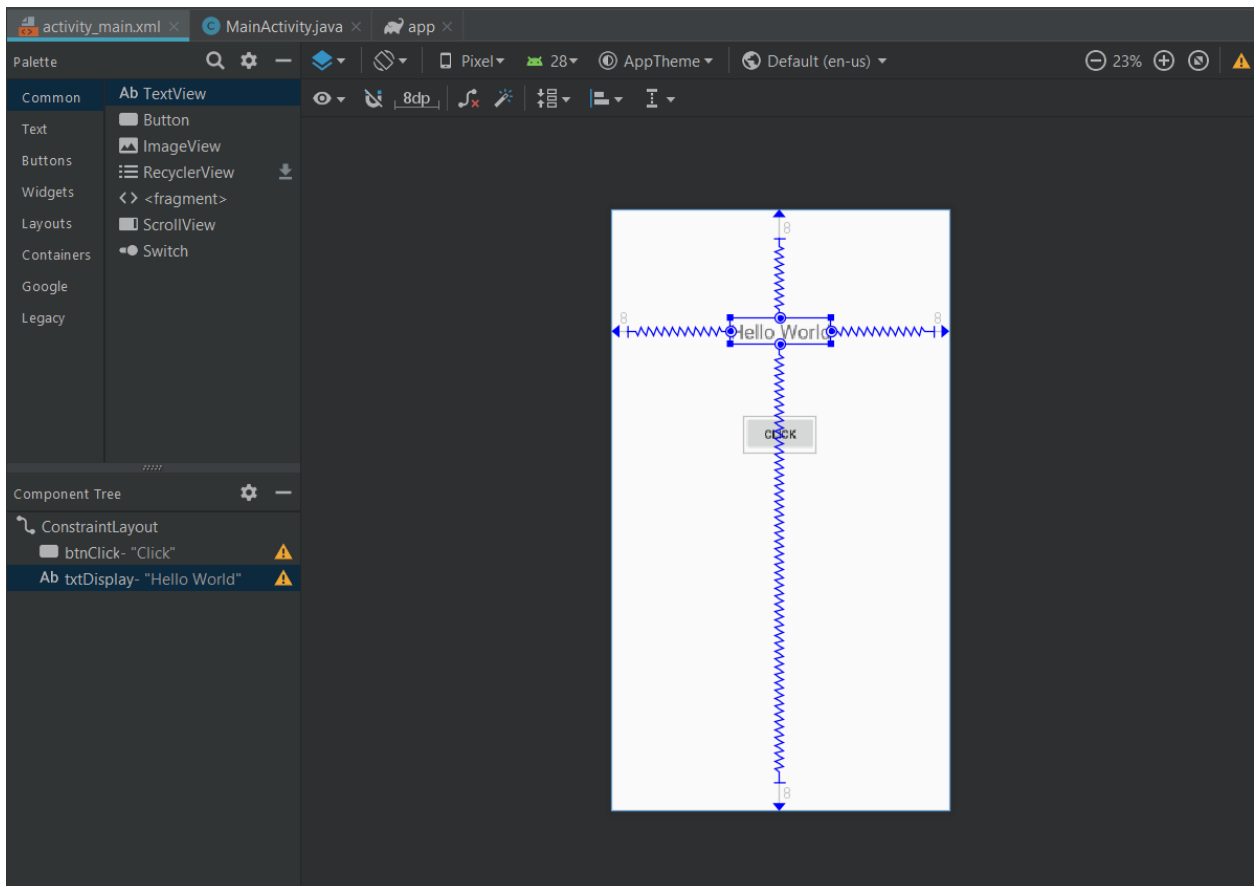


BICT- Level III – Semester II
Topic – Mobile Application Development (ICT3233)
Lab Sheet 02

Example 01

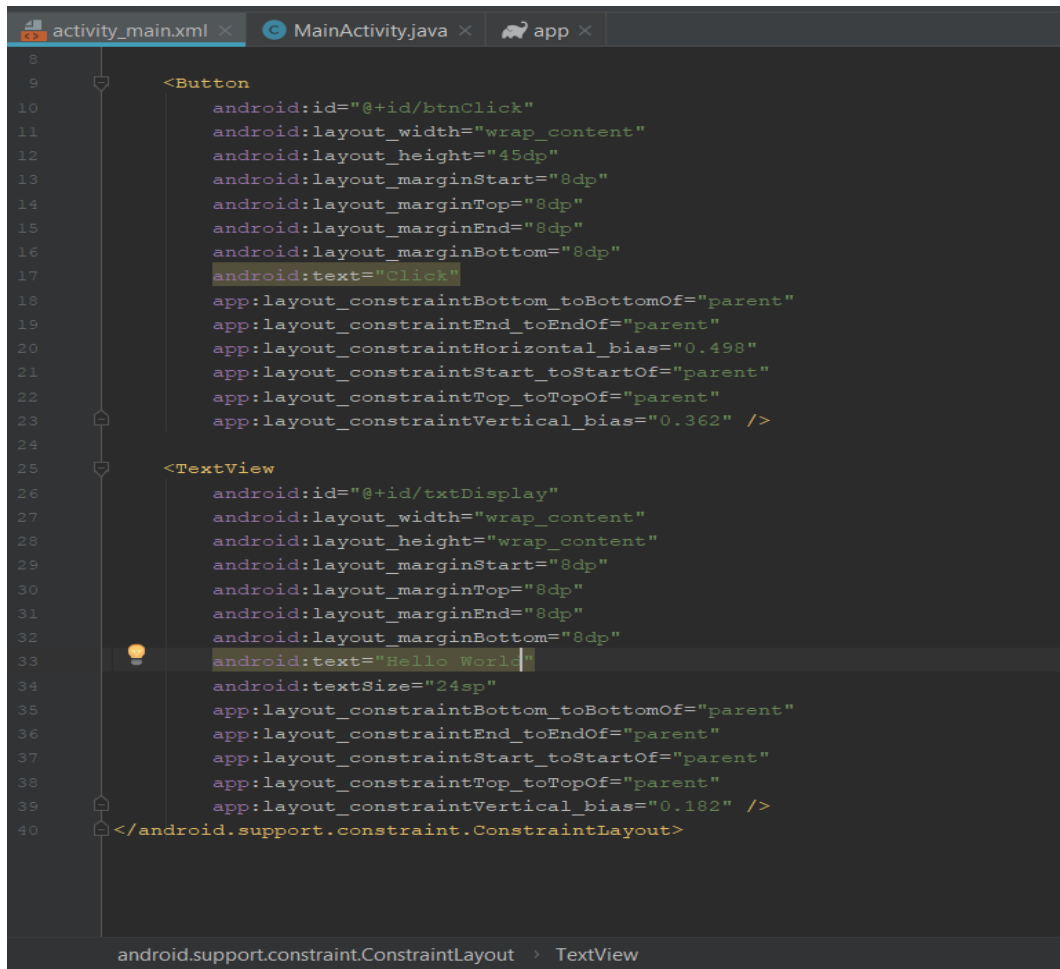
Create a program that has a **TextView** and a **Button**. The **TextView** should first display “Hello World” and when the user clicks the button the text should be changed as “Welcome to Android Studio”.

- Create a new project.
 - Insert a Button and a TextView and constraint them.
- 1) You can drag and drop a Button and TextView to the layout by using the Design. And constraint them.



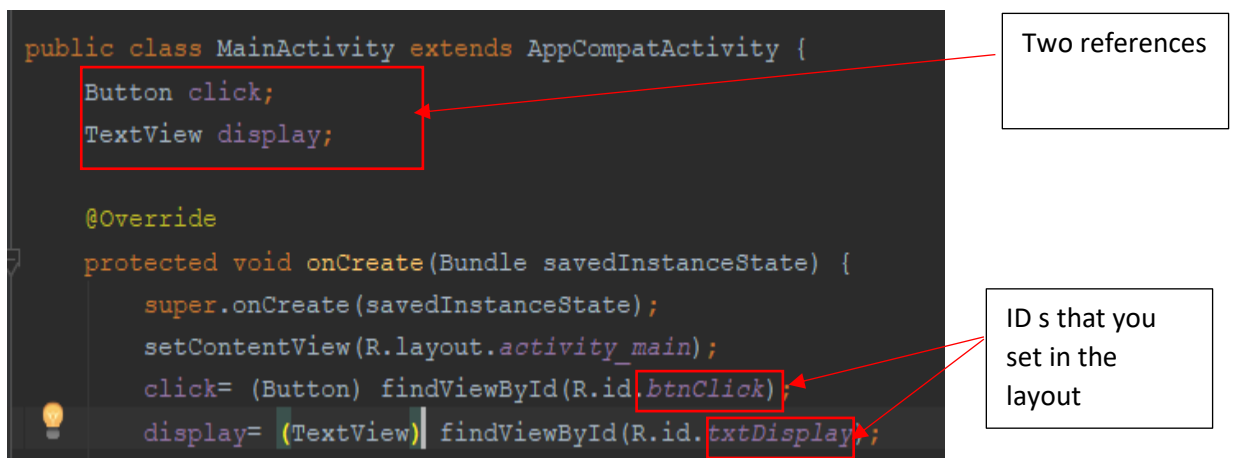
Important: You have to set ids for both Button and TextView

2) As well as you can use the XML code to insert the Button and TextView.



```
8
9      <Button
10          android:id="@+id/btnClick"
11          android:layout_width="wrap_content"
12          android:layout_height="45dp"
13          android:layout_marginStart="8dp"
14          android:layout_marginTop="8dp"
15          android:layout_marginEnd="8dp"
16          android:layout_marginBottom="8dp"
17          android:text="Click"
18          app:layout_constraintBottom_toBottomOf="parent"
19          app:layout_constraintEnd_toEndOf="parent"
20          app:layout_constraintHorizontal_bias="0.498"
21          app:layout_constraintStart_toStartOf="parent"
22          app:layout_constraintTop_toTopOf="parent"
23          app:layout_constraintVertical_bias="0.362" />
24
25      <TextView
26          android:id="@+id/txtDisplay"
27          android:layout_width="wrap_content"
28          android:layout_height="wrap_content"
29          android:layout_marginStart="8dp"
30          android:layout_marginTop="8dp"
31          android:layout_marginEnd="8dp"
32          android:layout_marginBottom="8dp"
33          android:text="Hello World"
34          android:textSize="24sp"
35          app:layout_constraintBottom_toBottomOf="parent"
36          app:layout_constraintEnd_toEndOf="parent"
37          app:layout_constraintStart_toStartOf="parent"
38          app:layout_constraintTop_toTopOf="parent"
39          app:layout_constraintVertical_bias="0.182" />
40  </android.support.constraint.ConstraintLayout>
```

- After adding the views to the layout you have to create two references to them and after that you can access those elements in your java code using **findViewById**.



```
public class MainActivity extends AppCompatActivity {
    Button click;
    TextView display;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        click= (Button) findViewById(R.id.btnClick);
        display= (TextView) findViewById(R.id.txtDisplay);
    }
}
```

Two references

IDs that you set in the layout

- Then you have to set the action for the button which you added earlier.
- For that you can use either **setOnClickListener()** method and **onClick** attribute.

1. setOnClickListener() Method

```
public class MainActivity extends AppCompatActivity {  
    Button click;  
    TextView display;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        click= (Button) findViewById(R.id.btnClick);  
        display= (TextView) findViewById(R.id.txtDisplay);  
        click.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                display.setText("Welcome to Android Studio");  
            }  
        });  
    }  
}
```

You can use the
setText()
method

2. Using onClick Attribute

- First insert a **TextView** and a **Button** to the layout and set IDs for them.
- Next move to the XML code and insert the **onClick** attribute and inside the attribute you have to give the method name that you wish to create.

```
<Button
    android:id="@+id/btnChange"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="8dp"
    android:onClick="change"
    android:text="Click"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

- Then you can click on the Red bulb and select **Create 'Change(View)' in 'Main activity'**. In here **Change** is the method name that I have given and it will change depending on the name that you will give for the method.
- Then it will create a public method in your java code.
- After that inside the method that you have created you can use **setText()** method to set the new text to the TextView.

```
public class MainActivity extends AppCompatActivity {
    TextView display;

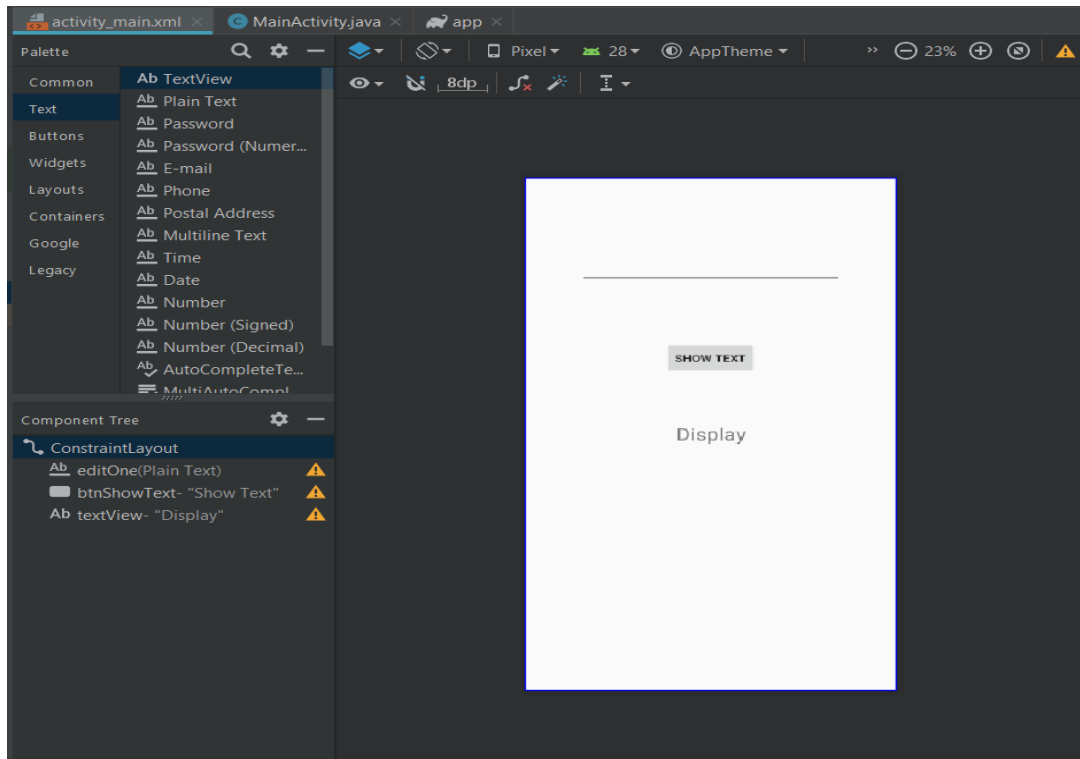
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void Change(View view) {
        display= (TextView) findViewById(R.id.txtOne);
        display.setText("Welcome to Android Studio");
    }
}
```

Example 02

Create a program that has an **EditText** (Plain Text element), a **Button** and a **TextView**. When the user enters a text to the **EditText** and clicks the **Button** the entered text should be displayed on the **TextView**.

- First insert an **EditText**, a **Button** and a **TextView** to the layout, constraint them and set IDs for them.



- After adding the views to the layout as in the previous example you have to create references to them and then you can access those elements in your java code using **findViewById**.
- In here you have to create another String variable to store the user input.

```

public class MainActivity extends AppCompatActivity {
    EditText input;
    Button show;
    TextView output;
    String data;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        input= (EditText) findViewById(R.id.editOne);
        show= (Button) findViewById(R.id.btnShowText);
        output= (TextView) findViewById(R.id.txtDisplay);
    }
}

```

- Then in the **onClick** method of the button you can get the user input using **getText()** method and store it in the String variable that you have created.
- And you can set that user input to the **TextView** by using **setText()** method.

```

public class MainActivity extends AppCompatActivity {
    EditText input;
    Button show;
    TextView output;
    String data;

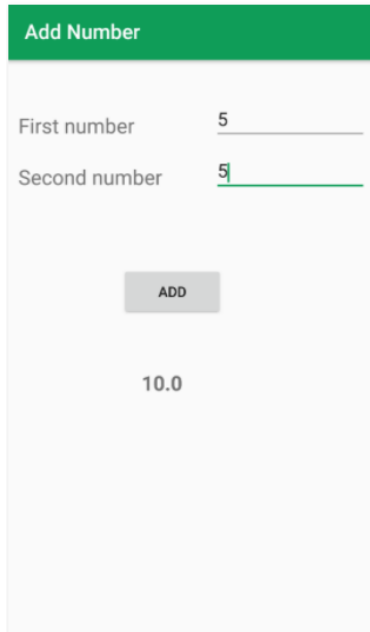
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        input= (EditText) findViewById(R.id.editOne);
        show= (Button) findViewById(R.id.btnShowText);
        output= (TextView) findViewById(R.id.txtDisplay);
        show.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                data=input.getText().toString();
                output.setText(data);
            }
        });
    }
}

```

Exercise 01

Create an application to take two numbers from the user and display their sum in a TextView.

Out Put:



The screenshot shows a mobile application interface with a green header bar labeled "Add Number". Below the header, there are two text input fields. The first field is labeled "First number" and contains the value "5". The second field is labeled "Second number" and contains the value "5". Below these fields is a grey button labeled "ADD". At the bottom of the screen, a TextView displays the result "10.0".

Exercise 02

Create a calculator that has addition, subtraction, Division and Multiplication.

- When the user enters two numbers and click Add button it should display the sum.
- When the user enters two numbers and click Subtraction button it should display the difference.
- When the user enters two numbers and click Division button it should display the division.
- When the user enters two numbers and click Multiplication button it should display the product.

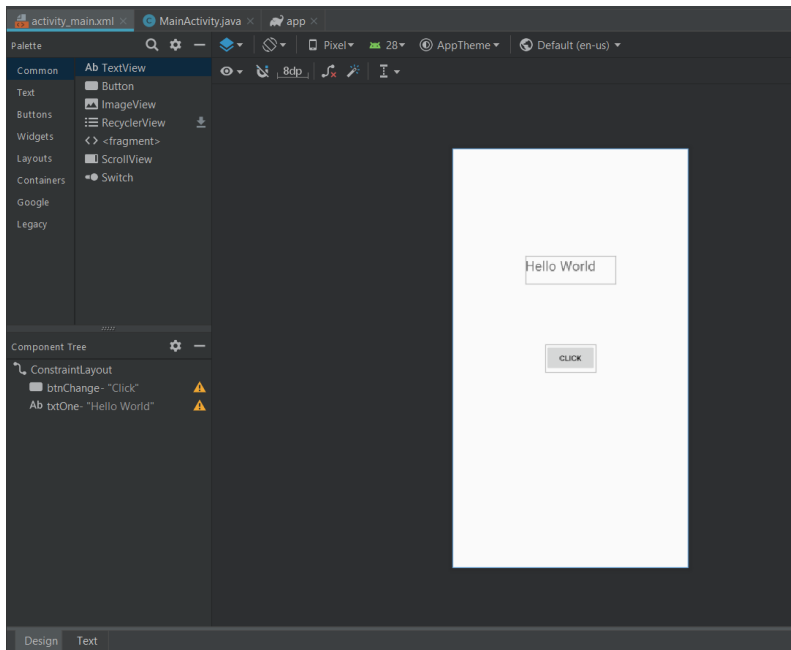
You can use a Reset button too to clear the data that you entered for the two numbers.

Exercise 03

Create a Log in application that takes User name and Password from the user and if the user name and password match with some specific values that you provide (you can use values you prefer) a TextView should display “Login successful” and if they don’t match TextView should display “Invalid user”.

Activity

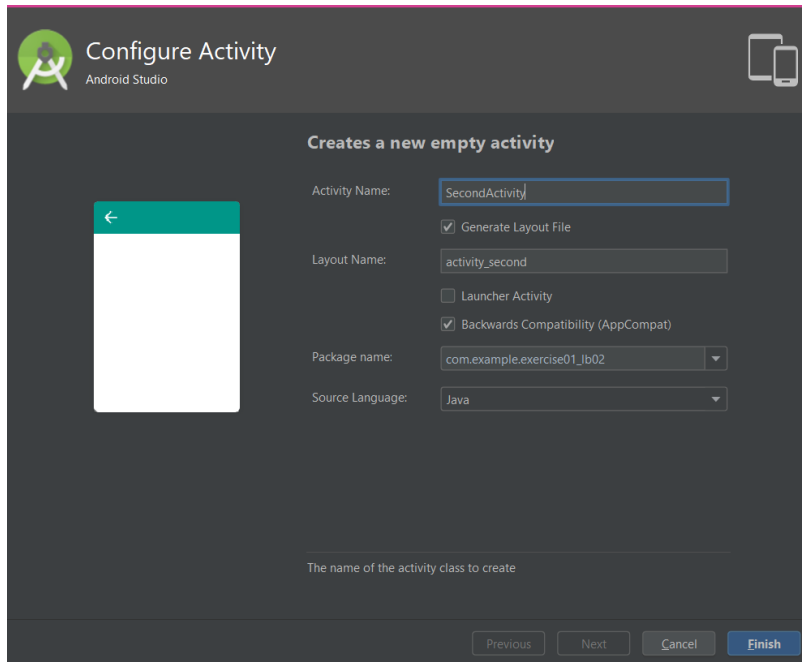
- An activity is the entry point for interacting with the user. It represents a single screen with a user interface.



- When user creates the project, an activity is created by default (Main Activity) as in the above figure.

How to create another activity in your application

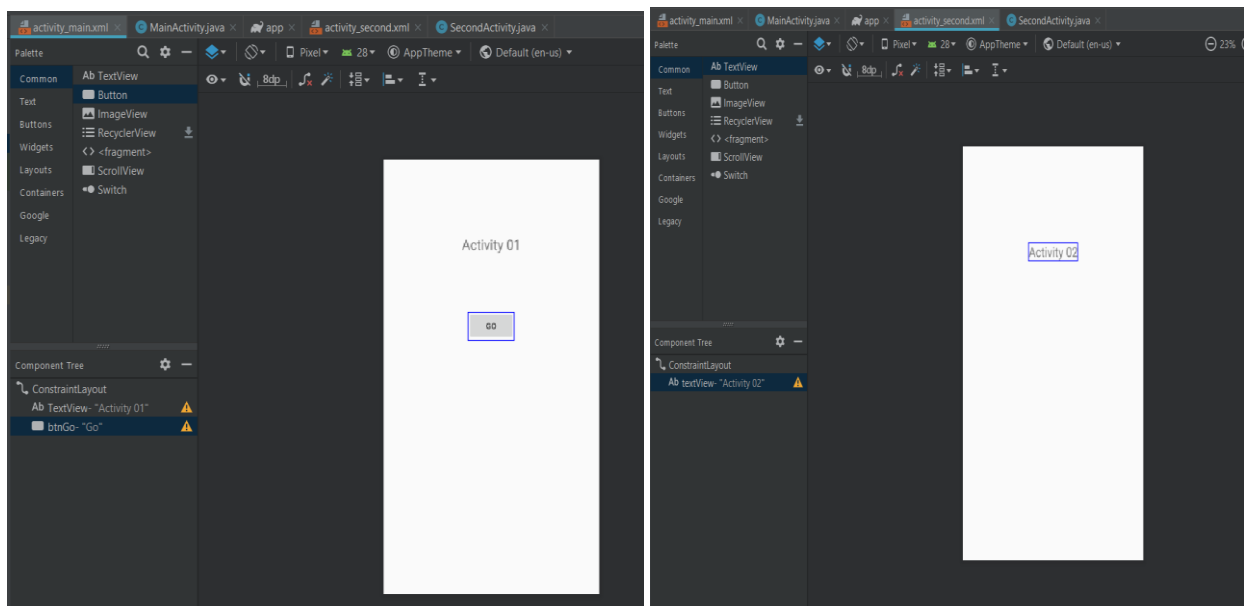
- Right click the package inside java folder → New → Activity → Empty Activity
- Give a name for your activity and Click Finish.



Example 03 – Moving from One activity to another

Create an application that has two activities. When the user clicks a button in the Main activity he should be directed to the second activity.

- Create the layouts of the two activities as below.



- You have to use Intent in here.
- First create an Intent reference.

```
public class MainActivity extends AppCompatActivity {
    Button go;
    Intent intent;
```

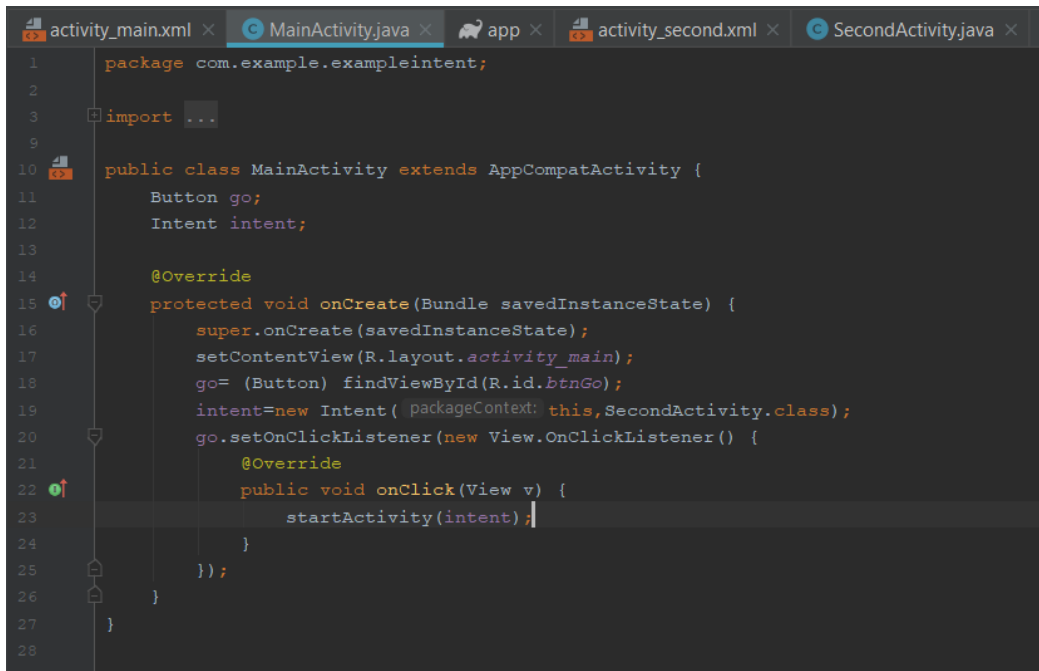
- Then you have to create the Intent object for the reference that you have created.
- You have to give two parameters for the object.

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

Application
Context

The Activity that
the app should be
directed to

- **this** – is used to specify to take the context of the same activity.
- Then by using **StartActivity()** method you can start the next activity.
- Inside the **StartActivity()** method give the name of the intent object that you created.



```
1 package com.example.exampleintent;
2
3 import ...
4
5
6
7
8
9
10 public class MainActivity extends AppCompatActivity {
11     Button go;
12     Intent intent;
13
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17         setContentView(R.layout.activity_main);
18         go= (Button) findViewById(R.id.btnGo);
19         intent=new Intent( packageContext: this,SecondActivity.class);
20         go.setOnClickListener(new View.OnClickListener() {
21             @Override
22             public void onClick(View v) {
23                 startActivity(intent);
24             }
25         });
26     }
27 }
28
```

Example 04 – Passing data from one activity to another

- There are several methods that can be used to pass data from one Activity to another.
- And here we are using Intents.
- You can find more on the other methods.

We are discussing two methods in Intent.

- 1) Extras
- 2) Bundle Object

Extras

- Create a new project.
- Create two Activities.
- Insert a Button to the first activity.
- Insert two TextViews to the second activity.

```
public class SecondActivity extends AppCompatActivity {
    TextView name;
    TextView city;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        name= (TextView) findViewById(R.id.txtOne);
        city= (TextView) findViewById(R.id.txtTwo);
    }
}
```

- Create an **Intent Object** in the **Main activity**.
- You can use **putExtra()** method to pass the data between activities.
- In **putExtra()** method we have to pass two parameters (**Key, Value**).

```

public class MainActivity extends AppCompatActivity {
    Button go;
    Intent intent;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        go= (Button) findViewById(R.id.btnGo);
        intent=new Intent( packageContext: this, SecondActivity.class);
        go.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                intent.putExtra( name: "NAME", value: "Ann");
                intent.putExtra( name: "CITY", value: "Galle");
                startActivity(intent);
            }
        });
    }
}

```

- By using **putExtra()** method we can pass data types such as String, Integer, Float, Double etc...

Then we have to display the data that have been passed from the Main activity to the second activity. For that,

- In the second activity insert the following code.

```

public class SecondActivity extends AppCompatActivity {
    TextView name;
    TextView city;
    Intent intent;
    String data_name;
    String data_city;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        name= (TextView) findViewById(R.id.txtOne);
        city= (TextView) findViewById(R.id.txtTwo);
        intent=getIntent();
        data_name=intent.getStringExtra( name: "NAME");
        data_city=intent.getStringExtra( name: "CITY");
        name.setText(data_name);
        city.setText(data_city);
    }
}

```

- In here we have created an Intent reference and we have assigned **getIntent()** method for that reference.
- The Intent object of the Main activity can be retrieved by using this **getIntent()** method.
- Then we have used **getStringExtra()** method to get the String type data that have been passed from the Main activity.
- There you have to use the “**Key**” of the **putExtra()** method which you used in the Main Activity.
- Then using **setText()** method you can set the text to TextView.

Home Work

- 1) **Create an application to pass data from one activity to another by using Bundle Object.**
- 2) **Find and get an idea about how to use Global Variables to pass data from one activity to another.**