

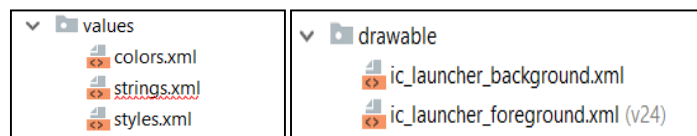
## Pertemuan 4

### View dan Resource File

Ketika mengembangkan aplikasi, ada baiknya dalam penulisan kode program dilakukan secara terpisah agar memudahkan fungsi kelola. Dalam hal desain tampilan UI pada Android, kita juga dapat melakukan hal tersebut. Pembuatan values seperti string, warna, dan dimensi dalam file resource yang terpisah akan membuat lebih mudah untuk mengelolanya, terutama jika kita menggunakan value tersebut lebih dari satu kali dalam layout kita.

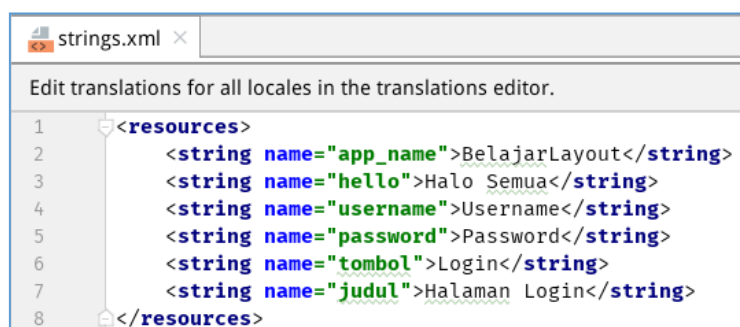
Sebagai contoh, cukup penting untuk menjaga string dalam file resource yang terletak terpisah untuk menerjemahkan dan melokalisasi aplikasi, sehingga kita dapat membuat file resource string untuk setiap bahasa tanpa perlu mengubah kode. File resource untuk gambar, warna, dimensi, dan atribut lainnya berguna untuk mengembangkan aplikasi untuk berbagai ukuran dan orientasi pada layar perangkat.

Ketika kita membuat sebuah proyek aplikasi Android, secara default telah tersedia tiga buah file pada folder values yang dapat kita tambah isinya. Serta terdapat folder drawable sebagai tempat pengumpulan file gambar yang terkait dengan tampilan aplikasi.



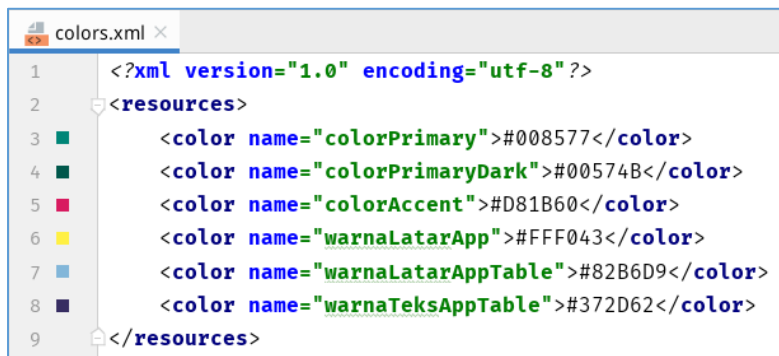
Gambar 1. Folder Resource

String resource terletak pada file **string.xml** di dalam folder **value**. Ketika folder **res** di expand, maka kita akan menemukan folder value ini. Pada file ini kita dapat menambahkan model string yang kira-kira akan kita gunakan berkali-kali pada tampilan sehingga tidak perlu menulis dari awal. Tambahkan kode ini pada file **string.xml**.



Gambar 2. Modifikasi file string.xml

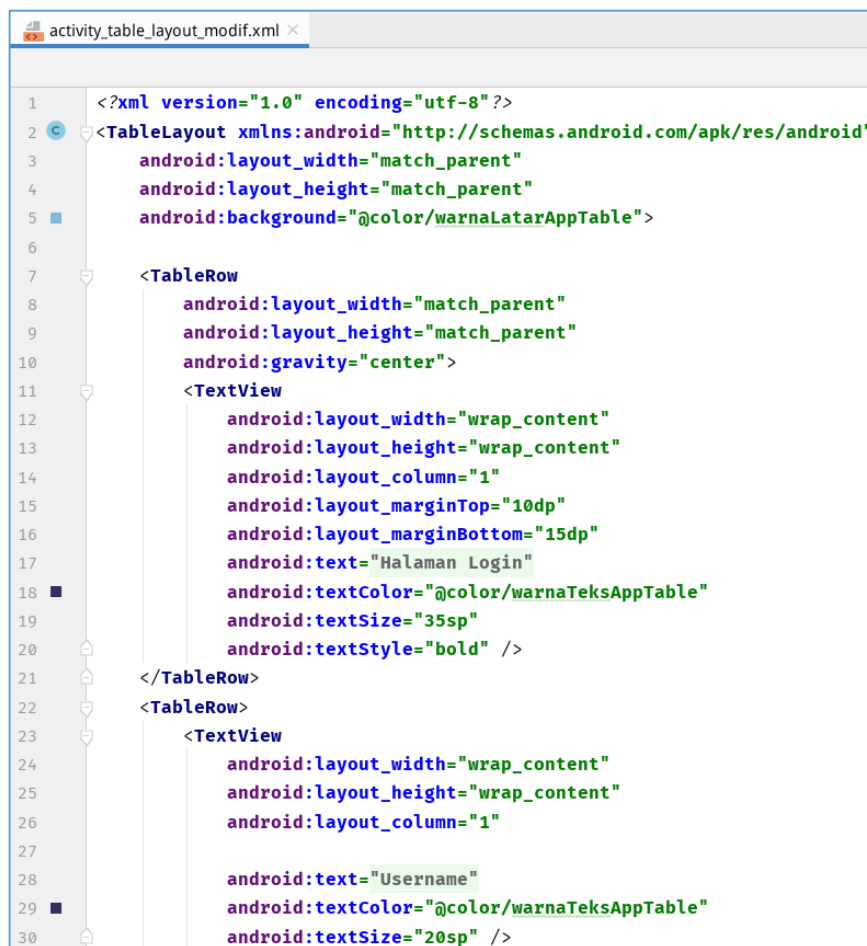
Kemudian tambahkan warna sesuai keinginan pada file **color.xml**.

A screenshot of an IDE showing the 'colors.xml' file. The file contains XML code defining a set of colors. The code is as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <color name="colorPrimary">#008577</color>
  <color name="colorPrimaryDark">#00574B</color>
  <color name="colorAccent">#D81B60</color>
  <color name="warnaLatarApp">#FFF043</color>
  <color name="warnaLatarAppTable">#82B6D9</color>
  <color name="warnaTeksAppTable">#372D62</color>
</resources>
```

Gambar 3. Modifikasi file colors.xml

Buka proyek sebelumnya yaitu “BelajarLayout”, buat sebuah file .xml baru bernama **activity\_layout\_table\_modif**, kemudian ketikkan kode berikut. Perhatikan bahwa pada contoh kode dibawah, kita menggunakan data yang telah kita simpan pada file **color.xml** dan **string.xml**. Perhatikan cara pemanggilannya.

A screenshot of an IDE showing the 'activity\_table\_layout\_modif.xml' file. The file contains XML code defining a table layout with two rows. The code is as follows:

```
<?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"
  android:background="@color/warnaLatarAppTable">
  <TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center">
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_column="1"
      android:layout_marginTop="10dp"
      android:layout_marginBottom="15dp"
      android:text="Halaman Login"
      android:textColor="@color/warnaTeksAppTable"
      android:textSize="35sp"
      android:textStyle="bold" />
    </TableRow>
  <TableRow>
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_column="1"
      android:text="Username"
      android:textColor="@color/warnaTeksAppTable"
      android:textSize="20sp" />
    </TableRow>
</TableLayout>
```

```
31 <EditText
32     android:id="@+id/edtUsername"
33     android:layout_width="wrap_content"
34     android:layout_height="wrap_content"
35     android:layout_column="2"
36     android:width="200dp"
37     android:textSize="20sp" />
38 </TableRow>
39
40 <TableRow>
41
42     <TextView
43         android:layout_width="match_parent"
44         android:layout_height="wrap_content"
45         android:layout_column="1"
46         android:text="Password"
47         android:textColor="@color/warnaTeksAppTable"
48         android:textSize="20sp" />
49
50     <EditText
51         android:id="@+id/edtPass"
52         android:layout_width="wrap_content"
53         android:layout_height="wrap_content"
54         android:layout_column="2"
55         android:width="200dp"
56         android:inputType="textPassword"
57         android:textSize="20sp" />
58 </TableRow>
59
60 <TableRow>
61     android:layout_width="match_parent"
62     android:layout_height="match_parent"
63     android:gravity="center">
64     <Button
65         android:id="@+id/btn_login"
66         android:layout_width="wrap_content"
67         android:layout_height="70dp"
68         android:layout_column="1"
69         android:layout_marginTop="15dp"
70         android:background="@color/warnaTeksAppTable"
71         android:text="Login"
72         android:textColor="@android:color/white"
73         android:textSize="20sp"
74         android:textStyle="bold" />
75 </TableRow>
</TableLayout>
```

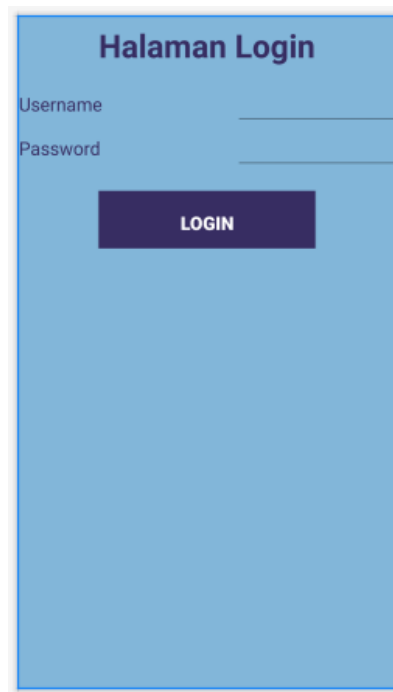
Gambar 4. activity\_table\_layout\_modif.xml

Setelah selesai melakukan proses desain tampilan halaman dengan menggunakan TableLayout, maka langkah selanjutnya adalah membuat class java baru bernama **"CustomTable"**. Kemudian ketikkan di bawah ini.

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

Gambar 5. CustomTable.kt

Jangan lupa mengganti activity pada **AndroidManifest.xml**. Jika sudah, maka silahkan jalankan aplikasi. Jika berhasil, maka tampilannya akan seperti pada gambar di bawah ini.



Gambar 6. Hasil dari CustomTable

Pada contoh yang kedua, kita akan membuat sebuah aplikasi kalkulator balok sederhana. Silahkan buat sebuah proyek baru bernama **"BelajarKalkulator"**. Kemudian jangan lupa tambahkan fitur **viewBinding true** pada **build.gradle (Module:..)**. Setelahnya silahkan buka file **strings.xml** untuk mendaftarkan teks yang akan digunakan pada aplikasi.



Gambar 7. File strings.xml

Setelah selesai mendaftarkan seluruh teks pada folder resources, maka buka halaman **activity\_main.xml** atau sesuai nama file yang dibuat (dalam modul ini **activity\_layout\_calculator.xml**). Lalu buat desain layout seperti kode xml dibawah ini untuk menambahkan tiga buah field inputan, dua buah tombol dan sebuah teks untuk menampilkan output menggunakan layout Linear.

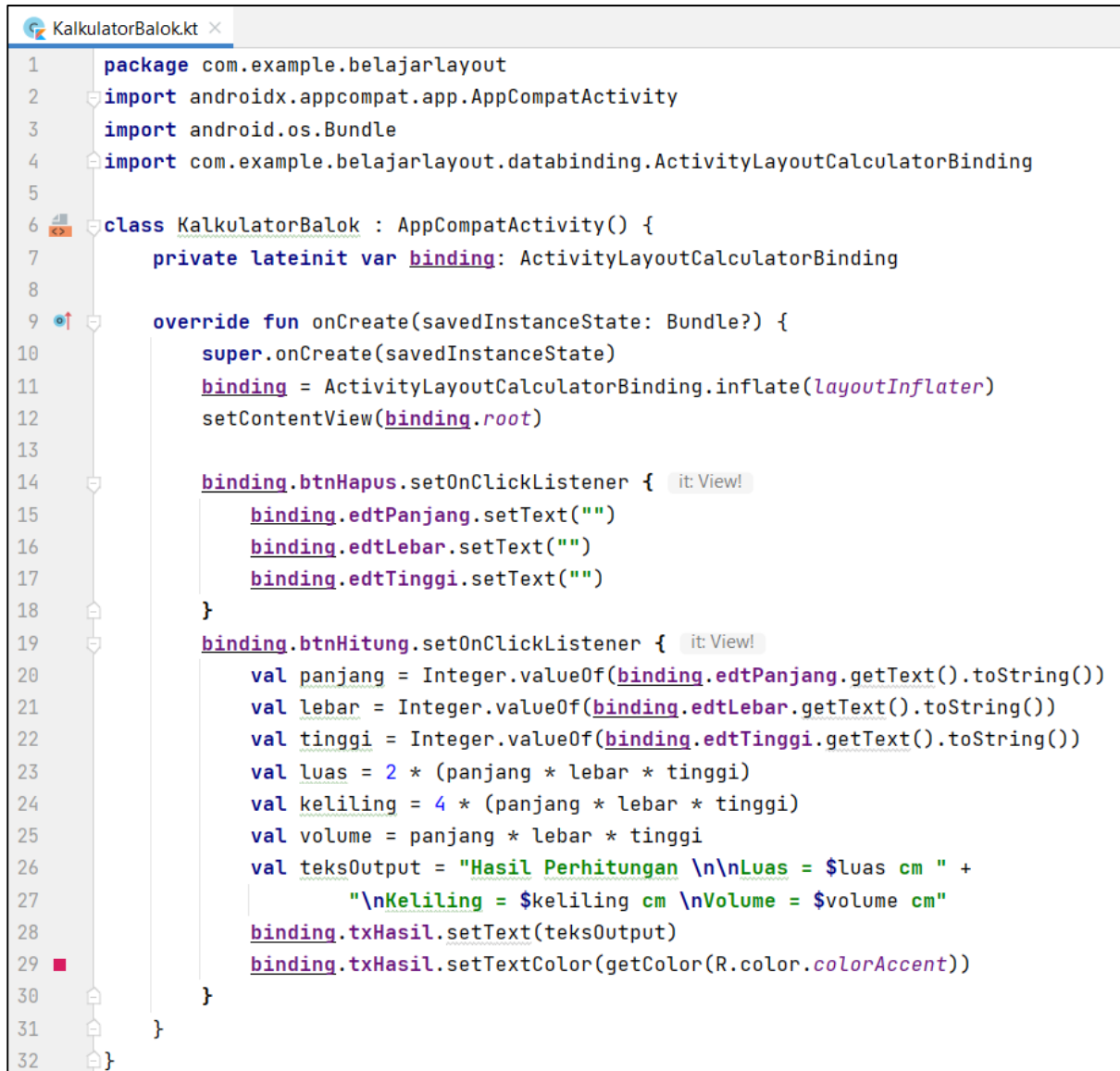


```
33 <EditText
34     android:id="@+id/edt_panjang"
35     android:layout_width="300dp"
36     android:layout_height="wrap_content"
37     android:hint="cm"
38     android:textSize="20sp"/>
39 </LinearLayout>
40
41 <LinearLayout
42     android:layout_width="wrap_content"
43     android:layout_height="wrap_content"
44     android:orientation="horizontal">
45
46     <TextView
47         android:layout_width="100dp"
48         android:layout_height="wrap_content"
49         android:paddingStart="5dp"
50         android:paddingEnd="15dp"
51         android:text="@string/txt_lebar"
52         android:textSize="18sp"
53         android:textStyle="bold"/>
54
55     <EditText
56         android:id="@+id/edt_lebar"
57         android:layout_width="300dp"
58         android:layout_height="wrap_content"
59         android:hint="cm"
60         android:textSize="20sp"/>
61
62 </LinearLayout>
```

```
64 <LinearLayout
65     android:layout_width="wrap_content"
66     android:layout_height="wrap_content"
67     android:orientation="horizontal">
68     <TextView
69         android:layout_width="100dp"
70         android:layout_height="wrap_content"
71         android:paddingStart="5dp"
72         android:paddingEnd="15dp"
73         android:text="Tinggi"
74         android:textSize="18sp"
75         android:textStyle="bold"/>
76
77     <EditText
78         android:id="@+id/edt_tinggi"
79         android:layout_width="300dp"
80         android:layout_height="wrap_content"
81         android:hint="cm"
82         android:textSize="20sp"/>
83 </LinearLayout>
84
85 <Button
86     android:id="@+id/btn_hitung"
87     android:layout_width="match_parent"
88     android:layout_height="wrap_content"
89     android:layout_marginTop="8dp"
90     android:backgroundTint="@color/colorPrimary"
91     android:text="Hitung"
92     android:textSize="20sp"/>
93
94 <Button
95     android:id="@+id/btn_hapus"
96     android:layout_width="match_parent"
97     android:layout_height="wrap_content"
98     android:layout_marginTop="8dp"
99     android:backgroundTint="@color/colorPrimary"
100    android:text="Hapus"
101    android:textSize="20sp"/>
102
103 <TextView
104     android:id="@+id/tx_hasil"
105     android:layout_width="wrap_content"
106     android:layout_height="wrap_content"
107     android:layout_gravity="center"
108     android:layout_marginTop="20dp"
109     android:text="Hasil Perhitungan"
110     android:textSize="20sp"
111     android:textStyle="bold"/>
112
113 </LinearLayout>
```

Gambar 8. Halaman activity\_layout\_calculator.xml

Jika sudah selesai merancang desain tampilan, maka tahap selanjutnya adalah membuat kode program kotlin nya. Silahkan buka halaman **MainActivity.kt** atau dalam modul ini halaman bernama **KalkulatorBalok.kt**. Kemudian kita akan membuat reaksi apabila tombol hitung dan tombol hapus ditekan oleh pengguna.



```
1 package com.example.belajarlayout
2 import androidx.appcompat.app.AppCompatActivity
3 import android.os.Bundle
4 import com.example.belajarlayout.databinding.ActivityLayoutCalculatorBinding
5
6 class KalkulatorBalok : AppCompatActivity() {
7     private lateinit var binding: ActivityLayoutCalculatorBinding
8
9     override fun onCreate(savedInstanceState: Bundle?) {
10         super.onCreate(savedInstanceState)
11         binding = ActivityLayoutCalculatorBinding.inflate(layoutInflater)
12         setContentView(binding.root)
13
14         binding.btnHapus.setOnClickListener { it: View!
15             binding.edtPanjang.setText("")
16             binding.edtLebar.setText("")
17             binding.edtTinggi.setText("")
18         }
19
20         binding.btnHitung.setOnClickListener { it: View!
21             val panjang = Integer.valueOf(binding.edtPanjang.getText().toString())
22             val lebar = Integer.valueOf(binding.edtLebar.getText().toString())
23             val tinggi = Integer.valueOf(binding.edtTinggi.getText().toString())
24             val luas = 2 * (panjang * lebar * tinggi)
25             val keliling = 4 * (panjang * lebar * tinggi)
26             val volume = panjang * lebar * tinggi
27             val teksOutput = "Hasil Perhitungan \n\nLuas = $luas cm " +
28                 "\nKeliling = $keliling cm \nVolume = $volume cm"
29             binding.txHasil.setText(teksOutput)
30             binding.txHasil.setTextColor(getColor(R.color.colorAccent))
31         }
32     }
33 }
```

Gambar 9. Halaman MainActivity.kt

Jika sudah selesai maka silahkan jalankan aplikasi pada emulator. Akan muncul halaman dimana pengguna bias menginputkan panjang, lebar dan tinggi. Ketika tombol hitung di klik maka akan menampilkan output hasil berupa luas, keliling dan volume. Jika tombol hapus di klik, maka konten pada field akan dibersihkan.



BelajarLayout	
<b>Kalkulator Balok Sederhana</b>	
Panjang	cm
Lebar	cm
Tinggi	cm
<b>HITUNG</b>	
<b>HAPUS</b>	
Hasil Perhitungan	

BelajarLayout	
<b>Kalkulator Balok Sederhana</b>	
Panjang	2
Lebar	3
Tinggi	1
<b>HITUNG</b>	
<b>HAPUS</b>	
Hasil Perhitungan	
Luas = 12 cm Keliling = 24 cm Volume = 6 cm	

Gambar 10. Hasil Implementasi

**Tugas Analisa :**

1. Apa manfaat dari penggunaan Resource File? Jelaskan.
2. Silahkan tambahkan juga pembersihan teks output pada kode program kotlin BelajarKalkulator diatas ketika tombol hapus di tekan.

## Activity dan Intent

*Intent* merupakan cara untuk menghubungkan antar *activity* di Android. *Intent* juga dapat membawa dan mengirimkan data ke *activity* yang lain atau bahkan ke aplikasi lainnya (Google Maps, Gmail, dsb). Secara umum, sebuah aplikasi memiliki lebih dari sebuah *activity* yang terhubung satu sama lain. Sebuah *activity* terhubung dengan sebuah *Layout*. Ada dua jenis *Intent* :

1. **Intent implicit** mendeklarasikan *Intent* untuk *activity* pada AndroidManifest.xml (<intent-filter>). Dengan begitu, maka android akan mengetahui *Intent* seperti apa yang dibutuhkan pada aplikasi. *Intent* jenis ini digunakan ketika kita tidak secara eksplisit tahu kegiatan mana yang harus dimulai dan kita ingin agar sistem Android yang memutuskan komponen mana yang akan dimulai. Jika sistem menemukan beberapa komponen yang dapat menangani *Intent* tersebut, maka kita dapat memilih yang dibutuhkan.

```
Intent i=new Intent ("com.example.counter.MainAction");
startActivity(i);
```

Gambar 11. Contoh Intent Implicit

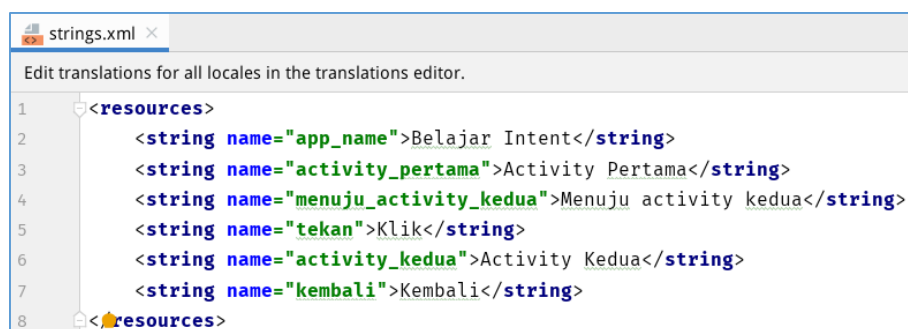
2. **Intent explicit** digunakan karena kita sudah tahu Aktivitas mana yang ingin dimulai. Jadi, dengan *intent* jenis ini, kita dapat secara langsung menulis nama *class* yang kita tuju pada kode program.

```
Intent i=new Intent (this,MainActivity.class);
startActivity(i);
```

Gambar 12. Contoh Intent Explicit

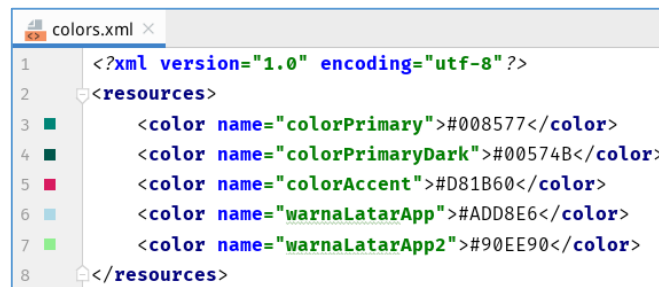
Untuk mencoba *Intent*, mari kita coba buat aplikasi sederhana untuk menampilkan halaman baru ketika tombol ditekan. Kita juga akan mengirim nilai dari sebuah halaman ke halaman lainnya. Semua *activity* yang kita miliki dan terkait dengan *interface*, maka harus didaftarkan pada AndroidManifest.xml agar proyek tersebut mengenal *activity* apa saja yang akan dipakai.

Buat sebuah proyek baru, beri nama “BelajarIntent” → kemudian buka file **strings.xml**. Ketikkan modifikasi string di bawah ini.



Gambar 13. Modifikasi Strings.xml

Setelah itu, silahkan buka file **colors.xml** untuk menambahkan jenis warna yang diinginkan. Anda dapat menggunakan warna yang anda sukai.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3     <color name="colorPrimary">#008577</color>
4     <color name="colorPrimaryDark">#00574B</color>
5     <color name="colorAccent">#D81B60</color>
6     <color name="warnaLatarApp">#ADD8E6</color>
7     <color name="warnaLatarApp2">#90EE90</color>
8 </resources>
```

Gambar 14. Modifikasi Colors.xml

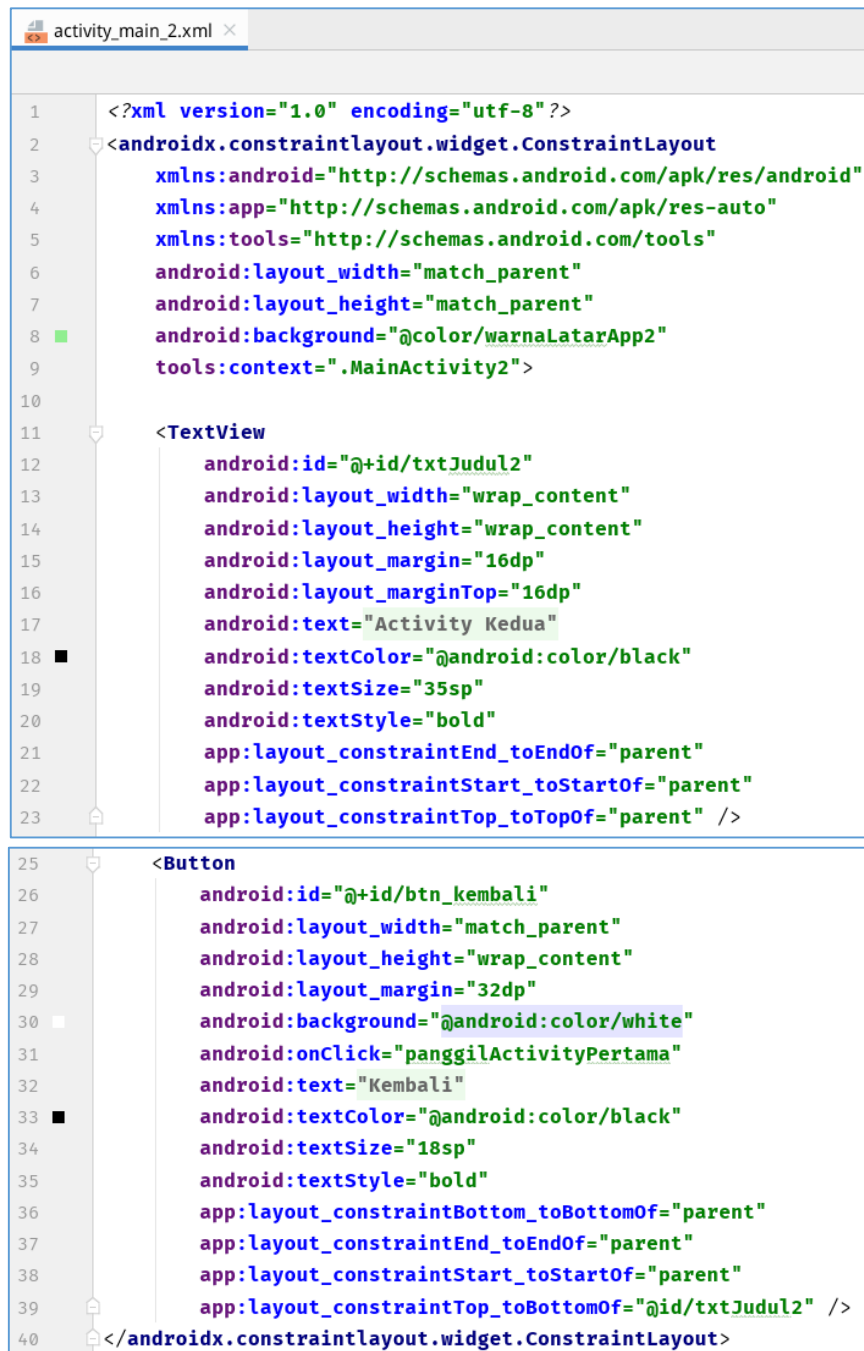
Selanjutnya, buka file **activity\_main.xml**. Kemudian buat tampilan dengan menggunakan kode xml berikut ini.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     xmlns:tools="http://schemas.android.com/tools"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     android:background="@color/warnaLatarApp"
9     tools:context=".MainActivity">
10
11     <TextView
12         android:id="@+id/txtJudul"
13         android:layout_width="wrap_content"
14         android:layout_height="wrap_content"
15         android:layout_margin="16dp"
16         android:layout_marginTop="16dp"
17         android:text="Activity Pertama"
18         android:textColor="@android:color/black"
19         android:textSize="35sp"
20         android:textStyle="bold"
21         app:layout_constraintEnd_toEndOf="parent"
22         app:layout_constraintStart_toStartOf="parent"
23         app:layout_constraintTop_toTopOf="parent" />
24
25     <Button
26         android:id="@+id/btn_klik"
27         android:layout_margin="32dp"
28         android:layout_width="match_parent"
29         android:layout_height="wrap_content"
30         android:background="@android:color/white"
31         android:onClick="panggilActivityKedua"
32         android:text="Klik"
33         android:textColor="@android:color/black"
34         android:textSize="18sp"
35         android:textStyle="bold"
36         app:layout_constraintBottom_toBottomOf="parent"
37         app:layout_constraintEnd_toEndOf="parent"
38         app:layout_constraintStart_toStartOf="parent"
39         app:layout_constraintTop_toBottomOf="@id/txtJudul" />
40 </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 15. activity\_main.xml

Kemudian tambahkan sebuah resource file bernama **activity\_main\_2.xml**, kemudian tambahkan kode program xml berikut ini.



Gambar 16. activity\_main2.xml

Jika sudah selesai, maka silahkan buka file **MainActivity.kt**. Kemudian tambahkan logika program berikut ini. Halaman ini akan membuka file MainActivity2 ketika tombol klik ditekan.

```
1 package com.example.shumaya.belajarintent
2
3 import android.content.Intent
4 import android.os.Bundle
5 import android.view.View
6 import androidx.appcompat.app.AppCompatActivity
7
8 class MainActivity : AppCompatActivity() {
9     override fun onCreate(savedInstanceState: Bundle?) {
10         super.onCreate(savedInstanceState)
11         setContentView(R.layout.activity_main)
12     }
13
14     fun panggilActivityKedua(view: View?) {
15         val i = Intent(applicationContext, MainActivity2::class.java)
16         i.putExtra(name: "Value1", value: "Belajar Android")
17         i.putExtra(name: "Value2", value: "Pemrograman Mobile")
18         startActivity(i)
19     }
20 }
```

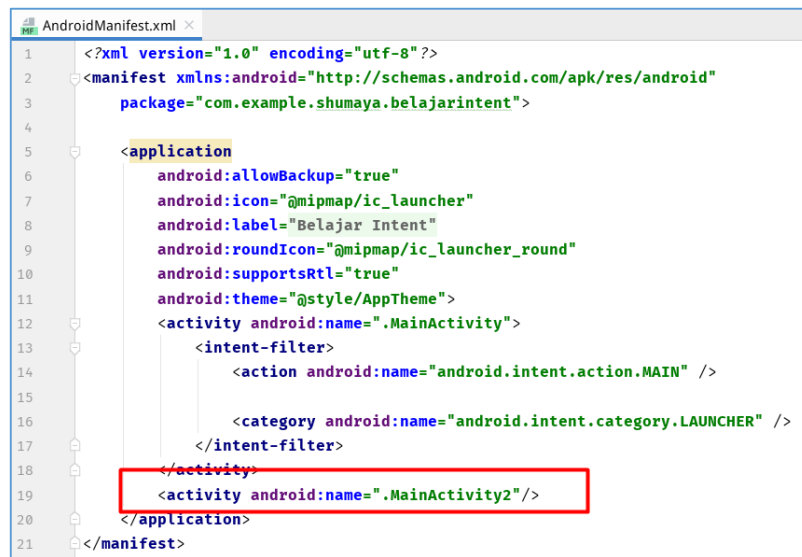
Gambar 17. MainActivity.kt

Kemudian, agar halaman kedua dapat dibuka maka kita akan menambahkan sebuah file kotlin lagi dengan nama **MainActivity2.kt**. Kemudian kita akan membuat logika program yang akan mengembalikan halaman ke MainActivity ketika tombol kembali ditekan.

```
1 package com.example.shumaya.belajarintent
2
3 import android.content.Intent
4 import android.os.Bundle
5 import android.view.View
6 import android.widget.Toast
7 import androidx.appcompat.app.AppCompatActivity
8
9 class MainActivity2 : AppCompatActivity() {
10     override fun onCreate(savedInstanceState: Bundle?) {
11         super.onCreate(savedInstanceState)
12         setContentView(R.layout.activity_main_2)
13         val extras : Bundle? = intent.extras
14         val value1 : String? = extras!!.getString(key: "Value1")
15         val value2 : String? = extras.getString(key: "Value2")
16         Toast.makeText(applicationContext, text: ""Value pertama adalah :
17             First value: $value1
18             Value kedua adalah: $value2"", Toast.LENGTH_LONG).show()
19     }
20
21     fun panggilActivityPertama(view: View?) {
22         val i = Intent(applicationContext, MainActivity::class.java)
23         startActivity(i)
24     }
25 }
```

Gambar 18. MainActivity2.kt

Selanjutnya jangan lupa mendaftarkan file **MainActivity2.kt** ke dalam **AndroidManifest.xml** agar file dapat dikenali oleh aplikasi.



Gambar 19. AndroidManifest.xml

Jika selesai, maka silahkan jalankan aplikasi. Jika aplikasi berhasil, maka ketika tombol klik ditekan akan membuka halaman kedua. Selain itu, data yang berjalan dari halaman satu ke halaman dua akan muncul dengan menggunakan pesan Toast seperti pada gambar dibawah ini.



Gambar 20. Hasil dari Belajar Intent

### Latihan :

Silahkan design tampilan menarik untuk sebuah halaman **registrasi** yang berisi tampilan registrasi berisi **username, password, nama lengkap dan alamat**. Kemudian ketika tombol submit ditekan akan masuk ke halaman login yang berisi form berjudul **Login**. Didalamnya terdapat dua buah form inputan yaitu username dan password. Ketika tombol „**login**“ di tekan akan masuk ke halaman kedua yang menampilkan informasi „**Selamat Datang, (username)**“. Silahkan gunakan warna dan icon terbaik yang anda mampu untuk membuat halaman registrasi dan login yang menarik.