Pertemuan 3
Perkenalan Kotlin Lanjutan

1. Function

Fungsi adalah blok pernyataan yang saling terkait guna melakukan tugas tertentu. Dengan menggunakan function, maka kita tidak perlu menulis kode dengan fungsi yang sama berulang kali. Cukup buat sekali function, lalu dipanggil dilokasi yang diperlukan. Sebagai contoh jika kita harus menulis tiga baris kode untuk menemukan rata-rata dua angka, jika kita membuat fungsi untuk mencari rata-rata maka daripada menulis kode yang sama berulang kali, kita bisa memanggil fungsi yang telah kita buat.

a. Standard Library Function

Fungsi yang sudah tersedia pada librari standar Kotlin disebut fungsi bawaan atau fungsi yang telah ditentukan sebelumnya. Misalnya ketika kita perlu menggunakan fungsi Math.floor () kita tidak mendefinisikan fungsinya karena sudah ada dan kita bisa langsung memanggilnya dalam kode kita.

BelajarKotlin11.kt

```
package com.example.belajarkodekotlin

fun main() {
    val angka = 2
    println("Pangkat 3 dari angka $angka : ${Math.pow(3.0,angka.toDouble())}")

    val angka2 = 25
    println("Akar dari $angka2 adalah: ${Math.sqrt(angka2.toDouble())}")

    println("Nilai Phi : ${Math.PI}")
}
```

Gambar 1. Program 12 Standard Lib Function

b. User Defined-Function

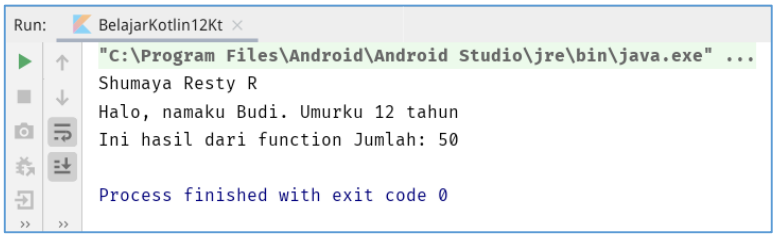
Fungsi yang kita definisikan dalam program sebelum digunakan dikenal sebagai fungsi yang ditentukan pengguna. Sebagai contoh, jika kita ingin sebuah fungsi memeriksa angka genap atau ganjil dalam program, maka kita dapat membuat fungsi untuk tugas ini dan kemudian memanggil fungsi di mana kita memerlukannya. Silahkan buat sebuah file kotlin baru bernama **BelajarKotlin12.kt**, kemudian ketikkan kode program di bawah ini.

BelajarKotlin12.kt

```
package com.example.belajarkodekotlin

fun main() {
    namaKu()
    contohReturn( nama: "Budi",  umur: 12)
    println("Ini hasil dari function Jumlah: ${jumlah( ...angka: 5, 10, 15, 20)}")
}

fun namaKu() {
    println("Shumaya Resty R")
}

fun contohReturn(nama: String, umur: Int) {
    println("Halo, namaku $nama. Umurku $umur tahun")
}

fun jumlah(vararg angka: Int): Int {
    var hasilJumlah = 0
    angka.forEach { nilai -> hasilJumlah += nilai }

    return hasilJumlah
}
```

Gambar 2. Program 13 User-Defined Function

Jika sudah selesai, maka silahkan coba jalankan kode program tersebut. Silahkan coba modifikasi kode program diatas.

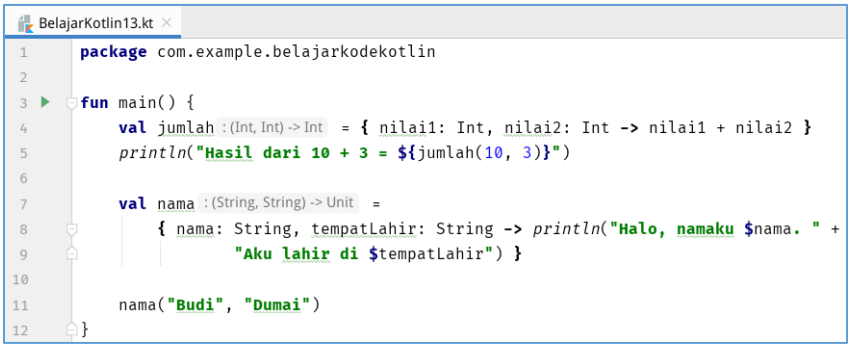


Gambar 3. Hasil Program 13

Inline Function

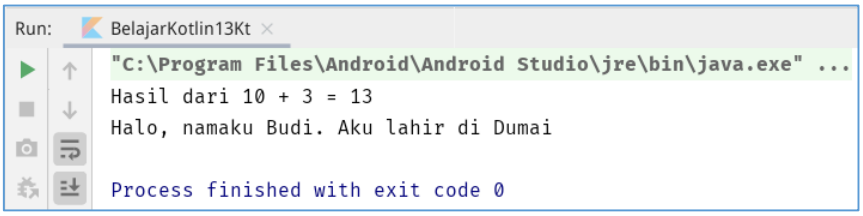
Fungsi Inline atau sebaris ini dapat langsung didefinisikan di dalam fungsi main(). Mari kita ambil contoh fungsi sebaris. Dalam contoh berikut kami telah mendefinisikan jumlah fungsi sebaris yang menerima dua argumen bilangan bulat num1 dan num2 dan tipe kembalian adalah bilangan bulat.

Buat sebuah file baru bernama **BelajarKotlin13.kt**. Lalu ketikkan kode program dibawah ini untuk mempelajari Inline Function.



Gambar 4. Program 14 Inline Function

Jika sudah selesai, maka silahkan coba jalankan kode program tersebut. Silahkan pahami kode program tersebut.



Gambar 5. Hasil Program 14

Tugas Analisa :

1. Silahkan buat sebuah user-defined function yang menampilkan berat badan dan hasil BMI nya. (Silahkan cari rumus di google).
2. Silahkan buat sebuah contoh lagi untuk inline function sesuai kreasi anda.

2. Recursive Function

Suatu fungsi disebut fungsi rekursif jika memanggil dirinya sendiri dan proses ini disebut rekursi. Pada contoh dibawah adalah contoh sederhana faktorial. Di sini kita telah mendefinisikan fungsi faktorial() untuk menghitung nilai faktorial dari sebuah bilangan yang diteruskan ke fungsi ini sebagai parameter. Pada fungsi faktorial tersebut kita memanggil fungsi ini lagi, nah proses inilah yang kemudian disebut dengan rekursi. Kita akan memasukkan bilangan bulat dan program menemukan faktorial nomor inputan tersebut dengan meneruskan nomor masukan sebagai argumen ke fungsi yang ditentukan pengguna faktorial ().

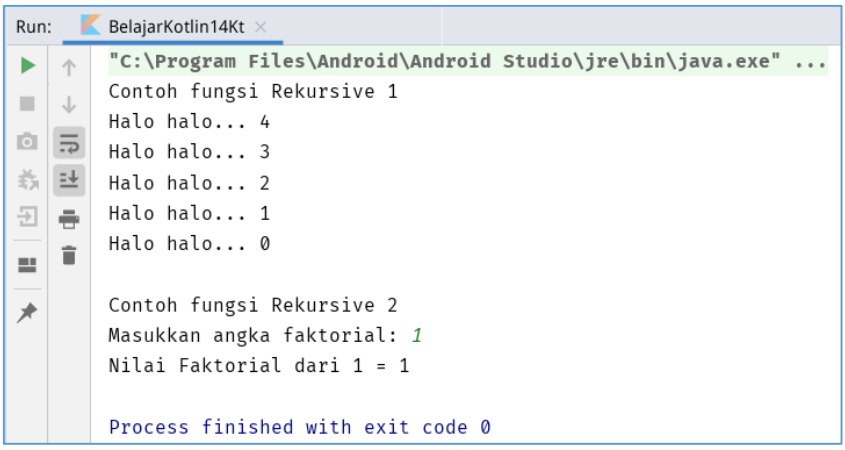
Silahkan buat file kotlin baru bernama BelajarKotlin14.kt, kemudian tuliskan kode program di bawah ini untuk mencobakan.

BelajarKotlin14.kt

```kotlin
package com.example.belajarkodekotlin

fun main() {
    println("Contoh fungsi Rekursive 1")
    fungsiRekursive1()

    println("\nContoh fungsi Rekursive 2")
    print("Masukkan angka faktorial: ")
    val angka :Int  = Integer.valueOf(readLine())
    val hasil :Long  = faktorial(angka)
    println("Nilai Faktorial dari $angka = $hasil")
}

var hitung = 5
fun fungsiRekursive1() {
    hitung--;
    if (hitung >= 0) {
        println("Halo halo... " + hitung);
        fungsiRekursive1();
    }
}

fun faktorial(n: Int): Long {
    return if (n == 1)
        n.toLong()
    else n * faktorial( n: n - 1)
}
```

Gambar 6. Program 15 Recursive Function

Jika sudah selesai, maka silahkan anda jalankan kode program tersebut. Jika berhasil maka hasilnya akan seperti di bawah ini.



Gambar 7. Hasil Program 15

Tugas Analisa :

1. Coba jelaskan langkah-langkah yang terjadi pada fungsi recursive faktorial.
2. Apa perbedaan head recursive dengan tail recursive? Jelaskan dengan menggunakan contoh.

3. Lambda

Fungsi lambda juga dikenal sebagai fungsi anonim karena tidak memiliki nama. Parameter ada di sisi kiri panah dan kode logika nya ada di sisi kanan panah.

```
{variable/parameter -> implementasi_kode}
```

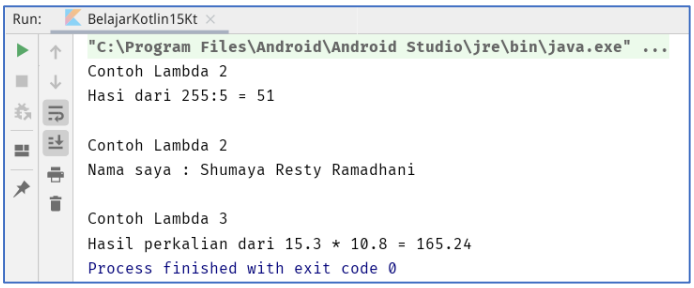
Pada **contoh Lambda 1** di bawah ini dapat terlihat sebuah fungsi lambda untuk melakukan pembagian sederhana. Terdapat dua buah variable nilai1 dan nilai2 disisi kiri panah lengkap dengan tipe datanya, dan kode program untuk kalkulasi disebelah kanan. Silahkan buat file kotlin baru bernama **BelajarKotlin15.kt**. Cobakan kode program di bawah ini.

BelajarKotlin15.kt

```kotlin
package com.example.belajarkodekotlin

fun main() {
    println("Contoh Lambda 1")
    val jumlah : (Int, Int) -> Int  = { nilai1: Int, nilai2: Int ->
        nilai1 / nilai2
    }
    println("Hasi dari 255:5 = ${jumlah(255, 5)}")
/*-------------------------------------------------------------*/
    println("\nContoh Lambda 2")
    val contohLambda: (String) -> Unit = { nama: String ->
        print("Nama saya : $nama")
    }
    val namaKu = "Shumaya Resty Ramadhani"
    contohLambda(namaKu)
/*-------------------------------------------------------------*/
    println("\n\nContoh Lambda 3")
    val myLambda : (String) -> Unit = {s:String -> print(s)}
    perkalian( angka1: 15.3, angka2: 10.8, myLambda)
}

fun perkalian(angka1: Double, angka2: Double, myLambda: (String) -> Unit) {
    val hasil : Double  = angka1 * angka2
    val output= "Hasil perkalian dari $angka1 * $angka2 = $hasil"
    myLambda(output)
}
```

Gambar 8. Program 16 Lambda

Jika sudah selesai, silahkan jalankan kode program tersebut. Seharusnya jika sesuai maka akan tampil seperti pada contoh di bawah ini. Silahkan dipahami bentuk penulisannya. Kedepannya anda akan sering menemukan model penulisan seperti ini pada saat pengembangan aplikasi.



Gambar 9. Hasil Program 16

4. Exception Handling

Exception merupakan masalah yang tidak diinginkan dan dapat terjadi pada saat runtime program dan keadaan tersebut dapat menyebabkan program anda berhenti tiba-tiba. Dengan menggunakan Exception Handling, maka kita dapat terhindar dari program yang keluar tiba-tiba. Bentuk kode pemrograman ini mirip dengan bahasa Java.

Ada dua jenis Exception yang digunakan :

1. Checked Exceptions → diperiksa pada waktu kompilasi, misalnya IOException
2. Unchecked Exceptions → diperiksa pada waktu proses, misalnya NullPointerException.

Kita juga dapat membuat pengecualian menggunakan **Throw**. Pernyataan sebelum Exception akan dieksekusi, akan tetapi pernyataan setelah Exception tidak akan dijalankan karena kontrol dipindahkan ke blok catch. Silahkan buat sebuah file kotlin baru bernama **BelajarKotlin16.kt**.

```kotlin
fun main() {

    println("Coba Exception Handling 1")
    try {
        val testError : Int = 10 / 0
        println("Contoh Error")
        println(testError)
    } catch (e: ArithmeticException) {
        println("Arithmetic Exception")
    } catch (e: Exception) {
        println(e)
    } finally {
        println(
            "Pada block Finally, apapun itu yang terjadi ya tetap di print"
        )
    }
/*-------------------------------------------------------------*/
    println("\nCoba Exception Handling 2")
    contohError()
}

fun contohError() {
    try {
        println("Sebelum Exception")
        throw Exception("Hayoooo masalahnya masuk ke Throw")
        println("Sesudah Exception")
    } catch (e: Exception) {
        println(e)
    } finally {
        println("Masih error, tapi finally tetap muncul ya")
    }
}
```

Gambar 10. Program 17 Exception Handling

Jika sudah selesai, silahkan jalankan kode program tersebut. Seharusnya jika sesuai maka akan tampil seperti pada contoh di bawah ini. Silahkan dipahami bentuk penulisannya. Anda juga dapat coba mengarahkan *mouse* ke bagian kode program yang diberi *highlight* kuning pada baris 7 dan 28. Kotlin cukup pintar untuk mendeteksi permasalahan sehingga kita bisa lebih *aware* sebelum *error* terjadi.

```
Run:  BelajarKotlin16Kt
"C:\Program Files\Android\Android Studio\jre\bin\java.exe" ...
Coba Exception Handling 1
Arithmetic Exception
Pada block Finally, apapun itu yang terjadi ya tetap di print

Coba Exception Handling 2
Sebelum Exception
java.lang.Exception: Hayoooo masalahnya masuk ke Throw
Masih error, tapi finally tetap muncul ya

Process finished with exit code 0
```

Gambar 11. Hasil Program 17

Tugas Analisa :

1. Silahkan coba buat sebuah contoh Exception Handling dengan ArrayIndexBoundsException dengan menggunakan Bahasa Kotlin.

8. Higher Order Function

Fungsi fungsi level lebih tinggi dapat memiliki fungsi lain yaitu sebagai parameter, mengembalikan fungsi atau bisa melakukan keduanya. Pada contoh dibawah ini kita akan belajar bagaimana

meneruskan sebuah fungsi ke fungsi lainnya dan juga bagaimana fungsi mengembalikan fungsi lainnya.

Silahkan buat sebuah file kotlin bernama **BelajarKotlin17.kt**. Selanjutnya silahkan ketikkan kode program berikut ini agar lebih dapat memahami.

BelajarKotlin17.kt

```
1  package com.example.belajarkodekotlin
2
3  fun main() {
4      println("Contoh High Order 1")
5      contohFungsi( kls: 3, ::contohFungsi2)
6
7      println("\nContoh High Order 2")
8      val jumlah :(Int) -> Int  = contohFungsi3( nilai1: 25)
9      println("Hasil Penjumlahan 25 + 30 = ${jumlah(30)}")
10
11 }
12
13 fun contohFungsi(kls: Int, fungsi: (Int) -> Unit) {
14     print("Selamat kalian sudah berhasil naik ke kelas ")
15     fungsi(kls)
16 }
17
18 fun contohFungsi2(kls: Int) {
19     println(kls)
20 }
21
22 /*-------------------------------------------------------*/
23 fun contohFungsi3(nilai1: Int): (Int) -> Int = {
24         nilai2 -> nilai2 + nilai1
25 }
```

Gambar 12. Program 18 Higher Order Function

Jika sudah selesai, silahkan jalankan kode program tersebut. Seharusnya jika sesuai maka akan tampil seperti pada contoh di bawah ini.

Run: BelajarKotlin17Kt

```
"C:\Program Files\Android\Android Studio\jre\bin\java.exe" ...
Contoh High Order 1
Selamat kalian sudah berhasil naik ke kelas 3

Contoh High Order 2
Hasil Penjumlahan 25 + 30 = 55

Process finished with exit code 0
```

Gambar 13. Hasil Program 18

Tugas Analisa :

1. Silahkan jelaskan tahapan yang terjadi pada contoh High Order 1 dan High Order 2 pada contoh di atas.

Android Layout

Android menyediakan dukungan penuh untuk pengembangan aplikasi berbasis UI. Android menyediakan berbagai widget yang dapat digunakan oleh pengembang aplikasi untuk membuat tata letak dan tampilan antarmuka yang diinginkan. Elemen tata letak ini dapat dibuat dengan menggunakan bahasa pemrograman secara langsung maupun dengan melalui file layout XML.

Pada android, layout berbasis-XML adalah file yang mendefinisikan berbagai widget yang akan digunakan di UI dan hubungan antara widget serta penampungnya. Android memperlakukan file layout sebagai resource. Oleh karena itu, tata letak disimpan dalam folder res. File layout bertindak sebagai masukan ke Android Asset Packaging Tool (AAPT), yang membentuk file R.java untuk semua sumber.

Android menyediakan model layout standar (viewgroups) yang dapat digunakan dalam aplikasi Android.

1. **AbsoluteLayout**

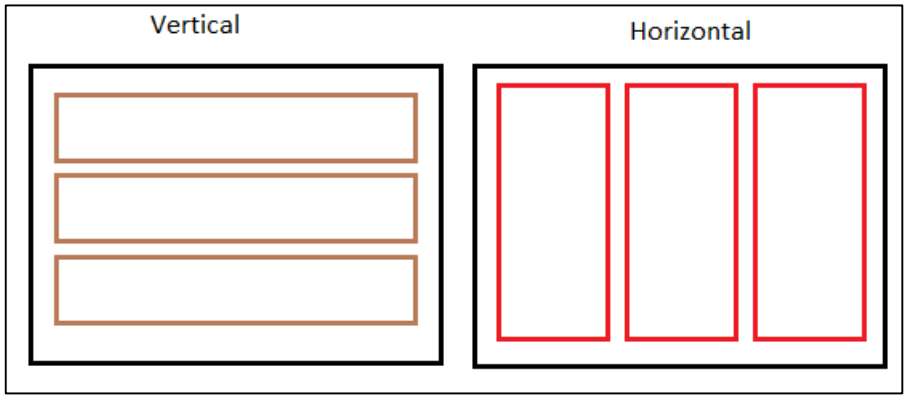
   Layout ini disediakan oleh Google sejak API versi 1 untuk desain tampilan aplikasi yang memungkinkan kita untuk mengatur posisi objek secara horizontal dan vertikal sesuai keinginan. Akan tetapi, semenjak adanya perkembangan API android versi 3, model layout ini tidak lagi digunakan. Mari kita lihat contoh pembuatan tampilan UI lainnya dengan menggunakan kode xml.

2. **FrameLayout**

   Layout model ini digunakan ketika kita ingin menampilkan sebuah objek saja pada tampilan aplikasi. Tentu saja ini tidak berarti bahwa kita hanya bisa membentuk sebuah objek dengan model ini, tapi ketika kita membuat banyak objek, maka posisi objek tersebut pada layar akan menimpa satu sama lain. FrameLayout sangat bermanfaat ketika kita ingin membuat aplikasi yang memiliki animasi.

3. **LinearLayout**

   Model LinearLayout membuat objek terletak pada satu baris atau kolom. Sehingga posisi objek tersebut dapat tersusun dengan rapi dari atas ke bawah. Model ini lah yang paling banyak digunakan untuk membuat tampilan antar muka pada aplikasi android.



Gambar 14. Pemodelan Umum Linear Layout

4. **RelativeLayout**

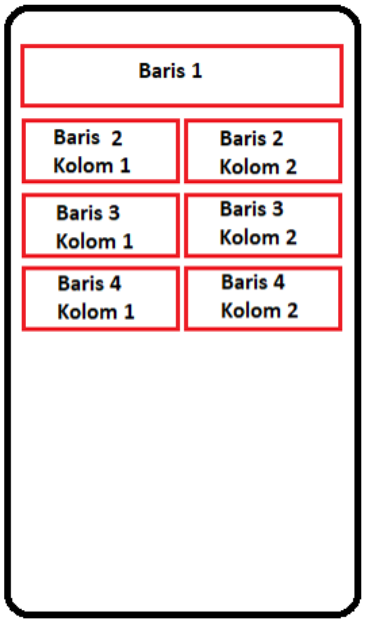
Dengan menggunakan layout model ini, kita dapat menentukan posisi suatu objek sesuai keinginan. Akan tetapi, penting untuk diperhatikan bahwa posisi suatu objek terhadap objek lainnya yang masih berada dalam grup yang sama harus dideklarasikan agar tampilan UI menjadi rapi.

5. **ConstraintLayout**

Layout ini merupakan tampilan baru yang disediakan oleh Android Studio dengan versi terbaru. ConstraintLayout membuat tampilan halaman menjadi lebih kompleks dan responsif. Hampir serupa dengan RelativeLayout, dengan layout jenis ini kita dapat melakukan desain UI dengan drag and drop saja.

6. **TableLayout**

TableLayout memungkinkan kita untuk menyusun tampilan halaman user kedalam bentuk baris dan kolom. Setiap baris dapat memiliki lebih dari sebuah kolom dan sebuah kolom dapat menyimpan sebuah objek pada Android. Untuk membuat tampilan table, Android menyediakan tag beruba <TableLayout> yang berisi susunan <TableRow>.



Gambar 15. Pemodelan Table Layout

Silahkan buat sebuah projek baru bernama **„BelajarLayout“.**

1. FrameLayout
   Buka file build.gradle (Module:…), kemudian tambahkan fitur viewBinding. Set nilainya menjadi true.

```
15      buildTypes {
16          release {
17              minifyEnabled false
18              proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'),
19          }
20      }
21      buildFeatures{
22          viewBinding true
23      }
24  }
```

Gambar 16. Build.gradle (Module)

Tambahkan sebuah *file* .xml baru. Caranya adalah buka folder res → klik kanan pada folder layout → pilih **Layout resource *file*** → beri nama **activity_frame_layout** → **Create** → kemudian ketikkan kode xml di bawah ini.

activity_frame_layout.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/txt_judul"
        android:layout_width="match_parent"
        android:layout_height="100dp"
        android:layout_margin="8dp"
        android:gravity="center|top"
        android:text="Frame Layout"
        android:textSize="40sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/btn_klik"
        android:layout_width="match_parent"
        android:layout_height="600dp"
        android:layout_gravity="center|bottom"
        android:layout_margin="8dp"
        android:onClick="onClick"
        android:text="Klik Disini\n\nHalaman 1"
        android:textSize="30sp" />

    <TextView
        android:id="@+id/txt_tombol"
        android:layout_width="match_parent"
        android:layout_height="600dp"
        android:layout_gravity="center|bottom"
        android:gravity="center"
        android:text="Halaman 2"
        android:textSize="30sp"
        android:textStyle="bold"
        android:visibility="gone" />
</FrameLayout>
```

Gambar 17. activity_frame_layout.xml

Kita membentuk sebuah tombol dan dua buah *TextView* yang menggunakan FrameLayout. Selanjutnya kita akan memberikan aksi kepada tombol tersebut, sehingga ketika di klik maka akan menampilkan *TextView* nya. Seperti yang terlihat pada teks, bahwa kita memberikan kode "onClick" pada objek *Button* di atas.

Langkah yang harus dilakukan yaitu buat sebuah *file* kotlin baru. Buka folder java, klik kanan pada package → **New** → **Kotlin File/*Class*** → beri nama **FrameLayout** → **Create** → ketik kode di bawah untuk menambahkan aksi "**onClick**" ketika tombol ditekan.

FrameLayout.kt

```kotlin
package com.example.belajarlayout

import android.os.Bundle
import android.view.View
import androidx.appcompat.app.AppCompatActivity
import com.example.belajarlayout.databinding.ActivityFrameLayoutBinding

class FrameLayout : AppCompatActivity() {

    private lateinit var binding : ActivityFrameLayoutBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityFrameLayoutBinding.inflate(layoutInflater)
        setContentView(binding.root)
    }

    fun onClick(v: View?) {
        binding.btnKlik.visibility = View.GONE
        binding.txtTombol.visibility = View.VISIBLE
    }
}
```

Gambar 18. FrameLayout.kt

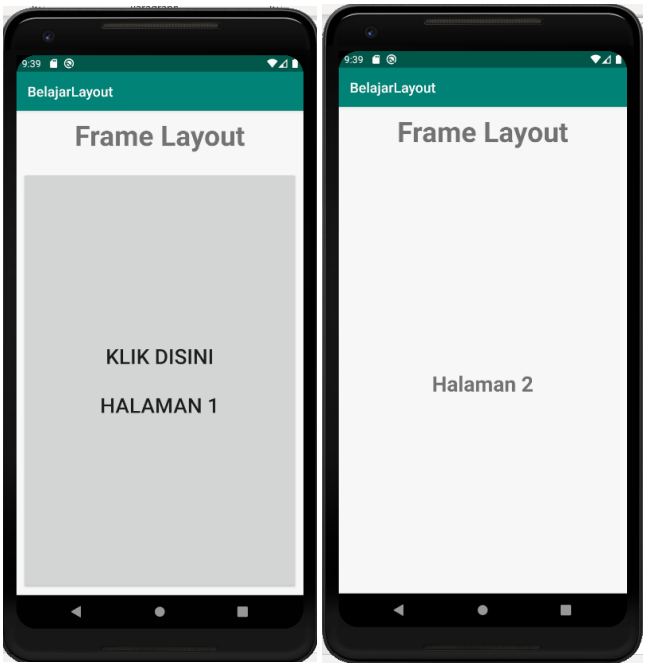
Jangan lupa ubah/tambahkan *activity* sesuai dengan nama *class* **pada AndroidManifest.xml** agar app dapat dijalankan pada emulator.

AndroidManifest.xml

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

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="BelajarLayout"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".FrameLayout">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

Gambar 19. AndroidManifest.xml

Lalu jalankan aplikasi tersebut. Jika berhasil, maka aplikasi akan muncul seperti pada tampilan di Gambar 20.



Gambar 20. Hasil dari FrameLayout

2. LinearLayout

Buat sebuah .xml baru → beri nama **activity_linear_layout** → ketik kode seperti di bawah. Anda dapat langsung melihat bentuk tampilan pada *tab design* tanpa perlu menjalankan aplikasi.

activity_linear_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Username"
        android:textSize="20sp" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="20sp" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Password"
        android:textSize="20sp" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="20sp" />

</LinearLayout>
```

Gambar 21. activity_linear_layout.xml

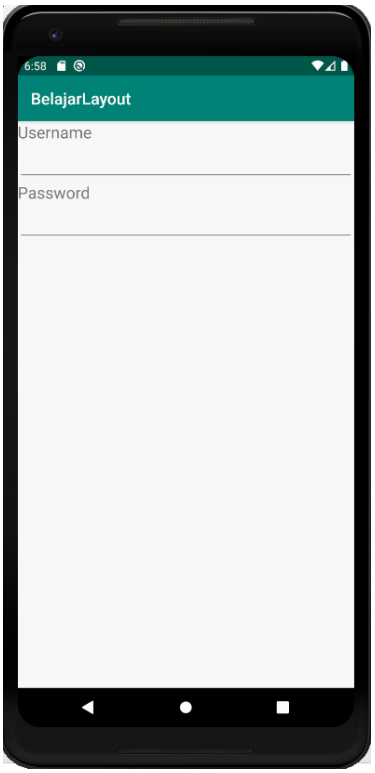
Kemudian buka tambahkan *class* kotlin baru bernama "**LinearLayout**". Ketikkan kode kotlin berikut ini. Kemudian, jangan lupa ubah *activity* android:name pada **AndroidManifest.xml** menjadi **.LinearLayout**.

LinearLayout.kt

```
1  package com.example.belajarlayout
2
3  import android.os.Bundle
4  import androidx.appcompat.app.AppCompatActivity
5
6  class LinearLayout : AppCompatActivity() {
7      override fun onCreate(savedInstanceState: Bundle?) {
8          super.onCreate(savedInstanceState)
9          setContentView(R.layout.activity_linear_layout)
10     }
11 }
```

Gambar 22. LinearLayout.kt

Jika selesai mengubah nama app pada manifest, maka jalankan aplikasi. Hasilnya akan muncul seperti pada contoh di bawah ini.



Gambar 23. Hasil dari LinearLayout

Tugas Analisa :

1. Ubah tampilan diatas menjadi layout LinearLayout secara **horizontal**.

3. RelativeLayout

Kali ini kita akan mencoba membuat sebuah halaman login sederhana dengan menggunakan RelativeLayout. Buat sebuah .xml baru → beri nama **activity_relative_layout** → ketikkan kode program seperti pada gambar di bawah ini.

activity_relative_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:id="@+id/txJudul"
        android:text="Halaman Login"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30sp"
        android:textStyle="bold"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="5dp"
        android:layout_marginBottom="20dp"/>

    <TextView
        android:id="@+id/txUsername"
        android:layout_width="120dp"
        android:layout_height="40dp"
        android:layout_below="@+id/txJudul"
        android:text="Nama User"
        android:textSize="20sp" />

    <EditText
        android:id="@+id/username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/txJudul"
        android:layout_toEndOf="@+id/txUsername"
        android:textSize="20sp"
        android:layout_marginBottom="10dp"/>

```

activity_relative_layout.xml

```
35      <TextView
36          android:id="@+id/txPass"
37          android:text="Kata Kunci"
38          android:layout_width="120dp"
39          android:layout_height="wrap_content"
40          android:layout_below="@+id/username"
41          android:textSize="20sp"/>
42
43      <EditText
44          android:id="@+id/pass"
45          android:layout_width="match_parent"
46          android:layout_height="wrap_content"
47          android:layout_below="@+id/username"
48          android:layout_toEndOf="@+id/txPass"
49          android:textSize="20sp"
50          android:layout_marginBottom="10dp"/>
51
52      <Button
53          android:id="@+id/btnLogin"
54          android:layout_width="120dp"
55          android:layout_height="wrap_content"
56          android:text="Masuk"
57          android:textSize="20sp"
58          android:layout_below="@id/pass"
59          android:layout_marginStart="90dp"/>
60
61      <Button
62          android:id="@+id/btnCancel"
63          android:layout_width="120dp"
64          android:layout_height="wrap_content"
65          android:text="Batal"
66          android:textSize="20sp"
67          android:layout_below="@id/pass"
68          android:layout_toEndOf="@+id/btnLogin"/>
69
70  </RelativeLayout>
```

Gambar 24. activity_relative_layout.xml

Kemudian tambahkan *file* kotlin baru bernama "**RelativeLayout**". Lalu ketikkan kode berikut ini.

RelativeLayout.kt

```
1   package com.example.belajarlayout
2
3   import android.os.Bundle
4   import androidx.appcompat.app.AppCompatActivity
5
6   class RelativeLayout : AppCompatActivity() {
7       override fun onCreate(savedInstanceState: Bundle?) {
8           super.onCreate(savedInstanceState)
9           setContentView(R.layout.activity_relative_layout)
10      }
11  }
```

Gambar 25. RelativeLayout.kt

Setelah selesai, maka jangan lupa untuk mengubah ***activity*** **android:name** pada **AndroidManifest.xml** menjadi **.RelativeLayout**. Kemudian anda dapat mencoba untuk menjalankan aplikasi tersebut. Tampilan akan muncul seperti pada contoh di bawah ini.

Gambar 26. Hasil dari RelativeLayout

4. TableLayout

Silahkan buat sebuah .xml baru bernama **activity_table_layout**. Lalu tambahkan kode xml seperti pada contoh di bawah ini.

activity_table_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_column="1"
            android:text="Halaman Rating"
            android:textSize="20sp"
            android:textStyle="bold" />

    </TableRow>

    <TableRow>

        <TextView
            android:text="Judul Film"
            android:layout_width="100dp"
            android:layout_height="wrap_content"
            android:layout_column="1" />

        <EditText
            android:width="200dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_column="2"/>
    </TableRow>
```

activity_table_layout.xml

```
37      <TableRow>
38
39          <TextView
40              android:text="Review"
41              android:layout_width="100dp"
42              android:layout_height="wrap_content"
43              android:layout_column="1" />
44
45          <EditText
46              android:width="200dp"
47              android:layout_width="wrap_content"
48              android:layout_height="wrap_content"
49              android:layout_column="2"/>
50      </TableRow>
51
52      <TableRow
53          android:layout_width="match_parent"
54          android:layout_height="match_parent">
55
56          <RatingBar
57              android:layout_width="match_parent"
58              android:layout_height="wrap_content"
59              android:id="@+id/ratingBar"
60              android:layout_column="2" />
61      </TableRow>
62
63      <TableRow
64          android:layout_width="match_parent"
65          android:layout_height="match_parent">
66
67          <Button
68              android:id="@+id/button"
69              android:layout_width="wrap_content"
70              android:layout_height="wrap_content"
71              android:layout_column="2"
72              android:text="Submit" />
73      </TableRow>
74
75  </TableLayout>
```

Gambar 27. activity_table_layout.xml

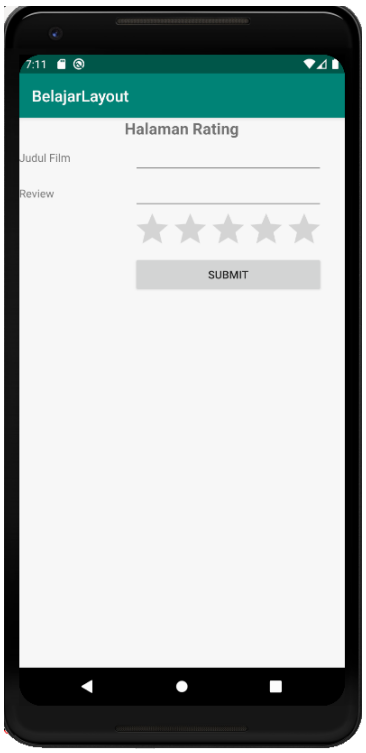
Setelah selesai menyusun kode xml, silahkan tambahkan kode Java berikut. Buat sebuah *file* kotlin baru bernama "**TableLayout**".

TableLayout.kt

```
1   package com.example.belajarlayout
2
3   import android.os.Bundle
4   import androidx.appcompat.app.AppCompatActivity
5
6   class TableLayout : AppCompatActivity() {
7       override fun onCreate(savedInstanceState: Bundle?) {
8           super.onCreate(savedInstanceState)
9           setContentView(R.layout.activity_table_layout)
10      }
11  }
```

Gambar 28. TableLayout.kt

Kemudian jangan lupa untuk mengubah ***activity*** **android:name** pada **AndroidManifest.xml** menjadi **.TableLayout**. Lalu jalankan aplikasi pada emulator Android anda. Jika berhasil, maka emulator akan menampilkan aplikasi seperti pada contoh di bawah ini.



Gambar 29. Hasil dari TableLayout

5. Constraint Layout

Silahkan buat sebuah .xml baru bernama **activity_constraint_layout**. Lalu tambahkan kode xml seperti pada contoh di bawah ini. Anda boleh menggunakan gambar untuk *profile* dengan men-*download* dari google. Lalu drag dan drop saja ke folder ***drawable***. Pastikan untuk **tidak menggunakan spasi** dalam penulisan nama gambar.

activity_constrain_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="16dp"
    android:background="#ADEFD1FF">

    <TextView
        android:id="@+id/txt_cl_judul"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:gravity="center"
        android:text="Halaman Profil"
        android:textSize="25sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

```
23      <ImageView
24          android:id="@+id/img_cl_foto"
25          android:layout_width="150dp"
26          android:layout_height="150dp"
27          android:layout_marginTop="24dp"
28          android:src="@drawable/user"
29          app:layout_constraintEnd_toEndOf="parent"
30          app:layout_constraintStart_toStartOf="parent"
31          app:layout_constraintTop_toBottomOf="@id/txt_cl_judul" />
32
33      <TextView
34          android:id="@+id/txt_cl_nama"
35          android:layout_width="wrap_content"
36          android:layout_height="wrap_content"
37          android:layout_marginTop="24dp"
38          android:text="Valentino Rossi"
39          android:textSize="18sp"
40          android:textStyle="bold"
41          app:layout_constraintEnd_toEndOf="parent"
42          app:layout_constraintStart_toStartOf="parent"
43          app:layout_constraintTop_toBottomOf="@id/img_cl_foto" />
```

```
45      <TextView
46          android:id="@+id/txt_cl_hobi"
47          android:layout_width="wrap_content"
48          android:layout_height="wrap_content"
49          android:layout_marginStart="24dp"
50          android:layout_marginTop="40dp"
51          android:text="Hobi"
52          android:textSize="18sp"
53          android:textStyle="bold"
54          app:layout_constraintStart_toStartOf="parent"
55          app:layout_constraintTop_toBottomOf="@id/txt_cl_nama" />
56
57      <TextView
58          android:id="@+id/txt_cl_fav"
59          android:layout_width="wrap_content"
60          android:layout_height="wrap_content"
61          android:layout_marginStart="24dp"
62          android:layout_marginTop="40dp"
63          android:text="Favorit"
64          android:textSize="18sp"
65          android:textStyle="bold"
66          app:layout_constraintStart_toStartOf="parent"
67          app:layout_constraintTop_toBottomOf="@id/edt_cl_hobi" />
```

```
69      <EditText
70          android:id="@+id/edt_cl_hobi"
71          android:layout_width="200dp"
72          android:layout_height="wrap_content"
73          android:layout_marginTop="24dp"
74          android:layout_marginEnd="32dp"
75          android:inputType="textPersonName"
76          app:layout_constraintEnd_toEndOf="parent"
77          app:layout_constraintStart_toEndOf="@id/txt_cl_hobi"
78          app:layout_constraintTop_toBottomOf="@id/txt_cl_nama" />
79
80      <EditText
81          android:id="@+id/edt_cl_fav"
82          android:layout_width="200dp"
83          android:layout_height="wrap_content"
84          android:layout_marginTop="24dp"
85          android:layout_marginEnd="32dp"
86          android:inputType="textPersonName"
87          app:layout_constraintEnd_toEndOf="parent"
88          app:layout_constraintHorizontal_bias="0.358"
89          app:layout_constraintStart_toEndOf="@id/txt_cl_fav"
90          app:layout_constraintTop_toBottomOf="@id/edt_cl_hobi" />
91
92      <Button
93          android:id="@+id/btn_simpan"
94          android:layout_width="match_parent"
95          android:layout_height="wrap_content"
96          android:layout_marginStart="16dp"
97          android:layout_marginTop="32dp"
98          android:layout_marginEnd="16dp"
99          android:background="#606060FF"
100         android:text="Simpan"
101         app:layout_constraintEnd_toEndOf="parent"
102         app:layout_constraintStart_toStartOf="parent"
103         app:layout_constraintTop_toBottomOf="@id/edt_cl_fav" />
104
105 </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 30. activity_constraint_layout.xml

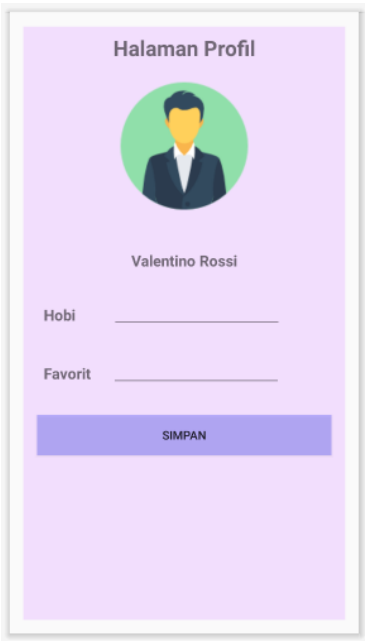
Kemudian tambahkan file kotlin baru bernama **"ConstraintLayout.kt"**. Atur layout seperti pada contoh di bawah ini.

ConstraintLayout.kt

```
1   package com.example.belajarlayout
2
3   import android.os.Bundle
4   import androidx.appcompat.app.AppCompatActivity
5
6   class ConstraintLayout:AppCompatActivity() {
7       override fun onCreate(savedInstanceState: Bundle?) {
8           super.onCreate(savedInstanceState)
9           setContentView(R.layout.activity_constrain_layout)
10      }
11  }
```

Gambar 31. ConstraintLayout.kt

Kemudian jangan lupa untuk mengubah ***activity*** **android:name** pada **AndroidManifest.xml** menjadi **.ConstraintLayout**. Lalu jalankan aplikasi pada emulator Android anda. Jika berhasil, maka emulator akan menampilkan aplikasi seperti pada contoh di bawah ini.



Gambar 32. Hasil dari ConstraintLayout

Tugas Analisa :

1. Tambahkan aksi ketika tombol Simpan di klik akan menampilkan informasi Hobi dan Favorit pada Toast.
2. Tambahkan sebuah tombol lagi dengan posisi dibawah tombol simpan untuk melakukan hapus teks informasi Hobi dan Favorit.