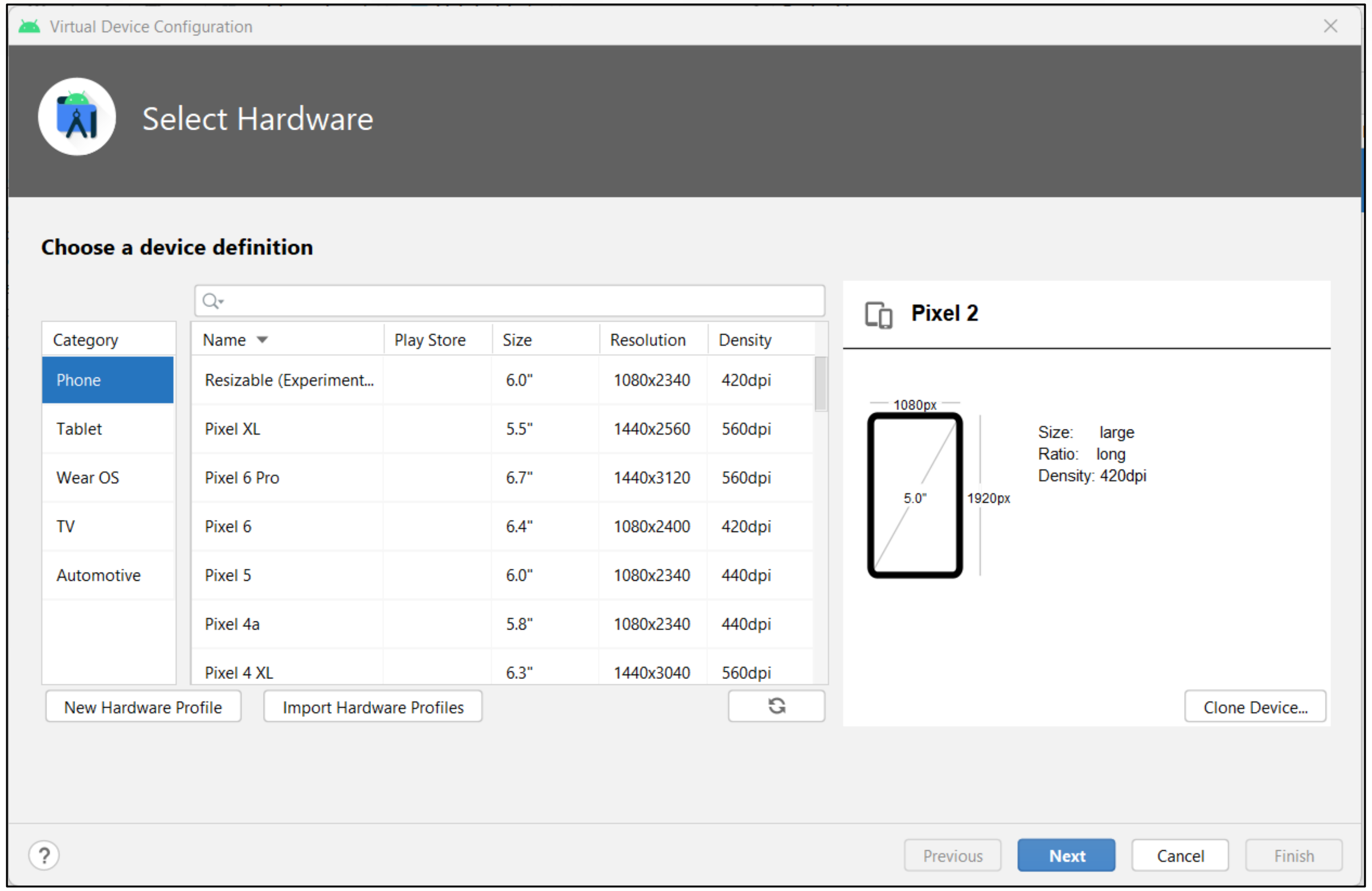
```
app
  manifests
  java
  res
Gradle Scripts
  build.gradle (Project: Lab3Layout)
  build.gradle (Module: Lab3Layout.app)
  gradle-wrapper.properties (Gradle Version)
  proguard-rules.pro (ProGuard Rules for Lab3
  gradle.properties (Project Properties)
  settings.gradle (Project Settings)
  local.properties (SDK Location)
```

```kotlin
package com.example.lab3layout

import ...

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

* daemon started successfully (13 minutes ago) 11:2 LF UTF-8 4 spaces

Run Apps

- Android Device
- Emulator: Android Virtual Device (AVD) Manager



Virtual Device Configuration

Select Hardware

Choose a device definition

Category
Phone
Tablet
Wear OS
TV
Automotive

Name	Play Store	Size	Resolution	Density
Resizable (Experiment...		6.0"	1080x2340	420dpi
Pixel XL		5.5"	1440x2560	560dpi
Pixel 6 Pro		6.7"	1440x3120	560dpi
Pixel 6		6.4"	1080x2400	420dpi
Pixel 5		6.0"	1080x2340	440dpi
Pixel 4a		5.8"	1080x2340	440dpi
Pixel 4 XL		6.3"	1440x3040	560dpi

Pixel 2

1080px
5.0"
1920px

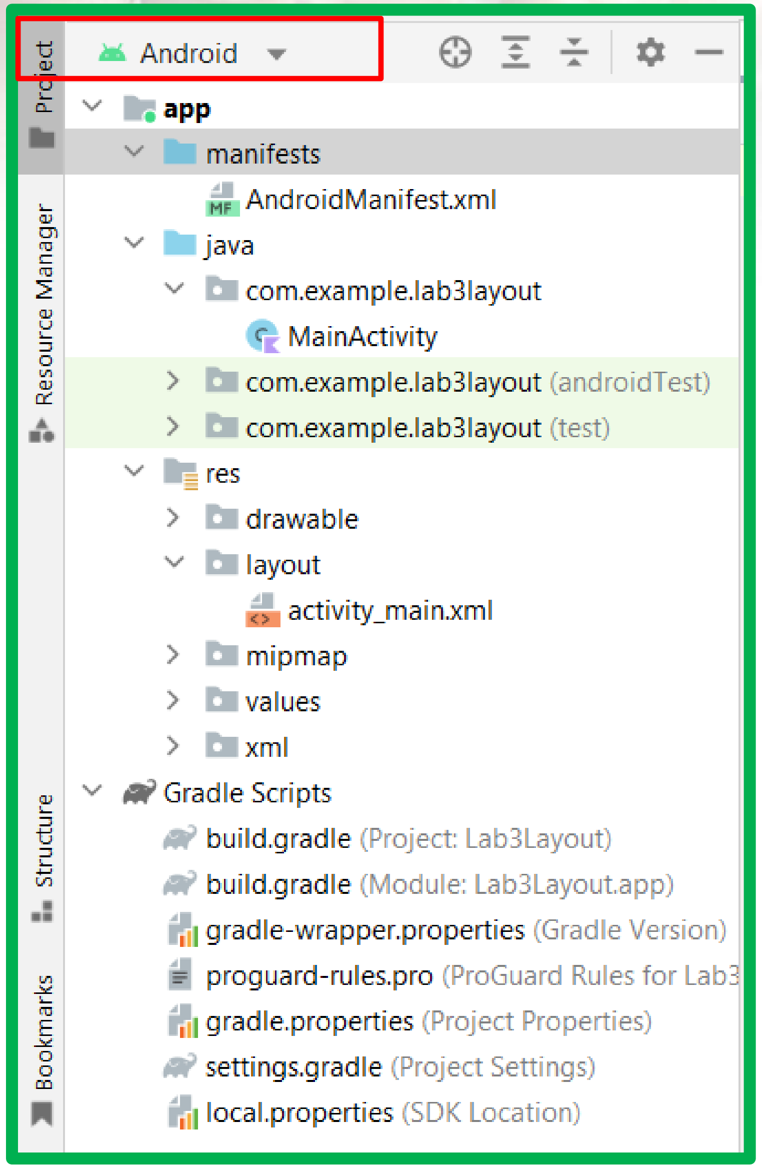
Size: large
Ratio: long
Density: 420dpi

Anatomy of an Android App project

- Activity
- Resources
- Gradle files

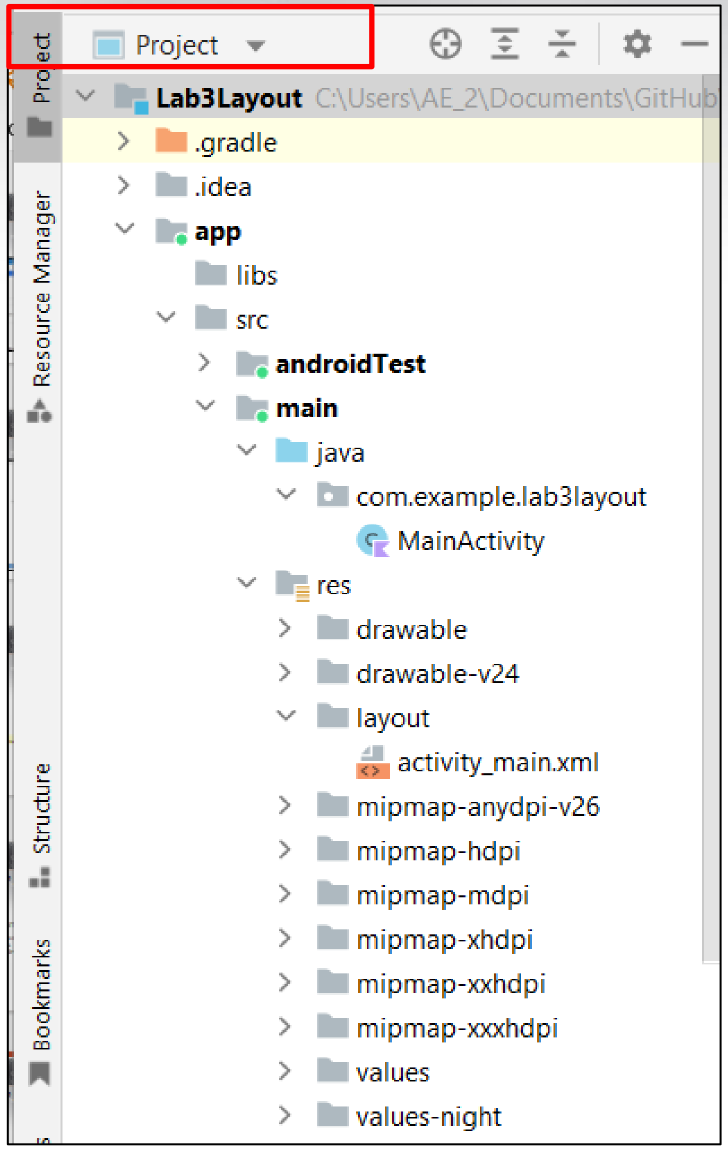
```
MyApplication
├── app
│   ├── libs
│   └── src
│       ├── androidTest
│       ├── main
│       │   ├── java
│       │   ├── res
│       │   └── AndroidManifest.xml
│       └── test
├── build.gradle
└── gradlew
```

Anatomy of an Android App project



Android

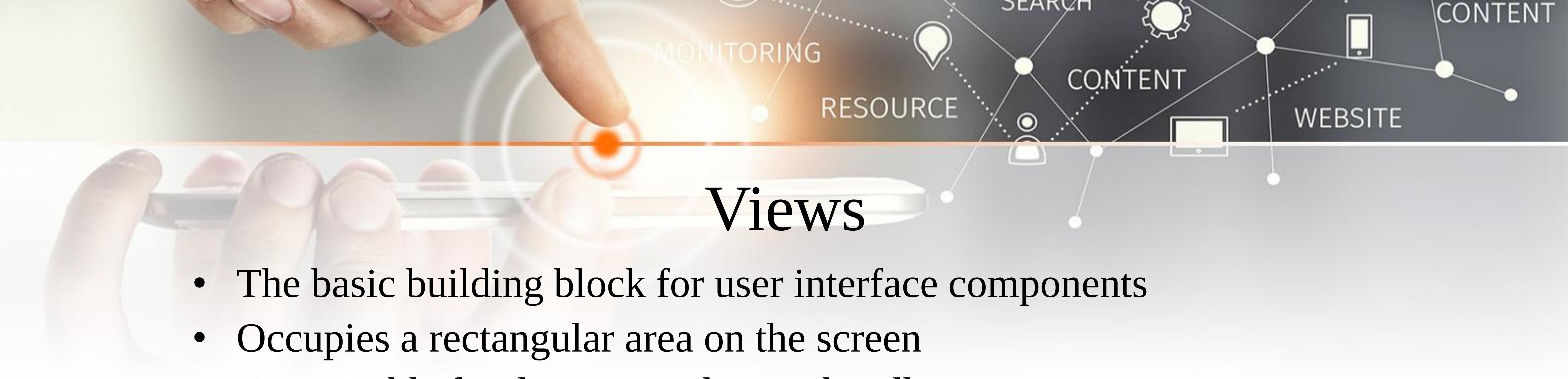
- app
  - manifests
    - AndroidManifest.xml
  - java
    - com.example.lab3layout
      - MainActivity
    - com.example.lab3layout (androidTest)
    - com.example.lab3layout (test)
  - res
    - drawable
    - layout
      - activity_main.xml
    - mipmap
    - values
    - xml
- Gradle Scripts
  - build.gradle (Project: Lab3Layout)
  - build.gradle (Module: Lab3Layout.app)
  - gradle-wrapper.properties (Gradle Version)
  - proguard-rules.pro (ProGuard Rules for Lab3
  - gradle.properties (Project Properties)
  - settings.gradle (Project Settings)
  - local.properties (SDK Location)



Project

- Lab3Layout C:\Users\AE_2\Documents\GitHub
  - .gradle
  - .idea
  - app
    - libs
    - src
      - androidTest
      - main
        - java
          - com.example.lab3layout
            - MainActivity
        - res
          - drawable
          - drawable-v24
          - layout
            - activity_main.xml
          - mipmap-anydpi-v26
          - mipmap-hdpi
          - mipmap-mdpi
          - mipmap-xhdpi
          - mipmap-xxhdpi
          - mipmap-xxxhdpi
          - values
          - values-night

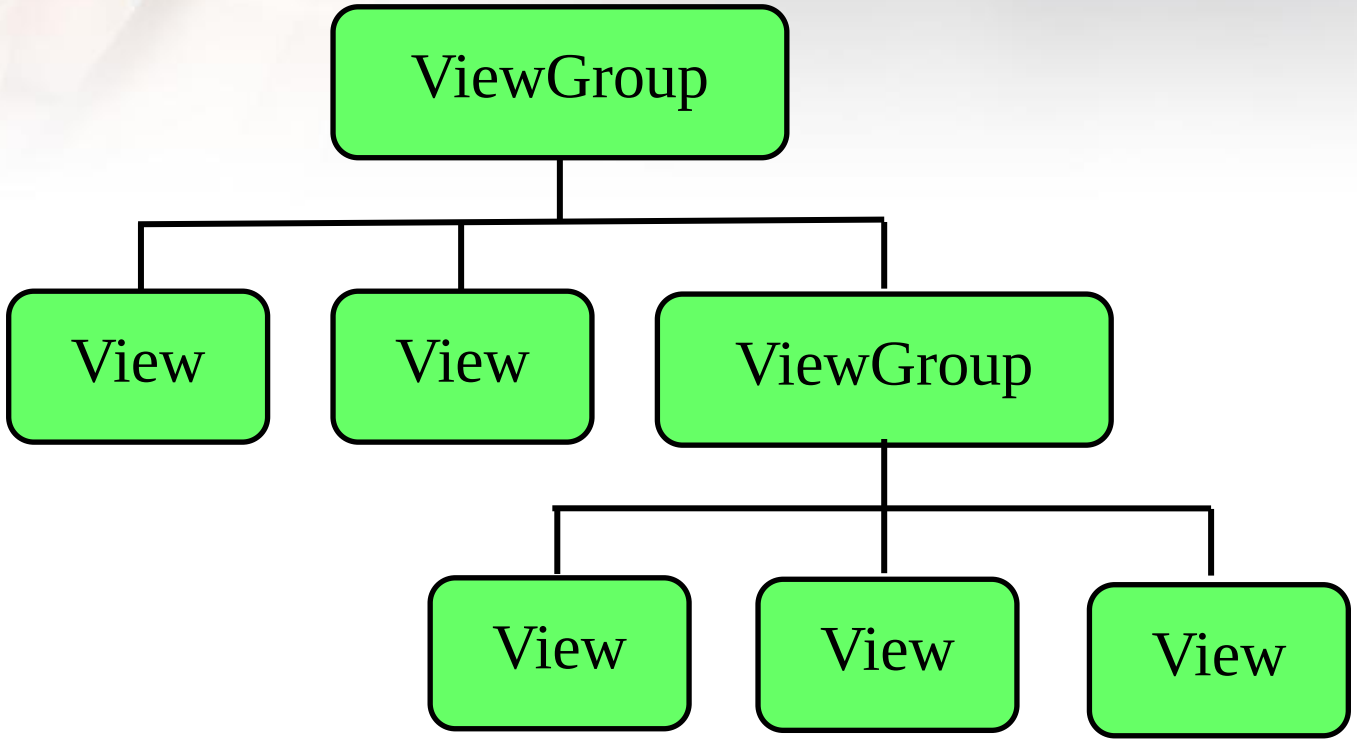
Views & Layout



Views

- The basic building block for user interface components
- Occupies a rectangular area on the screen
- Responsible for drawing and event handling
- The visual content of the window is provided by a hierarchy of views
- Parent views contain and organize the layout of their children
- Leaf views draw in the rectangles, control and respond to user actions directed at that space
- Activity's UI is defined by a hierarchy of View and ViewGroup nodes
- setContentView() - attaches the view hierarchy tree to the screen for rendering

View Hierarchy

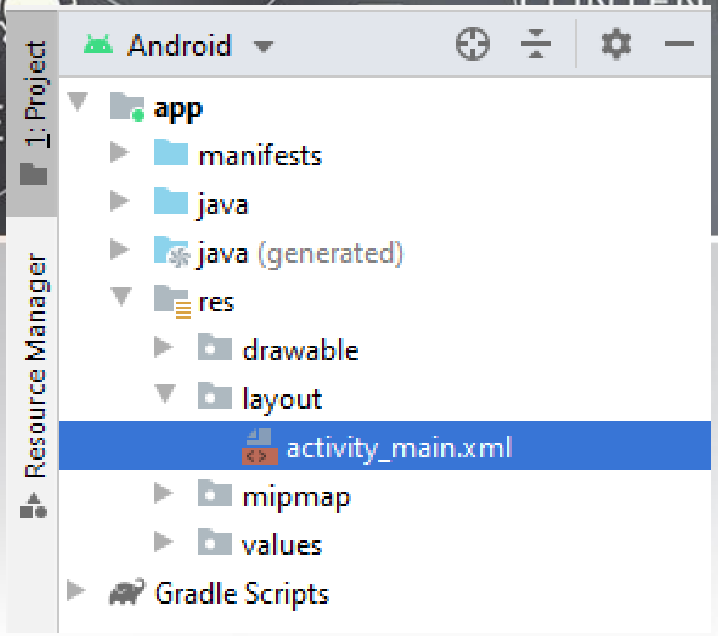


- Each element in XML is either a View or ViewGroup object
- View objects are leaves in the tree
- ViewGroup objects are branches in the tree

Layout

- Layout is the architecture for the user interface in an Activity
- Defines the layout structure and holds all the elements that appear to the user
- Express the view hierarchy

Layout



Layout is declared in two ways:

1. Instantiate layout elements at runtime

- Write code using SDK with classes like LinearLayout, TextView

2. Declare UI elements in XML

– create XML files in res/Layout (i.e. activity_main.xml) that contain Android XML view tags like <LinearLayout> <TextView>, etc.

– File path: app>> res >>layout >> activity_main.xml

Advantage to declaring UI in XML

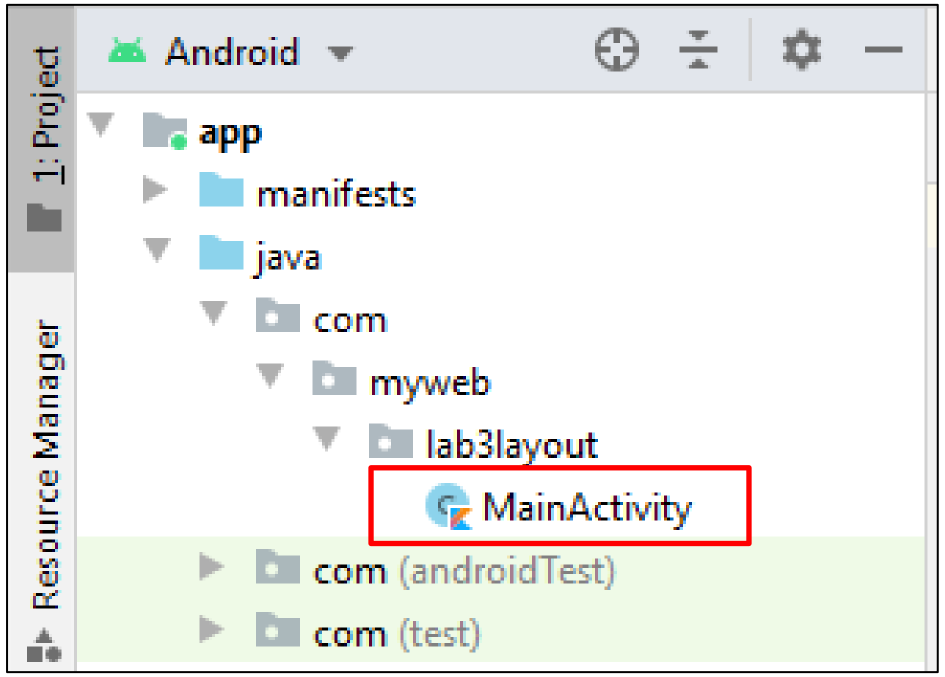
- Better separate the presentation of the application from the code that controls its behavior
- Can modify or adapt it without having to modify the source code and recompile
  - Create XML layouts for different screen orientations
  - Create XML layouts for different device screen sizes
  - Create XML layouts for different languages
- Makes it easier to visualize the structure of UI
- Easier to debug problems

Load the XML Resource

- The XML layout file is compiled into a View resource on MainActivity.kt file
- The layout resource is loaded in the Activity.onCreate() method
- The layout resource is loaded by calling setContentView() and passing the reference to the layout resource
- The layout resource reference is R.layout.*layout_file_name*
- Ex : setContentView(*R.layout.activity_main*)

Load the XML Resource

File path: app >> java >>com>> *package name*>> MainActivity.kt



```
override fun onCreate(savedInstanceState: Bundle?) {
     super.onCreate(savedInstanceState)
     setContentView(R.layout.activity_main)
  }
```

XML- eXtensible Markup Language

XML - Example

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity">
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Hello World!"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintLeft_toLeftOf="parent"
      app:layout_constraintRight_toRightOf="parent"
      app:layout_constraintTop_toTopOf="parent“
 />
</androidx.constraintlayout.widget.ConstraintLayout>
```

prolog

start tag

attribute name

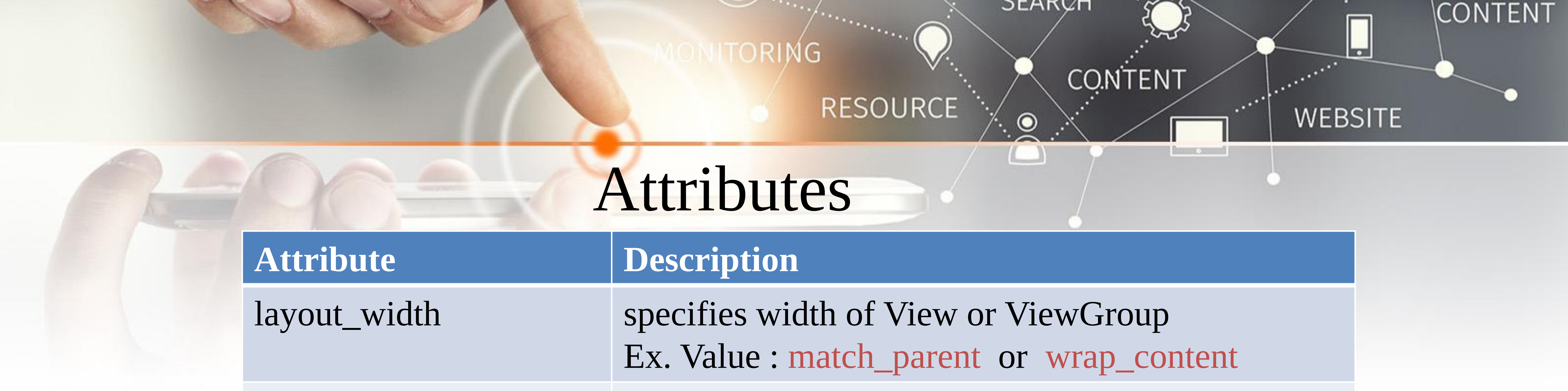
attribute value

empty tag
(no end tag)

end tag

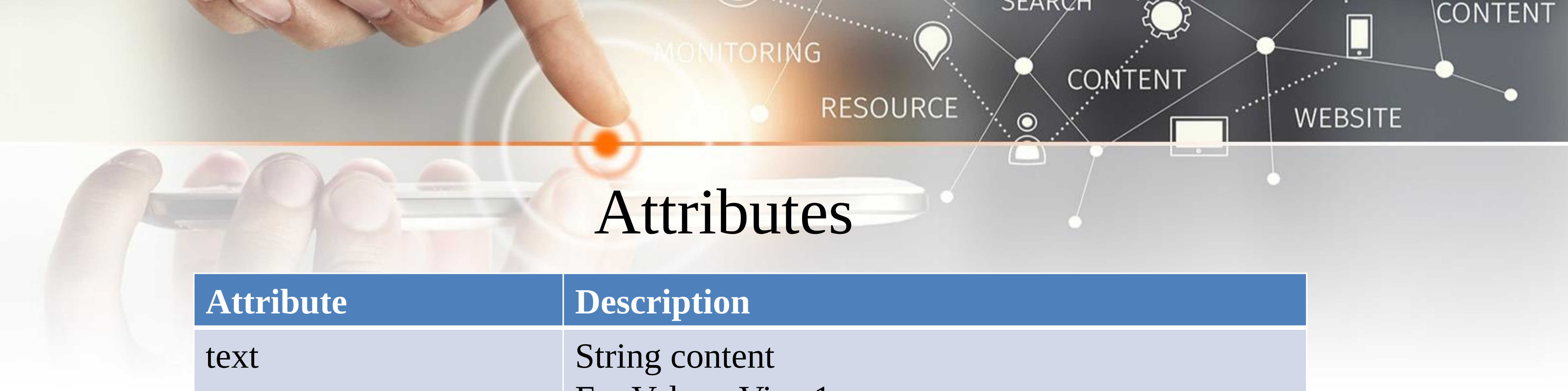
Attributes

- Every View and ViewGroup object supports their own variety of XML attributes
- Some attributes are specific to a View object , these attributes are inherited by any View objects that extend this class
- Other attributes are considered "layout parameters" that describe certain layout orientations of the View object



Attributes

Attribute	Description
layout_width	specifies width of View or ViewGroup Ex. Value : match_parent or wrap_content
layout_height	specifies height Ex. Value : 100dp
layout_marginTop	extra space on top Ex. Value : 100dp
layout_marginBottom	extra space on bottom side Ex. Value : 100dp
layout_marginLeft	extra space on left side Ex. Value : 100dp
layout_marginRight	extra space on right side Ex. Value : 100dp



Attributes

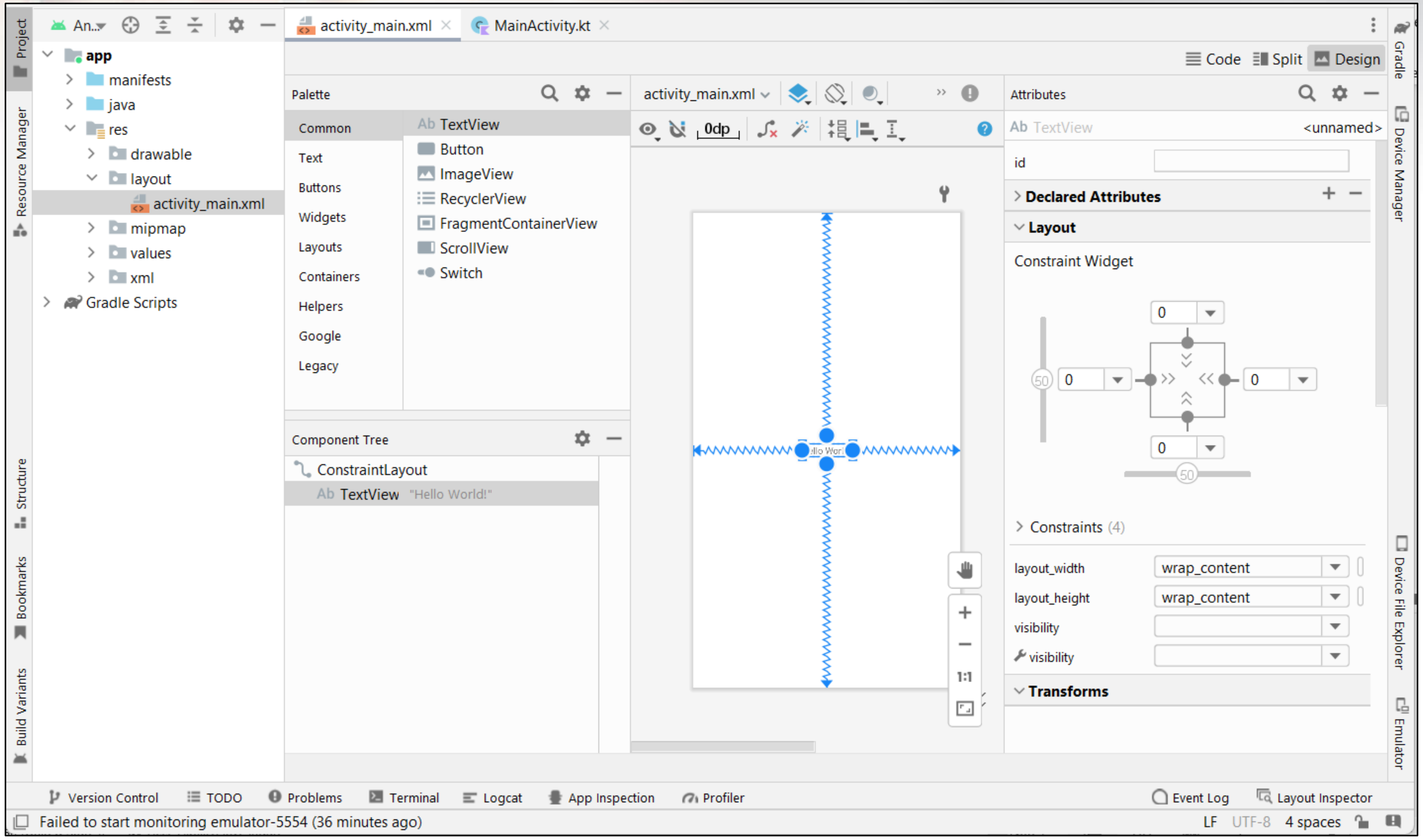
Attribute	Description
text	String content Ex. Value : View1
background	Background: color or image Ex. Value :#FFF5
textSize	Text Size Ex. Value : 20sp

Dimension

- Dimensions in any of the following units:
  - *px*: Pixels
  - *in*: Inches
  - *mm*: Millimeters
  - *pt*: Points
  - *dp*: Density-independent pixels based on a 160-dpi (pixel density per inch) screen (dimensions adjust to screen density)
  - *sp*: Scale-independent pixels (dimensions that allow for user sizing; helpful for use in fonts)

XML AND LAYOUTS

Layout Editor



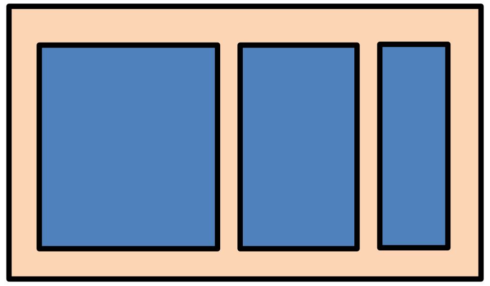
layout_width: wrap_content
layout_height: wrap_content
visibility:
visibility:

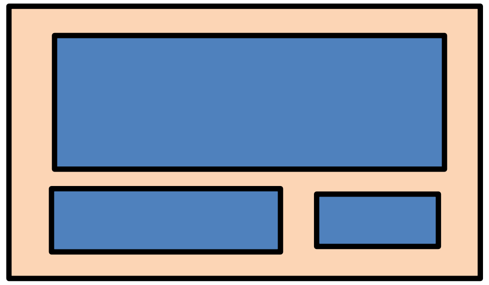
Failed to start monitoring emulator-5554 (36 minutes ago)

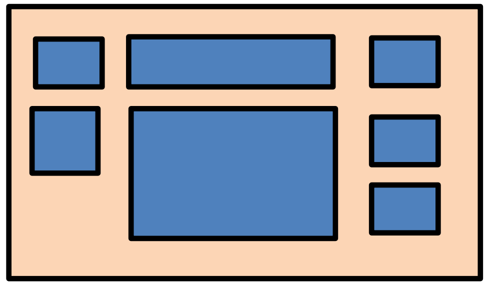
ViewGroup

Controls location of Views in that ViewGroup

- Constrain Layout
- LinearLayout
- RelativeLayout
- TableLayout


LinearLayout

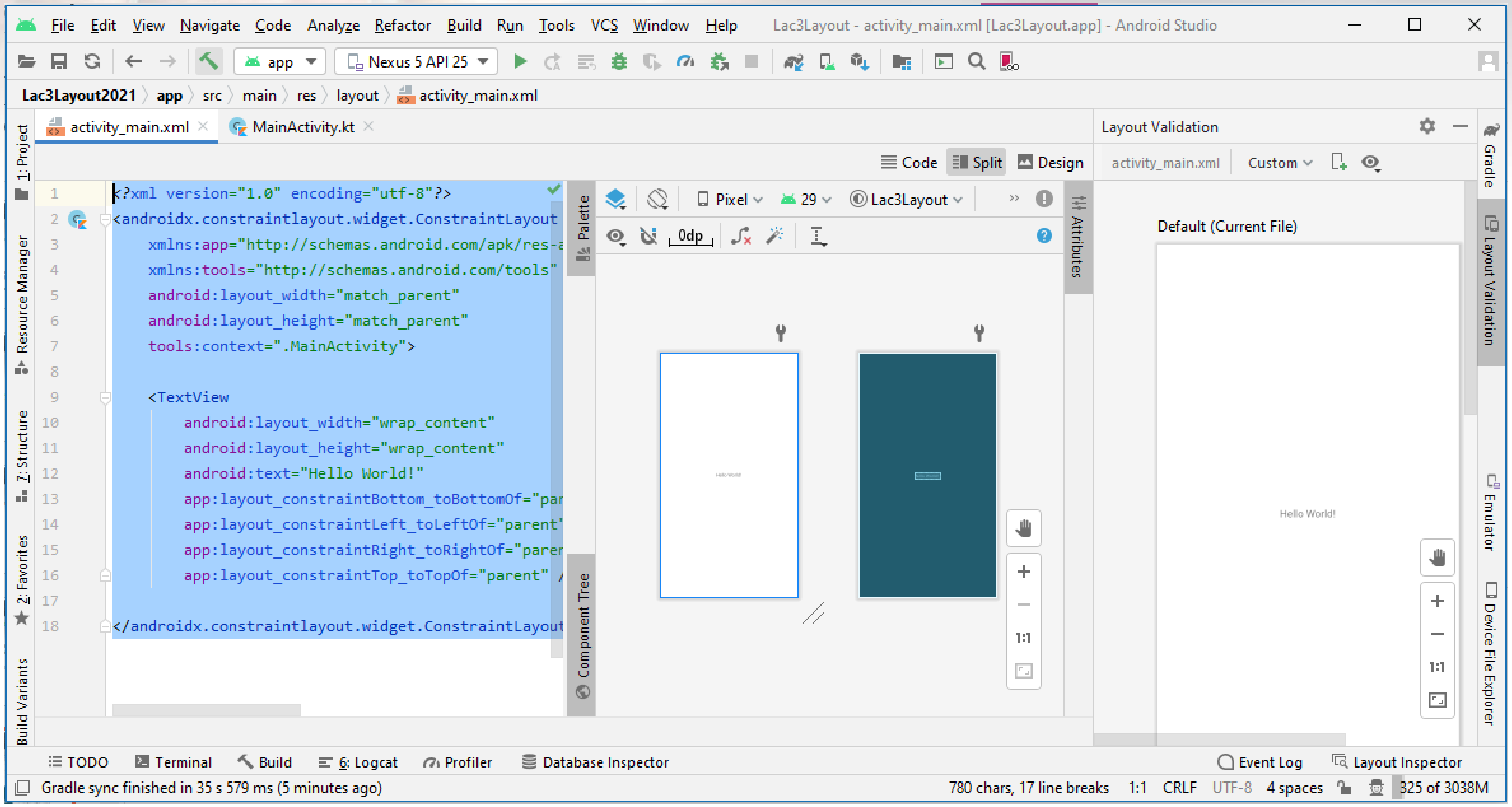

RelativeLayout


TableLayout

Constraint Layout

- New layout supported by Google
- A layout is a ViewGroup that is responsible for positioning its childViews. It calculates and set the position and size of those Views
- Allows you to create large and complex layouts with a flat view hierarchy (no nested view groups)

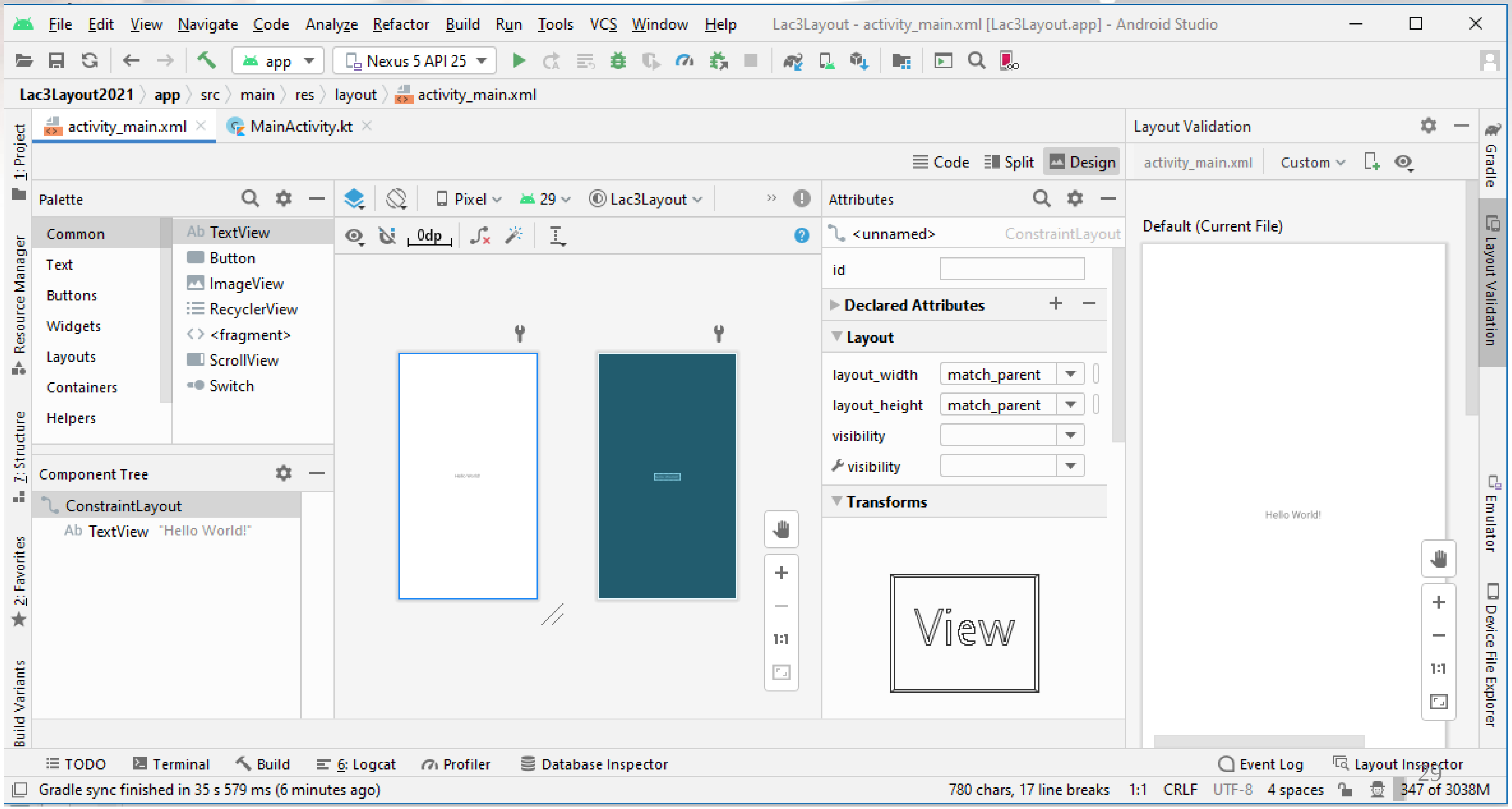
Constraint Layout



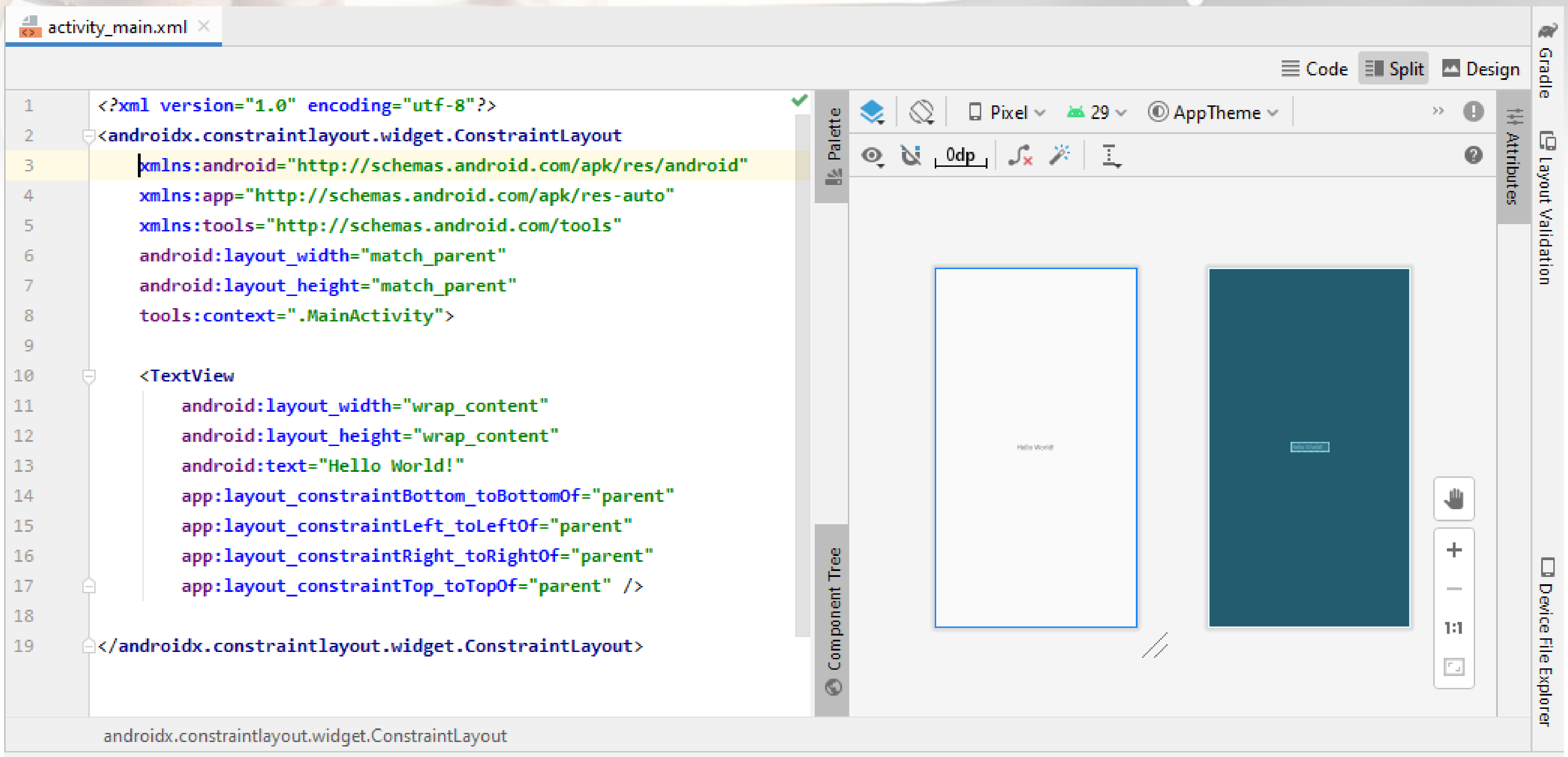
```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:app="http://schemas.android.com/apk/res-a
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="par
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parer
        app:layout_constraintTop_toTopOf="parent" /

</androidx.constraintlayout.widget.ConstraintLayout
```

Gradle sync finished in 35 s 579 ms (5 minutes ago)   780 chars, 17 line breaks   1:1   CRLF   UTF-8   4 spaces   325 of 3038M

Constraint Layout



Constraint Layout



activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

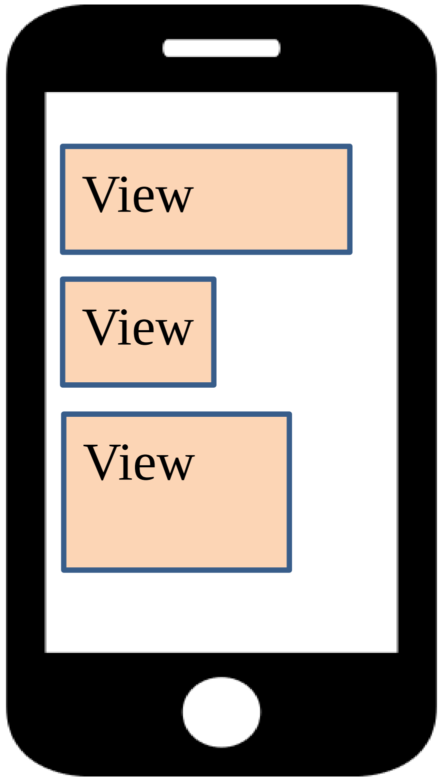
androidx.constraintlayout.widget.ConstraintLayout

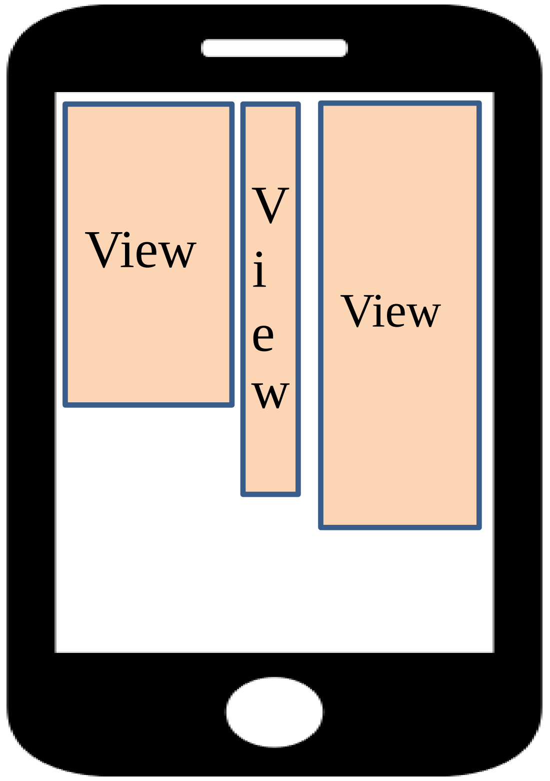
http://www.akexorcist.com/2016/06/constrain-layout-the-new-android-layout.html

LinearLayout

- Good for smaller devices or when simple interface makes sense
- Layout in Vertical or Horizontal one after another child View objects
- Examples:

Vertical





Horizontal

Linear Layout

Good:

- Simple
- Know exactly how it will look on every device

Bad:

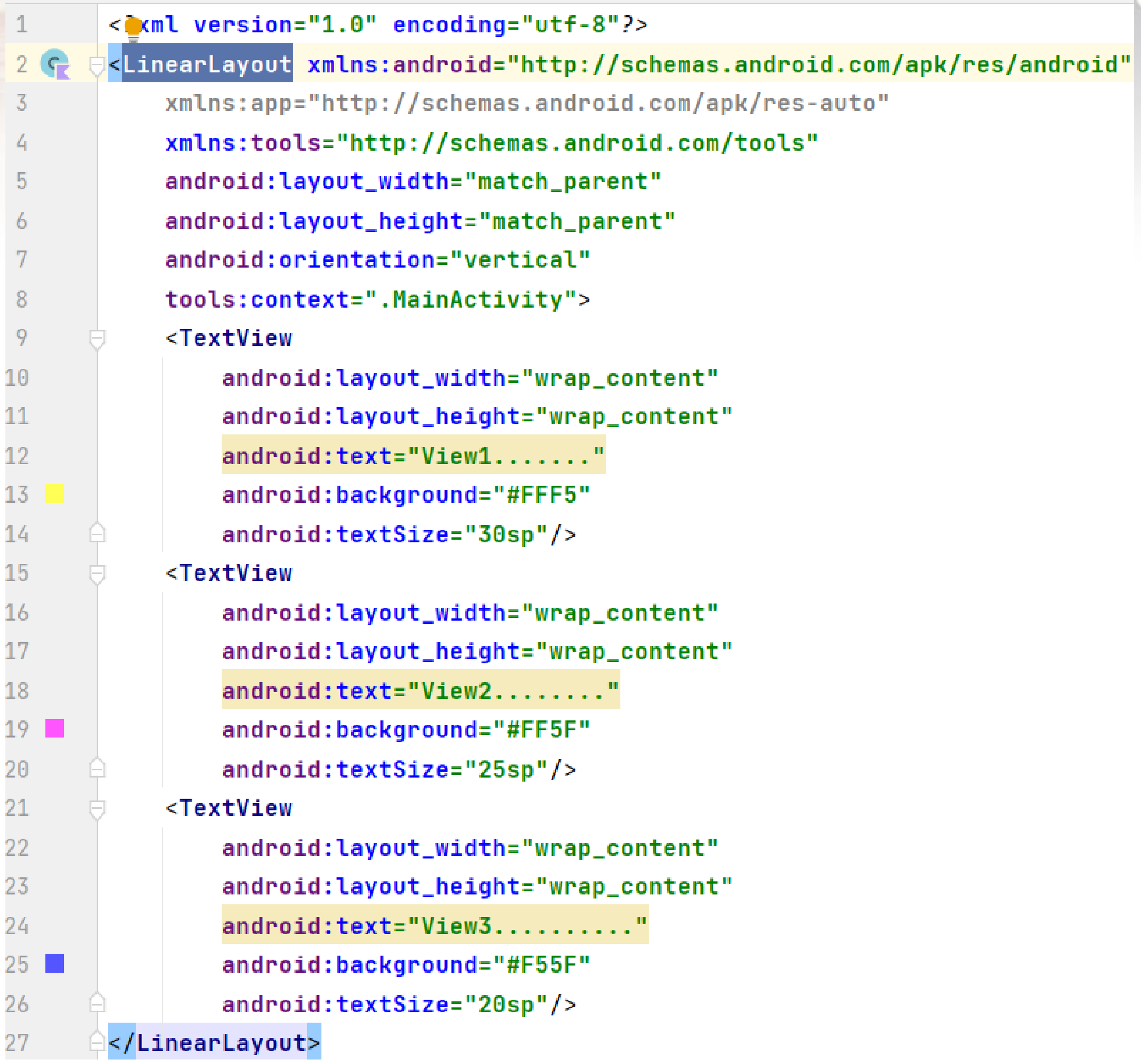
- Well for many interfaces too simple….

- ViewGroup (another Layout) inside as a member of the LinearLayout to make a more COMPLEX interface

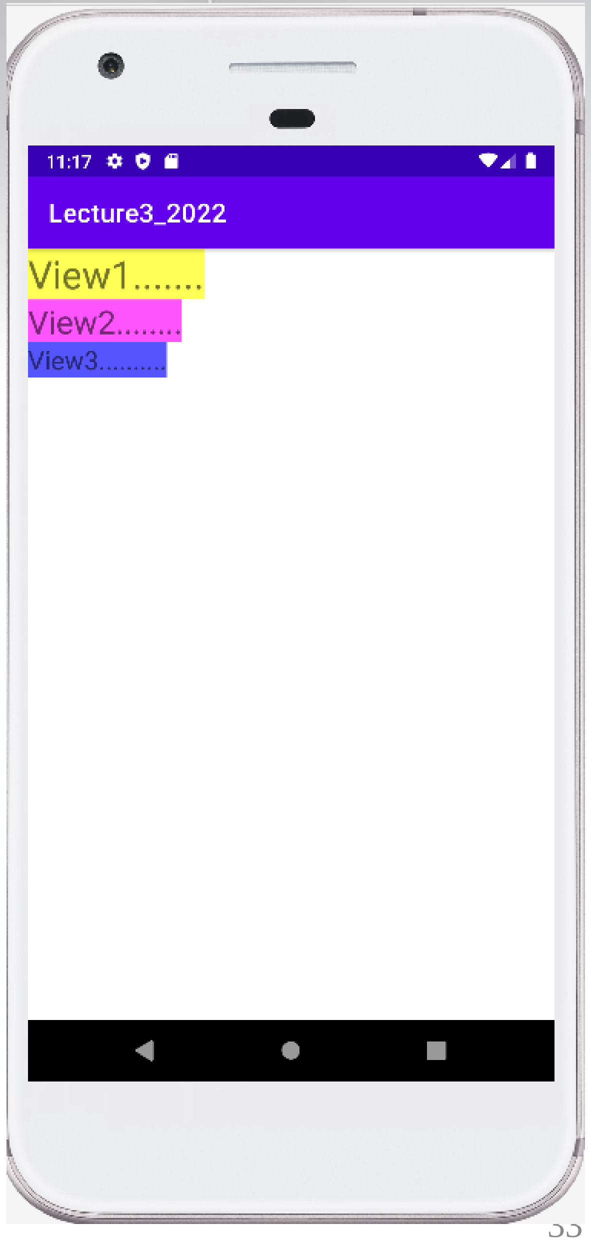
Example : Linear Layout

Activity_main.xml

Vertical
orientation



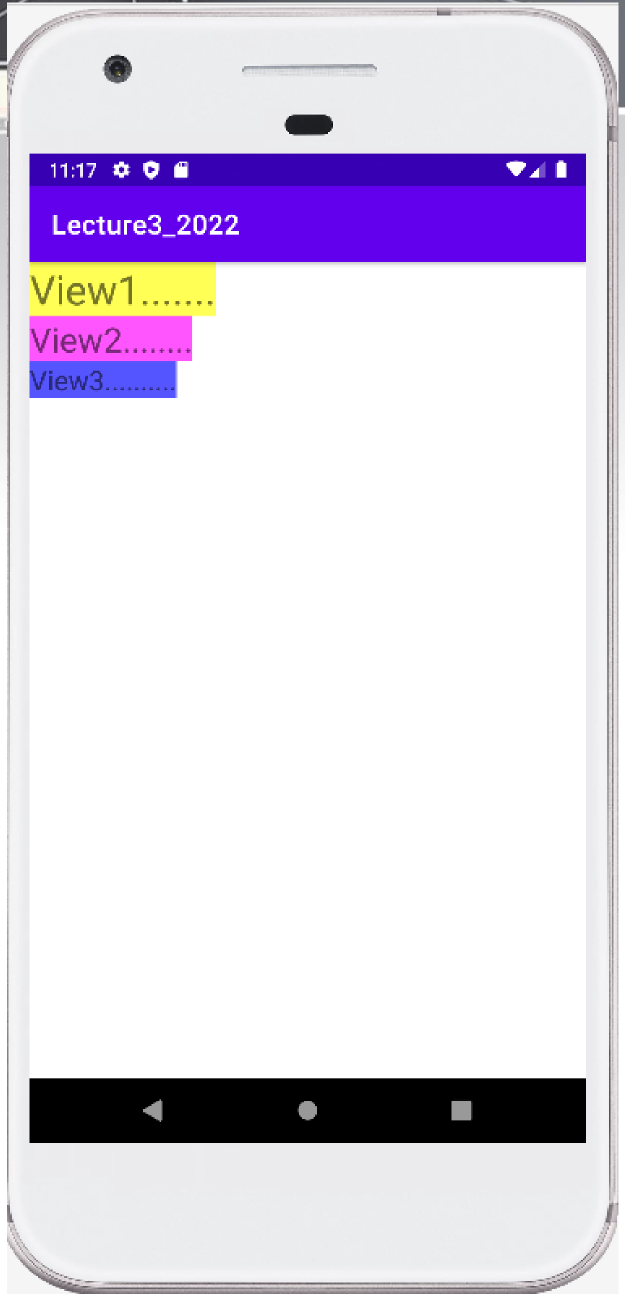
```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View1......."
        android:background="#FFF5"
        android:textSize="30sp"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View2........"
        android:background="#FF5F"
        android:textSize="25sp"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View3.........."
        android:background="#F55F"
        android:textSize="20sp"/>
</LinearLayout>
```



LinearLayout: Vertical orientation

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
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:orientation="vertical"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View1......."
        android:background="#FFF5"
        android:textSize= "30sp"/>
```

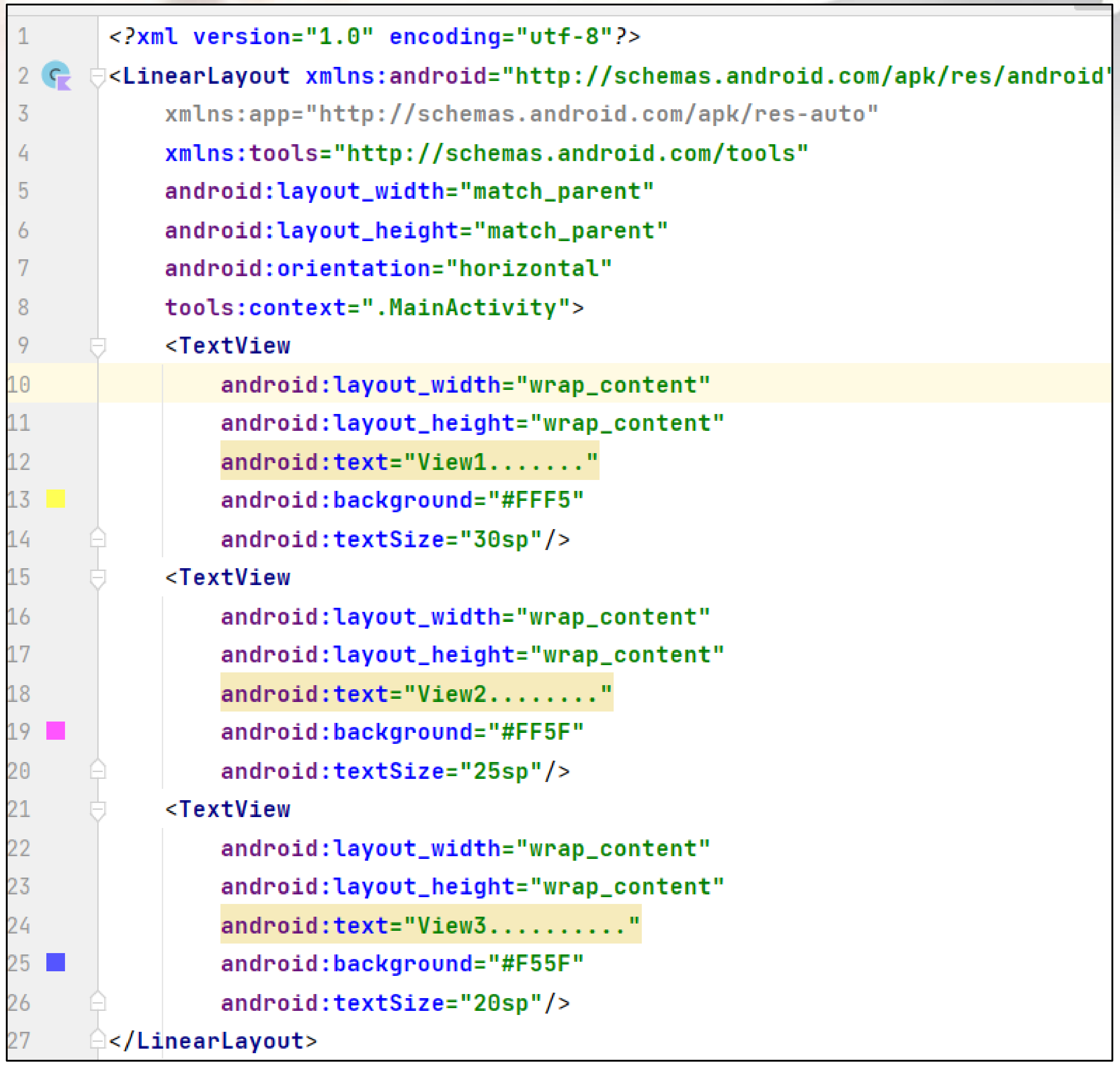
```
<TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View2........"
        android:background="#FF5F"
        android:textSize="25sp"/>
   <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View3.........."
        android:background="#F55F"
        android:textSize="20sp"/>
</LinearLayout>
```



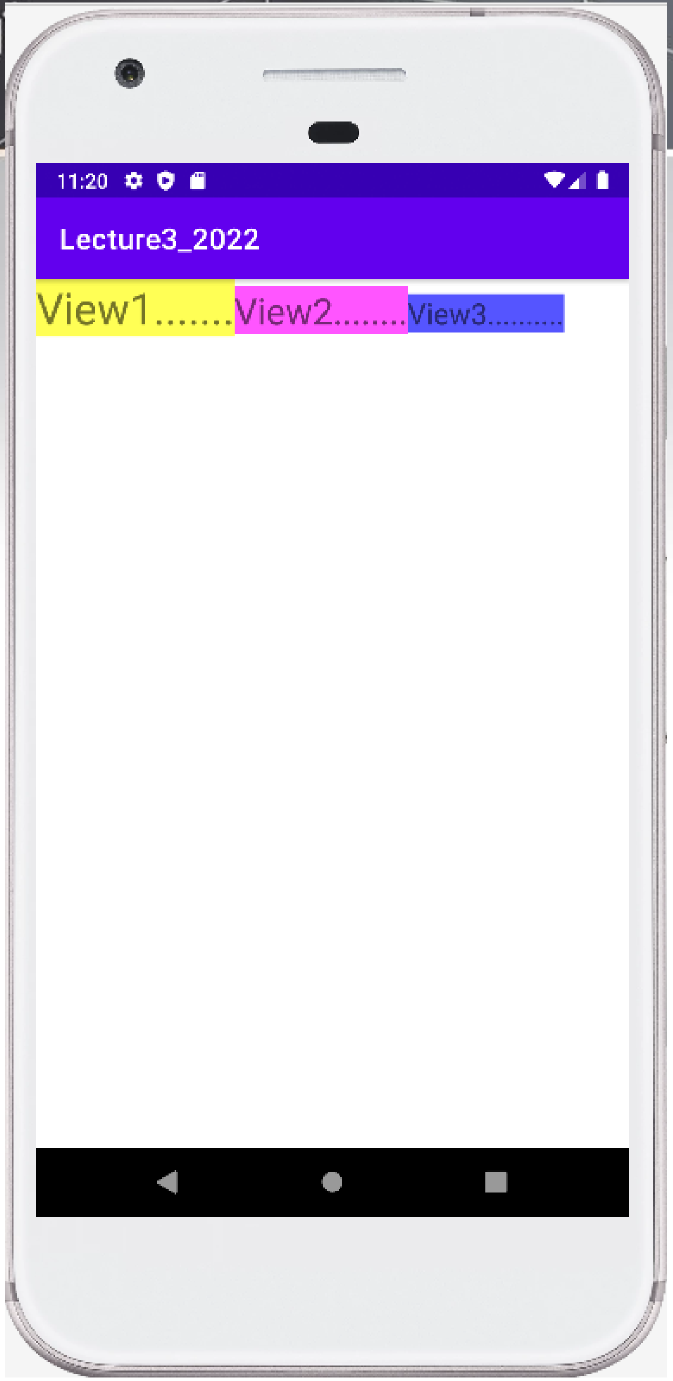
Example : Linear Layout

Activity_main.xml

Horizontal orientation



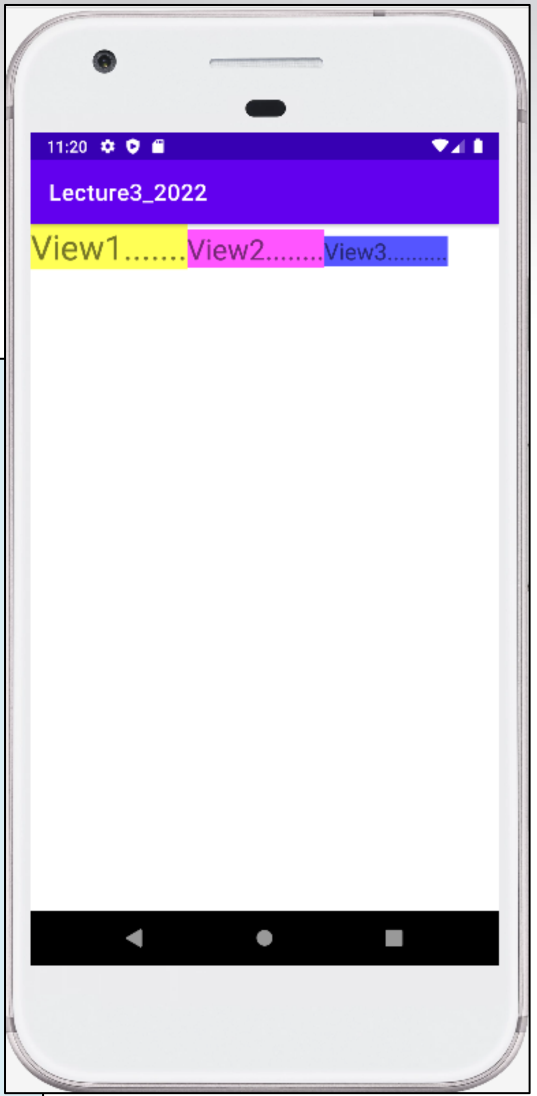
```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View1......."
        android:background="#FFF5"
        android:textSize="30sp"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View2........"
        android:background="#FF5F"
        android:textSize="25sp"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View3.........."
        android:background="#F55F"
        android:textSize="20sp"/>
</LinearLayout>
```



LinearLayout: Horizontall orientation

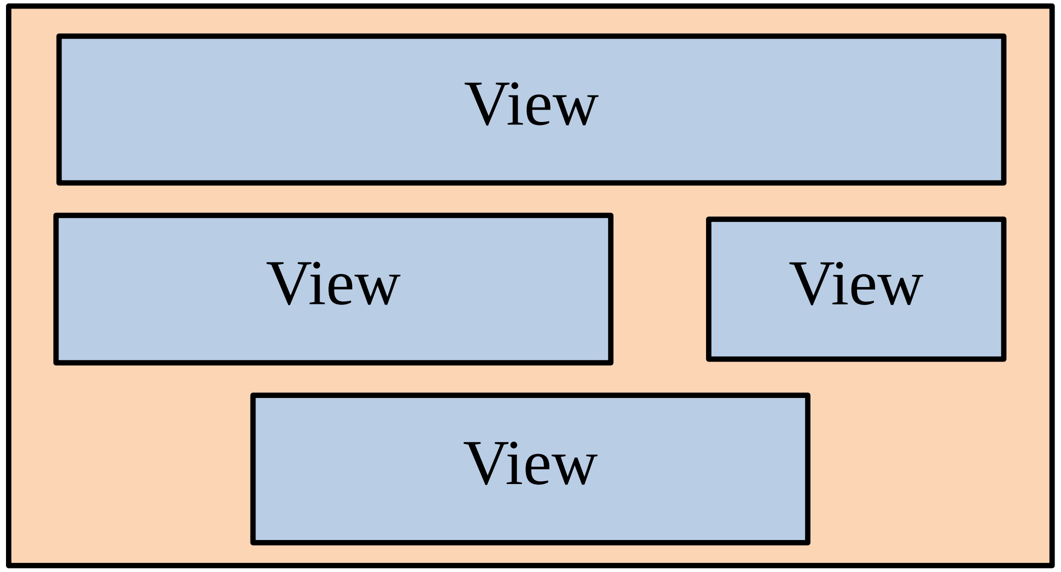
```
<?xml version="1.0" encoding="utf-8"?>
< LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View1......."
        android:background="#FFF5"
        android:textSize="30sp"/>
```

```
<TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View2........"
        android:background="#FF5F"
        android:textSize="25sp"/>
   <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View3......."
        android:background="#F55F"
        android:textSize="20sp"/>
</ LinearLayout >
```



RelativeLayout

– Position relative to another position
– Can give more complex interfaces
– Know what will look like on different sized devices



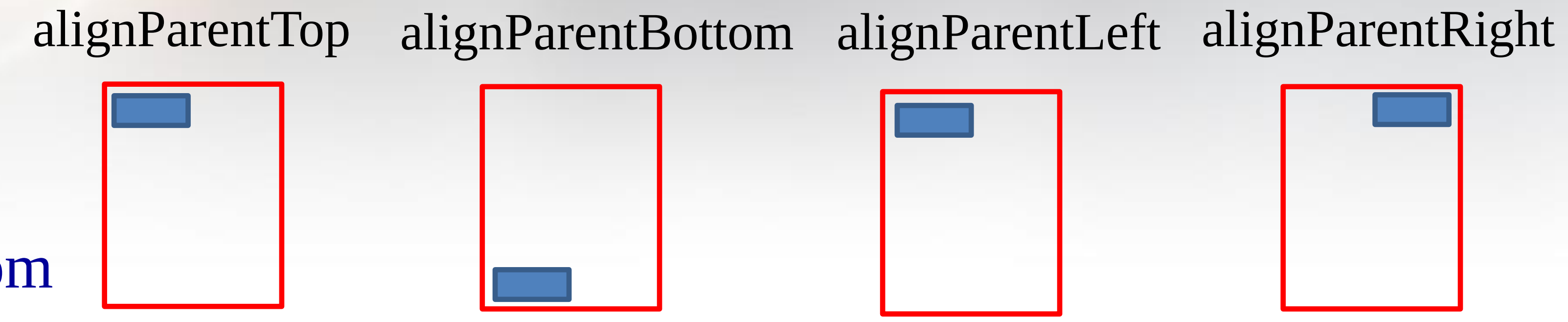
RelativeLayout

Parameters in XML

- Position relative to Parent

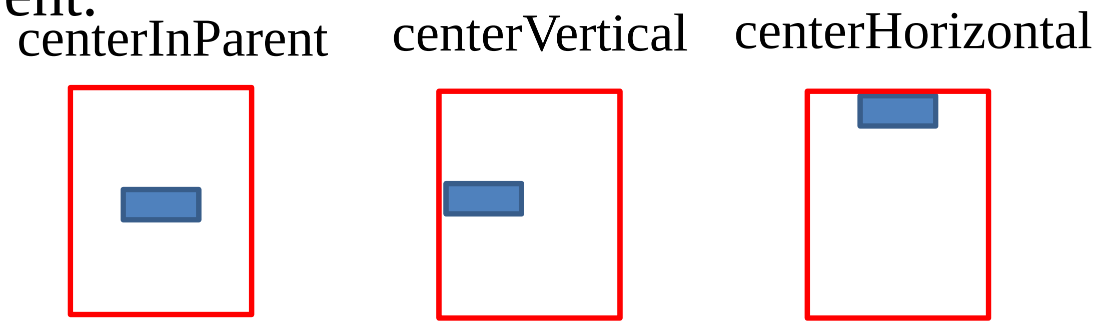
android:layout_alignParentTop
android:layout_alignParentBottom
android:layout_alignParentLeft
android:layout_alignParentRight

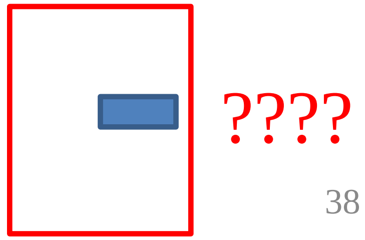
VALUE = "true“ -If "true", moves to that edge of Parent.



android:layout_centerInParent
android:layout_centerVertical
android:layout_centerHorizontal

VALUE= "true"-- If "true", centers this child vertically within its parent.





RelativeLayout

- Position relative to another view

  android:layout_below

  android:layout_above

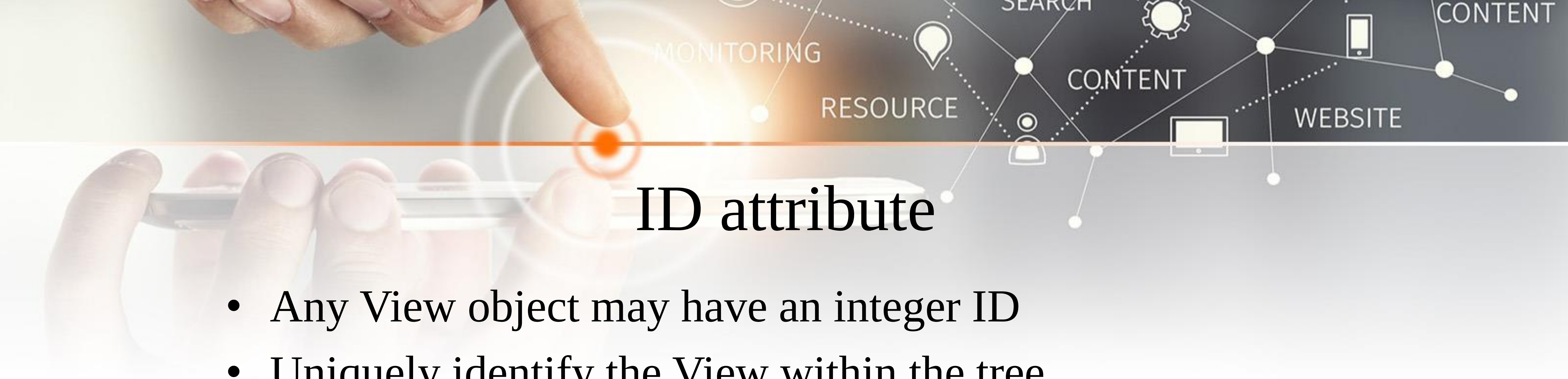
  android:layout_toLeftOf

  android:layout_toRightOf

  VALUE = “resource ID of other widget”

  -- Positions the top edge of this view below/above of the view specified with a resource ID.

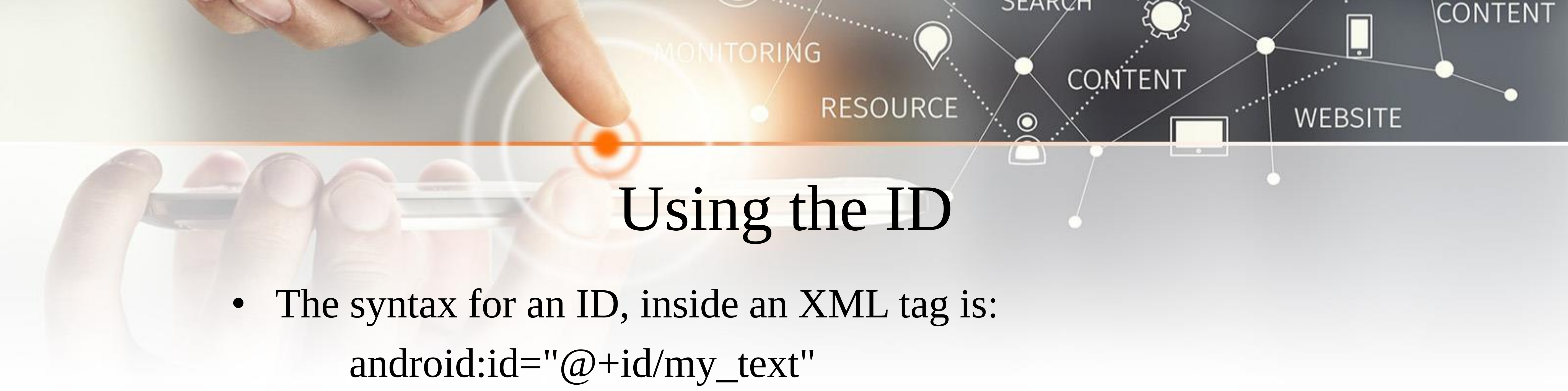
  OR Positions the left edge of this view to the left/right of the view specified with a resource ID.



ID attribute

- Any View object may have an integer ID
- Uniquely identify the View within the tree
- The ID is typically assigned in the layout XML file as a string
- This attribute is common to all View objects

```
android:id="@+id/id_of_view"
```



Using the ID

- The syntax for an ID, inside an XML tag is:

  android:id="@+id/my_text"

- Referencing an Android resource ID:

  android:layout_ ………..="@id/my_text"

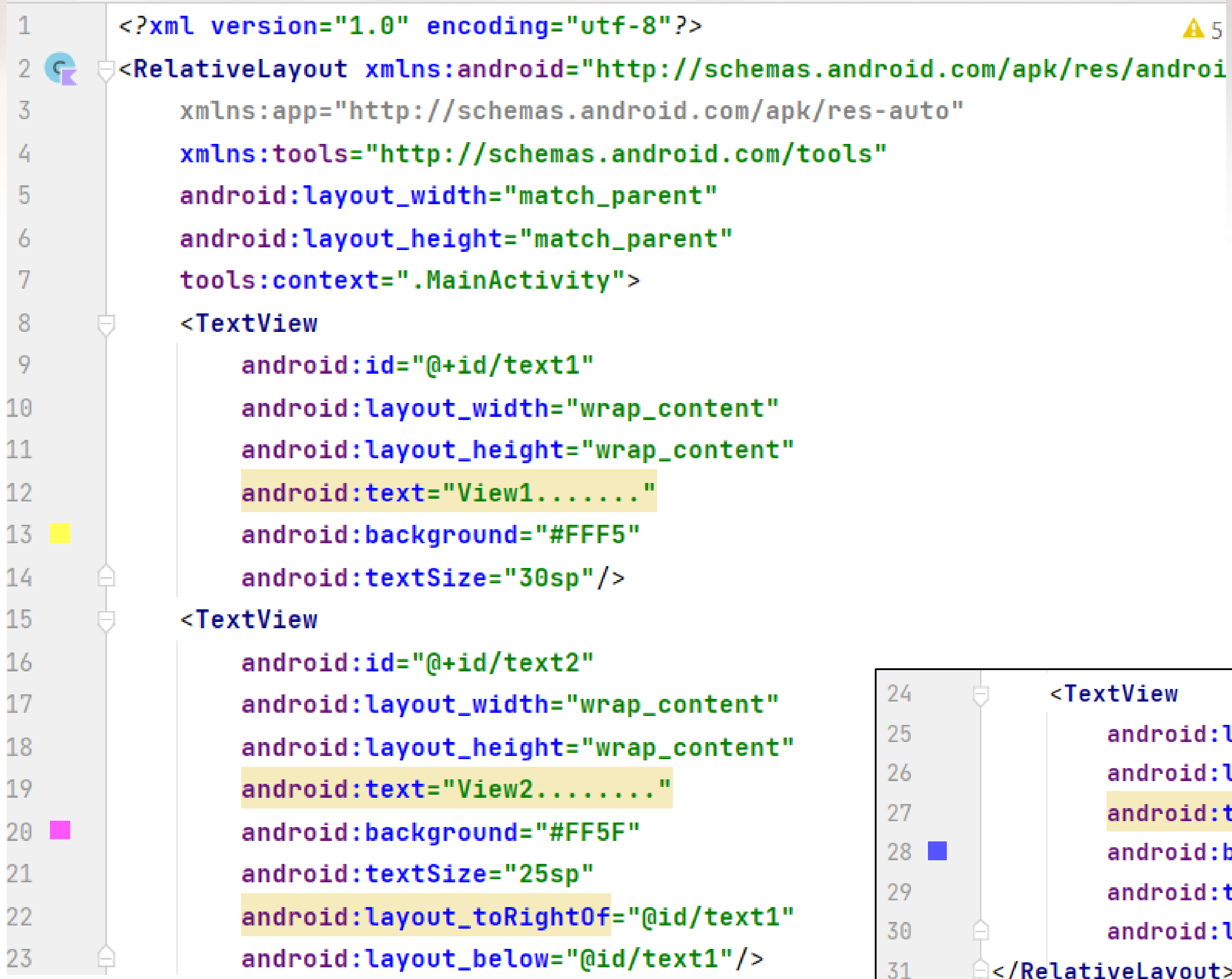
In the layout.xml file:

```
<TextView android:id="@+id/my_text"
android:layout_width="wrap_content"
android:layout_height="wrap_content" />
```

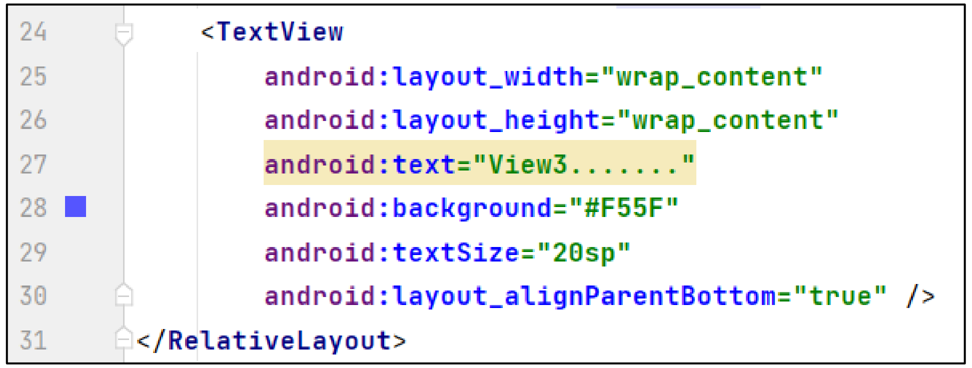
In the Kotlin code:

```
val textView: TextView = findViewById(R.id.my_text) as TextView
```

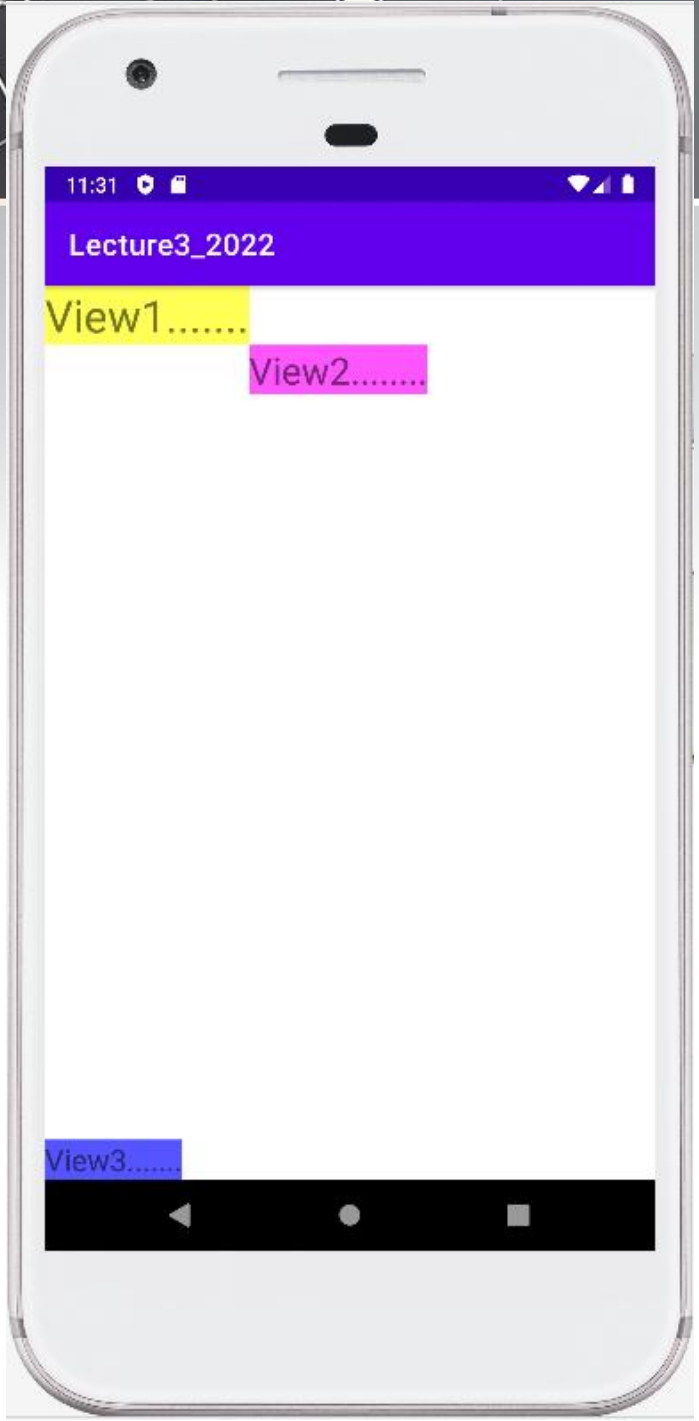
Example1: RelativeLayout



```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/androi
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/text1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View1......."
        android:background="#FFF5"
        android:textSize="30sp"/>
    <TextView
        android:id="@+id/text2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View2........"
        android:background="#FF5F"
        android:textSize="25sp"
        android:layout_toRightOf="@id/text1"
        android:layout_below="@id/text1"/>
```

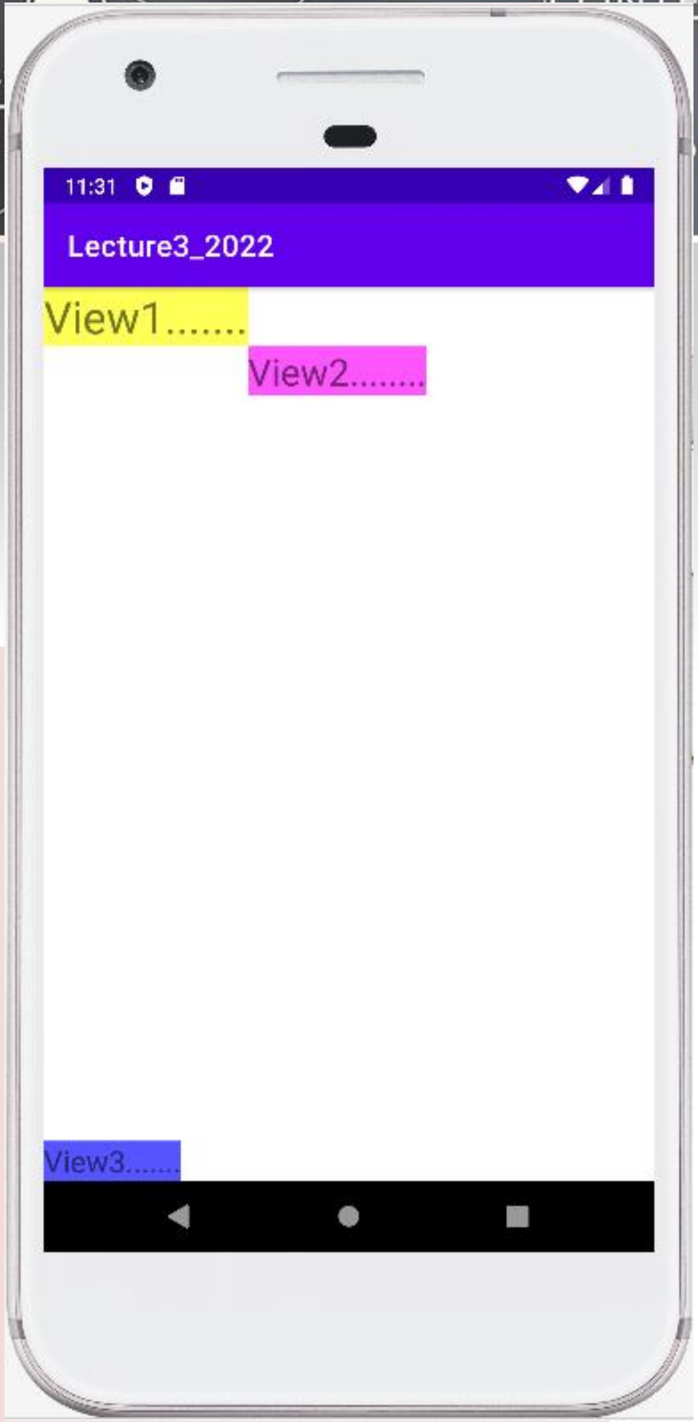


```xml
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View3......."
        android:background="#F55F"
        android:textSize="20sp"
        android:layout_alignParentBottom="true" />
</RelativeLayout>
```



Example1: RelativeLayout

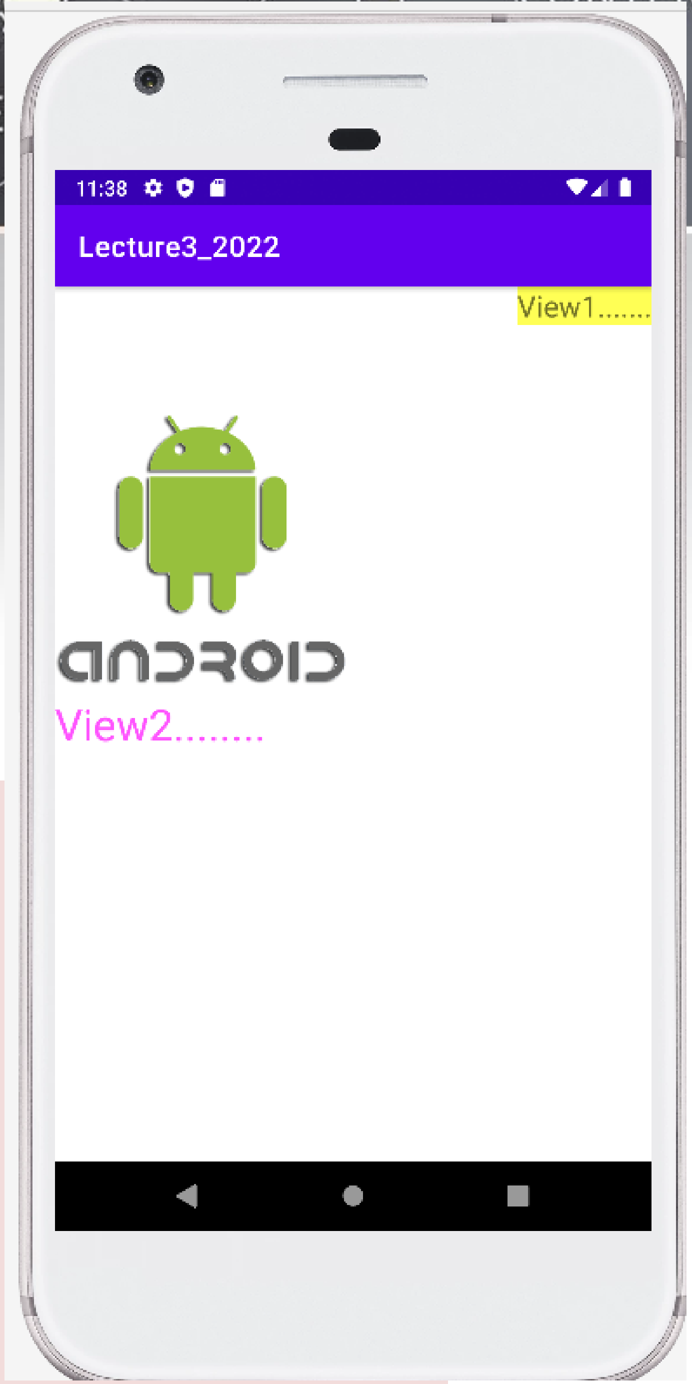
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/text1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View1......."
        android:background="#FFF5"
        android:textSize="30sp"/>
    <TextView
        android:id="@+id/text2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View2........"
        android:background="#FF5F"
        android:textSize="25sp"
        android:layout_toRightOf="@id/text1"
        android:layout_below="@id/text1"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View3......."
        android:background="#F55F"
        android:textSize="20sp"
        android:layout_alignParentBottom="true" />
</RelativeLayout>
```



Example2: RelativeLayout

```
<?xml version="1.0" encoding="utf-8"?>
<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"
   tools:context=".MainActivity">
   <TextView
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:text="View1......."
       android:background="#FFF5"
       android:layout_alignParentRight = "true"
       android:textSize="20sp"/>
```

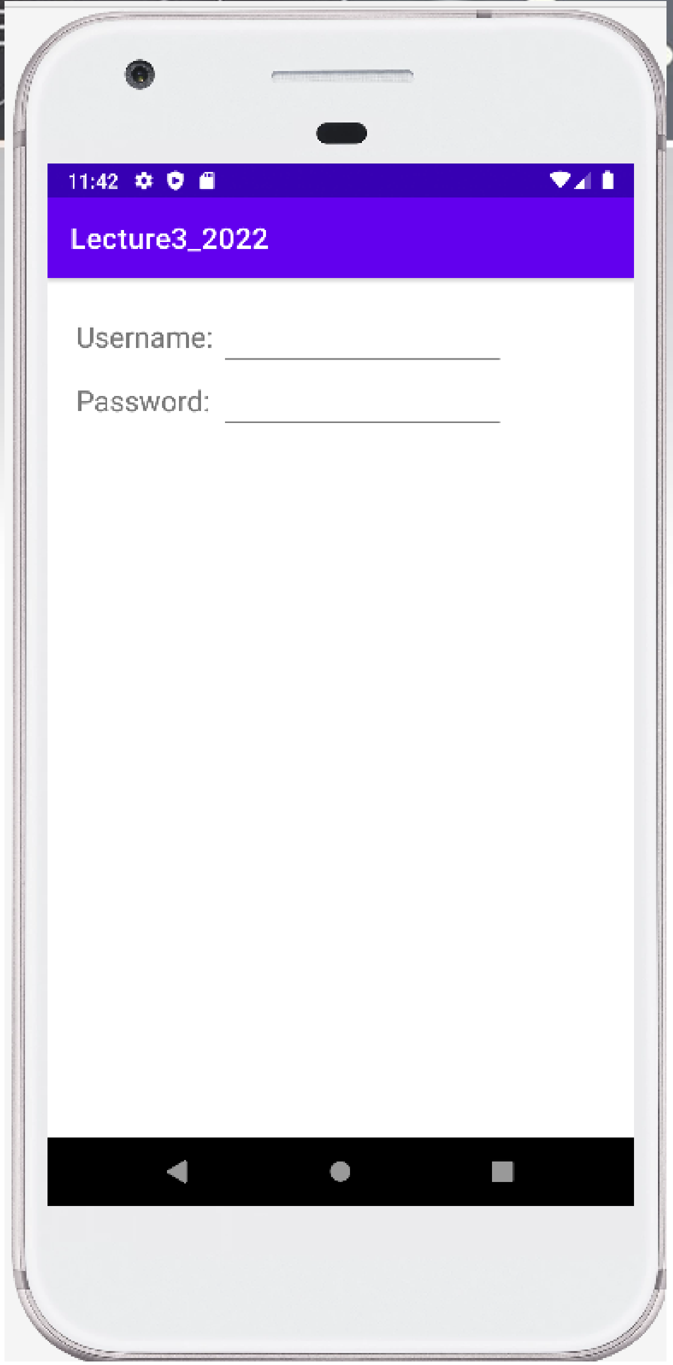
```
<TextView
      android:id="@+id/text2"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="View2........"
      android:textColor="#FF5F"
      android:textSize="30sp"
      android:layout_centerVertical="true"  />
  <androidx.appcompat.widget.AppCompatImageView
      android:layout_height="200dp"
      android:layout_width="200dp"
      android:layout_above="@id/text2"
      android:src="@drawable/android_logo"/>
</RelativeLayout >
```



TableLayout

- Extension of LinearLayout
- This layout structures its child controls into rows and columns

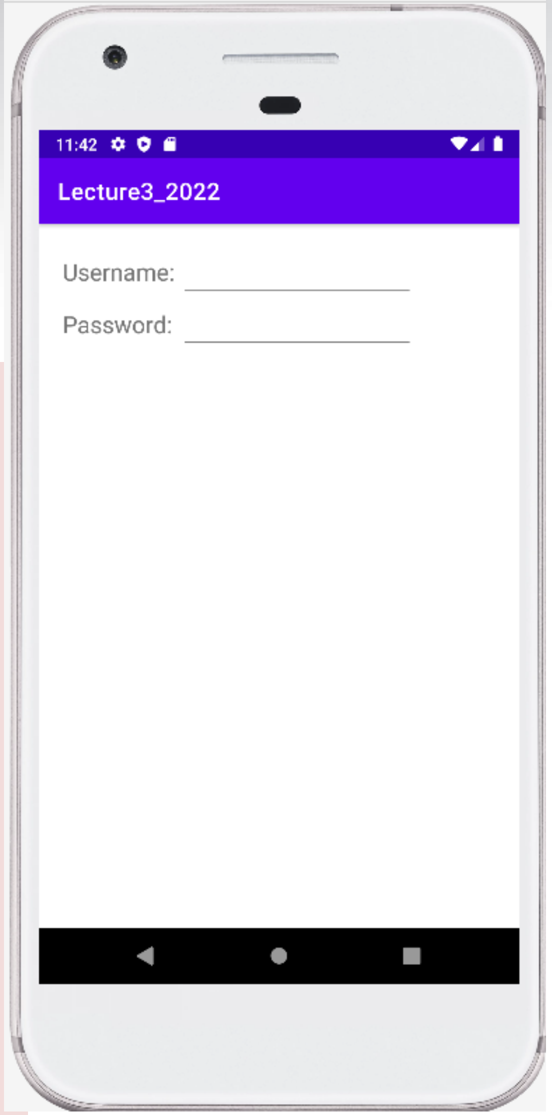
```
<TableLayout xmlns:android="…"
             android:layout_width="match_parent"
             android:layout_height="match_parent">
        <TableRow>
                <TextView … />
                <EditText … />
        </TableRow>
        <TableRow>
                <TextView … />
                <EditText … />
        </TableRow>
</TableLayout>
```



Username:
Password:

Example: TableLayout

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity">
   <TableRow>
      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Username: "
         android:textSize="20sp" />
      <EditText
         android:layout_width="200dp"
         android:layout_height="wrap_content"
         android:inputType="textShortMessage"
         android:hint=""/>
   </TableRow>
   <TableRow>
      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Password: "
         android:textSize="20sp" />
      <EditText
         android:layout_width="200dp"
         android:layout_height="wrap_content"
         android:inputType="textPassword"
         android:hint=""/>
   </TableRow>
</TableLayout>
```



End of Chapter

References

- http://www1.lasalle.edu/~blum/c349wks/ConstraintLayout.ppt
- http://people.sju.edu/~ggrevera/se/intro2android.ppt
- http://www.cse.bgu.ac.il/common/download.asp?FileName=Lecture%203.ppt&AppID=2&MainID=552&SecID=4666&MinID=3
- http://algebra.sci.csueastbay.edu/~grewe/CS4521/Mat/Lectures/Android_Layout.ppt
- https://research.ece.ncsu.edu/wireless/MadeInWALAN/AndroidTutorial/PPTs/helloViews1.ppt