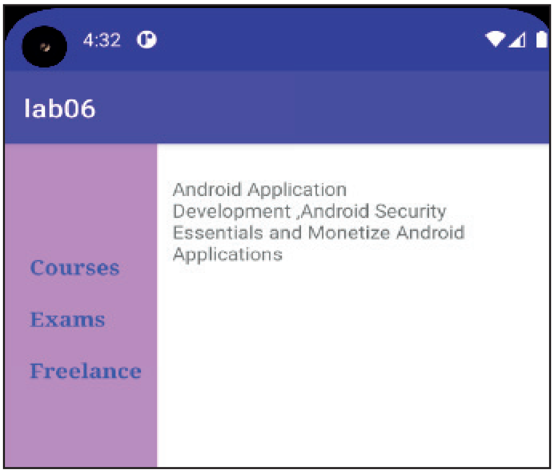
Lab 6

Android Application Layouts, Styles and Themes

Objectives:

- ➢ **Creating Your App Layout**
- ➢ **Configuring Your Styles and Themes**
- ➢ **Configuring Your App Icon**

Lab Scenario:
In this lab, you will create an Android application comprising a main constraint layout. You will then create a navigation frame layout which includes text links to display some information about these links in the main constraint layout, as illustrated in the figure on the right side.



4:32

lab06

Android Application Development ,Android Security Essentials and Monetize Android Applications

In the lab, you will create a style that has specific font styles and apply it to the text which will appear in the main constraint layout. You will then create your own app theme and apply it to your app using **AndroidMainefest.xml** file.

Finally, you will configure your app icon which will represent your app on the smart device.

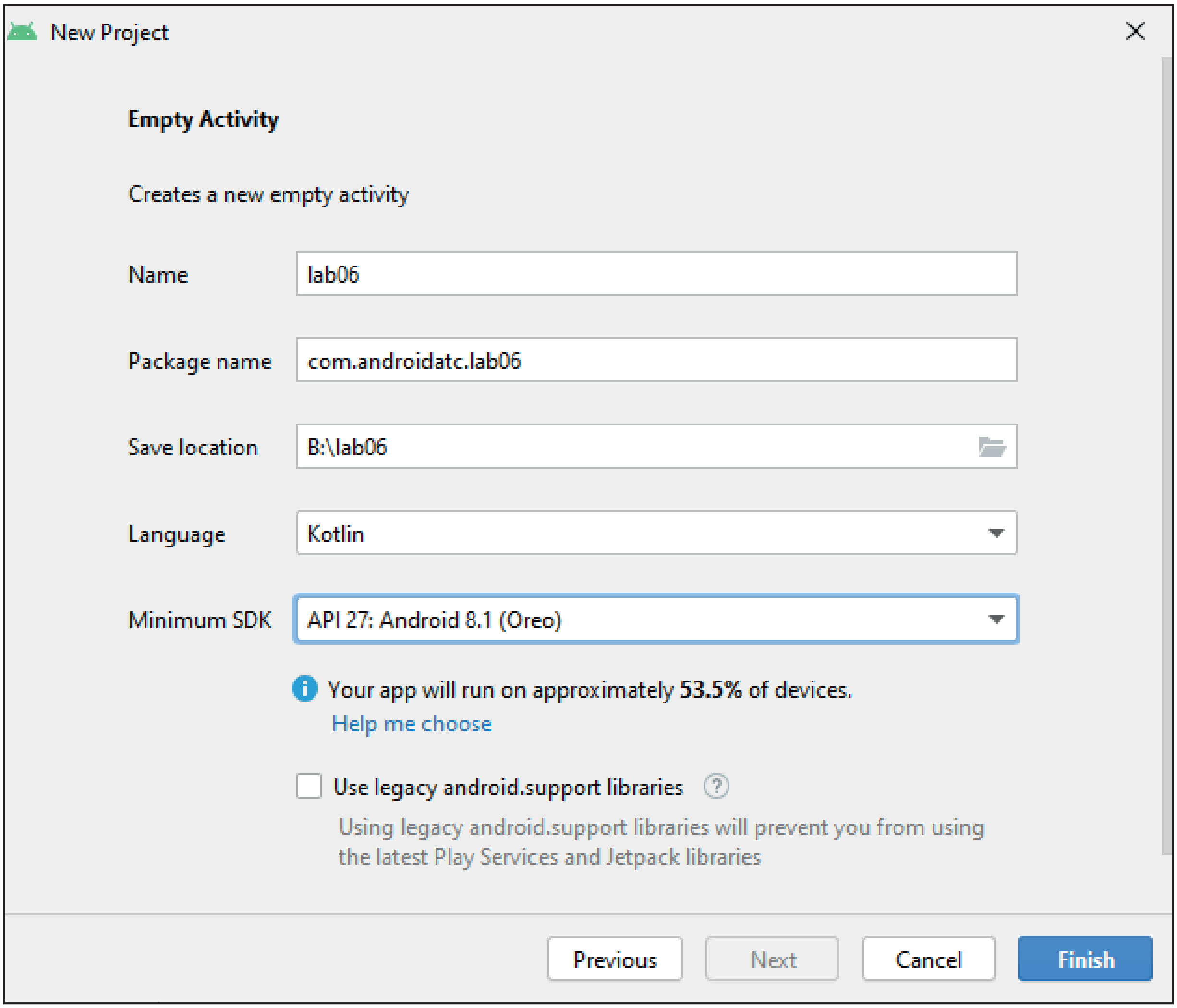
Create Your App Layout

The following are the steps of creating your Android App:

1- Open Android Studio, and then click **File → New → New Project.**

2- Select **Empty Activity**, and click **Next**.

3- Type **Lab03** for the project name as illustrated in the figure below, and click **Finish**.



New Project

Empty Activity

Creates a new empty activity

Name: lab06

Package name: com.androidatc.lab06

Save location: B:\lab06

Language: Kotlin

Minimum SDK: API 27: Android 8.1 (Oreo)

Your app will run on approximately **53.5%** of devices.
Help me choose

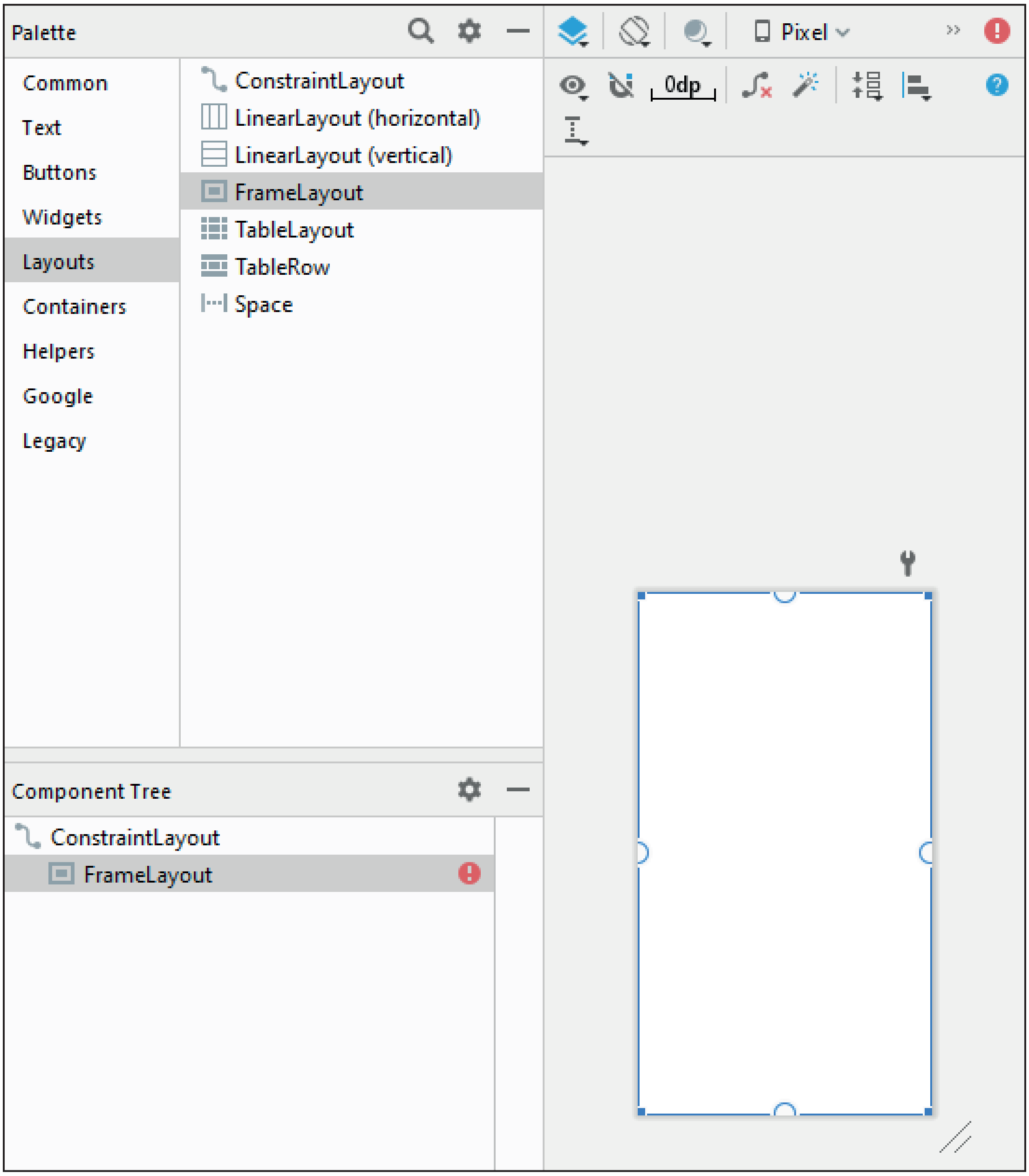
□ Use legacy android.support libraries

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

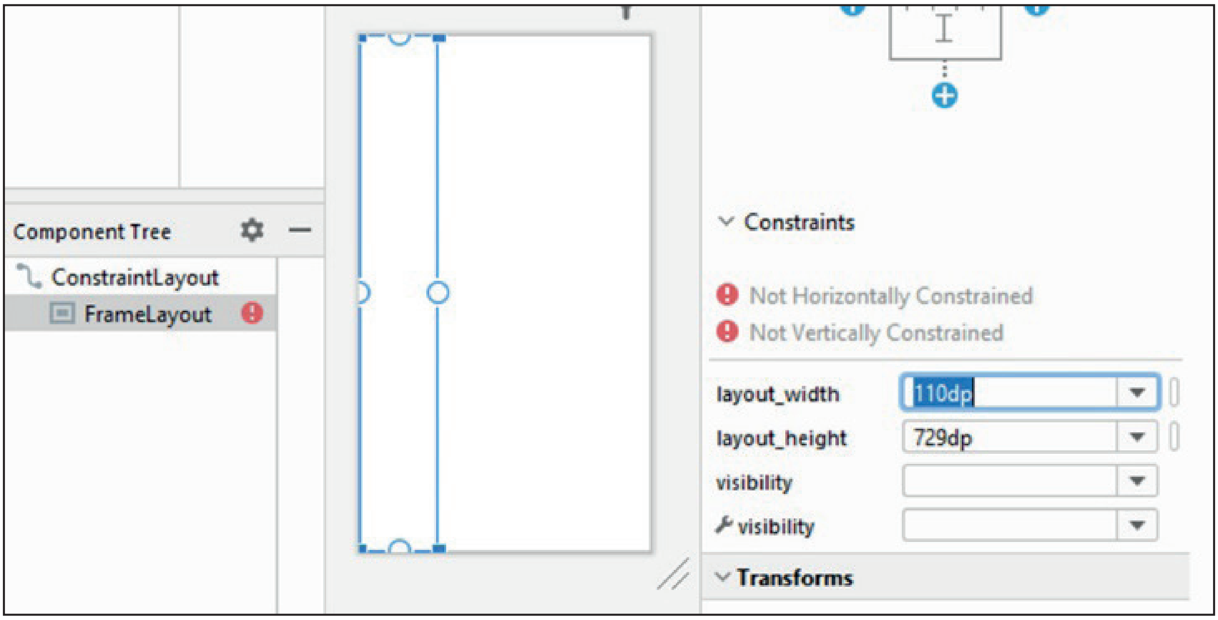
Configuring Your Styles and Themes

4- Open the **activity_main.xml** file (app → res → layout) in Design mode, and delete "**Hello World!**" **TextView**.

5- From the Palette panel (Layouts), drag and drop **FrameLayout** to your activity which is a constraint layout activity, as illustrated in the following figure:



6- Set the **FrameLayout** width to "**110 dp**" using its attributes (from the Attributes tab) so it takes the shape which is illustrated in the figure below:



Component Tree

ConstraintLayout

FrameLayout

Constraints

Not Horizontally Constrained

Not Vertically Constrained

layout_width 110dp

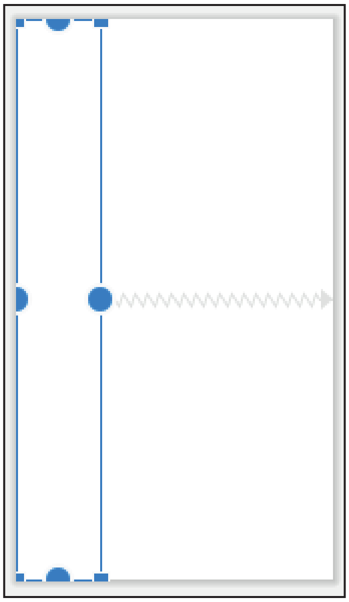
layout_height 729dp

visibility

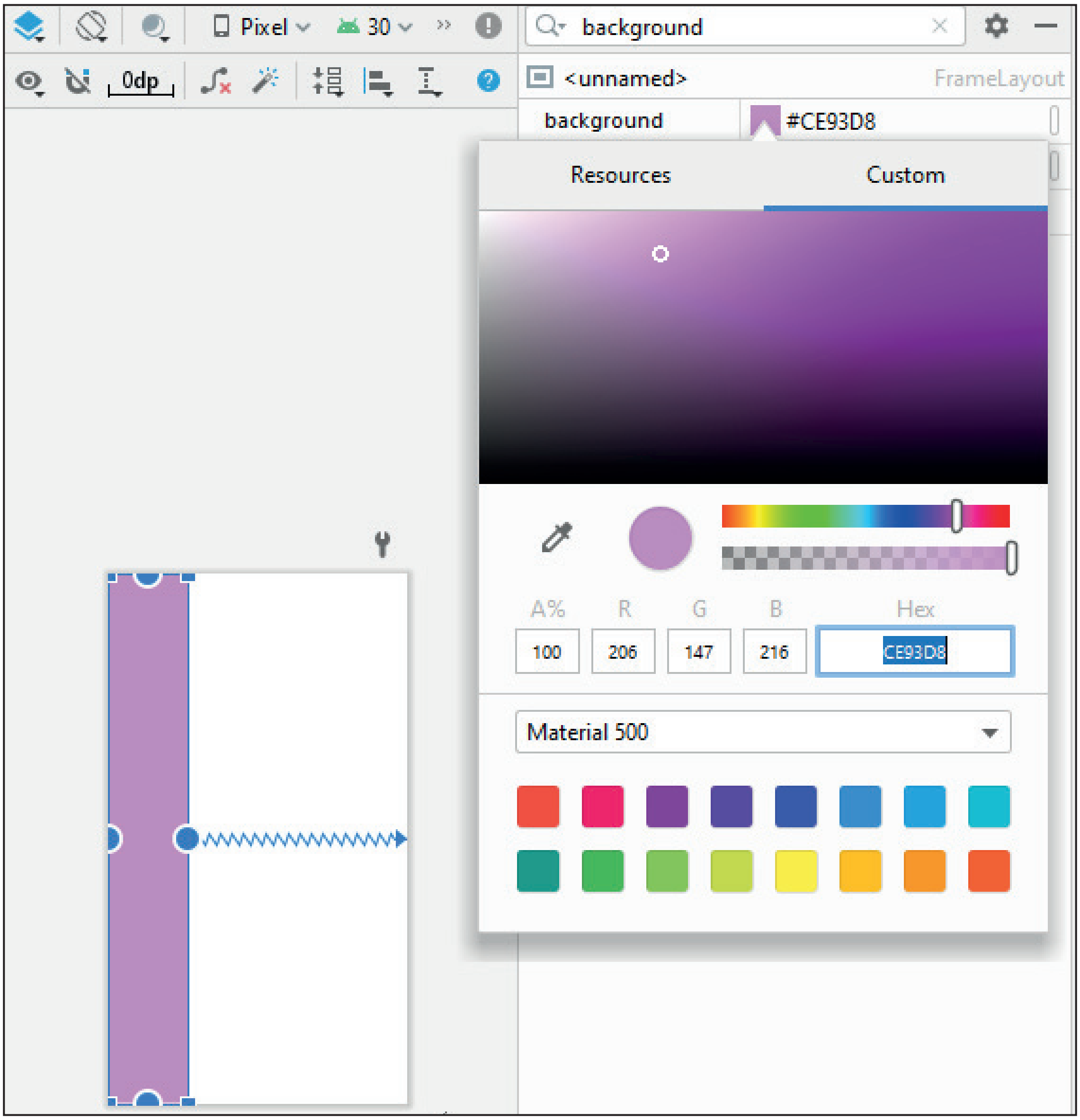
visibility

Transforms

7- Set the **FrameLayout** constraint relative to the main activity (ConstraintLayout) to ensure that it will not change location when you run your app as illustrated in the following figure (set the top, right, left and bottom constraints):



8- Change the **FrameLayout** background color to the following color: #CE93D8. You can do that from the **Attributes** tab. Select the **FrameLayout**, click the **Attributes** tab, type background in the search area, and then select the color you like (you can select any color or type: CE93D8 in the Hex area) as illustrated in the following figure:



9- You can do that also if you open the **activity_main.xml** file in the **Code** mode, and then add the following attribute `android:background="#CE93D8"`to **FrameLayout** tag. The **FrameLayout** tag code will be as follows:

```
<FrameLayout
    android:layout_width="110dp"
    android:layout_height="729dp"
    android:background="#CE93D8"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent">
```

10- Add the following three **TextView** widgets to your **FrameLayout** activity using **activity_main.xml** in **Code** mode. It is hard to use the drag and drop technique because it will be difficult to set these TextViews margins (Constraints) inside your **FrameLayout** activity.

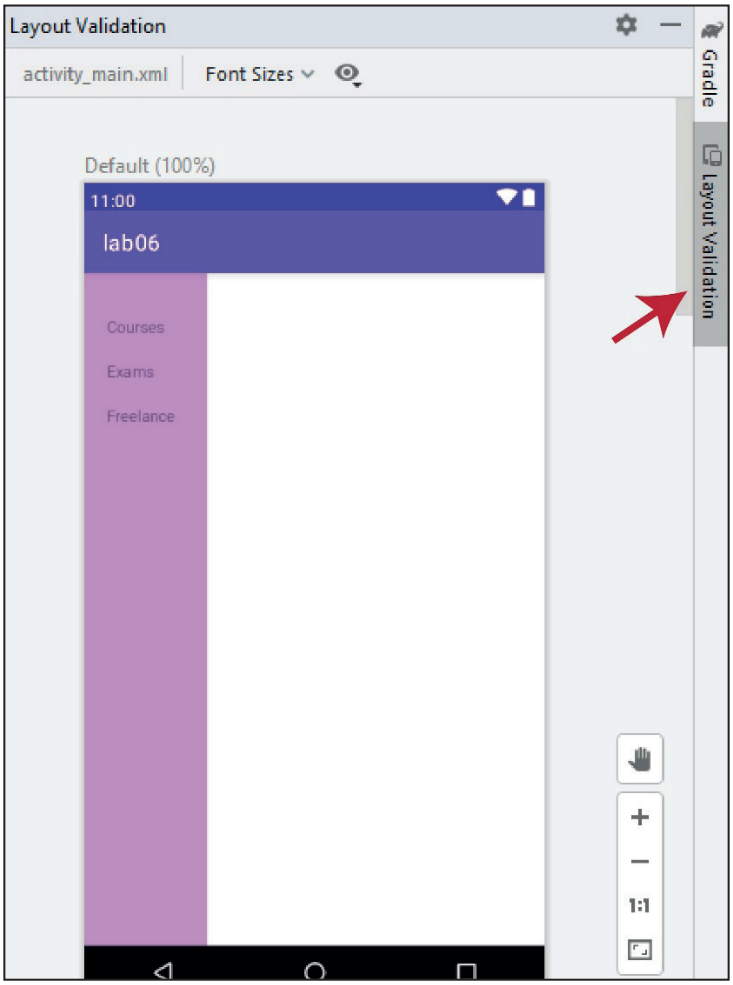
The three **TextViews** have the following characteristics:

ID	text
courses	Courses
Exams	Exams
Freelance	Freelance

The following is the **activity_main.xml** after adding the three **TextView**s to your **FrameLayout** activity:

```
<?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">

    <FrameLayout
        android:layout_width="110dp"
        android:layout_height="729dp"
        android:background="#CE93D8"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

    <TextView
        android:id="@+id/courses"
        android:layout_width="wrap_content"
        android:layout_height="21dp"
        android:layout_marginStart="20dp"
        android:layout_marginTop="100dp"
        android:text="Courses"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:layout_editor_absoluteY="0dp" />
    <TextView
        android:id="@+id/Exams"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="20dp"
        android:layout_marginTop="140dp"
        android:text="Exams"/>

    <TextView
        android:id="@+id/Freelance"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="20dp"
        android:layout_marginTop="180dp"
        android:text="Freelance"/>

</FrameLayout>

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

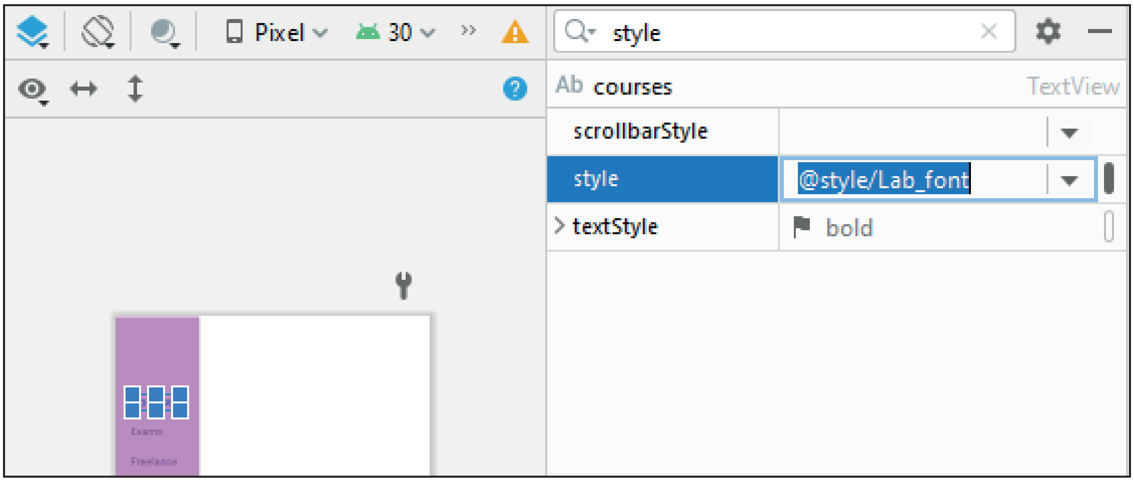
11 – Click the **Layout Validation** tab to check your activity layout. You should get the following figure:



12- To create a style that includes font family, font size, font style and other attributes and later apply this style to these three **TextView**s , open **themes.xml** file (app → res → values → themes → themes.xml), then add the following style tag which is called "**Lab_font**" as illustrated in the gray highlight part of the following code:

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.Lab06" parent="Theme.MaterialComponents.
DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor"
tools:targetApi="l">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>

    <style name="Lab_font">
        <item name="android:fontFamily">serif</item>
        <item name="android:textSize">16sp</item>
        <item name="android:textColor">#3F51B5</item>
        <item name="android:textStyle">bold</item>
    </style>

</resources>
```

13- Open the **activity_main.xml** file in the Design mode, and select the **Courses TextView**, click **Attributes** tab, type **Lab** in the search area, then select **Lab_Font** of the list as illustrated in the following figure:

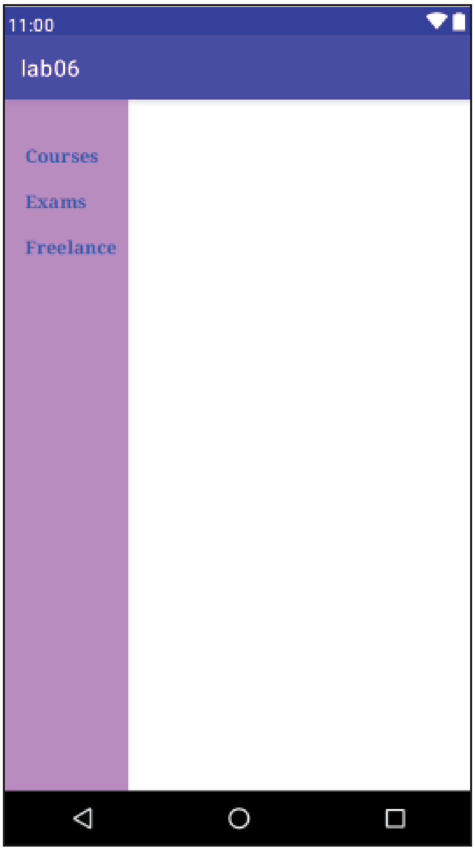


style

Ab courses TextView

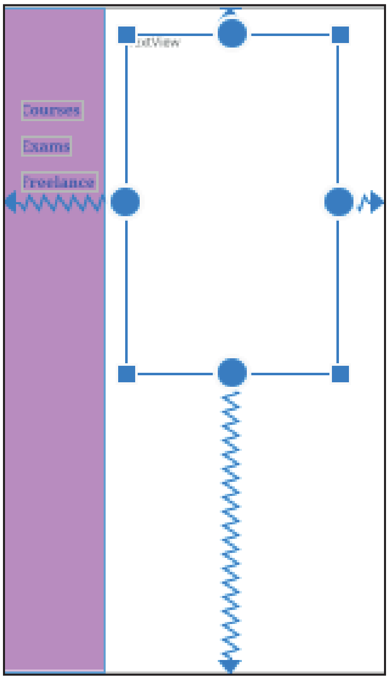
scrollbarStyle	
style	@style/Lab_font
textStyle	bold

Repeat this action on **Exams** and **Freelance TextViews**. If you click the **Layout Validation** on the top right of the Android Studio, you should get the following figure for your activity:



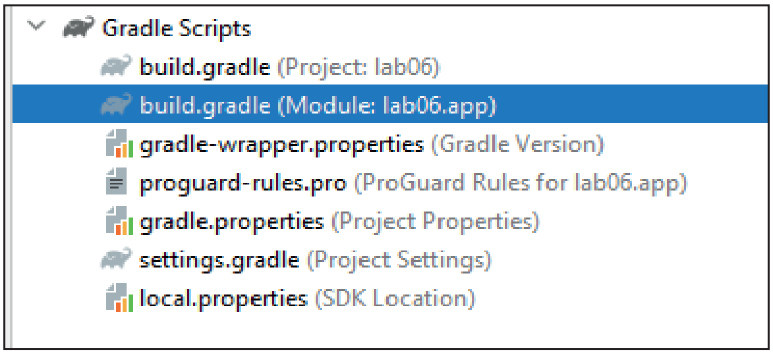
14- Now you will add a new empty **TextView** to the right side of your **FrameLayout** to display the information you need about the three items (Courses, Exams, and Freelance) on the left side.

Add **TextView** to the right side of your **FrameLayout** activity using drag and drop techniques and set its constraints relative to the main **ConstraintLayout**. (See illustration below.) This **TextView** will be used to display any information you want about the items in the left side like courses.



15- Using the **Attributes** tab, delete the **text** attribute value of this new TextView and set its **ID: info**

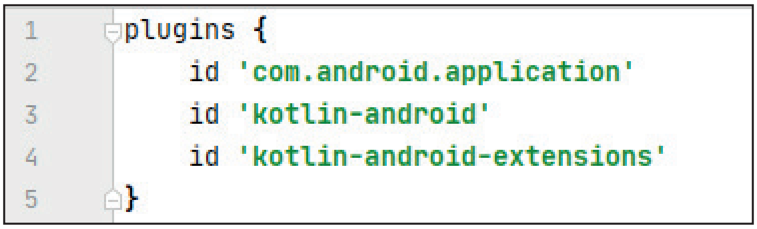
16- Before you start typing the Kotlin code in your app, check the **build.gardle (Module:lab06. app)** file content as illustrated in the figure below:



Gradle Scripts
- build.gradle (Project: lab06)
- build.gradle (Module: lab06.app)
- gradle-wrapper.properties (Gradle Version)
- proguard-rules.pro (ProGuard Rules for lab06.app)
- gradle.properties (Project Properties)
- settings.gradle (Project Settings)
- local.properties (SDK Location)

Be sure it has the Kotlin plugins. If not, add the following code : **id 'kotlin-android-extensions'**

And click the **Sync Now**. The configuration should be as follows:



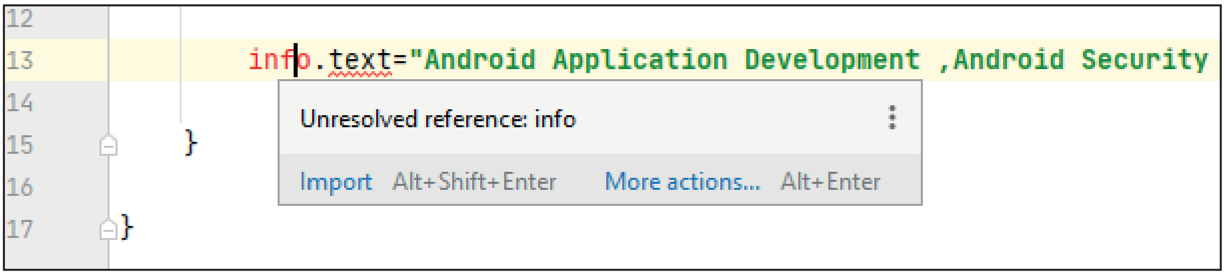
```
plugins {
    id 'com.android.application'
    id 'kotlin-android'
    id 'kotlin-android-extensions'
}
```

17- Now, you will configure your app in such a way that if the app user clicked on "**Courses**" on the left side, the courses' information will appear in the **TextView** which has ID = **info**.

You can do this by configuring the following Kotlin function in **MainActivity.kt** file as follows:

```
package com.androidatc.android_lab_lesson_06

import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

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

        info.text="Android Application Development ,Android
Security Essentials and Monetize Android Applications"

    }
}
```

You will find that the **info** and the **View** in the previous code are in red color. Click on **info**, then click **Import** to import the required classes to your app file as illustrated in the code below:



```
        info.text="Android Application Development ,Android Security

    }

}
```

Unresolved reference: info

Repeat the last step to import the required classed for the **View** class.

18- Open the **activity_main.xml** file in the **Code** mode, and link the “**display**” function with the **Courses TextView** by adding the attribute below the **Courses TextView**:

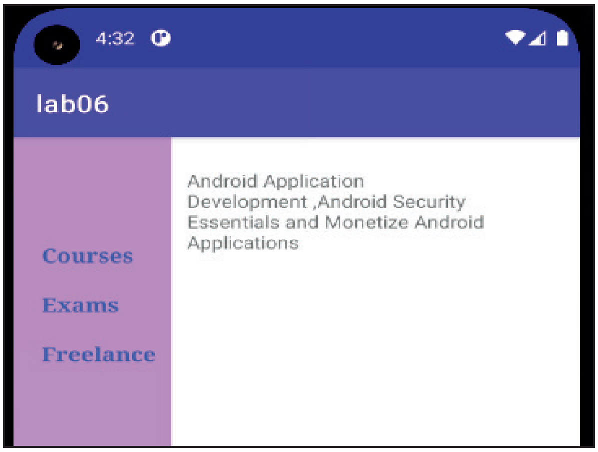
```
android:onClick="display"
```

This means that the function **display** will be called to run when your app users tap the Text: **Courses.**

The “**Courses**” **TextView** will be as follows. You should add only the gray highlighted attribute:

```
<TextView
    android:id="@+id/courses"
    style="@style/Lab_font"
    android:layout_width="wrap_content"
    android:layout_height="21dp"
    android:layout_marginStart="20dp"
    android:layout_marginTop="100dp"
    android:text="Courses"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    tools:layout_editor_absoluteY="0dp"
    android:onClick="display"
    />
```

19- Stop, then run your app. Tap the Courses in the left side of your layout. You should get the following figure:



4:32

lab06

Courses

Exams

Freelance

Android Application Development ,Android Security Essentials and Monetize Android Applications

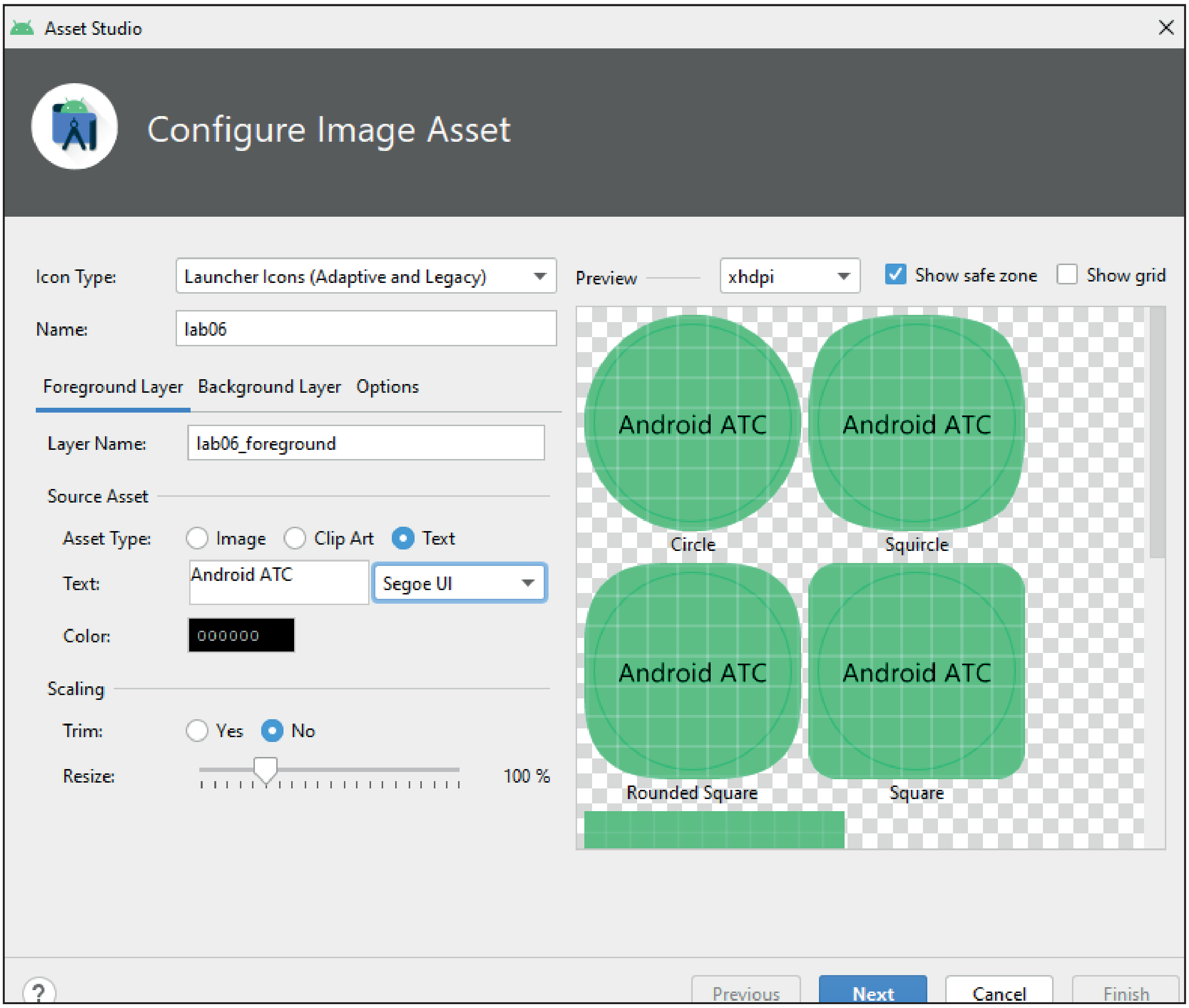
You may repeat the same configurations for the **TextView**s Exams and Freelance.

Configure Your App Icon

The final step of this lab is adding an icon which will represent your app on smart devices and at Google Play Store.

20- Right click **app** → **New** → **Image Asset** .

Change the **Name** to **lab06**, change the **Asset Type** to **Text,** and type: **Android ATC** in the **Text** box as illustrated in the following figure:



Asset Studio

Configure Image Asset

Icon Type: Launcher Icons (Adaptive and Legacy)

Name: lab06

Foreground Layer | Background Layer | Options

Layer Name: lab06_foreground

Source Asset

Asset Type: ○ Image ○ Clip Art ● Text

Text: Android ATC | Segoe UI

Color: 000000

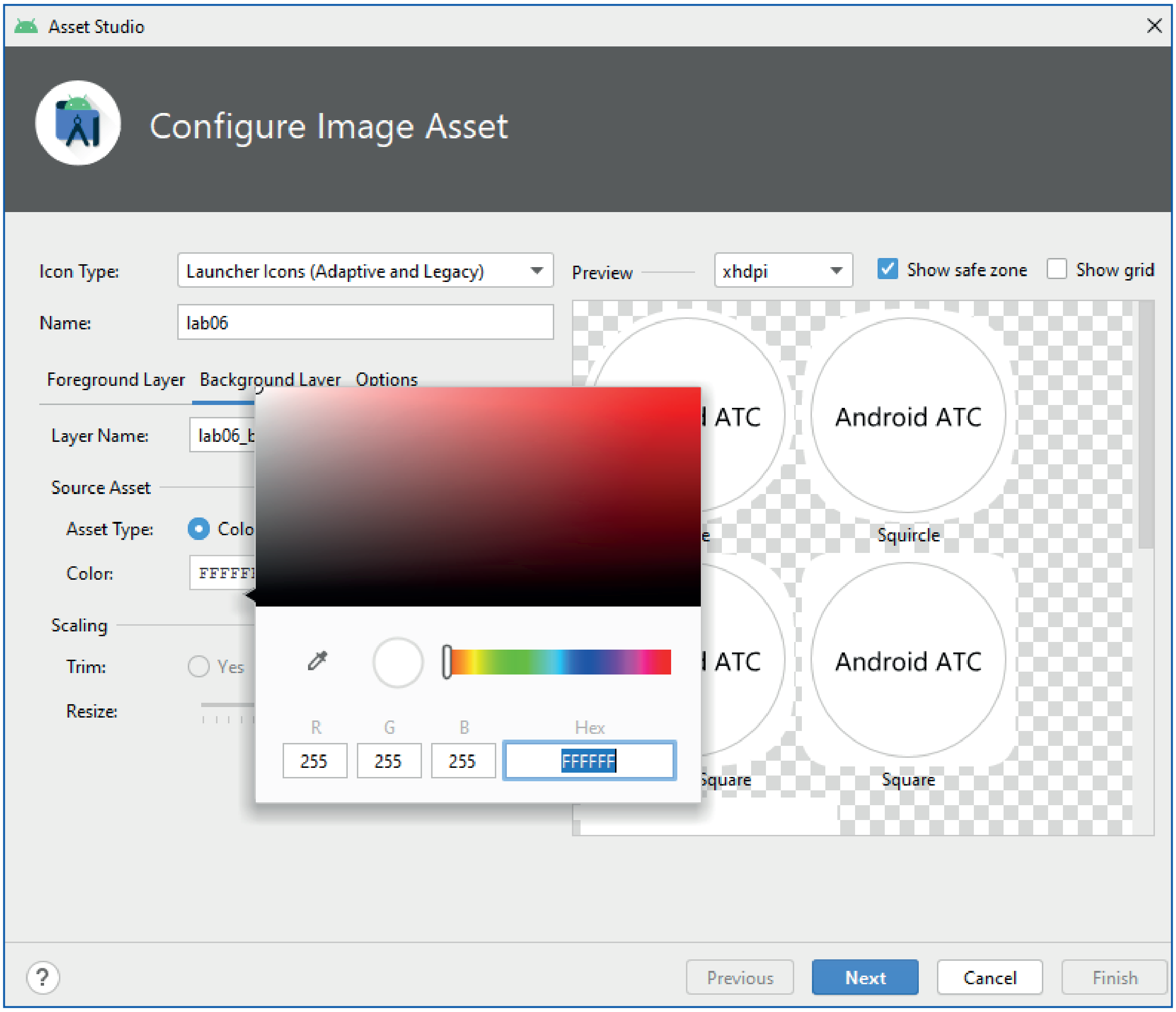
Scaling

Trim: ○ Yes ● No

Resize: 100 %

Preview: xhdpi ☑ Show safe zone □ Show grid

21- Click the **Background Layer** tab, select the **Color** radio button for the **Asset Type**, click the color box to set the color to White (or type the hexadecimal color code value: **FFFFFF**) or select any other color as illustrated in the following figure:



Icon Type: Launcher Icons (Adaptive and Legacy)

Name: lab06

Layer Name: lab06_b

Asset Type: ● Colo

Color: FFFFF

R 255 G 255 B 255 Hex FFFFFF

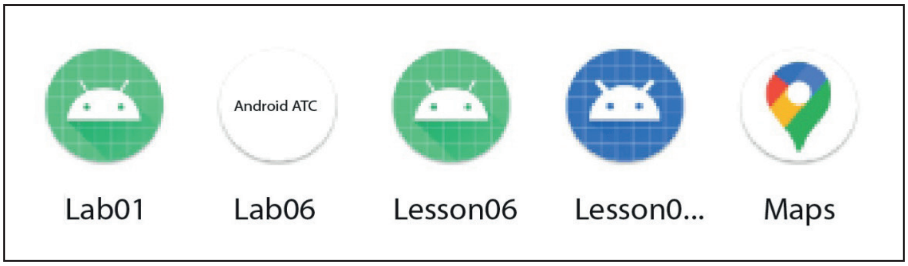
22- Click **Next**, then click **Finish**.

23- Now, open **AndroidManifest.xml** file and change the attributes values of "**android:icon**" and "**android:roundIcon**" to : **lab06** as illustrated in the gray highlighted part of the following code:

```
?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/
android"
    package="com.androidatc.lab06">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/lab06"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/lab06"
        android:supportsRtl="true"
        android:theme="@style/Theme.Lab06">
        <activity android:name=".MainActivity">
```

```
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.
LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

24- **Stop**, then **Run** your app. Click the home button of your phone emulator and check your phone emulator apps list. You should have an icon for **lab06** that has been represented by a text: **Android ATC** with a white background color as illustrated in the following figure:



That was the end of lab 6. Thank you.