Table of Contents

[Install Android Studio 1](#_Toc193462741)

[Software 1](#_Toc193462742)

[Android studio Download link: 1](#_Toc193462743)

[Project Creation 1](#_Toc193462744)

[New Project 1](#_Toc193462745)

[entering in the dashboard 1](#_Toc193462746)

[Note: 1](#_Toc193462747)

[Android Structure folder structure 1](#_Toc193462748)

[Create New Project: 2](#_Toc193462749)

[Manifest 3](#_Toc193462750)

[Activity\_main.xml 3](#_Toc193462751)

[Important: 4](#_Toc193462752)

[Chapter 3 6](#_Toc193462753)

[Android Layout Types: 6](#_Toc193462754)

[View: 6](#_Toc193462755)

[View Group: 6](#_Toc193462756)

[Layouts: 6](#_Toc193462757)

[1. Linear Layout: 6](#_Toc193462758)

[2. Relative Layout(Relative to other Views) 10](#_Toc193462759)

[3. Constraint Layout (Widely used) 13](#_Toc193462760)

[4. Absolute Layout 16](#_Toc193462761)

[5. Table Layout 17](#_Toc193462762)

[View (Widgets) 24](#_Toc193462763)

[Text View 24](#_Toc193462764)

[Edit Text 24](#_Toc193462765)

[Button 25](#_Toc193462766)

[Checkbox 25](#_Toc193462767)

[Radio Button 26](#_Toc193462768)

[Spinner 26](#_Toc193462769)

[Full code: 27](#_Toc193462770)

[File Name: widget\_example 27](#_Toc193462771)

[File Name: Activity\_Widget\_example 27](#_Toc193462772)

[Event Handling 28](#_Toc193462773)

[Event Listeners 28](#_Toc193462774)

[Event Listeners Registration 28](#_Toc193462775)

[Event Handlers 28](#_Toc193462776)

[Differences: 28](#_Toc193462777)

[1. String.xml 29](#_Toc193462778)

[2. Colors.xml 29](#_Toc193462779)

[Chapter 4: Android Activity 30](#_Toc193462780)

[Activity Lifecycle: 30](#_Toc193462781)

[Create new activity: 31](#_Toc193462782)

[Imp 5 Marks 32](#_Toc193462783)

[Creating Multiple Activities: 33](#_Toc193462784)

[What is an Android Intent? 34](#_Toc193462785)

[Types: 36](#_Toc193462786)

[Passing Data between activities 36](#_Toc193462787)

[IMPORTANT FROM THIS CHAPTER 4: 41](#_Toc193462788)

[ Android LifeCycle 41](#_Toc193462789)

[ Android Manifest 41](#_Toc193462790)

[ Android Intent 41](#_Toc193462791)

[ Passing data between activitie 41](#_Toc193462792)

[Chapter 5: UI Fragments, Menus and Dialogs 43](#_Toc193462793)

[Android Fragment: 43](#_Toc193462794)

[Fragment LifeCycle: 43](#_Toc193462795)

[Fragment Manager: 44](#_Toc193462796)

[Activity To Fragment 44](#_Toc193462797)

[Fragment To Fragment through one Activity 50](#_Toc193462798)

[Important Questions 54](#_Toc193462799)

[Menus 54](#_Toc193462800)

[1. Options Menu 54](#_Toc193462801)

[2. Context Menu 55](#_Toc193462802)

[3. Popup Menu 55](#_Toc193462803)

[Dialog: 56](#_Toc193462804)

[Different types of dialog: 56](#_Toc193462805)

# Install Android Studio

## Software

### Android studio Download link:

<https://developer.android.com/studio?gad_source=1&gclid=EAIaIQobChMIqfDAt-uZiwMVPh6DAx09VR1TEAAYASAAEgLetPD_BwE&gclsrc=aw.ds>

## Project Creation

New Project

* Project Name ( Two Language JAVA and Kotlin ) prefer to use JAVA for Now.

Package Name: com.example.projectname

Minimum SDK: (Nougat; Android 7.0)

Build Configuration Language: Kotlin DDL (Default)

*Project Created. May Loading Takes More Time…*

**Now after,**

### entering in the dashboard

Go to the Project File

*Select*

App -> Src -> Main -> java & res

* Java -> Java codes…s
* Res -> AndroidManifest.xml : Declares app components like activities, services, permissions, etc.

# Note:

## Android Structure folder structure

* App
  + Main: Contains main source code and resources
    - Java: stores java/kotlin source code for activities fragment and other logic.
    - Res: contains resources such as layout, strings, images, styles
      * Layout: contains XML Files for UI Designs
      * Drawable: image and vector graphics
      * Values: XML file for app-wide values ( strings, colors )
      * Mipmap: Launcher icons
      * Raw: Audio, Video
      * Menu: XML files menu
* Src
  + Main:
    - ­­AndroidManifest XML => Declares app components like activities, Services,permissions, etc

Build gradle: Defines module specific build settings like dependencies, SDK versions and plugins

1st Portion coverts installation of Android studio both folder/file structure

2nd portion covers creating project and building basic applications.

IMP:

* View Hierarchy
* View Attribute

3rd Portion: Running on Emulators

# Create New Project:

* To Create

Open Android Studio -> New -> New Project

Project Name: Class1Chapter2

Language: change from Kotlin to Java

As default…

Finish

Wait for completing the blue line of importing…

Java -> Com.example right click -> New -> Activity -> Empty Views Activity

Goto: res

## Manifest

and Change activity (Copy past) by removing last two line of code

<activity

android:name=".MainActivity"

android:exported="true"

android:label="Main Activity"

>

<intent-filter>

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

<category android:name="android.intent.category.LAUNCHER"/>

</intent-filter>

</activity>

*Now in*

## Activity\_main.xml

Change this at first place

<LinearLayout xmlns:android=<http://schemas.android.com/apk/res/android>

The codes to use in activity\_main.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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:gravity="center"

tools:context=".MainActivity">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Hello World"

android:textSize="34dp"

android:textColor="@color/design\_default\_color\_primary"

/>

<LinearLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:padding="16dp"

android:orientation="horizontal"

android:gravity="center"

android:background="#ff5733"

>

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Button1"

/>

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Button2"

android:layout\_marginLeft="18dp"

/>

</LinearLayout>

</LinearLayout>

***Explain these for 5 marks questions.***

# Important:

* View Hierarchy (above)
* View Attributes
  + android:layout\_width
  + android:layout\_height
  + etc…
* String Resources
  + String.xml
* Resources and Resources ID: (R.layout, … etc)

Id=> android:id="@+id/main"

* Intent Filters: used to initiates the main\_activity which activity is to be in action to lunch.
  + Android Manifest.xml
  + For launching activities

<intent-filter>

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

<category android:name="android.intent.category.LAUNCHER"/>

</intent-filter>

# Chapter 3

## Android Layout Types:

View

View

View

View

View

View Group

View Group

View Group: Layouts (Liner Layouts, Relative Layouts, Constraints Layouts)

View: Widgets (Button, TextView, ImageView, etc)

### View:

Usually draws something the user can see and Interact.

### View Group:

Invisible constrainers that defines layout structure for view and other view group obj.

## Layouts:

### Linear Layout:

It is a view Group that aligns children in a single direction: Vertically or Horizontally

android: orientation = ””

Vertical Orientation Horizontal Orientation

**Attributes:**

* + android:orientation
  + android:gravity
  + android:layout\_weight
  + android:height
  + android:weight
  + android:padding
  + android:margin

**add this in activity\_linear\_layout\_example or file name:**

<LinearLayout xmlns:android=<http://schemas.android.com/apk/res/android>

**In Androidmanifest.xml**

Change the intent line of code to activity of linear activity.

<activity

android:name=".LinearLayoutExample"

android:exported="true" /> //make it true

#### Code in linerar\_layout\_example

<?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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context=".LinearLayoutExample">

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="#000957"

android:layout\_marginTop="18dp"

android:layout\_weight="1"

/>

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="#344CB7"

android:layout\_marginTop="18dp"

/>

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:background="#577BC1"

android:layout\_marginTop="18dp"

/>

</LinearLayout>

#### LinearChallange: (Code)

<?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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context=".LinearChallange">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="center\_vertical"

>

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:layout\_margin="16dp"

android:background="#123524"

/>

<View

android:layout\_width="0dp"

android:layout\_height="100dp"

android:layout\_margin="16dp"

android:layout\_weight="1"

/>

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:layout\_margin="16dp"

android:background="#123524"

/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="center\_vertical"

android:layout\_marginTop="150dp" >

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:layout\_margin="16dp"

android:background="#123524"

/>

<View

android:layout\_width="0dp"

android:layout\_height="100dp"

android:layout\_margin="16dp"

android:layout\_weight="1"

/>

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:layout\_margin="16dp"

android:background="#123524"

/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_gravity="top"

android:layout\_marginTop="150dp"

>

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:layout\_margin="16dp"

android:background="#123524"

/>

<View

android:layout\_width="0dp"

android:layout\_height="100dp"

android:layout\_margin="16dp"

android:layout\_weight="1"

/>

<View

android:layout\_width="100dp"

android:layout\_height="100dp"

android:layout\_margin="16dp"

android:background="#123524"

/>

</LinearLayout>

</LinearLayout>

### Relative Layout(Relative to other Views)

#### RelativeLayout\_activity

<RelativeLayout 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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".RelativeLayoutExample"

android:padding="18dp"

>

<View

android:id="@+id/view1"

android:layout\_width="150dp"

android:layout\_height="180dp"

android:background="#780C28"

/>

<View

android:id="@+id/view5"

android:layout\_width="150dp"

android:layout\_height="180dp"

android:background="#780C28"

android:layout\_centerVertical="true"

/>

<View

android:id="@+id/view8"

android:layout\_width="150dp"

android:layout\_height="180dp"

android:background="#000"

android:layout\_centerHorizontal="true"

/>

<View

android:id="@+id/view9"

android:layout\_width="150dp"

android:layout\_height="180dp"

android:background="#000"

android:layout\_alignParentBottom="true"

android:layout\_centerHorizontal="true"

/>

<View

android:id="@+id/view2"

android:layout\_width="150dp"

android:layout\_height="180dp"

android:background="#780C28"

android:layout\_alignParentBottom="true">

</View>

<View

android:id="@+id/view6"

android:layout\_width="150dp"

android:layout\_height="180dp"

android:background="#000"

android:layout\_alignParentRight="true"

android:layout\_centerVertical="true"

/>

<View

android:id="@+id/view7"

android:layout\_width="150dp"

android:layout\_height="180dp"

android:background="#000"

android:layout\_centerInParent="true"

/>

<View

android:id="@+id/view3"

android:layout\_width="150dp"

android:layout\_height="180dp"

android:background="#000"

android:layout\_alignParentRight="true"

>

</View>

<View

android:id="@+id/view4"

android:layout\_width="150dp"

android:layout\_height="180dp"

android:background="#000"

android:layout\_alignParentBottom="true"

android:layout\_alignParentRight="true"

>

</View>

</RelativeLayout>

### Constraint Layout (Widely used)

start

start end

button







**Create New Activity**

constraintLayout

<?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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="10dp"

tools:context=".ConstraintLayoutExample">

<!-- <View-->

<!-- android:id="@+id/view1"-->

<!-- android:layout\_width="match\_parent"-->

<!-- android:layout\_height="180dp"-->

<!-- android:background="#FBA598"-->

<!-- app:layout\_constraintStart\_toStartOf="parent"-->

<!-- app:layout\_constraintStart\_toEndOf="parent"-->

<!-- app:layout\_constraintStart\_toTopOf="parent"-->

<!-- />-->

<!-- <View-->

<!-- android:id="@+id/view2"-->

<!-- android:layout\_width="220dp"-->

<!-- android:layout\_height="180dp"-->

<!-- android:background="#2196F3"-->

<!-- app:layout\_constraintStart\_toStartOf="parent"-->

<!-- app:layout\_constraintTop\_toBottomOf="@id/view1"-->

<!--/>-->

<!-- <View-->

<!--android:id="@+id/view3"-->

<!--android:layout\_width="150dp"-->

<!--android:layout\_height="180dp"-->

<!--android:background="#9C27B0"-->

<!-- app:layout\_constraintEnd\_toEndOf="parent"-->

<!-- app:layout\_constraintTop\_toBottomOf="@id/view1"-->

<!--/>-->

<View

android:id="@+id/view1"

android:layout\_width="match\_parent"

android:layout\_height="350dp"

android:background="#FBA598"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintStart\_toEndOf="parent"

app:layout\_constraintStart\_toTopOf="parent"

/>

<View

android:id="@+id/view2"

android:layout\_width="50dp"

android:layout\_height="50dp"

tools:context=".ConstraintLayoutExample"

android:background="#FBA598"

android:layout\_marginTop="10dp"

android:layout\_marginStart="10dp"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

/>

//for circle

<androidx.cardview.widget.CardView

android:layout\_width="50dp"

android:layout\_height="50dp"

app:cardCornerRadius="50dp"

app:layout\_constraintStart\_toStartOf="@id/view2"

app:layout\_constraintTop\_toTopOf="@id/view2"

app:layout\_constraintBottom\_toBottomOf="@id/view2"

app:cardElevation="14dp"

/>

<View

android:id="@+id/view3"

android:layout\_width="200dp"

android:layout\_height="20dp"

android:layout\_marginTop="10dp"

android:background="#FFFFFF"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

/>

<View

android:id="@+id/view4"

android:layout\_width="150dp"

android:layout\_height="20dp"

android:layout\_marginTop="38dp"

android:background="#FFFFFF"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintStart\_toStartOf="@id/view3"

/>

<View

android:id="@+id/view5"

android:layout\_width="320dp"

android:layout\_height="200dp"

android:layout\_marginTop="80dp"

android:layout\_marginRight="10dp"

android:background="#FFFFFF"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintStart\_toStartOf="@id/view2"

app:layout\_constraintEnd\_toEndOf="parent"

/>

<View

android:id="@+id/view6"

android:layout\_width="80dp"

android:layout\_height="20dp"

android:layout\_marginTop="310dp"

android:layout\_marginLeft="1dp"

android:background="#FFFFFF"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintStart\_toStartOf="@id/view2"

/>

<View

android:id="@+id/view7"

android:layout\_width="80dp"

android:layout\_height="20dp"

android:layout\_marginTop="310dp"

android:layout\_marginLeft="125dp"

android:background="#FFFFFF"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintStart\_toStartOf="@id/view2"

/>

<View

android:id="@+id/view8"

android:layout\_width="80dp"

android:layout\_height="20dp"

android:layout\_marginTop="310dp"

android:layout\_marginLeft="235dp"

android:background="#FFFFFF"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintStart\_toStartOf="@id/view2"

/>

</androidx.constraintlayout.widget.ConstraintLayout>

### Absolute Layout

* Specify exact locations (x/y) coordinates of its children
* Less flexible and harder to maintain

Android:layout\_x

Android:layout\_y

(15,25)

(200,200)

(200,100)

#### AbsoluteLayout\_activity

<?xml version="1.0" encoding="utf-8"?>

<AbsoluteLayout 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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".AbsoluteLayout">

<View

android:layout\_width="200dp"

android:layout\_height="100dp"

android:background="@color/purple\_200"

android:layout\_x="15dp"

android:layout\_y="25dp"

/>

<View

android:layout\_width="200dp"

android:layout\_height="100dp"

android:background="@color/teal\_200"

android:layout\_x="200dp"

android:layout\_y="200dp"

/>

<View

android:layout\_width="200dp"

android:layout\_height="100dp"

android:background="@color/purple\_700"

android:layout\_x="15dp"

android:layout\_y="400dp"

/>

</AbsoluteLayout>

### Table Layout

* Arranges children into row and columns
* Consist of number of table row objects, each defining a row.
* Each row has zero or more cells

Row 1

Row 1

Row 3

Column 2

Row 3

Column 1

Row 2

Column 1

Row 2

Column 2

Row 2

Column 3

Row 2

Row 3

#### TableLayout\_activity

<?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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="10dp"

tools:context=".TableLayout">

<TableRow

android:layout\_height="match\_parent"

android:layout\_width="wrap\_content">

<View

android:layout\_height="50dp"

android:layout\_width="match\_parent"

android:layout\_weight="1"

android:background="#468"

/>

</TableRow>

<TableRow

android:layout\_height="wrap\_content"

android:layout\_width="match\_parent"

>

<View

android:layout\_width="1dp"

android:layout\_height="100dp"

android:layout\_weight="1"

android:background="#FFEB3B" />

<View

android:layout\_height="100dp"

android:layout\_width="0dp"

android:layout\_weight="1"

android:background="#673AB7"

/>

<View

android:layout\_height="100dp"

android:layout\_width="0dp"

android:layout\_weight="1"

android:background="#F44336"

/>

</TableRow>

<TableRow

android:layout\_height="wrap\_content"

android:layout\_width="match\_parent"

>

<View

android:layout\_width="1dp"

android:layout\_height="100dp"

android:layout\_weight="1"

android:background="#65FF3B" />

<View

android:layout\_height="100dp"

android:layout\_width="0dp"

android:layout\_weight="1"

android:background="#A3041B40"

/>

</TableRow>

</TableLayout>

#### TableLayoutChallange

<?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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".TableLayoutChallange"

android:padding="18dp"

android:stretchColumns="1,2,3">

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<TextView

android:layout\_weight="1"

android:padding="8dp"

android:text="ICC Ranking of Players:"

android:textSize="20sp"

android:textStyle="bold" />

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#7E7777">

<TextView

android:textSize="18sp"

android:text="Rank"

android:padding="8dp"

android:textStyle="bold"/>

<TextView

android:textSize="18sp"

android:text="Player"

android:padding="8dp"

android:textStyle="bold"/>

<TextView

android:textSize="18sp"

android:text="Team"

android:padding="8dp"

android:textStyle="bold"/>

<TextView

android:textSize="18sp"

android:text="Points"

android:padding="8dp"

android:textStyle="bold"/>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<TextView

android:textSize="16sp"

android:text="1"

android:padding="8dp"

/>

<TextView

android:textSize="16sp"

android:text="Virat Kohli"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="India"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="895"

android:padding="8dp"/>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<TextView

android:textSize="16sp"

android:text="2"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="Rohit Sharma"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="India"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="863"

android:padding="8dp"/>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<TextView

android:textSize="16sp"

android:text="3"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="Faf du Plessis"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="Pakistan"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="834"

android:padding="8dp"/>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<TextView

android:textSize="16sp"

android:text="4"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="Steve Smith"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="Australia"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="820"

android:padding="8dp"/>

</TableRow>

<TableRow

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<TextView

android:textSize="16sp"

android:text="5"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="Ross Taylor"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="NewZealand"

android:padding="8dp"/>

<TextView

android:textSize="16sp"

android:text="817"

android:padding="8dp"/>

</TableRow>

</TableLayout>

# View (Widgets)

## Text View

* **TextView** is used to display text to the user.
* **android:id="@+id/textView1":** The id attribute gives the TextView a unique identifier.
* **android:layout\_width="wrap\_content**": The width of the TextView will adjust to fit the content (text) inside it.
* **android:layout\_height="wrap\_content":** The height will also adjust based on the content.
* **android:text="Registration":** This sets the text that will be displayed.
* **android:textSize="40sp":** The text size is set to 40 scaled pixels, which is a unit of measurement for text size in Android that scales with the user's preferred font size.
* **android:textColor="@color/black":** Sets the text color to black using a color resource **(@color/black).**

<TextView

android:id="@+id/textView1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Registration"

android:textSize="40sp"

android:textColor="@color/black"

/>

## Edit Text

* **EditText** allows the user to input text (e.g., their name).
* **android:id="@+id/editText1":** The id for this EditText field.
* **android:layout\_marginTop="10dp":** Adds a margin of 10dp on the top.
* **android:hint="Enter Your Name":** The hint is displayed when the field is empty, guiding the user to enter their name.

<EditText

android:id="@+id/editText1"

android:layout\_width="296dp"

android:layout\_height="wrap\_content"

android:layout\_marginTop="10dp"

android:hint="Enter Your Name" />

## Button

* A **Button** allows the user to perform an action when clicked.
* **android:id="@+id/btn1":** A unique ID for the Button.
* **android:layout\_width="wrap\_content"** and **android:layout\_height="wrap\_content":** The size of the button adjusts based on content.
* **android:text="CLICK":** The text displayed on the button.
* **android:layout\_margin="10dp":** Adds a margin of 10dp around the button.

<Button

android:id="@+id/btn1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="CLICK"

android:layout\_margin="10dp"

/>

## Checkbox

* A **CheckBox** allows the user to select or deselect an option.
* **android:id="@+id/checkbox1":** A unique ID for the CheckBox.
* **android:text="Accept Terms and Condition":** Sets the text label next to the checkbox.

<CheckBox

android:id="@+id/checkbox1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Accept Terms and Condition"

/>

## Radio Button

* **android:id="@+id/radioBtnFemale":** A unique ID for the second radio button.
* **android:text="Female":** Sets the text next to the radio button.
* **android:padding="20dp":** Adds 20dp padding around the radio button.

<RadioGroup

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:orientation="horizontal"

android:layout\_margin="10dp"

>

<RadioButton

android:id="@+id/radioBtnMale"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Male"

android:padding="20dp"

/>

<RadioButton

android:id="@+id/radioBtnFemale"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Female"

android:padding="20dp"

/>

</RadioGroup>

## Spinner

* A **Spinner** is a dropdown menu, allowing the user to select an item from a list of options.
* **android:id="@+id/spinner":** A unique ID for the Spinner.
* **android:entries="@array/spinner\_items":** Refers to a list of items stored in the **@array/spinner\_items** resource (likely defined in a strings.xml file).

<Spinner

android:id="@+id/spinner"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:entries="@array/spinner\_items"

/>

## Full code:

### File Name: widget\_example Registration Form

#### Activity\_Main.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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:gravity="center"

tools:context=".MainActivity"

android:background="#E4E4E4"

>

<LinearLayout

android:layout\_width="350dp"

android:layout\_height="wrap\_content"

android:background="@drawable/main\_layout"

android:elevation="20dp"

android:translationZ="10dp"

android:orientation="vertical"

android:padding="30dp"

>

<TextView

android:layout\_width="match\_parent"

android:textAlignment="center"

android:layout\_height="30sp"

android:layout\_margin="20dp"

android:textStyle="bold"

android:textSize="20sp"

android:text="Student Registration"

/>

<EditText

android:id="@+id/edit1"

android:layout\_width="match\_parent"

android:textSize="15sp"

android:layout\_height="wrap\_content"

android:hint="Enter Your Name"

android:padding="10dp"

android:background="@drawable/edit\_background"

/>

<EditText

android:layout\_width="match\_parent"

android:id="@+id/edit2"

android:textSize="15sp"

android:layout\_height="wrap\_content"

android:hint="Enter Your Email"

android:layout\_marginTop="10dp"

android:padding="10dp"

android:background="@drawable/edit\_background"

/>

<EditText

android:id="@+id/edit3"

android:layout\_width="match\_parent"

android:textSize="15sp"

android:layout\_height="wrap\_content"

android:hint="Enter Your Age"

android:layout\_marginTop="10dp"

android:padding="10dp"

android:background="@drawable/edit\_background"

/>

<TextView

android:layout\_width="50dp"

android:layout\_height="20dp"

android:text="Gender"

/>

<RadioGroup

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:orientation="horizontal"

android:layout\_margin="10dp">

<RadioButton

android:id="@+id/radioBtnMale"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Male"

/>

<RadioButton

android:id="@+id/radioBtnFemale"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Female"

/>

</RadioGroup>

<TextView

android:text="Select Course"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

/>

<Spinner

android:id="@+id/spinner"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal"

android:entries="@array/spinner\_items"

/>

<CheckBox

android:id="@+id/checkbox1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="I accept Terms &amp; Condition"

/>

<Button

android:id="@+id/btn1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Submit"

/>

</LinearLayout>

</LinearLayout>

#### MainActivity.java

package com.example.studentregistrationform;

import android.content.Intent;

import android.os.Bundle;

import android.widget.EditText;import android.view.View;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.CheckBox;

import android.widget.CompoundButton;

import android.widget.EditText;

import android.widget.RadioButton;

import android.widget.Spinner;

import android.widget.TextView;

import android.widget.Toast;

import androidx.activity.EdgeToEdge;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.graphics.Insets;

import androidx.core.view.ViewCompat;

import androidx.core.view.WindowInsetsCompat;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

EdgeToEdge.enable(this);

setContentView(R.layout.activity\_main);

EditText editOne =findViewById(R.id.edit1);

EditText editTwo =findViewById(R.id.edit2);

EditText editThree =findViewById(R.id.edit3);

CheckBox checkBoxOne = findViewById(R.id.checkbox1);

RadioButton radioButtonMale = findViewById(R.id.radioBtnMale);

RadioButton radioButtonFemale = findViewById(R.id.radioBtnFemale);

Spinner spin = findViewById(R.id.spinner);

Button btnOne = findViewById(R.id.btn1);

btnOne.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

String gender="Not selected";

String selectedOption;

selectedOption= spin.getSelectedItem().toString();

gender=radioButtonMale.isChecked() ?" Male": radioButtonFemale.isChecked() ? " Female": " not selected";

boolean isChecked=checkBoxOne.isChecked();

if(isChecked){

Intent intent = new Intent(MainActivity.this, DisplayList.class);

intent.putExtra("Name",editOne.getText().toString());

intent.putExtra("Email",editTwo.getText().toString());

intent.putExtra("Age",editThree.getText().toString());

intent.putExtra("Gender",gender);

intent.putExtra("Course",selectedOption);

startActivity(intent);

}

}

});

}

}

#### Activity\_Display\_List.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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:background="#E4E4E4E4"

tools:context=".DisplayList">

<Button

android:id="@+id/btnBack"

android:layout\_width="100dp"

android:layout\_height="wrap\_content"

android:text="← Back"

android:backgroundTint="@android:color/transparent"

android:textColor="@color/purple\_500"

android:textSize="14sp"

android:layout\_marginTop="30dp"

/>

<LinearLayout

android:layout\_width="350dp"

android:layout\_height="450dp"

android:background="@drawable/main\_layout"

android:orientation="vertical"

android:padding="20dp"

android:layout\_gravity="center"

android:layout\_marginTop="60dp"

>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="35dp"

android:textSize="30sp"

android:textColor="@color/black"

android:text="Student Details: "

android:textAlignment="center"

/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:text="Name: "

android:layout\_marginTop="50dp"

android:textColor="@color/black"

/>

<TextView

android:id="@+id/view1"

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:text="Email: "

android:textColor="@color/black"

/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:id="@+id/view2"

/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:text="Age: "

android:textColor="@color/black"

/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:id="@+id/view3"

/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:text="Gender"

android:textColor="@color/black"

/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:id="@+id/view4"

/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:text="Selectected Course"

android:textColor="@color/black"

/>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:id="@+id/view5"

/>

</LinearLayout>

</LinearLayout>

#### DisplayList.java

package com.example.studentregistrationform;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

import androidx.activity.EdgeToEdge;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.graphics.Insets;

import androidx.core.view.ViewCompat;

import androidx.core.view.WindowInsetsCompat;

public class DisplayList extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

EdgeToEdge.enable(this);

setContentView(R.layout.activity\_display\_list);

String name = getIntent().getStringExtra("Name");

String email = getIntent().getStringExtra("Email");

String age = getIntent().getStringExtra("Age");

String gender = getIntent().getStringExtra("Gender");

String course = getIntent().getStringExtra("Course");

Button btnBack = findViewById(R.id.btnBack);

btnBack.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

finish(); // Finish DisplayList activity

}

});

TextView view1 = findViewById(R.id.view1);

view1.setText(name);

TextView view2 = findViewById(R.id.view2);

view2.setText(email);

TextView view3 = findViewById(R.id.view3);

view3.setText(age);

TextView view4 = findViewById(R.id.view4);

view4.setText(gender);

TextView view5 = findViewById(R.id.view5);

view5.setText(course);

}

}

# Event Handling

* Events are ways to collect data about users
* Interactions with interactive components of Applications
* Event queue in FIFO(first in first out) order
* Captures events -> Process Events
  + Capture events -> EventListner
  + Process the Event -> Handles Event
* Event Listeners vs Event Handlers
  + Event Listners

Void.OnclickListner() -> interface

* + Event Handlers

Onclick()

Called inside Listeners

### Event Listeners

* + Interface in the view class that contains a single callback method.
  + Methods will be called by Android Framework where the view to which the listeners is registered -> triggered by user.

### Event Listeners Registration

* + Event handler gets registered with an Event Listener.

### Event Handlers

* + Event Listener calls the event handlers -> Methods that actually handles for event.

## Differences:

|  |  |
| --- | --- |
| Event Handlers | Event Listners |
| 1. Onclick() | OnclickListener()  **User** -> touch/clicks/focuses upon any widget (Button, text, image, etc). |
| 1. onLongClick() | onLongClickListerner()  For one or more **users:** like above |
| 1. onFocusChange() | onFocusChangeListener()  Wdigat looses its focus i.e. User goes away from the view item. |
| 1. onMenuItemClick() | OnMenuItemClickListerner() |
| 1. onTouch()   onKey() | ---------------------------------------------- |

Goto main -> res -> then below for string and colors

## String.xml

* String -> Basically used for string throughout your application.
* String Array -> Spinner Items

## Colors.xml

* Drawable: Concept of graphics that can be ( Png, Jpg, Svg, etc) drawn on the screen.
* Android:src=”@drawable/imagename”

**Example:** Android:src=”@drawable/city\_nature\_svgreen.com”

# Chapter 4: Android Activity

## Activity Lifecycle:

onCreate()

onStart() onRestart()

onResume()

onPause()

onStop()

onDestroy

User navigates to the activity

OnRestart()

App with higher priority need memory

App process kill

Activity is finished or destroying by the system

Activity Shutdown

onDestroy()

User returns to activity

OnStop()

User returns to activity

Activity Launched

On Pause()

On Activity Running

On Resume()

OnStart()

On Create()

Another activity comes in fragment

### Create new activity:

LifecycleActivity:

In LifecycleActivity.java

package com.example.chapter4;

import android.os.Bundle;

import android.util.Log;

import androidx.activity.EdgeToEdge;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.graphics.Insets;

import androidx.core.view.ViewCompat;

import androidx.core.view.WindowInsetsCompat;

public class LifecycleActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

EdgeToEdge.enable(this);

setContentView(R.layout.activity\_lifecycle);

Log.d("BCA\_ON\_CREATE","Activity is created");

}

@Override

protected void onStart() {

super.onStart();

Log.d("BCA\_ON\_START","Activity is startes");

}

@Override

protected void onRestart() {

super.onRestart();

Log.d("BCA\_ON\_RESTART","Activity is Restarted");

}

@Override

protected void onPause() {

super.onPause();

Log.d("BCA\_ON\_PAUSE","Activity is Paused");

}

@Override

protected void onResume() {

super.onResume();

Log.d("BCA\_ON\_RESUME","Activity is Resumed");

}

@Override

protected void onStop() {

super.onStop();

Log.d("BCA\_ON\_STOP","Activity is stopped");

}

@Override

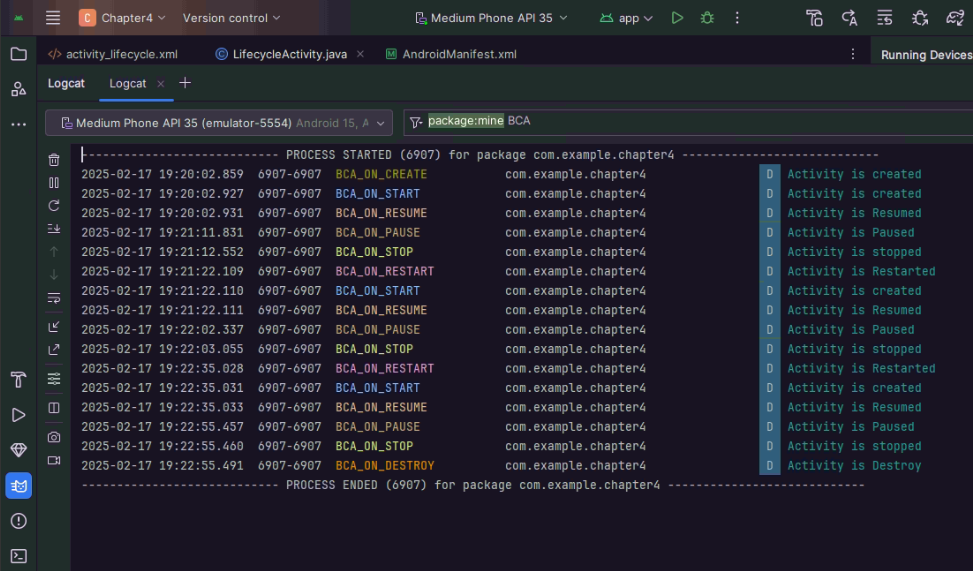
protected void onDestroy() {

super.onDestroy();

Log.d("BCA\_ON\_DESTROY","Activity is Destroy");

}

}



## Imp 5 Marks

LifeCycle (with code necessary as above)

onCreate() -> onStart() -> onResume() -> onPause() -> onStop() -> onDestroy()

onCreate()

* Triggers when the system first creates the activity
* onCreation, the activity enters the created state
* used for startup logic

onStart()

* makes the activity visible to the user
* this method is where the app initializes the code that maintains OI.

onResume()

* Enters resumed state -> it comes to foreground -> system invokes this callback.
* User can interact with the app

onPause()

* Called when we take the focus away from the app like receiving phone cells, device turning off, users navigating (leaving your activity) to another activity.

onStop()

* when activity is no longer visible to the user.

onDestroy()

* when the activity is finishing (user dismissing or terminating the app) or during window rotations.

onRestart()

* when app is minimized and enters in the app again the status changes from onPause and onStop to start again known as Restart.

## Creating Multiple Activities:

Goto Second

Second Activity.java

This is second Activity !!

First Activity.java

This is first Activity

* Set manifest as first activity.
* Run to get empty in emulator without any error.

### What is an Android Intent?

An **Intent** in Android is like a message that lets one part of an app communicate with another part (like starting a new screen or activity). You can also use intents to send data or ask other apps to do something for you.

* it is a messaging object used to request an action from another component (such as activity, service or broadcast receiver).
* It allows communication between different parts of an app or between different apps.
* Explicit Intent: You know exactly which component you want to use (like opening a specific screen).
* Implicit Intent: You just describe the action you want, and Android decides which app can handle it (like opening a webpage).
* Intents are used for navigation, data passing, and starting other tasks in Android apps.

#### To start and send from one activity to another activity

Intent intent = new Intent(this, NewActivity.class);

startActivity(intent);

#### To parse data from one activity to another

Intent intent = new Intent(this, SecondActivity.class);

intent.putExtra("message", "Hello!");

startActivity(intent);

#### To open a website

Intent intent = new Intent(Intent.ACTION\_VIEW, Uri.parse("http://www.example.com"));

startActivity(intent);

#### To make a phone call

Intent intent = new Intent(Intent.ACTION\_DIAL, Uri.parse("tel:+1234567890"));

startActivity(intent);

#### To send an Email

Intent intent = new Intent(Intent.ACTION\_SENDTO);

intent.setData(Uri.parse("mailto:")); // This ensures only email apps can handle the intent

intent.putExtra(Intent.EXTRA\_EMAIL, new String[]{"recipient@example.com"}); // Set recipient email

intent.putExtra(Intent.EXTRA\_SUBJECT, "Subject of the email"); // Set subject

intent.putExtra(Intent.EXTRA\_TEXT, "Body of the email"); // Set the email body

// If there's an email app available, it will open to send the email

if (intent.resolveActivity(getPackageManager()) != null) {

startActivity(intent);

}

#### To pick image

Intent intent = new Intent(Intent.ACTION\_PICK);

intent.setType("image/\*");

startActivity(intent);

#### To View in Map

Intent intent = new Intent(Intent.ACTION\_VIEW, Uri.parse("geo: 37.7749, -122.4194?q= **San,** **Francisco** "));

startActivity(intent);

**My Application:**

Android OS

Start Activity

Activity Message

Intent

First Activity

(Intent)

Second Activity

Content

=Second Activiyty

***Figure: Flow of Application between two Activity***

***To Receive the Connection from one activity to another activity***

Intent intent = getIntent();

### Types:

#### Explict Intent

* Used to start specific components within the same app. (Navigation from one activity to another)
* You know exactly which component you want to use (like opening a specific screen).

#### Implicit Intent:

* Used when you donot specify the exact component, instead you decleare an action to be performed.
* Eg: sharing a file, opening URl to the browser, etc.
* You just describe the action you want, and Android decides which app can handle it (like opening a webpage).

### Passing Data between activities

* Getting result back from child activity

Child Activity.java

This is Child Activity

Parent Activity.java

This is parent Activity

Back with child message

Goto Parent Activity with child message

Goto Child Activity

**Parent Activity**

*Parentactivity.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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:gravity="center"

tools:context=".ParentActivity">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Parent Activity"

android:textSize="32sp"

android:textStyle="bold"/>

<Button

android:id="@id/parentBtn"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="go to child activity"

android:layout\_marginTop="12dp"/>

<TextView

android:id="@+id/resultTextView"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textSize="32sp"/>

</LinearLayout>

*Parent\_Activity.java*

package com.example.chapter4;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

import androidx.activity.EdgeToEdge;

import androidx.activity.result.ActivityResultLauncher;

import androidx.activity.result.contract.ActivityResultContracts;

import androidx.annotation.Nullable;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.graphics.Insets;

import androidx.core.view.ViewCompat;

import androidx.core.view.WindowInsetsCompat;

public class ParentActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

EdgeToEdge.enable(this);

setContentView(R.layout.activity\_parent);

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {

Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());

v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);

return insets;

});

// **above method is this**

//Create activityResultLauncher Object

ActivityResultLauncher<Intent> activityResultLauncher;

//code to view the message or data retrived from the child activity

activityResultLauncher=registerForActivityResult(

new ActivityResultContracts.StartActivityForResult(),

//lamda method to view in parent activity retrived from the child activity

result->{

// to view the message by id textview

TextView resultTextView=findViewById(R.id.resultTextView);

//to get the data from child activity

Intent data=result.getData();

// if result code meet from child activity

if(result.getResultCode()==100)

{

String message=data.getStringExtra("Message");

resultTextView.setText(message);

}

}

);

Button parentBtn=findViewById(R.id.parentBtn);

parentBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent= new Intent(ParentActivity.this,ChildActivity.class);

// startActivityForResult(intent,100);

activityResultLauncher.launch(intent);

}

});

}

**//Important also can use the above method for the same task below**

// @Override

// protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {

// super.onActivityResult(requestCode, resultCode, data);

// TextView resultTextView=findViewById(R.id.resultTextView);

// if(resultCode==100)

// {

// String message=data.getStringExtra("Message");

// resultTextView.setText(message);

// }

// }

}

**Child Activity**

*Child\_Activity.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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:gravity="center"

tools:context=".ChildActivity">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Child Activity"

android:textSize="32sp"

android:textStyle="bold"/>

<Button

android:id="@id/childBtn"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="go to Parent activity with child message"

android:layout\_marginTop="12dp"/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textSize="32sp"/>

</LinearLayout>

*Child\_Activity.java*

package com.example.chapter4;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import androidx.activity.EdgeToEdge;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.graphics.Insets;

import androidx.core.view.ViewCompat;

import androidx.core.view.WindowInsetsCompat;

public class ChildActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

EdgeToEdge.enable(this);

setContentView(R.layout.activity\_child);

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {

Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());

v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);

return insets;

});

Button childBtn=findViewById(R.id.childBtn);

childBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent resultIntent=new Intent();

resultIntent.putExtra("Message","Message from child activity");

setResult(100,resultIntent);

finish();

}

});

}

}

## IMPORTANT FROM THIS CHAPTER 4:

### Android LifeCycle

### Android Manifest

### Android Intent

### Passing data between activitie

# Chapter 5: UI Fragments, Menus and Dialogs

## Android Fragment:

* It is a part of activity, which is also known as sub-activity
* There can be more than one fragment in the activity
* Fragment represents multiple screen in one activity

## Fragment LifeCycle:

onCreateView()

onCreate()

onAttach()

Fragment is added

Fragment returns to the layout from the stack

Fragment is active

onResume()

onStart()

onActivityCreated()

onDestroy()

onDestroyView()

onStop()

onPause()

Fragment is destroyed.

onDetach()

onAttach(): Called only once when attached to activity.

onCreate(): Initialize fragment.

onStart(): makes fragment Visible.

onResume(): makes fragment interactive.

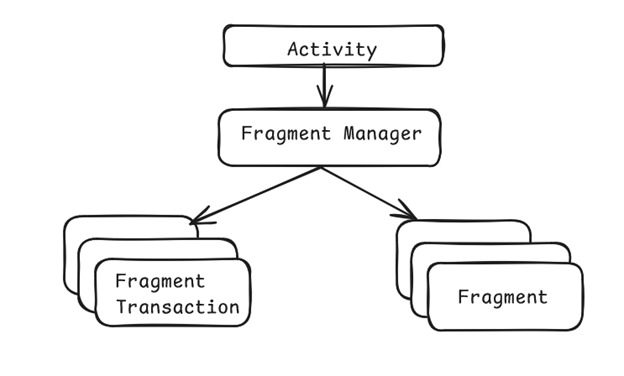
fragment is active:

* + user navigate backwards or fragment is removed/replaced.
  + the fragment returns to the layout from back stack.

onPause(): fragment no longer interactive.

onStop(): Fragment no longer visible.

## Fragment Manager:



* Responsible for managing fragments and adding the view to activity’s view hierarchy.
* Dynamically add, replace, remove fragments at runtime
* Also maintains a back stack of fragment transactions that we navigate

### Activity To Fragment

#### Fragment\_Activity.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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context=".fragment">

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/btntransitionOne"

android:text="Transition to first fragment"

android:layout\_marginHorizontal="24dp" />

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/btntransitionTwo"

android:text="Transition to second fragment"

android:layout\_marginHorizontal="24dp" />

<FrameLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/transitionFrameLayout"

/>

</LinearLayout>



#### Fragment\_Activity.java

package com.example.chapter5;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import androidx.activity.EdgeToEdge;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.graphics.Insets;

import androidx.core.view.ViewCompat;

import androidx.core.view.WindowInsetsCompat;

import androidx.fragment.app.FragmentManager;

public class fragment extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

EdgeToEdge.enable(this);

setContentView(R.layout.activity\_fragment);

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {

Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());

v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);

return insets;

});

Button btnOne=findViewById(R.id.btntransitionOne);

Button btnTwo=findViewById(R.id.btntransitionTwo);

btnOne.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

OneFragment oneFragment=new OneFragment();

FragmentManager fragmentManager=getSupportFragmentManager();

fragmentManager.beginTransaction()

.add(R.id.transitionFrameLayout,oneFragment)

.commit();

}

});

btnTwo.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

TwoFragment TwoFragment=new TwoFragment();

FragmentManager fragmentManager=getSupportFragmentManager();

fragmentManager.beginTransaction()

.add(R.id.transitionFrameLayout,TwoFragment)

.commit();

}

});

}

}

#### Fragment\_one.xml

<?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:gravity="center"

android:orientation="vertical"

android:background="#CD08C3"

tools:context=".OneFragment">

<!-- TODO: Update blank fragment layout -->

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="FragmentOne"

android:textSize="32dp"/>

</LinearLayout>



#### FragmentOne.java

*(No Change anything)*

package com.example.chapter5;

import android.os.Bundle;

import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

/\*\*

\* A simple {@link Fragment} subclass.

\* Use the {@link OneFragment#newInstance} factory method to

\* create an instance of this fragment.

\*/

public class OneFragment extends Fragment {

// TODO: Rename parameter arguments, choose names that match

// the fragment initialization parameters, e.g. ARG\_ITEM\_NUMBER

private static final String ARG\_PARAM1 = "param1";

private static final String ARG\_PARAM2 = "param2";

// TODO: Rename and change types of parameters

private String mParam1;

private String mParam2;

public OneFragment() {

// Required empty public constructor

}

/\*\*

\* Use this factory method to create a new instance of

\* this fragment using the provided parameters.

\*

\* @param param1 Parameter 1.

\* @param param2 Parameter 2.

\* @return A new instance of fragment OneFragment.

\*/

// TODO: Rename and change types and number of parameters

public static OneFragment newInstance(String param1, String param2) {

OneFragment fragment = new OneFragment();

Bundle args = new Bundle();

args.putString(ARG\_PARAM1, param1);

args.putString(ARG\_PARAM2, param2);

fragment.setArguments(args);

return fragment;

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

if (getArguments() != null) {

mParam1 = getArguments().getString(ARG\_PARAM1);

mParam2 = getArguments().getString(ARG\_PARAM2);

}

}

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container,

Bundle savedInstanceState) {

// Inflate the layout for this fragment

return inflater.inflate(R.layout.fragment\_one, container, false);

}

}

#### Fragment\_two.xml

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout 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:gravity="center"

android:orientation="vertical"

android:background="#CD03"

tools:context=".TwoFragment">

<!-- TODO: Update blank fragment layout -->

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_gravity="center"

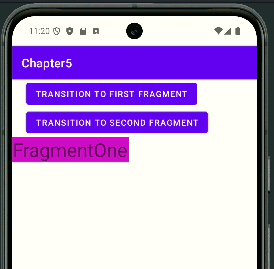
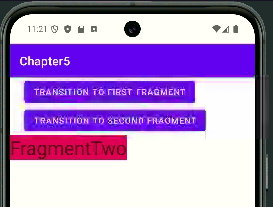
android:text="FragmentTwo"

android:textSize="32dp" />

</FrameLayout>



#### Output:



## Fragment To Fragment through one Activity

#### FragmentToFragment\_Activity.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:id="@+id/main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context=".FragementToFragmentActivity">

<FrameLayout

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/transitionFrameLayout"

/>

</LinearLayout>

#### FragmentToFragmentActivity.Java

package com.example.chapter5;

import android.os.Bundle;

import android.view.View;

import androidx.activity.EdgeToEdge;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.graphics.Insets;

import androidx.core.view.ViewCompat;

import androidx.core.view.WindowInsetsCompat;

import androidx.fragment.app.FragmentManager;

public class FragementToFragmentActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

EdgeToEdge.enable(this);

setContentView(R.layout.activity\_fragement\_to\_fragment);

//create an fragmentone object

FragmentToFragmentOne fragmentToFragmentOne = new FragmentToFragmentOne();

//fragment manager to manage the transition of fragment on activity

FragmentManager fragmentManager = getSupportFragmentManager();

//for transition in activity

fragmentManager.beginTransaction()

.add(R.id.transitionFrameLayout, fragmentToFragmentOne)

.commit();

}

}

#### Fragment\_To\_Fragment\_one.xml

<?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=".FragmentToFragmentOne">

<!-- TODO: Update blank fragment layout -->

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="This is the first fragement"

android:layout\_marginTop="100dp"

android:textSize="32dp"

android:id="@+id/fragmentOne"

/>

<Button

android:id="@+id/btnClickForward"

android:layout\_width="wrap\_content"

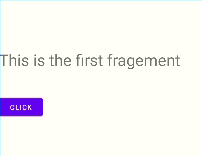
android:layout\_height="wrap\_content"

android:text="Click"

android:layout\_marginTop="50dp"

/>

</LinearLayout>



#### FragmentToFragmentone.java

package com.example.chapter5;

import android.os.Bundle;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentManager;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

/\*\*

\* A simple {@link Fragment} subclass.

\* Use the {@link FragmentToFragmentOne#newInstance} factory method to

\* create an instance of this fragment.

\*/

public class FragmentToFragmentOne extends Fragment {

// TODO: Rename parameter arguments, choose names that match

// the fragment initialization parameters, e.g. ARG\_ITEM\_NUMBER

private static final String ARG\_PARAM1 = "param1";

private static final String ARG\_PARAM2 = "param2";

// TODO: Rename and change types of parameters

private String mParam1;

private String mParam2;

public FragmentToFragmentOne() {

// Required empty public constructor

}

/\*\*

\* Use this factory method to create a new instance of

\* this fragment using the provided parameters.

\*

\* @param param1 Parameter 1.

\* @param param2 Parameter 2.

\* @return A new instance of fragment FragmentToFragmentOne.

\*/

// TODO: Rename and change types and number of parameters

public static FragmentToFragmentOne newInstance(String param1, String param2) {

FragmentToFragmentOne fragment = new FragmentToFragmentOne();

Bundle args = new Bundle();

args.putString(ARG\_PARAM1, param1);

args.putString(ARG\_PARAM2, param2);

fragment.setArguments(args);

return fragment;

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

if (getArguments() != null) {

mParam1 = getArguments().getString(ARG\_PARAM1);

mParam2 = getArguments().getString(ARG\_PARAM2);

}

}

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container,

Bundle savedInstanceState) {

**// Inflate the layout for this fragment**

View viewOne= inflater.inflate(R.layout.fragment\_to\_fragment\_one, container, false);

//now we have write from here

//button taken by their id

Button btnClick=viewOne.findViewById(R.id.btnClickForward);

//onclick event

btnClick.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

FragmentToFragmentTwo fragmentToFragmentTwo= new FragmentToFragmentTwo();

FragmentManager fragmentManager=requireActivity().getSupportFragmentManager();

fragmentManager.beginTransaction()

.replace(R.id.transitionFrameLayout,fragmentToFragmentTwo)

.commit();

}

});

return viewOne;

}

}

#### FragmentToFragmentTwo.xml

<?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=".FragmentToFragmentTwo">

<!-- TODO: Update blank fragment layout -->

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="This is the second fragement"

android:layout\_marginTop="400sp"

android:textSize="24dp"

/>

</LinearLayout>

#### FragmentToFragmentTwo.Java

//No Change

## Develop Calculator

* Develop an Android Application to calculate area and perimeter of rectangle. Application must calculate and display area in one fragment and perimeter in another fragment.
* We have to create one Activity (eg. : CalculatorAppActivity) and Three Fragments.
  + CalculatorFormFragment
  + CalculatorAreaFragment
  + CalculatorPerimeterFragemnt

### CalculatorAreaFragment

#### CalculatorAreaFragment.xml

<?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=".CalculatorAreaFragment">

<!-- TODO: Update blank fragment layout -->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:layout\_marginTop="50dp">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Calculated Area:"

android:layout\_marginTop="50dp"

android:textSize="40dp"

android:textStyle="bold"

android:layout\_marginLeft="60dp"

/>

<TextView

android:id="@+id/calculatedArea"

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:layout\_marginTop="150dp"

/>

</LinearLayout>

</LinearLayout>

#### CalculatorAreaFragment.java

package com.example.chapter5;

import android.os.Bundle;

import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.TextView;

/\*\*

\* A simple {@link Fragment} subclass.

\* Use the {@link CalculatorAreaFragment#newInstance} factory method to

\* create an instance of this fragment.

\*/

public class CalculatorAreaFragment extends Fragment {

// TODO: Rename parameter arguments, choose names that match

// the fragment initialization parameters, e.g. ARG\_ITEM\_NUMBER

private static final String ARG\_PARAM1 = "param1";

private static final String ARG\_PARAM2 = "param2";

// TODO: Rename and change types of parameters

private String mParam1;

private String mParam2;

public CalculatorAreaFragment() {

// Required empty public constructor

}

/\*\*

\* Use this factory method to create a new instance of

\* this fragment using the provided parameters.

\*

\* @param param1 Parameter 1.

\* @param param2 Parameter 2.

\* @return A new instance of fragment CalculatorAreaFragment.

\*/

// TODO: Rename and change types and number of parameters

public static CalculatorAreaFragment newInstance(String param1, String param2) {

CalculatorAreaFragment fragment = new CalculatorAreaFragment();

Bundle args = new Bundle();

args.putString(ARG\_PARAM1, param1);

args.putString(ARG\_PARAM2, param2);

fragment.setArguments(args);

return fragment;

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

if (getArguments() != null) {

mParam1 = getArguments().getString(ARG\_PARAM1);

mParam2 = getArguments().getString(ARG\_PARAM2);

}

}

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container,

Bundle savedInstanceState) {

// Inflate the layout for this fragment

View view= inflater.inflate(R.layout.fragment\_calculator\_area, container, false);

Bundle bundle = getArguments();

String area = bundle.getString("Area");

TextView resultText = view.findViewById(R.id.calculatedArea);

resultText.setText("Area: " + area);

return view;

}

}

### CalculatorPerimeterFragment

#### CalculatorPerimeterFragment.xml

<?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"

tools:context=".CalculatorPerimeterFragment">

<!-- TODO: Update blank fragment layout -->

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:layout\_marginTop="50dp">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Calculated Perimeter:"

android:layout\_marginTop="50dp"

android:textSize="30dp"

android:textStyle="bold"

android:layout\_marginLeft="60dp"

/>

<TextView

android:id="@+id/calculatedPerimeter"

android:layout\_width="match\_parent"

android:layout\_height="30dp"

android:textSize="20sp"

android:layout\_marginTop="150dp"

/>

</LinearLayout>

</LinearLayout>

#### CalculatorPerimeterFragment.Java

package com.example.chapter5;

import android.os.Bundle;

import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.TextView;

/\*\*

\* A simple {@link Fragment} subclass.

\* Use the {@link CalculatorPerimeterFragment#newInstance} factory method to

\* create an instance of this fragment.

\*/

public class CalculatorPerimeterFragment extends Fragment {

// TODO: Rename parameter arguments, choose names that match

// the fragment initialization parameters, e.g. ARG\_ITEM\_NUMBER

private static final String ARG\_PARAM1 = "param1";

private static final String ARG\_PARAM2 = "param2";

// TODO: Rename and change types of parameters

private String mParam1;

private String mParam2;

public CalculatorPerimeterFragment() {

// Required empty public constructor

}

/\*\*

\* Use this factory method to create a new instance of

\* this fragment using the provided parameters.

\*

\* @param param1 Parameter 1.

\* @param param2 Parameter 2.

\* @return A new instance of fragment CalculatorPerimeterFragment.

\*/

// TODO: Rename and change types and number of parameters

public static CalculatorPerimeterFragment newInstance(String param1, String param2) {

CalculatorPerimeterFragment fragment = new CalculatorPerimeterFragment();

Bundle args = new Bundle();

args.putString(ARG\_PARAM1, param1);

args.putString(ARG\_PARAM2, param2);

fragment.setArguments(args);

return fragment;

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

if (getArguments() != null) {

mParam1 = getArguments().getString(ARG\_PARAM1);

mParam2 = getArguments().getString(ARG\_PARAM2);

}

}

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container,

Bundle savedInstanceState) {

// Inflate the layout for this fragment

View view= inflater.inflate(R.layout.fragment\_calculator\_perimeter, container, false);

Bundle bundle = getArguments();

String perimeter = bundle.getString("Perimeter");

TextView resultText = view.findViewById(R.id.calculatedPerimeter);

resultText.setText("Perimeter: " +perimeter);

return view; }

}

### CalculatorFormFragment

#### CalculatorFormFragment.xml

<?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=".CalculatorFormFragment">

<!-- TODO: Update blank fragment layout -->

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="50dp"

android:layout\_marginLeft="150dp"

android:textStyle="bold"

android:textSize="32dp"

android:text="Calculator" />

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Enter Length"

android:layout\_marginTop="20sp"

android:layout\_marginLeft="100dp"

/>

<EditText

android:id="@+id/enteredlength"

android:layout\_width="wrap\_content"

android:textSize="15sp"

android:layout\_height="wrap\_content"

android:hint="Enter the Length"

android:layout\_marginTop="10dp"

android:layout\_marginLeft="100dp"

android:padding="10dp"

/>

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Enter Breadth"

android:layout\_marginTop="20sp"

android:layout\_marginLeft="100dp"

/>

<EditText

android:id="@+id/enteredbreadth"

android:layout\_width="wrap\_content"

android:textSize="15sp"

android:layout\_height="wrap\_content"

android:hint="Enter the breadth"

android:layout\_marginTop="10dp"

android:layout\_marginLeft="100dp"

android:padding="10dp"

/>

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Calculate Area"

android:layout\_marginTop="40dp"

android:layout\_marginLeft="150dp"

android:id="@+id/btnArea"

/>

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Calculate Perimeter"

android:layout\_marginTop="0dp"

android:layout\_marginLeft="125dp"

android:id="@+id/btnPerimeter"

/>

</LinearLayout>

#### CalculatorFormFragment.java

package com.example.chapter5;

import android.content.Intent;

import android.os.Bundle;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentManager;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

import android.widget.EditText;

/\*\*

\* A simple {@link Fragment} subclass.

\* Use the {@link CalculatorFormFragment#newInstance} factory method to

\* create an instance of this fragment.

\*/

public class CalculatorFormFragment extends Fragment {

// TODO: Rename parameter arguments, choose names that match

// the fragment initialization parameters, e.g. ARG\_ITEM\_NUMBER

private static final String ARG\_PARAM1 = "param1";

private static final String ARG\_PARAM2 = "param2";

// TODO: Rename and change types of parameters

private String mParam1;

private String mParam2;

public CalculatorFormFragment() {

// Required empty public constructor

}

/\*\*

\* Use this factory method to create a new instance of

\* this fragment using the provided parameters.

\*

\* @param param1 Parameter 1.

\* @param param2 Parameter 2.

\* @return A new instance of fragment CalculatorFormFragment.

\*/

// TODO: Rename and change types and number of parameters

public static CalculatorFormFragment newInstance(String param1, String param2) {

CalculatorFormFragment fragment = new CalculatorFormFragment();

Bundle args = new Bundle();

args.putString(ARG\_PARAM1, param1);

args.putString(ARG\_PARAM2, param2);

fragment.setArguments(args);

return fragment;

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

if (getArguments() != null) {

mParam1 = getArguments().getString(ARG\_PARAM1);

mParam2 = getArguments().getString(ARG\_PARAM2);

}

}

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container,

Bundle savedInstanceState) {

// Inflate the layout for this fragment

View view = inflater.inflate(R.layout.fragment\_calculator\_form, container, false);

EditText elength =view.findViewById(R.id.enteredlength);

EditText ebreath =view.findViewById(R.id.enteredbreadth);

Button areabtn=view.findViewById(R.id.btnArea);

Button perimeterbtn=view.findViewById(R.id.btnPerimeter);

areabtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Bundle bundle = new Bundle();

float lengthC = Float.parseFloat(elength.getText().toString());

float breadthC = Float.parseFloat(ebreath.getText().toString());

float Area = (lengthC+breadthC);

bundle.putString("Area", String.valueOf(Area));

CalculatorAreaFragment cfa=new CalculatorAreaFragment();

cfa.setArguments(bundle);

getParentFragmentManager().beginTransaction().replace(R.id.transitionFrame,cfa).commit();

}

});

perimeterbtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Bundle bundle = new Bundle();

float lengthC = Float.parseFloat(elength.getText().toString());

float breadthC = Float.parseFloat(ebreath.getText().toString());

float Perimeter = 2\*(lengthC+breadthC);

bundle.putString("Perimeter", String.valueOf(Perimeter));

CalculatorPerimeterFragment cfp=new CalculatorPerimeterFragment();

cfp.setArguments(bundle);

getParentFragmentManager().beginTransaction().replace(R.id.transitionFrame,cfp).commit();

}

});

return view;

}

}

### CalculatorAppActivity

#### CalculatorAppActivity.xml

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout 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:id="@+id/transitionFrame"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".CalculatorAppActivity">

<!-- <FrameLayout-->

<!-- android:layout\_width="wrap\_content"-->

<!-- android:layout\_height="wrap\_content"-->

<!-- android:id="@+id/transitionFrame"-->

<!-- android:layout\_marginTop="100dp"-->

<!-- />-->

</FrameLayout>

#### CalculatorAppActivity.java

package com.example.chapter5;

import android.os.Bundle;

import androidx.activity.EdgeToEdge;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.graphics.Insets;

import androidx.core.view.ViewCompat;

import androidx.core.view.WindowInsetsCompat;

import androidx.fragment.app.FragmentManager;

public class CalculatorAppActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

EdgeToEdge.enable(this);

setContentView(R.layout.activity\_calculator\_app);

CalculatorFormFragment calculatorFormFragment=new CalculatorFormFragment();

FragmentManager fragmentManager = getSupportFragmentManager();

fragmentManager.beginTransaction()

.add(R.id.transitionFrame, calculatorFormFragment)

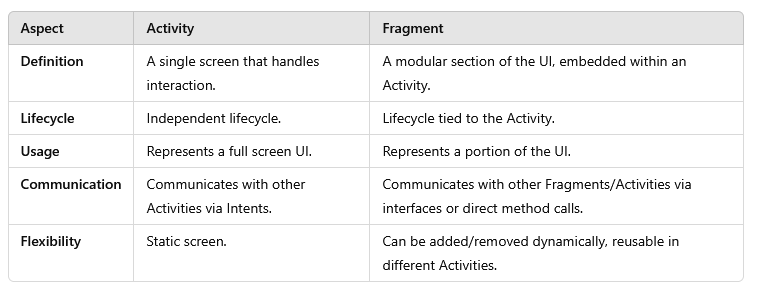
.commit();

}

}

## Important Questions

* **Activity vs Fragment[5 Marks]**



* **What is Fragment? Its importance flexibility and use case?**

A **Fragment** is a modular UI component that can be embedded within an **Activity**. It represents a portion of the screen and can be reused across multiple Activities. It has its own lifecycle but is closely tied to the Activity's lifecycle.

### ****Importance & Flexibility:****

* + **Modularity**: Fragments allow for code reuse and a more organized UI.
  + **Dynamic UI**: They can be added, removed, or replaced during runtime, making the UI more flexible.
  + **Responsive Design**: Useful for creating layouts that adapt to different screen sizes, especially tablets.
  + **Efficient Navigation**: Fragments enable in-Activity navigation without launching new Activities.

### ****Use Cases:****

* + **Multi-Pane Layouts**: Commonly used on tablets to show multiple fragments (e.g., list and details side by side).
  + **Reusable Components**: Can represent reusable sections like settings or user profiles.
  + **Master-Detail Interfaces**: Used for navigation patterns like displaying a list in one fragment and details in another.
  + **Dialog Fragments**: Used for showing dialogs without starting a new Activity.
* **Fragments and its lifecycle**
* **What do u mean by menu? Explain its types.**
* **Dialogs and custom dialogs[10marks]**

## Menus

* Menus are common user interface component in many types of applications.
* To present user actions and other options in your activities

### Types of Menus: (Three Types)

#### Options Menu

* Primary collections of menu items for an activity
* Actions that have a global impact on the app
* Search, setting and profile

|  |
| --- |
| Activity … |
| |  | | --- | | Items1 | | Items2 | | Items3 | |

#### Context Menu

* When users press long click on an Element
* Also known as floating menu

|  |
| --- |
| Activity |
| User1 |
| User2 |
| User3 |
|  |
|  |
|  |

#### Popup Menu

* Display list of items in vertical list that’s anchored to the view that invoke that menus
* Provides on overflow of actions that relates to specific context or provide second part of command.
* Used for extended actions.

|  |
| --- |
| Activity |
| |  | | --- | | Show Pages |   Onclick this   |  | | --- | |  | |  | |  | |

## Dialog:

* A small window that prompts uses to make a decision or enter additional information.
* Does not fill whole screen and is normally a modal events



### Different types of dialog:

* Alert Dialog:
* Shows title, upto three buttons, a list of selected items or a custion layout
* Datepicker Dialog or Time picker Dialog
* A dialog with pre-defined UI that allows the user to select a date or time
* Custom Dialog
* Custom Layout/Design

#### AlertDialog

#### DatePicker

#### Custom Dialog

|  |  |
| --- | --- |
| |  | | --- | | Show Custom Button | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | EditText1 |  |  | | --- | | EditText2 |  |  | | --- | | Button |   Custom\_dialog.xml | |

# Chapter 6: List View, Grid View, Recycler View

List View

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | |  |  | | --- | --- | |  |  | |  | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | | |  |  | | --- | --- | |  |  | |  | | | |

Grid View

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | |  |  |  |   ……………… |

Recycler View

* Combination of both T Both approach for showing logic set of dates

Cerate an new activity

Goto layout of activity

Create <List view> template

Attribute:

Height

Width

Id

## ListView

* View that groups several items and display them in vertical scorlable list
* Items are automatically inserted to the list using an adapter that pulls content from a source such as an array or database
* An adapter bridges between UI components and data source
* Adapter holds the data and send the data to adapter view

### List view with ArrayAdapter:

ArrayAdapter <string> adapter=new ArrayAdapter<> (this,R.layout.view, R.id.textview, string Array)

## Grid View:

-> Shows item in 2D scrolling grid (rows and column)  
Attributes:

android:numColumns->how many columns

android:horizontalSpacing

android:columnWidth

A white paper with black outline

AI-generated content may be incorrect.

# RecyclerView:

* More advanced and flexible version of ListView.
* Used for displaying large sets of data efficiently by recycling the views that are no longer visible in the screen.
* Efficient Memory Usages:
  + Instead of creating a new view for every item(like listView), RecyclerView resuses(or recycle) existings view that scroll of the screen while saves memory & performance.
* More customization :
  + lists, grids or staggered layouts
  + Animate items
  + Supports vertical and horizontal scrolling
* View Holder pattern:
  + User viewHolder pattern, which improves performance by avoiding unnecessary “findViewById()” calls
* Decouple Architecture:
  + Adapter: binds data to view
  + Layout Manager: Linear,Grid,etc
  + **ViewHolder: Describe and provides access to all view in one item(Each viewHolder is in charge of displaying single item with a view)**

Chapter 7: Advanced Android Concepts

* SQLite
  + Local Database Engine in Android
  + Stores structured table format similar to SQL database but is embedded in your app without the need of external server
  + User cases
    - Storing user info and preferences.
    - Offline caching.
    - Local app logs.
    - Small scale app data management.

|  |  |  |
| --- | --- | --- |
|  | SQLite | SQL |
| Type | Embedded, Local DB | Server based RDBMS |
| Server | No server needed | required |
| Storage | Local(inside app/device) | Remote or hosted server |
| Concurrency | Limited | High, Supports multiple users |
| Best for | Mobile app, light weighted system | Large, enterprise system |

Features:

* Lightweight and fast (zero config)
* Serverless and free
* Single file database(stores all data in one db file)
* SQL syntax support
* Low memory footprint
* ACID- complaint (support transaction)
* Cross-platform(android, iOS, desktop)

SQLiteOpenHelper Class

public abstract void onCreate() -> called once when DB is created for the first time.

public abstract void onUpgrade -> called when DB needs to be upgraded

public synchronized void close() -> close the DB object

public void onDowngrade() -> called when DB needs to be downgraded

SQLiteDatabase class:

void execSQL(String sql) -> executes raw SQL commands

long insert(…) -> insert records in DB

update() -> update existing records

delete() -> delete data from table

query() -> retrieve data in structured way

rawQuery() -> returns the cursor over the result set

Database Related and API & JSON for 10 marks

API- Advanced Programming Interface

* It is a collection of protocols and tools that allows communication between different software programs.
* Simplifies programming by providing pre-built operations instead of writing complex code from scratch.
* Request and Response between two applications.

Types of API: \*

\* Web API: Accessed over internet using HTTP protocols

\* Local API: Works on local machine or middleware services.

\* Program API: Makes remote programs seam local via RPC

* REST API (REPRESENTED STATE TRANSFER API)
  + It is a type if web service that allows different applications to communicate with each other over the internet using HTTP methods like GET, POST, PUT and DELETE

# Procedure for publishing Application in Google playstore:  
-> Generate Signed APK/AAB files

->Create App

-> Details-icons screenshots and description

-> Review Process – 2to 3 days pass or reject

END