

MODUL I

PENGENALAN *IDE ANDROID STUDIO*

A. TUJUAN PRAKTIKUM

- Mengetahui dan memahami *IDE Android Studio* beserta komponen-komponennya.

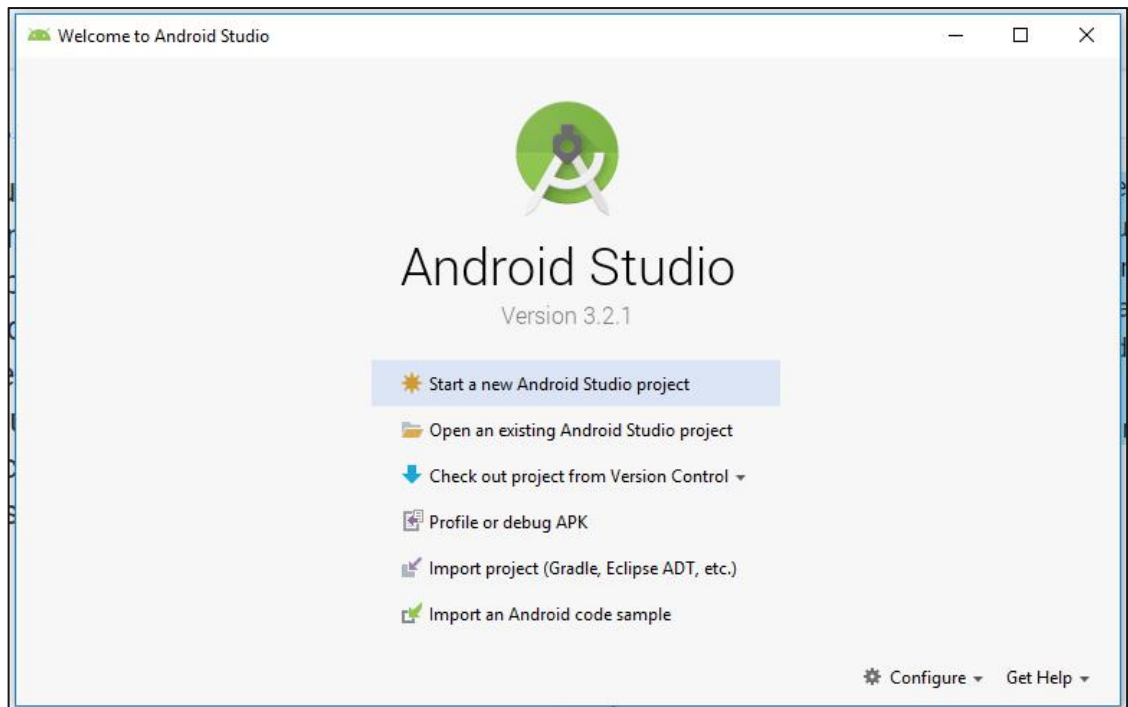
B. DASAR TEORI

1. IDE ANDROID STUDIO

Android Studio adalah Lingkungan Pengembangan Terpadu - Integrated Development Environment (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA . Selain merupakan editor kode IntelliJ dan alat pengembang yang berdaya guna, Android Studio menawarkan fitur lebih banyak untuk meningkatkan produktivitas kita saat membuat aplikasi Android, misalnya:

- a. Sistem versi berbasis Gradle yang fleksibel.
- b. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat Android.
- c. Instant Run untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru.
- d. Template kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh.
- e. Alat pengujian dan kerangka kerja yang ekstensif.
- f. Alat Lint untuk meningkatkan kinerja, kegunaan, kompatibilitas versi, dan masalah-masalah lain.
- g. Dukungan C++ dan NDK.
- h. Dukungan bawaan untuk Google Cloud Platform, mempermudah pengintegrasian Google Cloud Messaging dan App Engine.

Kali pertama menjalankan Android Studio, kita akan melihat tampilan seperti berikut ini.



Untuk memulai proyek baru pilihlah **“Start a new Android Studio project”**.

Setelah menginstall Android Studio, mari kita mulai untuk membuat sebuah aplikasi **Hallo World** atau aplikasi pertama kita.

1. Setelah melakukan **“Start a new Android Studio project”**, kita diminta untuk melakukan konfigurasi dalam pembuatan proyek baru kita. Dalam dialog ini kita bisa memberi nama dari aplikasi kita, lokasi proyek kita dan company domain. Company domain akan digunakan dalam identifikasi unik dari aplikasi kita ketika sudah di-publish. Kita juga dapat mengganti dari direktori dimana proyek kita akan disimpan.
2. Dialog selanjutnya adalah target *devices*, di dalam dialog ini kita bisa memilih target *devices* dari aplikasi yang akan kita buat. Kita juga bisa mengganti nilai minimum SDK, yang berfungsi untuk membatasi penggunaan API pada sebuah Aplikasi.

Create New Project

Create Android Project

Application name
My Application

Company domain
com.fti.helloworld

Project location
C:\Users\yahma\Desktop\MyApplication2

Package name
helloworld.fti.com.myapplication Edit

☐ Include C++ support

☐ Include Kotlin support

Previous Next Cancel Finish

Create New Project

Target Android Devices

Select the form factors and minimum SDK

Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.

☒ **Phone and Tablet**
API 21: Android 5.0 (Lollipop)
By targeting **API 21 and later**, your app will run on approximately **85.0%** of devices. [Help me choose](#)

☐ Include Android Instant App support

☐ **Wear OS**
API 23: Android 6.0 (Marshmallow)

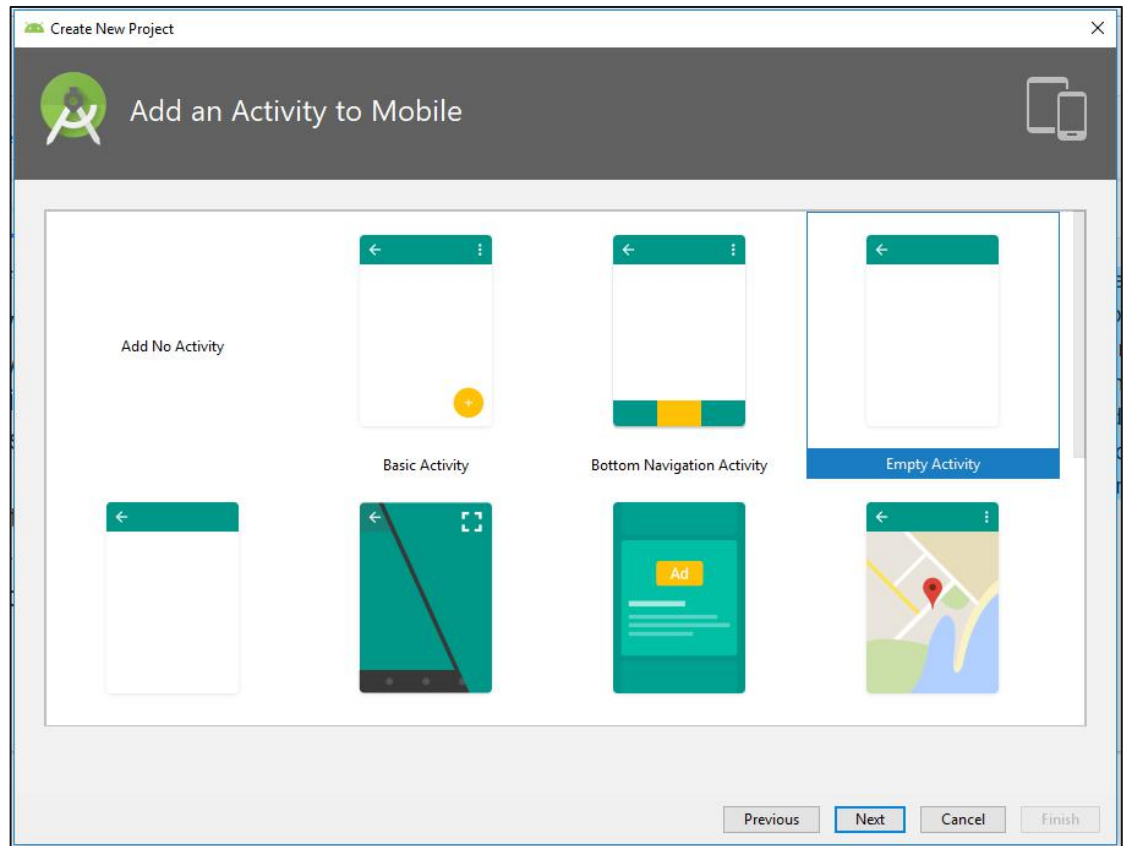
☐ **TV**
API 21: Android 5.0 (Lollipop)

☐ **Android Auto**

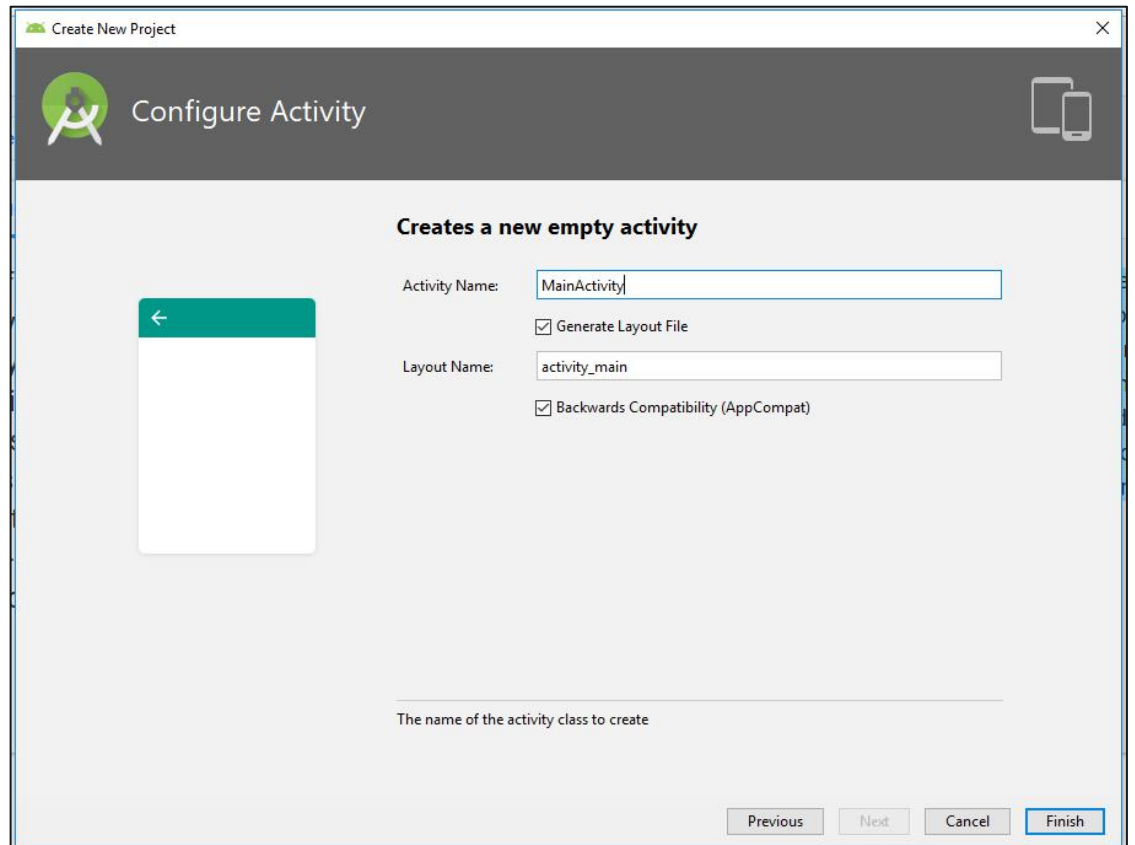
☐ **Android Things**
API 24: Android 7.0 (Nougat)

Previous Next Cancel Finish

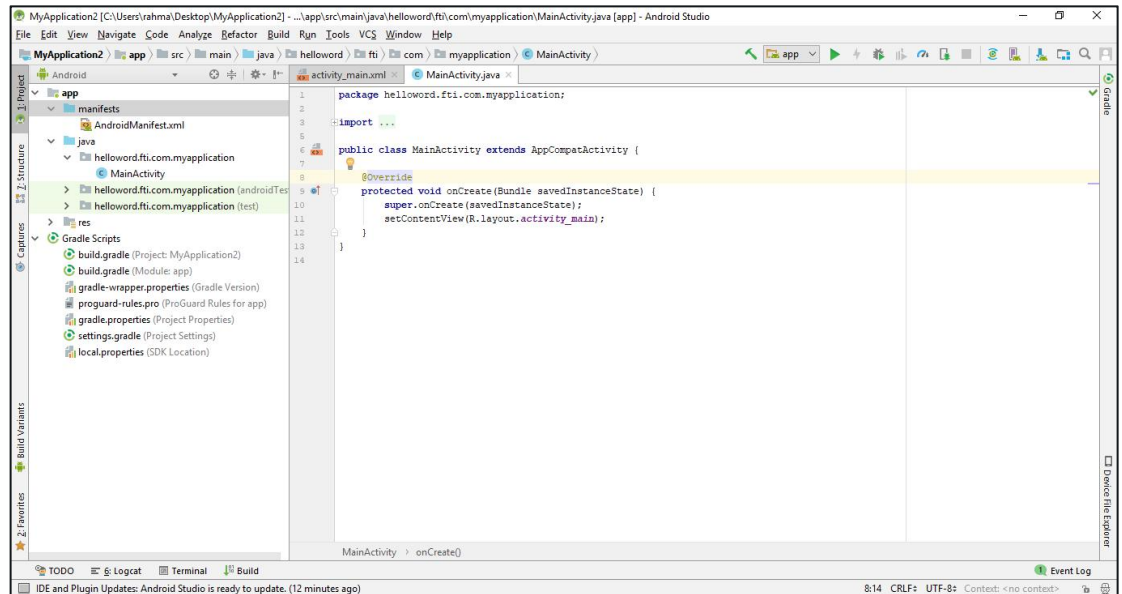
3. Dialog selanjutnya adalah **default template**. Terdapat beberapa template yang bisa kita gunakan seperti **Empty Activity**, **Login Activity**, dan lain-lain.



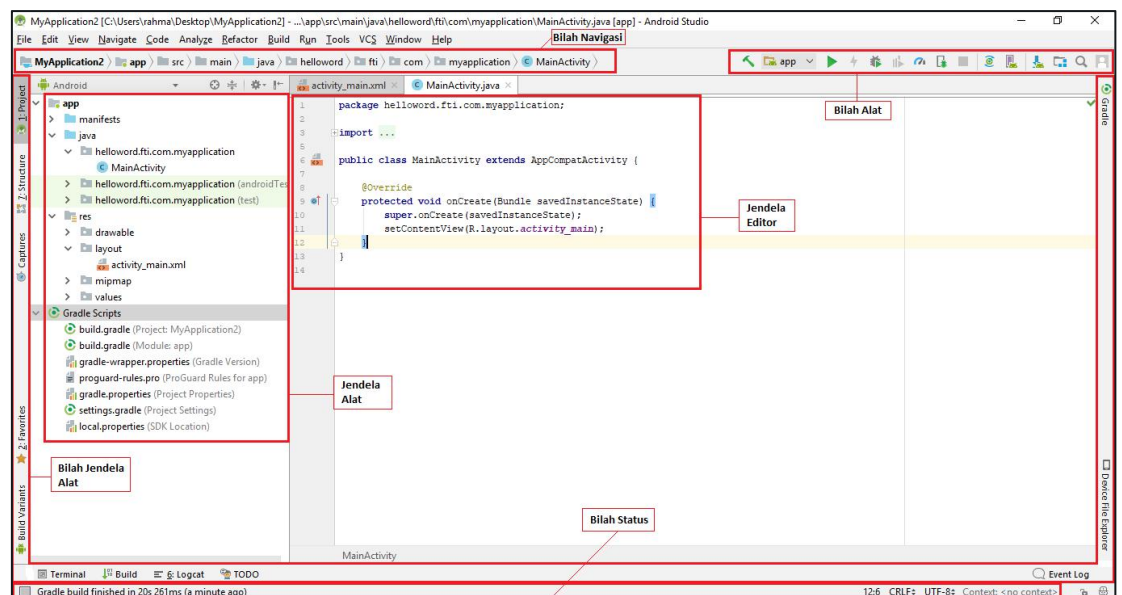
4. Dialog selanjutnya adalah nama dari Activity yang pertama kali kita buat.



5. Setelah proses create project selesai, maka akan ditampilkan antar muka android studio.



2. KOMPONEN IDE ANDROID STUDIO



a. Bilah Alat

Bilah alat memungkinkan kita untuk melakukan berbagai jenis tindakan, termasuk menjalankan aplikasi dan meluncurkan alat Android.

b. Bilah Navigasi

Bilah navigasi membantu kita bernavigasi di antara proyek dan membuka file untuk diedit. Bilah ini memberikan tampilan struktur yang terlihat lebih ringkas dalam jendela Project.

c. Jendela Editor

Jendela editor adalah tempat kita membuat dan memodifikasi kode. Bergantung pada jenis file saat ini, editor dapat berubah. Misalnya, ketika melihat file tata letak, editor menampilkan Layout Editor.

d. Bilah Jendela Alat

Bilah jendela alat muncul di luar jendela IDE dan berisi tombol yang memungkinkan kita meluaskan atau menciutkan jendela alat individual.

e. Jendela Alat

Jendela alat memberi kita akses ke tugas tertentu seperti pengelolaan proyek, penelusuran, kontrol versi, dan banyak lagi. Kita bisa meluaskan dan juga menciutkannya.

f. Bilah Status

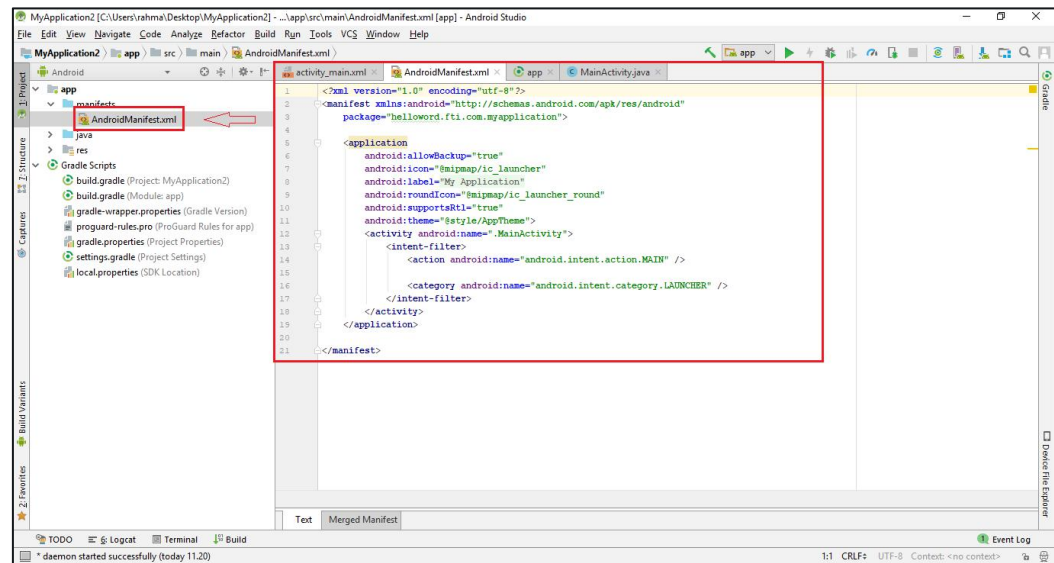
Bilah status menampilkan status proyek kita dan IDE itu sendiri, serta setiap peringatan atau pesan.

3. ANDROID MANIFEST

Setiap aplikasi harus memiliki file `AndroidManifest.xml` (bernama persis seperti ini) di direktori akar. File manifest menyediakan informasi penting tentang aplikasi ke sistem Android, yang harus dimiliki sistem agar bisa menjalankan setiap kode aplikasi. Yang dilakukan file manifest di antaranya:

- Menamai paket Java untuk aplikasi. Nama paket berfungsi sebagai identifier unik untuk aplikasi.
- Menjelaskan berbagai komponen aplikasi, yang menyertakan aktivitas, layanan, penerima siaran, dan penyedia materi yang membentuk aplikasi. Juga menamai kelas yang mengimplementasikan masing-masing komponen dan menerbitkan kemampuannya, seperti pesan Intent yang dapat mereka tangani. Deklarasi ini menginformasikan sistem Android mengenai komponen dan kondisi yang memungkinkan peluncurannya.
- Menentukan proses yang menjadi host komponen aplikasi.
- Mendeklarasikan izin aplikasi yang harus dimiliki aplikasi untuk mengakses bagian yang dilindungi pada API dan berinteraksi dengan aplikasi lain. Juga mendeklarasikan izin lain yang harus dimiliki untuk berinteraksi dengan komponen aplikasi, seperti izin menggunakan internet, izin membuka gallery, dll.
- Mencantumkan daftar kelas `Instrumentation` yang memberikan profil dan informasi lain saat aplikasi berjalan. Deklarasi ini hanya ada di manifest saat aplikasi dibuat dan dihapus sebelum aplikasi dipublikasikan.
- Mendeklarasikan level minimum Android API yang diperlukan aplikasi.

- Mencantumkan daftar pustaka yang harus ditautkan aplikasi.

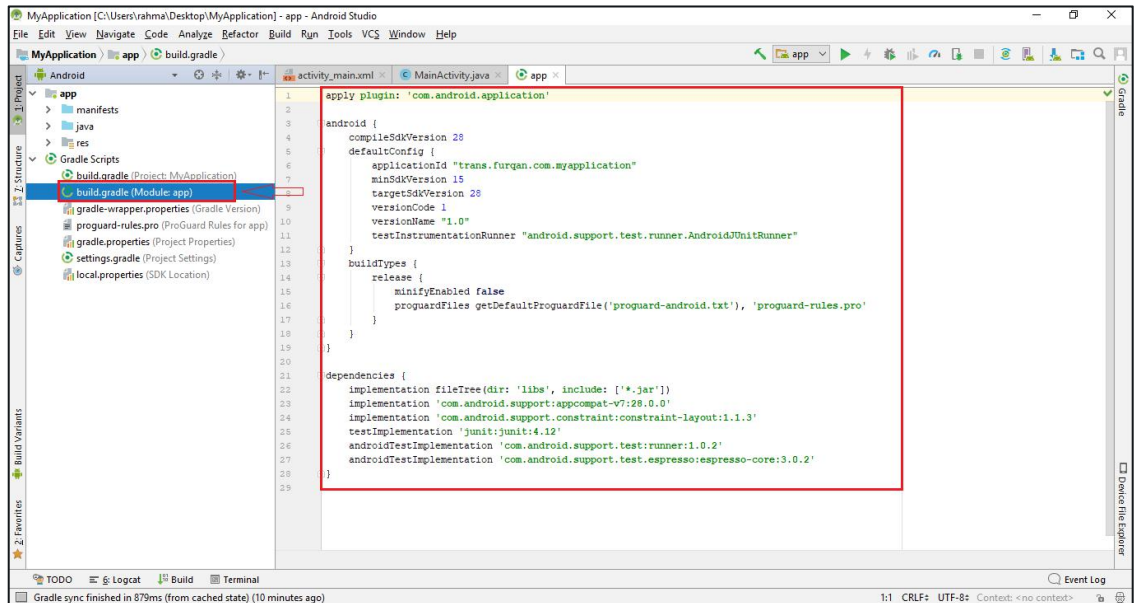


4. GRADLE

Android Studio menggunakan Gradle sebagai dasar sistem versi, dengan kemampuan khusus Android yang disediakan oleh Plugin Android untuk Gradle. Sistem ini bisa dijalankan sebagai alat terpadu dari menu Android Studio dan secara independen dari baris perintah. Kita bisa menggunakan fitur-fitur sistem versi untuk melakukan yang berikut:

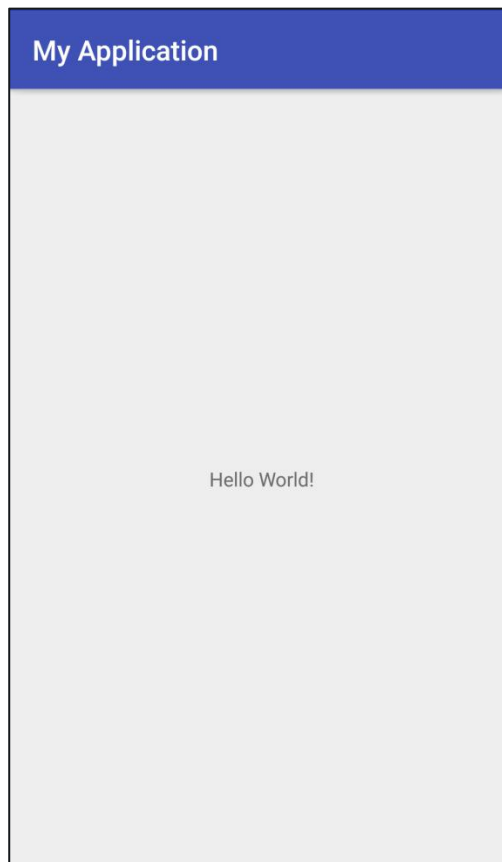
- Menyesuaikan, mengonfigurasi, dan memperluas proses pembangunan.
- Membuat beberapa APK untuk aplikasi Android kita, dengan aneka fitur menggunakan proyek dan modul yang sama. Menggunakan kembali kode dan sumber daya pada seluruh set sumber.

Dengan menerapkan fleksibilitas Gradle, kita dapat mencapai semua ini tanpa mengubah file sumber inti aplikasi. File versi Android Studio diberi nama `build.gradle`. File ini adalah teks biasa yang menggunakan Groovy mengonfigurasi versi dengan elemen yang disediakan oleh plugin Android untuk Gradle. Masing-masing proyek memiliki file versi level atas untuk seluruh proyek dan file versi level modul terpisah untuk setiap modul. Saat kita mengimpor proyek saat ini, Android Studio otomatis menghasilkan file versi yang diperlukan.



C. TUGAS PRAKTIKUM

Agar dapat melakukan proses run pada hp android, maka hp tersebut harus dapat terhubung dengan computer dan menghidupkan mode debugging usb. Cari tau bagaimana cara menghidupkan mode debugging usb hp android kalian.



MODUL II

LAYOUT

A. TUJUAN PRAKTIKUM

- Memperkenalkan apa itu layout.
- Dapat membuat sebuah layout menggunakan Bahasa pemrograman xml..

B. DASAR TEORI

1. PENGENALAN LAYOUT

Layout mendefinisikan struktur visual untuk antarmuka pengguna, seperti UI sebuah aktivitas atau widget aplikasi. kita dapat mendeklarasikan layout dengan dua cara:

- a. Deklarasikan elemen UI dalam XML.** Android menyediakan sebuah kosakata XML sederhana yang sesuai dengan kelas dan subkelas View, seperti halnya untuk widget dan layout.
- b. Buat instance elemen layout saat waktu proses.** Aplikasi kita bisa membuat objek View dan ViewGroup (dan memanipulasi propertinya) lewat program.

Kerangka kerja Android memberi kita fleksibilitas untuk menggunakan salah satu atau kedua metode ini guna mendeklarasikan dan mengelola UI aplikasi kita. Misalnya, kita bisa mendeklarasikan layout default aplikasi kita dalam XML, termasuk elemen-elemen layar yang akan muncul di dalamnya dan di propertinya. kita nanti bisa menambahkan kode dalam aplikasi yang akan memodifikasi status objek layar, termasuk yang dideklarasikan dalam XML, saat waktu proses.

Keuntungan mendeklarasikan UI dalam XML adalah karena hal ini memungkinkan kita memisahkan penampilan aplikasi dari kode yang mengontrol perilakunya dengan lebih baik. Keterangan UI kita bersifat eksternal bagi kode aplikasi kita, yang berarti bahwa kita bisa memodifikasi atau menyesuaikannya tanpa harus memodifikasi dan mengompilasi ulang kode sumber. Misalnya, kita bisa membuat layout XML untuk berbagai orientasi layar, berbagai ukuran layar perangkat, dan berbagai bahasa. Selain itu, mendeklarasikan layout dalam XML akan mempermudah kita memvisualisasikan struktur UI, sehingga lebih mudah men-debug masalahnya.

2. KOMPONEN LAYOUT

Sebuah layout terdiri dari beberapa gabungan view, view dibagi menjadi dua bagian yaitu :

a. Parent View

Merupakan sebuah view yang didalamnya dapat diisi oleh child view ataupun oleh parent view lainnya, beberapa bentuk dari parent view :

- Linear Layout

LinearLayout adalah sekelompok view yang menyejajarkan semua anak atau child view dalam satu arah, secara vertikal atau horizontal. kita bisa menetapkan arah layout dengan atribut `android:orientation`. Semua anak *LinearLayout* akan ditumpuk satu sama lain, jadi daftar vertikal hanya akan memiliki satu anak per baris, seberapa pun lebarnya, dan daftar horizontal hanya akan setinggi satu baris (tinggi anak yang tertinggi, ditambah pengisi). *LinearLayout* akan mengikuti *margin* antara anak dan *gravity* (sejajar kanan, tengah, atau kiri) setiap anak (child).

```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
</LinearLayout>
```

- Relative Layout

RelativeLayout adalah Layout ini menampilkan komponen dalam posisi relatif. Posisi masing-masing Layout dapat ditentukan sebagai relatif terhadap elemen saudara (seperti di sebelah kiri atau di bawah tampilan lain) atau pada posisi relatif terhadap area *RelativeLayout* induk (seperti sejajar dengan bagian bawah, kiri atau tengah). Jadi, dengan menggunakan *RelativeLayout*, kita dapat menyusun suatu komponen secara relatif terhadap komponen lainnya.

```
<?xml version="1.0" encoding="utf-8" ?>
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
</RelativeLayout>
```

b. Child View

Merupakan sebuah view yang letaknya harus berada didalam sebuah parent view, terdapat banyak child view, diantaranya :

- `TextView` : Sebuah view untuk menampilkan tulisan.
- `ImageView` : Sebuah view untuk menampilkan gambar.
- `Button` : Sebuah tombol untuk melakukan aksi/proses.

- EditText : Sebuah form untuk input tulisan.

C. LANGKAH PRAKTIKUM

Pada praktikum kali ini kita akan mencoba membuat sebuah tampilan layout menggunakan *LinearLayout* dan *RelativeLayout*.

1. DESAIN DENGAN *LINEARLAYOUT*

Buka kelas *activity_main.xml* pada project yang telah kita buat sebelumnya, kelas tersebut dapat ditemukan pada *app->res->layout->activity_main.xml*, pada kelas tersebut tuliskan code di bawah ini.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/tv_pertama"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Baris pertama" />
    <TextView
        android:id="@+id/tv_kedua"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Baris ke dua"/>
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <Button
            android:id="@+id/btn_satu"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="kolom satu"/>
        <Button
            android:id="@+id/btn_dua"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="kolom dua"/>
    </LinearLayout>
</LinearLayout>
```

Pada Listing code diatas terdapat dua *LinearLayout* sebagai parent view, Pada *LinearLayout* pertama terdapat dua *TextView* sebagai child view yang memiliki orientasi vertical, atau sejajar kebawah. Tetapi didalam *LinearLayout* tersebut juga terdapat sebuah linear layout lagi yang mendefinisikan setiap child yang ada didalamnya memiliki arah horizontal, sehingga button dengan nama kolom satu dan kolom dua sejajar kesamping. Setiap view dapat diberi id yang nanti akan digunakan pada kelas activity agar dapat mengedit view yang memiliki id tersebut.



2. DESAIN DENGAN *RELATIVELAYOUT*

Kali ini kita akan melakukan edit pada *activity_main.xml* yang telah kita buat sebelumnya dari *LinearLayout* menjadi *RelativeLayout*,

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

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=".MainActivity">
    <TextView
        android:id="@+id/tv_pertama"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Baris pertama" />
```

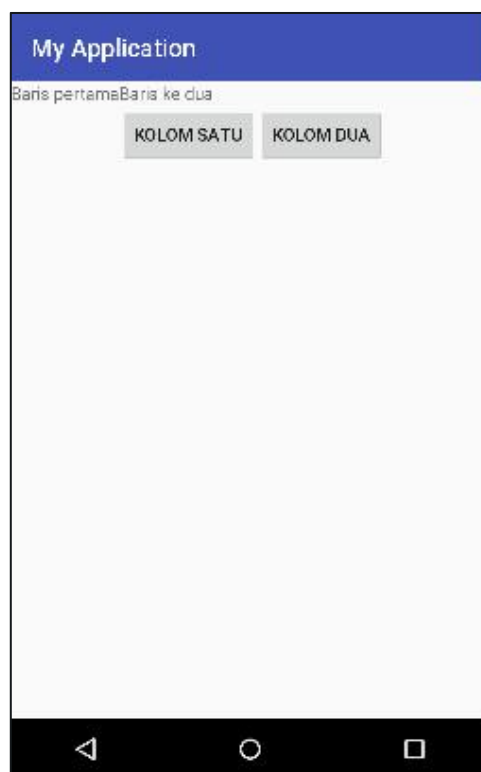
```

<TextView
    android:layout_toRightOf="@+id/tv_pertama"
    android:id="@+id/tv_kedua"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Baris ke dua"/>
<Button
    android:layout_toRightOf="@+id/tv_pertama"
    android:layout_below="@+id/tv_kedua"
    android:id="@+id/btn_satu"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="kolom satu"/>
<Button
    android:layout_below="@+id/tv_kedua"
    android:layout_toRightOf="@id/btn_satu"
    android:id="@+id/btn_dua"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="kolom dua"/>

</RelativeLayout>

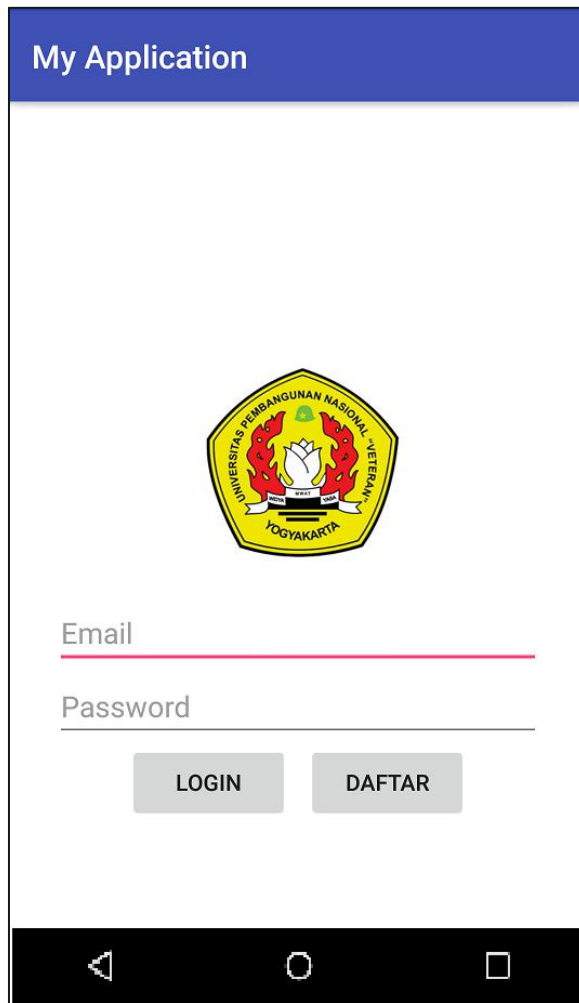
```

Pada listing code diatas kita menampilkan sesuat yang sama seperti pada *LinearLayout* sebelumnya, tetapi textview dengan id tv_kedua diletakan pada sebelah kanan textview yang memiliki id tv_pertama dengan atribut *layout_toRightOf="@+id/tv_pertama"*. Pada button dengan id btn_pertama, diletakkan di bawah textview dengan id tv_kedua, dan dikanan textview dengan id tv_pertama, begitu juga dengan button kedua dengan pengaturan yang telah kita tetapkan.



D. TUGAS PRAKTIKUM

1. Buat lah sebuah tampilan layout seperti dibawah ini.



The screenshot shows a mobile application interface with a blue header bar containing the text "My Application". Below the header, there is a yellow hexagonal logo of Universitas Pembangunan Nasional Veteran Yogyakarta. Under the logo, there are two text input fields: "Email" and "Password", each with a red underline. Below the "Password" field, there are two gray buttons labeled "LOGIN" and "DAFTAR". At the bottom of the screen, there is a black navigation bar with three white icons: a triangle, a circle, and a square.

2. Praktikan dibebaskan untuk menggunakan *LinearLayout* atau *RelativeLayout*, disarankan untuk menggabungkan kedua hal tersebut.

MODUL III

PENYAJIAN DATA

A. TUJUAN PRAKTIKUM

- Memahami cara penyajian data pada Aplikasi Android.

B. DASAR TEORI

Penyajian data adalah cara bagaimana seorang programmer mampu menyajikan berbagai data dan proses tertentu pada aplikasi mobile yang dibangun agar dapat ditampilkan pada view yang telah dirancang oleh programmer tersebut.

Pada pertemuan kali ini, praktikan akan diberikan berbagai materi tertentu tentang bagaimana cara mengolah data yang diberikan, proses pengolahan yang akan dilakukan dapat berupa bagaimana mengolah angka dengan operasi aritmatika seperti penjumlahan, pengurangan, perkalian dan pembagian sehingga praktikan mampu membangun sebuah aplikasi mobile sederhana seperti kalkulator.

Berikut langkah-langkah membangun sebuah aplikasi mobile kalkulator menghitung volume balok sederhana :

1. Buatlah sebuah project *empty basic* baru dengan nama “Kalkulator Penjumlahan” atau sesuai keinginan kita.
2. Setelah proses build selesai, buka direktori “res” dan pilih *activity_main.xml* dan IDE akan menampilkan *source code* seperti berikut :

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
```

3. Ganti *parent view* yang semula *ConstraintLayout* menjadi *LinearLayout*, hal ini bertujuan untuk mengubah metode *layouting* yang semula *constraint* menjadi *Linear*, dan tambahkan *android:orientation="vertical"* didalam tag *LinearLayout* pertama sehingga akan terlihat seperti berikut :

```
<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:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:orientation="vertical"
>
```

4. Hapus Tag *<TextView>* yang berada didalam *view group* tersebut dan tambahkan tag berikut secara berurutan , *EditText* , *EditText*, *EditText*, *Button*, *TextView* dengan masing-masing lebar adalah *layout_width="match_parent"* dan tinggi *layout_height="wrap_content"*.

5. Berikan id pada masing-masing komponen tersebut secara berurutan "*et_panjang*" , "*et_lebar*" , "*et_tinggi*" , "*btn_jumlah*" , "*tv_hasil*" lalu berikan atribut *textAlignment="center"* dan *hint* pada masing-masing komponen dan isi sesuai keinginan, sehingga *source code* akan terlihat seperti berikut :

```
<EditText
    android:id="@+id/et_panjang"
    android:hint="Panjang"
    android:inputType="number"
    android:textAlignment="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<EditText
    android:id="@+id/et_lebar"
    android:hint="Lebar"
    android:inputType="number"
    android:textAlignment="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<EditText
    android:id="@+id/et_tinggi"
    android:hint="Tinggi"
    android:inputType="number"
    android:textAlignment="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<Button
    android:id="@+id/btn_hasil"
```



```

        android:text="Jumlahkan"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
<TextView
    android:id="@+id/tv_hasil"
    android:text="0.0"
    android:textSize="40sp"
    android:textAlignment="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

```

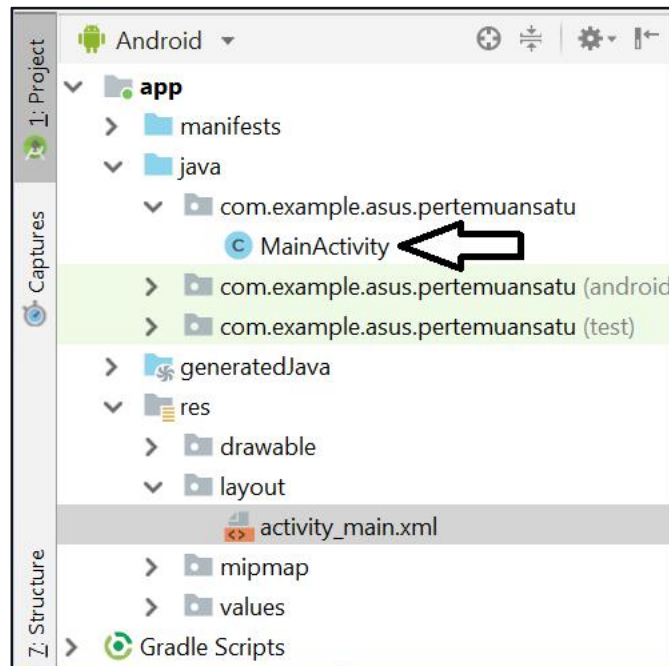
Pada preview layout, akan menampilkan output seperti berikut :



6. Pada tag <TextView> terdapat atribut textSize yang berguna untuk mengatur size dari tulisan dengan satuan “sp”

7. Proses yang dilakukan pada program tersebut adalah, nilai dari masing-masing EditText panjang, lebar dan tinggi akan diambil untuk kemudian diproses menjadi proses perhitungan volume balok, proses tersebut akan berjalan apabila Button “Jumlahkan” di klik dan hasil dari perkalian tersebut akan ditampilkan pada TextView hasil.

8. Setelah proses *layouting* dilakukan, buka direktori java yang terdapat pada bagian kiri IDE dan pilih *class MainActivity.java* :



9. Maka akan terlihat *source code* seperti berikut :

```
package com.example.asus.pertemuansatu;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

Pada class *activity_main.xml* programmer telah membuat 3 EditText , 1 Button dan 1 TextView, supaya dapat dilakukan proses pada masing-masing komponen tersebut , maka deklarasikan kembali didalam *class MainActivity* diluar *method onCreate*, dengan menambahkan code seperti berikut :

```
private Button btnHasil;
private TextView tvHasil;
private EditText etPanjang, etLebar, etTinggi;
```

Setelah *programmer* menambahkan code tersebut, pada masing-masing tipe data akan terdapat error, untuk mengatasi hal tersebut, lakukan perintah sesuai dengan pop-up yang ditampilkan IDE dengan menekan kombinasi tombol Alt+Enter , bertujuan untuk menambahkan library secara otomatis pada *class* tersebut

```
? android.widget.EditText? Alt+Enter
private TextView tvHasil;
private EditText etPanjang, etLebar, etTinggi;
```

10. Supaya masing-masing variabel yang telah dideklarasikan tersebut dapat terhubung dengan `activity_main.xml` dan dapat digunakan, perintah yang dilakukan adalah menuliskan kode `findViewById(R.id.id_komponen)` didalam `method onCreate` sehingga code akan terlihat seperti berikut :

```
public class MainActivity extends AppCompatActivity {

    private Button btnHasil;
    private TextView tvHasil;
    private EditText etPanjang, etLebar, etTinggi;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        btnHasil = findViewById(R.id.btn_hasil);
        tvHasil = findViewById(R.id.tv_hasil);
        etPanjang = findViewById(R.id.et_panjang);
        etLebar = findViewById(R.id.et_lebar);
        etTinggi = findViewById(R.id.et_tinggi);
    }
}
```

11. Langkah selanjutnya adalah membuat code untuk melakukan proses pengambilan nilai dari `activity_main.xml` ke class `MainActivity.java` apabila button Hasil di klik, proses tersebut klik tersebut dilakukan dengan menambahkan `setOnClickListener` pada variabel `btnHasil` , source code seperti berikut :

```
btnHasil.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //write your code here
    }
});
```

12. Kode diatas akan dijalankan apabila user mengklik `button` Hasil yang terdapat pada layar aplikasi mobile yang dibangun, kemudian langkah selanjutnya adalah membuat proses perhitungan volume balok, tambahkan kode berikut didalam `public void onClick(View v)` :

```

public void onClick(View v) {
    String sPanjang = etPanjang.getText().toString();
    String sLebar = etLebar.getText().toString();
    String sTinggi = etTinggi.getText().toString();

    double panjang = Double.parseDouble(sPanjang);
    double lebar = Double.parseDouble(sLebar);
    double tinggi = Double.parseDouble(sTinggi);

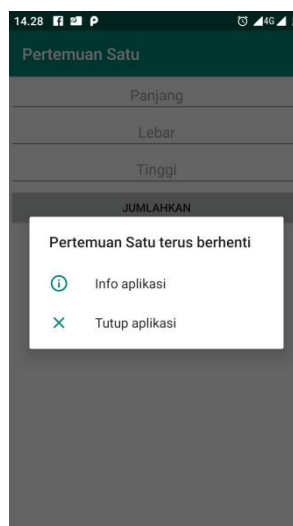
    double hasil = panjang*lebar*tinggi;

    String sHasil = String.valueOf(hasil);
    tvHasil.setText(sHasil);
}

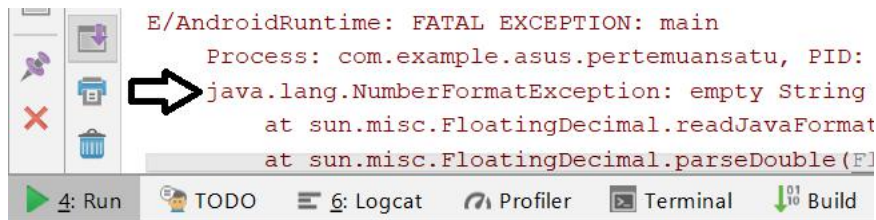
```

13. Proses yang dilakukan pada kode diatas adalah pada saat *button* Hasil di klik, program pertama kali melakukan proses pengambilan nilai yang telah diinputkan oleh *user* ke EditText panjang, lebar dan tinggi yang bertipe String , kemudian supaya dapat diproses , terlebih dahulu merubah tipe String tersebut ke *double*, tujuan dirubah ke *double* supaya apabila terjadi perkalian yang memiliki koma, hasil akan tetap berwujud desiman , setelah dilakukan konversi dari String ke double, dilakukan proses perkalian *panjang x lebar x tinggi* , dan hasilnya disimpan ke variabel *hasil* yang bertipe double, setelah itu supaya dapat ditampilkan kembali kedalam layar aplikasi mobile, variabel *hasil* bertipe *double* tersebut dikonversikan kembali ke variabel *sHasil* bertipe String, dan tuliskan syntax *tvHasil.setText(sHasil);* tujuan dilakukannya konversi ke String terlebih dahulu supaya tidak terjadi *error*, karena TextView tersebut hanya bisa menampilkan data yang bertipe String.

14. Kelemahan yang masih terjadi adalah apabila user tidak mengisi salah satu EditText yang disediakan, maka akan terjadi *crash* seperti berikut :



Apabila *programmer* melihat ke bagian consol->Run, Error tersebut disebabkan oleh *NumberFormatException* :



,error tersebut terjadi dikarenakan satu atau lebih field tidak diisi sehingga menyebabkan terjadinya error, untuk mengatasi hal tersebut, tambahkan *error handling* seperti *try catch* kedalam kode sebelumnya, keseluruhan source code btnHasil terlihat seperti berikut :

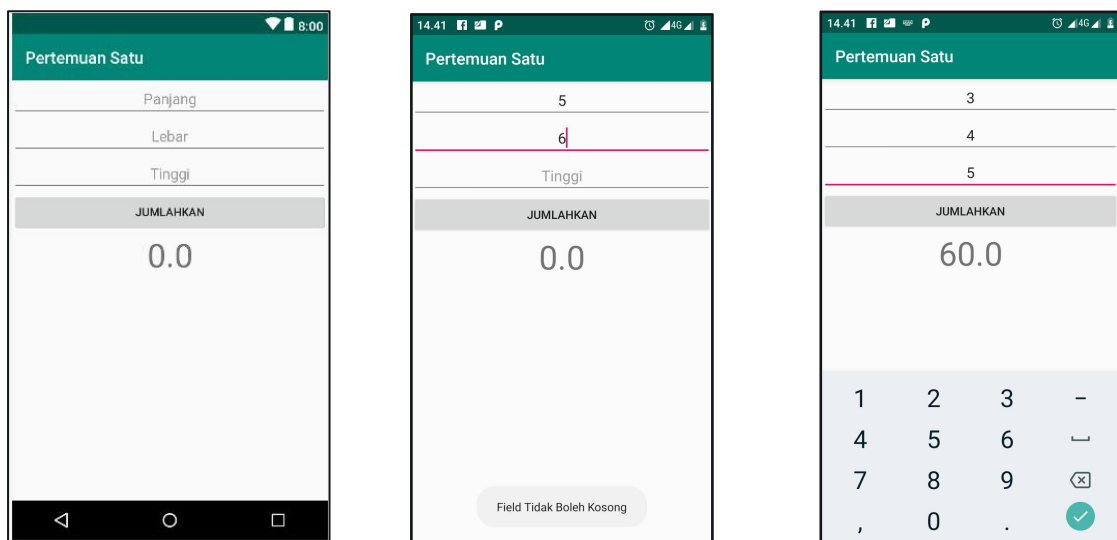
```
btnHasil.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try{
            String sPanjang = etPanjang.getText().toString();
            String sLebar = etLebar.getText().toString();
            String sTinggi = etTinggi.getText().toString();

            double panjang = Double.parseDouble(sPanjang);
            double lebar = Double.parseDouble(sLebar);
            double tinggi = Double.parseDouble(sTinggi);

            double hasil = panjang*lebar*tinggi;

            String sHasil = String.valueOf(hasil);
            tvHasil.setText(sHasil);
        }catch (NumberFormatException nfe){
            Toast.makeText(getApplicationContext(), "Field Tidak Boleh
Kosong", Toast.LENGTH_SHORT).show();
        }
    }
});
```

15. Program telah selesai dibuat, berikut output dari aplikasi kalkulator menghitung volume balok :



C. TUGAS PRAKTIKUM

- a) Kembangkan sekreatif mungkin apa yang telah kita pelajari pada pembangunan aplikasi Kalkulator Menghitung Volume Balok, menjadi aplikasi Kalkulator yang lebih kompleks dengan minimal terdapat proses aritmatika Penjumlahan, Pengurangan, Perkalian dan Pembagian, dan minimal terdapat 2 EditText dan Satu TextView hasil.

MODUL IV

INTENT DAN DATA

A. TUJUAN PRAKTIKUM

- Memahami cara berpindah halaman antar layout pada Aplikasi Android

B. DASAR TEORI

1. INTENT

Intent adalah sebuah kelas dalam programming Android yang berfungsi untuk perpindahan halaman.

Intent juga merupakan suatu objek yang terdapat dalam suatu activity dimana objek tersebut dapat komunikasi dengan activity yang lain, baik activity pada fungsi internal android misal seperti memanggil activity dalam satu package atau beda package yang masih berada dalam satu project.

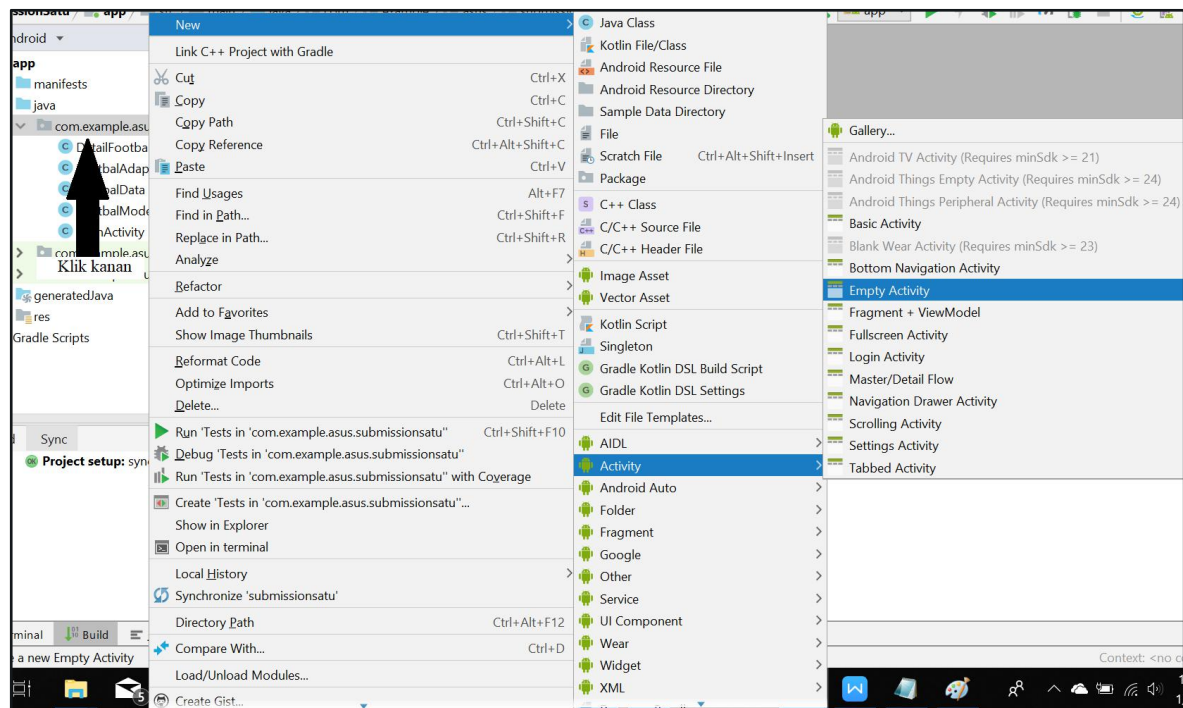
Intent merupakan objek tipe `android.content.Intent`. Melalui metode `startActivity()` yang digunakan untuk memulai sebuah activity lain.

Intent dibagi menjadi 2, yaitu :

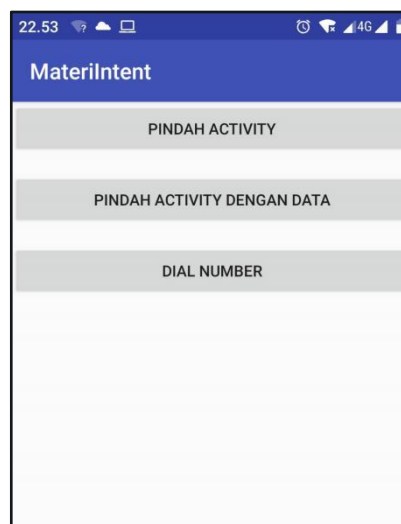
- **Explicit Intent** berfungsi untuk mengaktifkan komponen-komponen dalam satu aplikasi yang sama. Misalnya seperti : Berpindah Activity.
- **Implicit Intent** berfungsi untuk memanggil fungsi activity yang sudah ada di fungsi internal android seperti Dial Number, Open Browser dan lainnya

a) Contoh Pertama:

- 1) Buatlah sebuah project baru dengan nama sesuai keinginan kita menggunakan *empty activity*, disini penulis memberi nama "ModulEmpat".
- 2) Setelah project baru selesai dibuat, tambahkan *activity* baru dengan klik kanan pada direktori tempat *MainActivity.java* berada dengan cara seperti berikut, dan beri nama activity baru tersebut dengan *ActivityTujuan* :



- 3) Selanjutnya pada *activity_main.xml* Buatlah tampilan view dengan tampilan seperti berikut :



Kemudian berilah *id* pada masing-masing komponen di *activity_main.xml* tersebut lalu buka *class MainActivity.java* , deklarasikan masing-masing komponen tersebut dan hubungkan variabelnya menggunakan *findViewById()* , contoh : `btnPindahActivity = findViewById(R.id.btn_pindah_activity);`

- 4) Tambahkan `setOnClickListener(new View.OnClickListener())` pada variabel *btnPindahActivity*.

- 5) Selanjutnya tambahkan kode berikut pada method `public void onClick(View v) {}` yang ada didalam *setOnClickListener* :


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

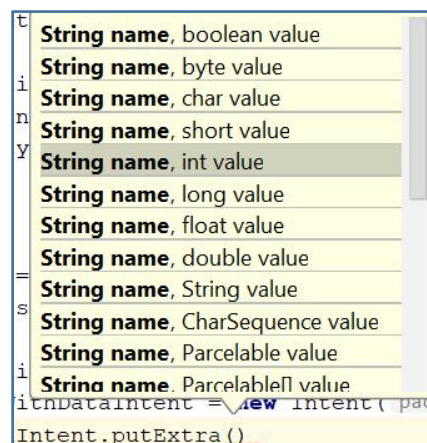
Sehingga kode akan terlihat seperti berikut :

```
btnPindahActivity.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent moveIntent = new Intent(MainActivity.this ,
        ActivityTujuan.class);
        startActivity(moveIntent);
    }
});
```

6) Run Aplikasi , semoga berhasil

b) Contoh Kedua :

Contoh kedua adalah pindah *activity* dengan data, pada contoh ini kita menggunakan *function putExtra()* , sederhananya parameter pada *function* tersebut ada 2 , yang pertama adalah *class* tujuan, dan yang kedua adalah *value* yang ingin dikirimkan,



- 1) Lanjutkan kembali project “ModulEmpat” , tambahkan *activity* baru dengan nama PindahDengkitataActivity.
- 2) Buka *activity_pindah_dengan_data.xml* pada direktori *res* , tambahkan *TextView* , dan berilah id dengan nama *tv_terima_data*.
- 3) Buka kembali *MainActivity.java* , tambahkan kode berikut didalam method *onCreate* :

```
btnPindahDengkitata = findViewById(R.id.btn_pindah_dengan_data);
btnPindahDengkitata.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
```

```

        Intent moveWithDataIntent = new Intent(MainActivity.this ,
PindahDengkitataActivity.class);
        moveWithDataIntent.putExtra("umur",20);
        moveWithDataIntent.putExtra("nama", "Bayu Saputra");
        startActivity(moveWithDataIntent);
    }
});

```

Pada *function putExtra* , parameter pertama berguna sebagai *key* sehingga pada *class* tujuan kita dapat menerima kembali nilai yang dikirimkan tersebut menggunakan *key* nya, dan parameter kedua merupakan nilai yang kita kirimkan.

- 4) Buka *class* PindahDengkitataActivity.java , deklarasikan variabel *tvTerimaData* dengan tipe *TextView*, hubungkan ke *class activity_pindah_dengan_data.xml* menggunakan *findViewById()* , tambahkan kode berikut :

```

public class PindahDengkitataActivity extends AppCompatActivity {

    private TextView tvTerimaData;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_pindah_dengan_data);

        tvTerimaData = findViewById(R.id.tv_terima_data);
        String nama = getIntent().getStringExtra("nama");
        int umur = getIntent().getIntExtra("umur",0);

        String text = "Nama kita "+nama+", Umur kita "+umur;
        tvTerimaData.setText(text);
    }
}

```

Jika nilai yang dikirimkan adalah *string* kita menggunakan *getIntent().getStringExtra("key");* jika nilai yang dikirimkan *integer* gunakan *getIntent().getIntExtra("key",0);* 0 sebagai *default value* jika terjadi kegagalan dalam pengiriman.

- 5) Klik Run, maka output akan seperti berikut :



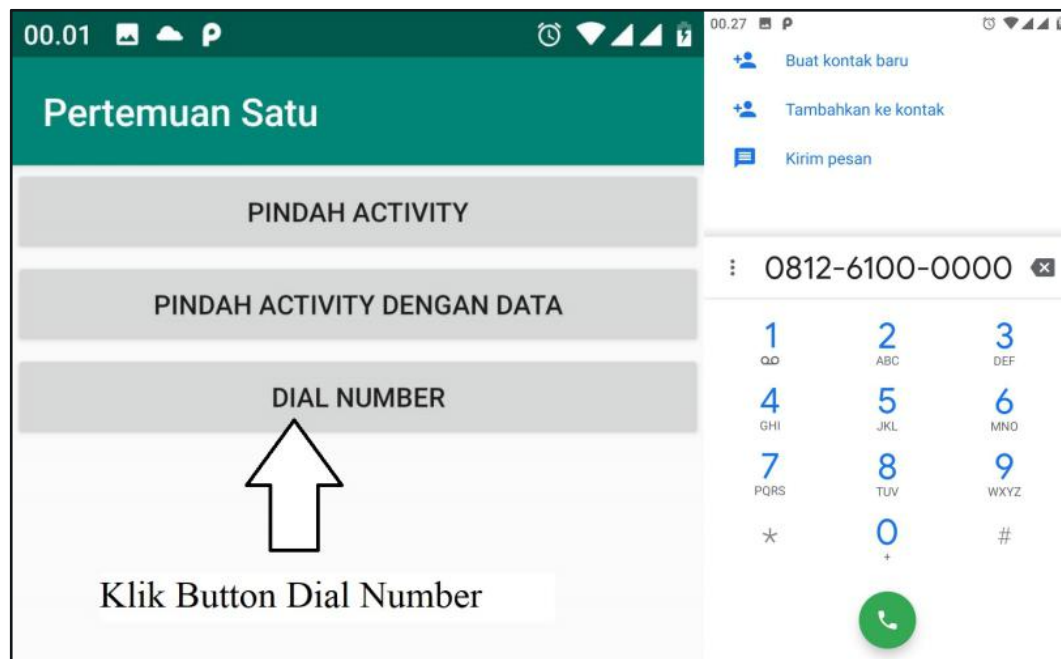
c) Contoh Ketiga :

Pada contoh ketiga kita akan memanfaatkan fungsi internal android menggunakan *intent eksplisit*.

- 1) Buka class MainActivity.java dan tuliskan kode berikut didalam method onCreate :

```
btnDialNumber = findViewById(R.id.btn_dial_number);
btnDialNumber.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String phoneNumber= "081261000000";
        Intent moveEksplisit = new Intent(Intent.ACTION_DIAL,
        Uri.parse("tel:"+ phoneNumber));
        startActivity(moveEksplisit);
    }
});
```

- 2) Perintah `new Intent(Intent.ACTION_DIAL, Uri.parse("tel:"+ phoneNumber));` parameter pertama didalam Intent tersebut merupakan perintah yang ingin dilakukan, dituliskan `Intent.ACTION_DIAL` berarti membuka fungsi internal Dial Number yang terdapat didalam android kita, kemudian pada parameter kedua dituliskan `Uri.parse("tel:"+ phoneNumber)` , berarti tuliskan secara otomatis value *phoneNumber* kedalam fungsi Internal Dial Number, sehingga output akan terlihat seperti berikut :



2. RECYCLER VIEW

Recyclerview adalah sebuah library yang digunakan sebagai tempat untuk menampilkan banyak data, mirip seperti ListView, namun RecyclerView lebih banyak memiliki keunggulan dibandingkan dengan ListView, sederhananya adalah RecyclerView digunakan apabila *programmer* ingin menampilkan sebuah data dengan tampilan yang sama tetapi memiliki konten berbeda secara berulang-kali.

Pada contoh kali ini ada tiga class java yang digunakan, yaitu :

1. MainActivity.java
2. FootballModel.java
3. FootballAdapter.java
4. FootballData.java

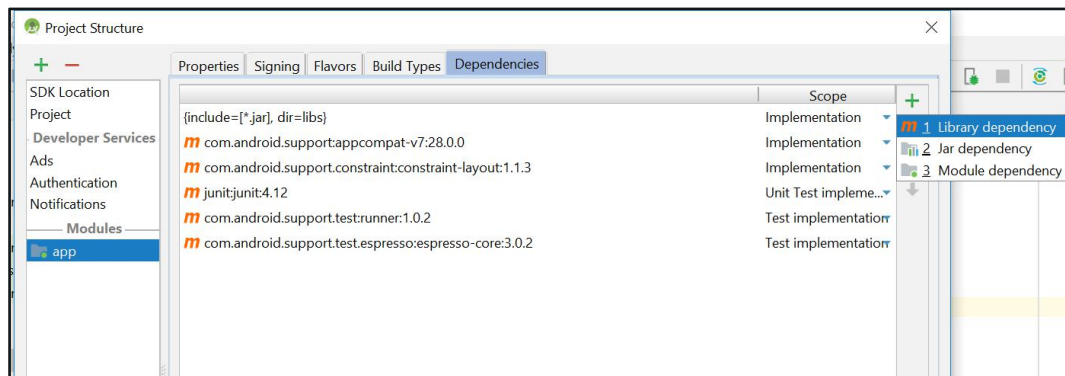
Dan menggunakan dua file xml sebagai layoutnya, yaitu :

1. activity_main.xml
2. item_football.xml

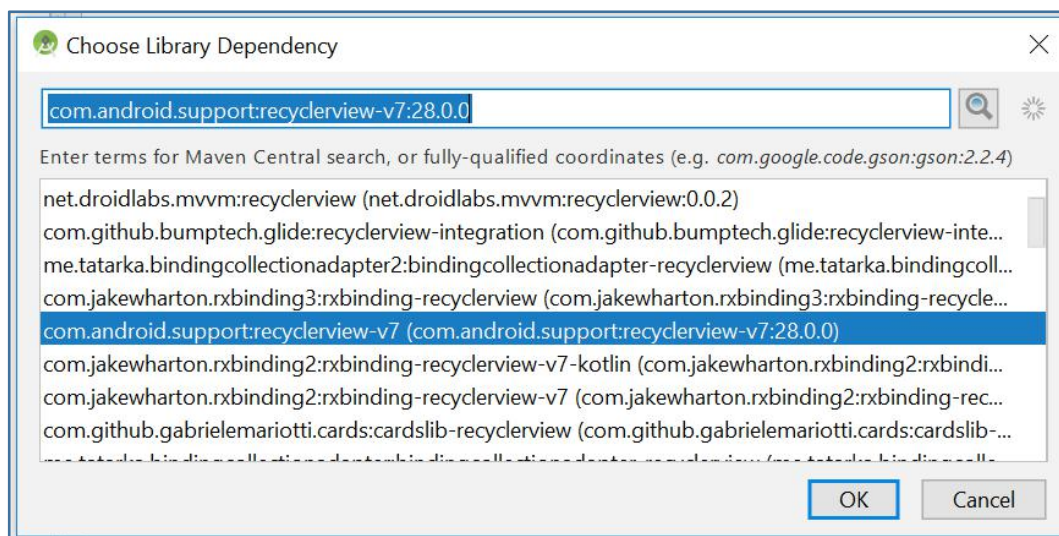
FootballModel.java berfungsi sebagai model data yang berisi method *setter()* dan *getter()* yang digunakan sebagai penyimpanan data sementara saat terjadi proses pengambilan data dari sumbernya.

FootballAdapter.java berfungsi sebagai *class* yang melakukan proses menerima, menampilkan data secara berulang-ulang sebanyak data yang ada, berikut langkah-langkah yang akan dilakukan :

- 1) Buatlah sebuah project baru dengan nama “TimSepakBola” atau sesuai keinginan kita.
- 2) Tambahkan *library recycler view* dengan cara buka File -> Project Structure , kemudian akan tampil sebuah window, pada kiri window pilih app -> Dependeciens -> Tombol + -> Library Dependencies

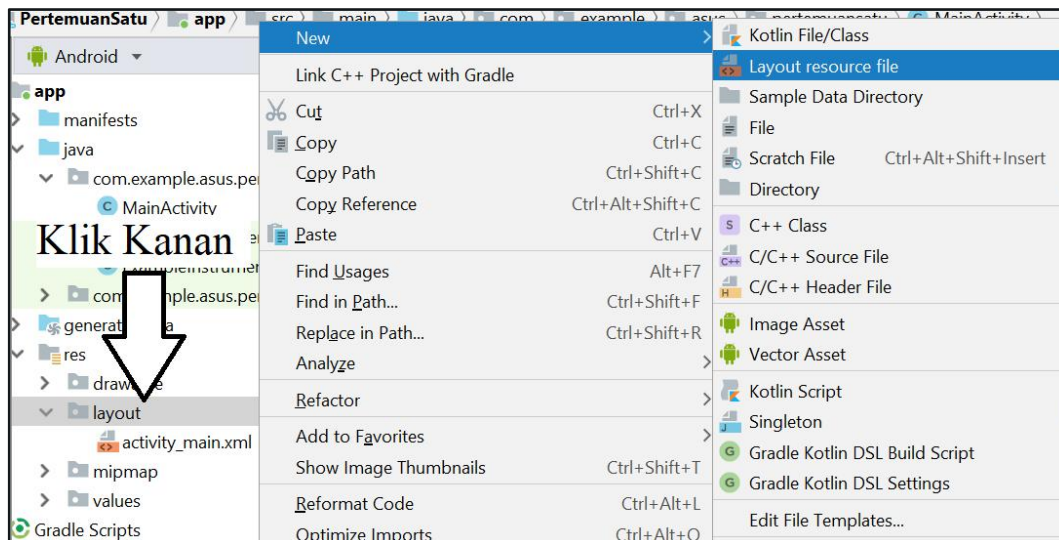


Akan muncul lagi window baru, ketik didalam kolom search recyclerview -> search, pilih com.android.support:recyclerview-v7:28.0.0

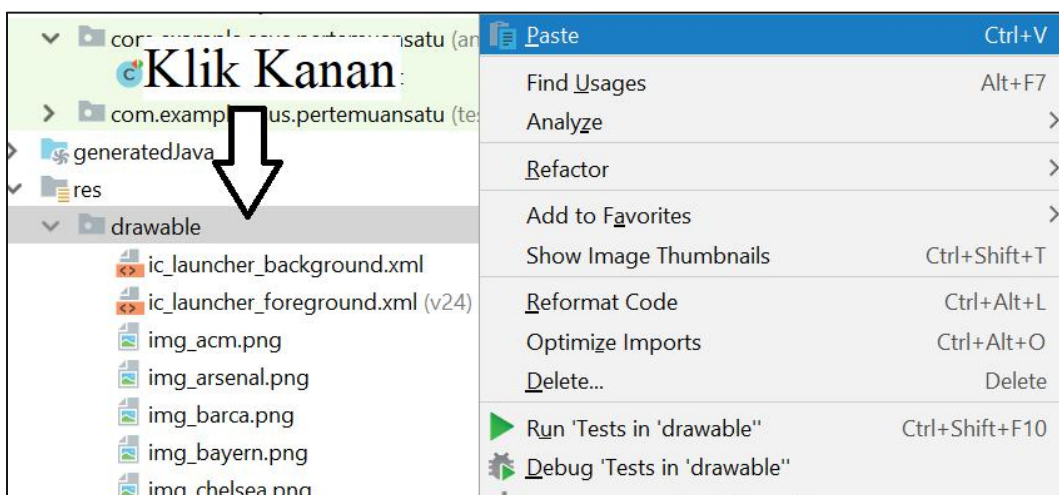


Library recycler view siap digunakan

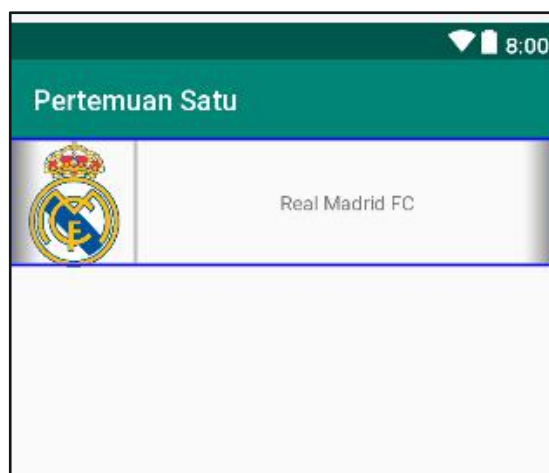
- 3) Klik kanan pada direktori Res -> Layout , dan buatlah sebuah *item_footbal.xml*



- 4) Tambahkan gambar yang kita inginkan kedalam direktori Res -> Drawable, dengan cara blok gambar-gambar yang kita inginkan dan *paste* kan kedalam direktori Res -> Drawable :



- 5) Kemudian buatlah tampilan seperti berikut pada *item_footbal.xml*



Untuk *class item_football.xml* , tinggi dari parent view diset menjadi *wrap content* , seperti berikut : `android:layout_height="wrap_content"` .

- 6) Layout yang kita buat ini akan digunakan sebagai holder terhadap view group yang akan kita tampilkan dalam sebuah *activity* , selanjutnya buka *activity_main.xml* tambahkan tag *RecyclerView* sehingga terlihat seperti berikut :

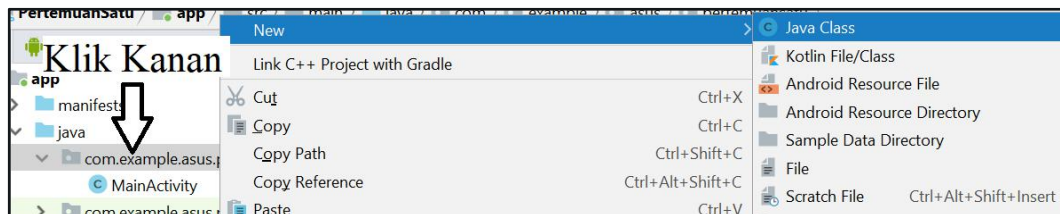
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"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

    <android.support.v7.widget.RecyclerView
        android:id="@+id/rv_team_list"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

    </android.support.v7.widget.RecyclerView>

</LinearLayout>
```

- 7) Klik kanan pada direktori *java* , buatlah sebuah *class* tanpa layout , beri nama *FootballModel.java*



- 8) Didalam *class FootballModel.java* , tambahkan variabel *namaTeam* bertipe *String* dan *lambangTeam* bertipe *int* karena gambar bertipe integer , kemudian lakukan kombinasi *Alt+insert*, pilih menu *Setter and Getter*, blok semua variabel yang ditampilkan, klik *OK*. secara otomatis IDE akan generate method tersebut

```
public class FootballModel {
    private String namaTeam;
    private int lambangTeam;
}

Generate
Constructor
Getter
Setter
Getter and Setter
equals() and hashCode()
toString()
Override Methods... Ctrl+O
Delegate Methods...
Copyright
```

9) Lakukan kembali perintah yang sama seperti poin nomor 7, buatlah sebuah *class* baru bernama *FootballAdapter.java* dan *FootballData.java*

10) Pada *class FootballData.java* berguna untuk kita melakukan load terhadap data-data yang ingin kita tampilkan, tambahkan kode seperti berikut :

```
public class FootballData {  
    private static String[] title = new String[] {"Real Madrid", "AC Milan", "FC Barcelona",  
        "Bayern Munchen", "Chelsea FC", "Manchester United", "Arsenal FC"};  
  
    private static int[] thumbnail = new  
int[] {R.drawable.img_madrid, R.drawable.img_acm,  
R.drawable.img_barca, R.drawable.img_bayern, R.drawable.img_chelsea, R.drawable.  
le.img_mu,  
        R.drawable.img_arsenal};  
  
    public static ArrayList<FootballModel> getListData() {  
        FootballModel footballModel = null;  
        ArrayList<FootballModel> list = new ArrayList<>();  
        for (int i = 0; i < title.length; i++) {  
            footballModel = new FootballModel();  
            footballModel.setLambangTeam(thumbnail[i]);  
            footballModel.setNamaTeam(title[i]);  
            list.add(footballModel);  
        }  
        return list;  
    }  
}
```

11) Selanjutnya buka *class FootballAdapter.java* dan ubah **public class** *FootbalAdapter* menjadi **public class** *FootbalAdapter* **extends** *RecyclerView.Adapter<FootbalAdapter.ViewHolder>* , *class* *FootballAdapter* di extends kan ke *class RecyclerView* sehingga dapat menggunakan berbagai fungsi-fungsi yang terdapat pada *class* tersebut.

12) Akan terjadi error pada *class FootballAdapter.java* , hal tersebut terjadi karena setelah kita melakukan *implements* ke *class ViewHolder* tetapi tidak mengimplementasi *method* yang dimilikinya, letakkan cursor pada bagian yang *error* kemudian lakukan kombinasi Alt+Enter, pilih menu *Implement Methods* untuk mengatasinya



sehingga secara keseluruhan kode akan terlihat seperti berikut :

```
public class FootballAdapter extends
RecyclerView.Adapter<FootballAdapter.ViewHolder> {
    @NonNull
    @Override
    public FootballAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup
viewGroup, int i) {
        return null;
    }

    @Override
    public void onBindViewHolder(@NonNull FootballAdapter.ViewHolder
viewHolder, int i) {

    }

    @Override
    public int getItemCount() {
        return 0;
    }

    public class ViewHolder extends RecyclerView.ViewHolder{
        public ViewHolder(@NonNull View itemView) {
            super(itemView);
        }
    }
}
```

13) Tambahkan kode berikut pada *class FootballAdapter.java* :

```
private Context context;
private ArrayList<FootballModel> footballModels;
```

Kemudian lakukan kembali kombinasi Alt+Enter dan pilih menu *constructor*, dan buatlah sebuah constructor dengan parameter *context*, dan klik OK, lakukan lagi kombinasi Alt+Enter dan pilih menu *Setter and Getter*, pilih *FootballModels* , sehingga kode tertuliskan seperti berikut :

```
private Context context;
private ArrayList<FootballModel> footballModels;

public FootballAdapter(Context context) {
    this.context = context;
}

public ArrayList<FootballModel> getFootballModels() {
    return footballModels;
}

public void setFootballModels(ArrayList<FootballModel> footballModels) {
    this.footballModels = footballModels;
}
```

- 14) Pada `FootballAdapter.ViewHolder` `onCreateViewHolder` , tambahkan kode berikut :

```
@NonNull
@Override
public FootballAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup
viewGroup, int i) {
    View itemRow = LayoutInflater.from(viewGroup.getContext()).
        inflate(R.layout.item_football,viewGroup,false);
    return new ViewHolder(itemRow);
}
```

- 15) Pada method `getItemCount()` ubah menjadi :

```
@Override
public int getItemCount() {
    return getFootballList().size();
}
```

- 16) Kemudian pada *inner class ViewHolder*, ubahlah kodenya hingga menjadi seperti berikut :

```
public class ViewHolder extends RecyclerView.ViewHolder{
    private ImageView ivLambangTeam;
    private TextView tvNamateam;
    public ViewHolder(@NonNull View itemView) {
        super(itemView);
        ivLambangTeam = itemView.findViewById(R.id.lambang_team);
        tvNamateam = itemView.findViewById(R.id.nama_team);
    }
}
```

Inner Class tersebut berguna untuk menghubungkan adapter dengan *item_football.xml* sehingga proses perulangan dapat dilakukan oleh method `onBindViewHolder()`.

- 17) Tambahkan library '**com.github.bumptech.glide:glide:4.8.0**' seperti yang dilakukan pada poin ke 2 , library ini berguna untuk melakukan load image kepada view yang sedang dibangun.
- 18) Tambahkan kode berikut pada `onBindViewHolder()` :

```
@Override
public void onBindViewHolder(@NonNull FootballAdapter.ViewHolder viewHolder,
int i) {

    Glide.with(context).load(getFootballModels().get(i).getLambangTeam()).into
    (viewHolder.ivLambangTeam);
    viewHolder.tvNamateam.setText(getFootballModels().get(i).getNamaTeam());
}
```

`onBindViewHolder` berguna untuk melakukan proses perulangan, sehingga `RecyclerView` tersebut dapat ditampilkan sebanyak data yang ada dengan konten berbeda-beda sesuai dengan data yang telah kita deklarasikan sebelumnya pada *class FootballData.java* dan menampilkannya kedalam view *activity_main.xml*.

- 19) Kemudian buka *class MainActivity.java* , ubahlah keseluruhan kode menjadi seperti berikut :

```
public class MainActivity extends AppCompatActivity {

    private RecyclerView rvTeam;
    private ArrayList<FootballModel> listTeam = new ArrayList<>();

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

        rvTeam = findViewById(R.id.rv_team_list);
        rvTeam.setHasFixedSize(true);
        listTeam.addAll(FootballData.getListData());

        showRecyclerView();
    }

    private void showRecyclerView() {
        rvTeam.setLayoutManager(new LinearLayoutManager(this));
        FootballAdapter footballAdapter = new FootballAdapter(this);
        footballAdapter.setFootballModels(listTeam);
        rvTeam.setAdapter(footballAdapter);
    }
}
```

Terlebih dahulu deklarasikan *variabel rvTeam* bertipe *RecyclerView* dan *variabel listTeam* bertipe *ArrayList<FootballModel>*, *rvTeam* berguna untuk menghubungkan *RecyclerView* yang dituliskan di *activity_main.xml* ke *class MainActivity.java*, dan *variabel listTeam* berguna sebagai objek untuk menyimpan data yang telah kita deklarasikan pada *FootballData.java* kedalam *ArrayList*, sehingga *listTeam* tersebut berisikan data-data yang akan ditampilkan oleh adapter kedalam *recycler view*.

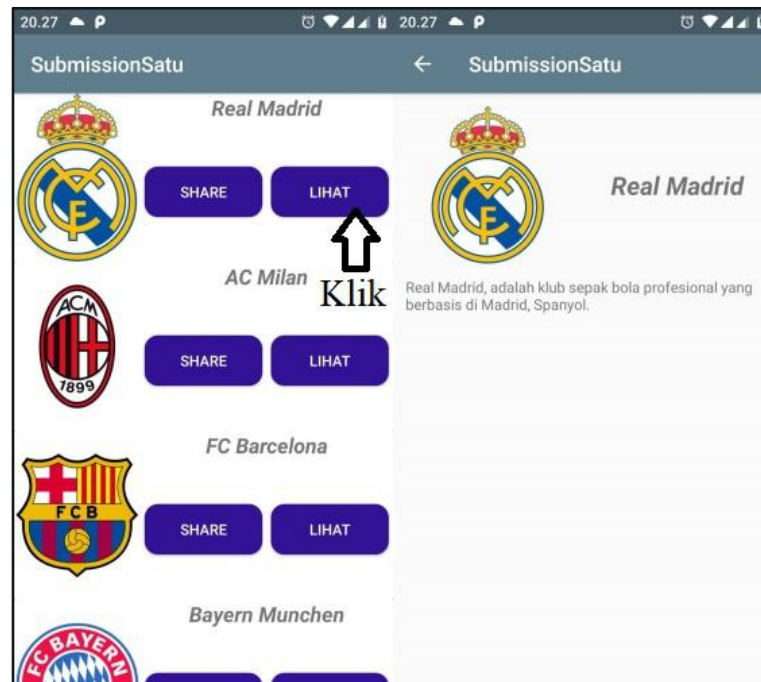
Method *showRecyclerView()* merupakan *method* yang berguna untuk mengatur bagaimana data tersebut ditampilkan (Horizontal, Vertical, Grid, dan sebagainya), pada kasus ini ditampilkan secara *vertical* (Default).

20) Run aplikasi tersebut, maka akan menampilkan output seperti berikut :

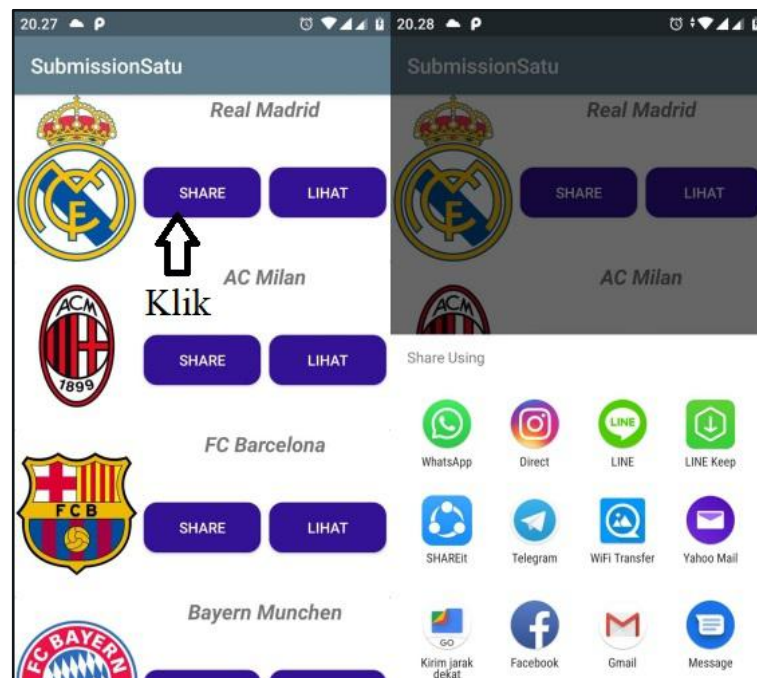


C. TUGAS PRATIKUM

Buatlah Sebuah Aplikasi Yang Mengkombinasikan RecyclerView dan Intent (Eksplisit dan Implisit) dengan semenarik mungkin, contoh seperti berikut :



Output Ketika Button Lihat di Klik



Output Ketika Button Share di Klik

MODUL V

MENGGUNAKAN FITUR BAWAAN ANDROID

A. TUJUAN PRAKTIKUM

- Mengetahui dan memahami cara menggunakan fungsi fungsi dasar di Android

B. DASAR TEORI

Pada materi ini, kita akan mencoba untuk membuat beberapa aplikasi yang menggunakan fitur2 dasar di Android kita.

4. Membuat Panggilan

Yang pertama, kita akan membuat aplikasi Android yang bisa membuat panggilan dari nomor yang kita input. Contoh dari aplikasi ini dapat kalian unduh di https://github.com/anggaaryas/Android_contoh_membuat_panggilan

Yang harus dicermati pertama kali, kita perlu mendapatkan izin untuk menggunakan fitur panggilan. Maka kita tambahkan kode pada `AndroidManifest.xml`

```
<uses-permission android:name="android.permission.CALL_PHONE" />
```

Setelah itu, kita juga perlu menambahkan kode untuk meminta izin secara eksplisit. Ini digunakan untuk Aplikasi yang akan dijalankan di Android **M** (6.0) ke atas. Mungkin kalian tahu di saat kita membuka aplikasi pertama kali, maka akan muncul permohonan izin yang dibutuhkan. Yang mengharuskan kita untuk memilih **allow**. Pada projek ini, kita tambahkan pada `MainActivity.java` .

```
if (ContextCompat.checkSelfPermission(getApplicationContext(), Manifest.permission.CALL_PHONE) == PackageManager.PERMISSION_GRANTED) {  
    Intent intent = new Intent(Intent.ACTION_CALL);  
    intent.setData(Uri.parse("tel:" + numberTelp));  
    startActivity(intent);  
} else {  
    ActivityCompat.requestPermissions(this, new  
        String[]{Manifest.permission.CALL_PHONE},  
        PHONE_REQUEST_CODE);  
}
```

Cara kerja kode tersebut adalah kita cek dahulu apakah kita sudah memiliki izin untuk menggunakan fitur panggilan. Jika belum, maka kita harus membuat izin dahulu.

```
ActivityCompat.requestPermissions(this, new  
    String[]{Manifest.permission.CALL_PHONE},  
    PHONE_REQUEST_CODE);
```

`Manifest.permission.CALL_PHONE` adalah izin yang kita minta dengan kode permintaan `PHONE_REQUEST_CODE`. Kode ini sebenarnya bebas mau kita isi berapa. Penggunaan variabel bertujuan agar memudahkan kita dalam mengingat kode permintaan berapa yang kita pakai.

```
private static final int PHONE_REQUEST_CODE = 986;
```

Dan untuk mengetahui hasil permintaan kita (Apakah diizinkan ataukah tidak, maka kita perlu meng Override fungsi `onRequestPermissionsResult`.

```
@Override  
public void onRequestPermissionsResult(int requestCode, @NonNull String[]  
    permissions, @NonNull int[] grantResults) {  
    if(requestCode == PHONE_REQUEST_CODE){  
        // Aplikasi di izinkan ...  
    } else {  
        Toast.makeText(getApplicationContext(), "Aplikasi tidak  
            diizinkan", Toast.LENGTH_SHORT).show();  
    }  
}
```

Setelah kita di izinkan untuk membuat panggilan maka kita bisa melakukan panggilan. Masih ingatkah kita dengan materi *intent*?

```
Intent intent = new Intent(Intent.ACTION_CALL);  
    intent.setData(Uri.parse("tel:" + numberTelp));  
    startActivity(intent);
```

5. Membuat Browser Sederhana

Selanjutnya, kita akan membuat aplikasi browser sederhana. Diharapkan kita bisa mengakses url yang kita inputkan dan di tampilkan dalam aplikasi kita. Contoh project aplikasi ini dapat kalian unduh di https://github.com/anggaaryas/Android_contoh_membuat_browser

Jika project sebelumnya kita memerlukan izin panggilan, maka pada aplikasi ini kita memerlukan izin untuk mengakses internet

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Setelah menambahkan kode pada `AndroidManifest.xml` , kita juga perlu menambahkan kode perizinan secara eksplisit pada aplikasi kita.

```
if (ContextCompat.checkSelfPermission(getApplicationContext(),
    Manifest.permission.INTERNET) ==
    PackageManager.PERMISSION_GRANTED) {

    webView.loadUrl("http://" + url.getText().toString());

} else {

    ActivityCompat.requestPermissions(this, new
        String[]{Manifest.permission.CALL_PHONE},
        INTERNET_REQUEST_CODE);

}
```

Setelah itu, kita tambahkan juga kode untuk mengecek hasil permintaan izin kita. Jika kita diizinkan, maka kita bisa mengambil halaman web dari url yang kita input.

```
@Override

public void onRequestPermissionsResult(int requestCode, @NonNull String[]
    permissions, @NonNull int[] grantResults) {

    if (requestCode == INTERNET_REQUEST_CODE) {

        aksesURL();

    } else {

        Toast.makeText(getApplicationContext(), "Aplikasi tidak
            diizinkan", Toast.LENGTH_SHORT).show();

    }

}
```


Sebelumnya kita Setting dahulu beberapa peraturan yang kita butuhkan pada `WebView` . `WebView` adalah komponen yang kita butuhkan untuk menampilkan halaman web.

```
private WebView webView;

webView = findViewById(R.id.wv_main);

webView.getSettings().setJavaScriptEnabled(true);

webView.setWebViewClient(new WebClient(this));
```

Pada Bab sebelumnya, kita tahu bahwa fungsi `findViewById` digunakan untuk menghubungkan variabel komponen di bahasa java dengan komponen yang berada di layout.

```
webView.getSettings().setJavaScriptEnabled(true);
```

Kode di atas di gunakan untuk mengaktifkan javascript pada webview kita. Kadang kita menemui halaman web yang tidak termuat semuanya karena tidak mendukung javascript.

```
webView.setWebViewClient(new WebClient(this));
```

Secara bawaan, ketika pengguna memilih link yang berada di halaman web, maka akan otomatis membuka browser bawaan smartphone kita. Agar kita tetap berada di aplikasi kita , maka kita perlu membuat custom kelas `WebViewClient` dan meng `Override` fungsi `shouldOverrideUrlLoading`. Pada kelas `WebClient.java` kita tambahkan kode berikut

```
public class WebClient extends WebViewClient {

    private Activity activity;

    public WebClient(Activity activity) {

        this.activity = activity;

    }

    @Override
```

```
public boolean shouldOverrideUrlLoading(WebView view, String url) {  
    return false;  
}  
}
```

Lalu kita set WebClient nya

```
webView.setWebViewClient(new WebClient(this));
```

Setelah itu, kita dapat memuat halaman yang kita inputkan dengan kode ini.
Jalankan ketika aplikasi kita telah di izinkan

```
webView.loadUrl("http://" + url.getText().toString());
```

Namun, disaat kita menekan tombol kembali, kita malah keluar dari aplikasi yang seharusnya aplikasi kita dapat kembali ke halaman web sebelumnya. Untuk memperbaiki ini kita perlu meng Override fungsi onKeyDown.

```
@Override  
public boolean onKeyDown(int keyCode, KeyEvent event) {  
    if((keyCode == KeyEvent.KEYCODE_BACK) && webView.canGoBack()){  
        webView.goBack();  
        return true;  
    }  
  
    return super.onKeyDown(keyCode, event);  
}
```

Cara kerja kode tersebut adalah kita menekan tombol, aplikasi akan mengecek apakah tombol yang kita pencet merupakan tombol kembali ataukah bukan. Jika iya, kita cek juga apakah kita bisa kembali ke halaman sebelumnya / adakah laman sebelumnya yang kita muat. Jika iya, maka jalankan kode

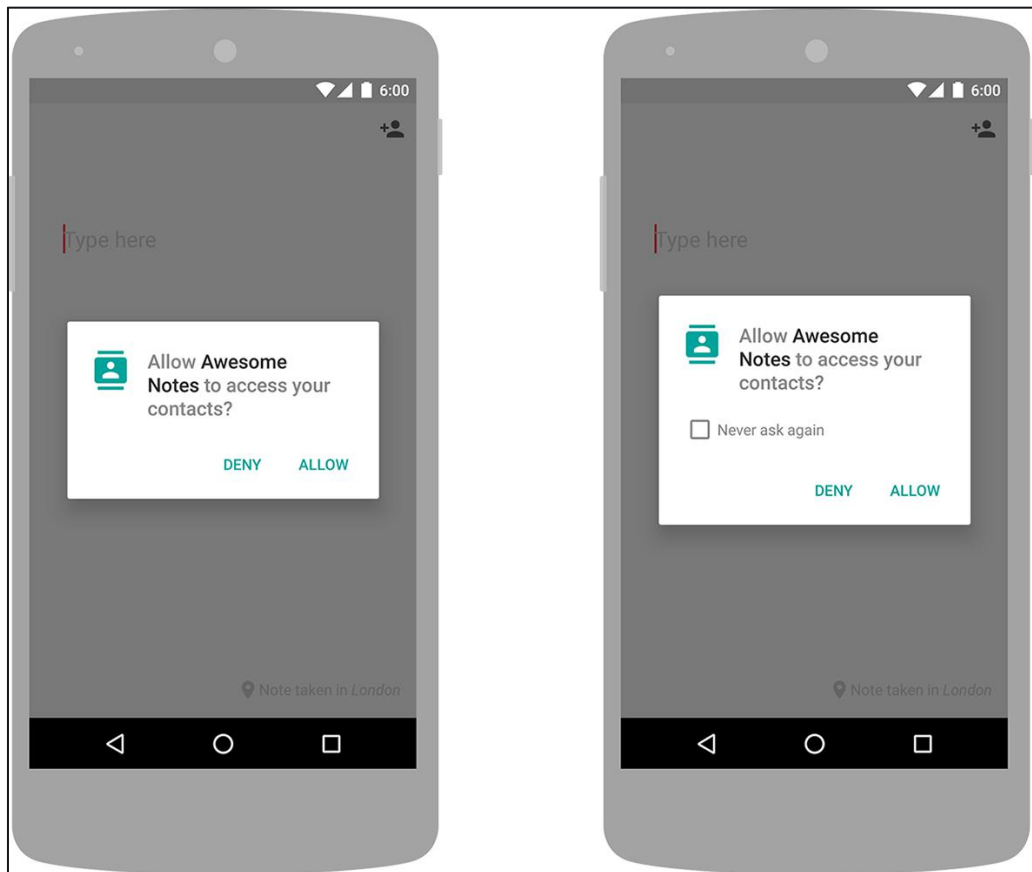
```
webView.goBack();
```

6. Lain Lain

Selain dapat membuat panggilan telepon dan membuat browser sederhana, masih banyak lagi yang dapat kita lakukan. Berikut ini adalah beberapa perizinan yang dapat kita lakukan

- a. ACCESS_LOCATION_EXTRA_COMMANDS
- b. ACCESS_NETWORK_STATE
- c. ACCESS_NOTIFICATION_POLICY
- d. ACCESS_WIFI_STATE
- e. BLUETOOTH
- f. BLUETOOTH_ADMIN
- g. BROADCAST_STICKY
- h. CHANGE_NETWORK_STATE
- i. CHANGE_WIFI_MULTICAST_STATE
- j. CHANGE_WIFI_STATE

Dan masih banyak lagi...



Contoh tampilan ketika meminta perizinan (Untuk Android 6.0 ke atas)

C. TUGAS PRATIKUM

Buatlah Aplikasi tebak tebakan matematika minimal berisi 5 Pertanyaan. Tiap pertanyaan harus 1 layout per pertanyaan, jadi kita akan mempunyai 5 layout soal. Jika jawaban salah, getarkan handphone. Jika benar, maka lanjut ke soal berikutnya. Aplikasi harus Landscape / Mendatar

$2 \times 5 = ?$

JAWAB

MODUL VI

DATABASE LOKAL

D. TUJUAN PRAKTIKUM

- Mengetahui dan memahami bagaimana cara mengakses database lokal di Android

E. DASAR TEORI

Kali ini kita akan belajar bagaimana cara mengakses database lokal di Android. Materi ini meliputi bagaimana cara membuat database, membaca database, menambahkan data ke database, mengupdate data di database, dan menghapus data pada database. Kita akan menggunakan library "Room Persistence Library". Contoh program Android pada bab ini dapat diunduh di https://github.com/anggaaryas/android_room_example

7. Room Persistence Library

Library ini merupakan bagian dari *Android Jetpack*. Jetpack adalah kumpulan komponen software Android yang memudahkan kita dalam mengembangkan aplikasi Android yang hebat. Komponen-komponen ini membantu kita mengikuti praktik terbaik, membebaskan kita dari kode boilerplate, dan menyederhanakan tugas yang kompleks, sehingga kita dapat fokus pada kode yang kita minati.

Untuk menggunakannya, kita perlu menambahkan komponennya di `build.gradle` (yang App). dan tambahkan kode berikut pada bagian dependencies.

```
def room_version = "1.1.1"

Implementation
"android.arch.persistence.room:runtime:$room_version"

annotationProcessor
"android.arch.persistence.room:compiler:$room_version"
```

Terdapat 3 komponen utama pada Room:

1. Database : kelas ini dianotasikan **@Database** dan harus memenuhi beberapa kondisi, yaitu:

- * Harus bersifat abstract class dan extends RoomDatabase .
- * Memuat entity yang berhubungan dengan database tersebut di dalam anotasi
- * mempunyai method yang mengembalikan kelas yang beranotasi

@Dao

2. Entity: Merepresentasikan tabel pada database

3. Dao : Berisi method untuk mengakses database. Dao = Data Access Object

Di bawah ini adalah contoh kode yang berisi 1 entity dan 1 Dao:

A. DataDiri

```
@Entity(tableName = "user_db")
public class DataDiri {

    @NonNull

    @PrimaryKey(autoGenerate = true)
    @ColumnInfo(name = "id")
    private int id ;

    @ColumnInfo(name = "name")
    private String name;

    @ColumnInfo(name = "adress")
    private String adress;

    @ColumnInfo(name = "gender")
    private char gender;

    @NonNull

    public int getId() {
        return id;
    }
}
```

```
public void setId(@NonNull int id) {  
    this.id = id;  
}  
  
public String getName() {  
    return name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public String getAddress() {  
    return address;  
}  
  
public void setAddress(String address) {  
    this.address = address;  
}  
  
public char getGender() {  
    return gender;  
}  
  
public void setGender(char gender) {  
    this.gender = gender;  
}  
}
```

B. DataDiriDAO

```
@Dao
public interface DataDiriDAO {

    @Insert
    Long insertData(DataDiri dataDiri);

    @Query("Select * from user_db")
    DataDiri[] getData();

    @Update
    int updateData(DataDiri item);

    @Delete
    void deleteData(DataDiri item);
}
```

C. AppDatabase

```
@Database(entities = {DataDiri.class} , version = 1)
public abstract class AppDatabase extends RoomDatabase {

    public abstract DataDiriDAO dao();

    private static AppDatabase appDatabase;

    public static AppDatabase iniDb(Context context){
        if(appDatabase == null)
            appDatabase = Room.databaseBuilder(context, AppDatabase.class,
                                                "dbUser").allowMainThreadQueries().build();

        return appDatabase;
    }
}
```



```

public static void destroyInstance() {
    appDatabase = null;
}
}

```

8. Inisialisasi Database

Untuk menginisialisasi database, cukup menggunakan kode ini:

```
appDatabase = AppDatabase.initDb(getApplicationContext());
```

kita bisa menaruh kode tersebut pada method onCreate suatu Activity

```

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

    appDatabase = AppDatabase.initDb(getApplicationContext());

    btnOK = findViewById(R.id.btn_submit);
    btnOK.setOnClickListener(this);
    ....
}

```

Oh ya, jangan lupa untuk menginisialisasi variabel appDatabase dahulu

```
AppDatabase appDatabase;
```

9. Read Database

Berikut ini adalah contoh method untuk membaca data suatu database.

```

public void readData(AppDatabase database) {
    List list;
}

```

```

        list = database.dao().getData();           // Untuk read database
        getData(list);                             // Untuk menampilkan
    }

```

Lalu bisa dipanggil dengan cara seperti ini

```
readData(appDatabase);
```

10. Update Database

```

public void editData(String nama, String alamat, char gender, int id,
                    final AppDatabase database) {
    final DataDiri dataDiri = new DataDiri();
    dataDiri.setAdress(alamat);
    dataDiri.setGender(gender);
    dataDiri.setName(nama);
    dataDiri.setId(id);

    new EditData(database, dataDiri).execute(); // Update Database
}

```

Lalu pada kelas EditData. AsyncTask adalah Sebuah class yang disediakan Android untuk proses/operasi pengambilan/pengiriman yang dilakukan secara background.

```

class EditData extends AsyncTask<Void, Void, Integer> {

    private AppDatabase database;

    private DataDiri dataDiri;

    public EditData(AppDatabase database, DataDiri dataDiri) {

        this.database = database;

        this.dataDiri = dataDiri;

    }
}

```

```

        @Override

        protected Integer doInBackground(Void... voids) {

            return database.dao().updateData(dataDiri);

        }

        @Override

        protected void onPostExecute(Integer integer) {

            super.onPostExecute(integer);

            Log.d("integer db", "onPostExecute: " + integer);

            view.successAdd();

        }

    }
}

```

11. Insert Database

```

public void insertData(String nama, String alamat, char gender,
                        final AppDatabase database) {

    final DataDiri dataDiri = new DataDiri();

    dataDiri.setAdress(alamat);

    dataDiri.setGender(gender);

    dataDiri.setName(nama);

    new InsertData(database, dataDiri).execute(); // Insert Db

}

```

Lalu pada kelas InsertData

```

class InsertData extends AsyncTask<Void, Void, Long>{

    private AppDatabase database;

    private DataDiri dataDiri;

```

```

public InsertData(AppDatabase database, DataDiri dataDiri) {

    this.database = database;

    this.dataDiri = dataDiri;

}

@Override

protected Long doInBackground(Void... voids) {

    return database.dao().insertData(dataDiri);

}

@Override

protected void onPostExecute(Long aLong) {

    super.onPostExecute(aLong);

    view.successAdd();

}

}

```

12. Delete Database

```

public void deleteData(final DataDiri dataDiri,

                        final AppDatabase database) {

    new DeleteData(database, dataDiri).execute();

}

```

Lalu pada kelas DeleteData

```

class DeleteData extends AsyncTask<Void, Void, Void>{

    private AppDatabase database;

    private DataDiri dataDiri;

```

```
public DeleteData(AppDatabase database, DataDiri dataDiri) {  
    this.database = database;  
    this.dataDiri = dataDiri;  
}  
  
@Override  
protected Void doInBackground(Void... voids) {  
    database.dao().deleteData(dataDiri);  
    return null;  
}  
  
@Override  
protected void onPostExecute(Void aVoid) {  
    super.onPostExecute(aVoid);  
}
```

C. TUGAS PRATIKUM

1. Buatlah Aplikasi Android untuk menyimpan kontak temanmu. Data hanya berisi nama dan alamat. Minimal terdapat fitur tambah, edit, baca, dan hapus data! Untuk layout se kreatif kalian.
2. Pelajari lagi kegunaan AsyncTask pada Android dan berikan contoh lain penggunaan AsyncTask!

MODUL VII

MEMANFAATKAN *WEBSERVICE*

A. TUJUAN PRAKTIKUM

- Mengetahui dan memahami bagaimana cara mengakses *webservice* di Android

B. DASAR TEORI

1. Retrofit

Retrofit adalah type-safe HTTP client untuk Android dan Java yang dibuat oleh orang-orang hebat di Square. Retrofit memudahkan untuk mengonsumsi data JSON atau XML yang diuraikan menjadi Plain Old Java Objects (POJOs).

Keuntungan dari retrofit adalah :

1. Mudah tersambung ke *webservice* dengan menerjemahkan API ke Java
2. Mudah untuk menambahkan *Header* dan Request
3. Mudah untuk digunakan

Untuk menggunakannya, kita perlu menambahkan komponennya di build.gradle (App). dan tambahkan kode berikut pada bagian dependencies.

```
implementation 'com.squareup.retrofit2:retrofit:2.4.0'  
implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
```

Selanjutnya tambahkan permission internet di manifest :

```
<uses-permission android:name="android.permission.INTERNET"/>
```

2. OkHttp-Logging

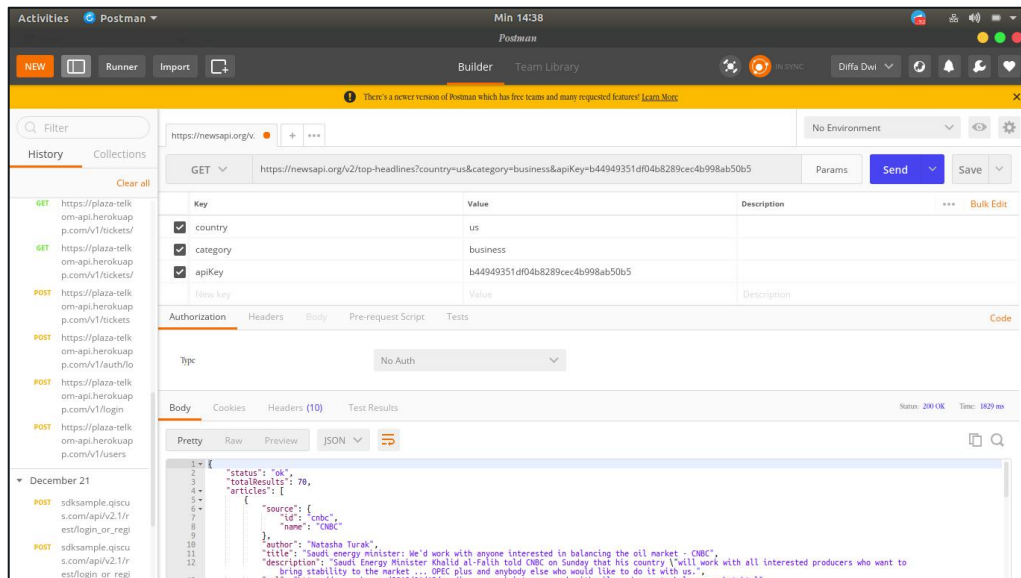
Terkadang saat bekerja dengan Retrofit dan memarsing data JSON mungkin akan mendapatkan beberapa kesalahan-kesalahan aneh. kita mempunyai dua opsi untuk menemukan penyebab masalah ini. Salah satunya menggunakan Postman untuk membuat permintaan dan memeriksa respons atau cara lain adalah dengan menggunakan **Okhttp logging interceptor** ke Retrofit dan memeriksa Log.

Buka build.gradle aplikasi kita dan tambahkan lagi dependencies berikut bersama dengan dependencies Retrofit lainnya. Kemudian sinkronkan proyek.

```
Implementation 'com.squareup.okhttp3:logging-interceptor:3.10.0'  
implementation 'com.squareup.okhttp3:okhttp:3.10.0'
```

3. Postman

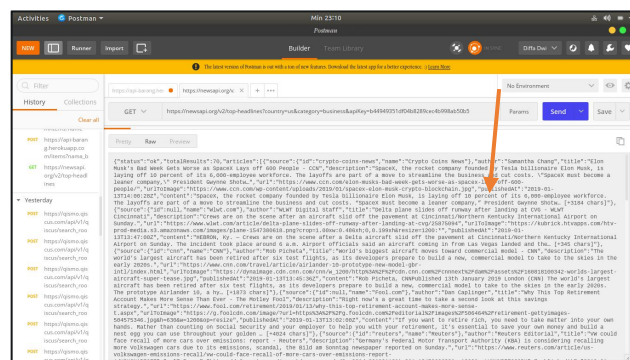
Fungsi utama postman ini adalah sebagai GUI API Caller namun sekarang postman juga menyediakan fitur lain yaitu Sharing Collection API for Documentation (free), Testing API (free), Realtime Collaboration Team (paid), Monitoring API (paid), Integration (paid). Untuk menggunakan Postman laptop atau komputer kita harus terinstal postman terlebih dahulu. Kemudian buka aplikasi postman tersebut :



1. Tambahkan URL <https://newsapi.org/> dan pilih metode GET :
2. Klik Params untuk menambahkan parameter. Key sebagai kata kunci dan Value sebagai nilai dari Key tersebut, tambahkan parameter yang dibutuhkan.

No	Key	Value
1	country	us
2	category	business
3	apiKey	b44949351df04b8289cec4b998ab50b5

3. Klik Send maka response dari Url tersebut adalah seperti berikut :



4. Design Pattern MVP (Model-View-Presenter)

Model-View-Presenter atau yang biasa disingkat menjadi **MVP** adalah sebuah konsep arsitektur pengembangan aplikasi yang memisahkan antara tampilan aplikasi dengan proses bisnis yang bekerja pada aplikasi. Arsitektur ini akan membuat pengembangan aplikasi kita menjadi lebih terstruktur, mudah di-*test* dan juga mudah di-*maintain*.

Berikut penjelasan masing-masing layer pada **MVP**

1. **View**, merupakan layer untuk menampilkan data dan interaksi ke *user*. View biasanya berupa *Activity*, *Fragment* atau *Dialog* di Android. View ini juga yang langsung berkomunikasi dengan *user*.
2. **Model**, merupakan layer yang menunjuk kepada objek dan data yang ada pada aplikasi.
3. **Presenter**, merupakan layer yang menghubungkan komunikasi antara Model dan View. Setiap interaksi yang dilakukan oleh *user* akan memanggil Presenter untuk memrosesnya dan mengakses Model lalu mengembalikan responnya kembali kepada View.

Latihan. Membuat aplikasi menggunakan Retrofit dan OkHttp

Kali ini kita akan belajar bagaimana cara mengakses *webservice* di Android. Materi ini meliputi bagaimana cara pembuatan class Plain Old Java Objects (POJOs) dan Mendapatkan data dari *webservice*. Dalam modul ini kita akan memanfaatkan library “Retrofit dan Okhttp” dan juga Design Pattern MVP. Contoh program ini dapat diunduh di <https://github.com/ddiffa/retrofit.git>

1. Tambahkan Gradle yang diperlukan dan sinkronkan

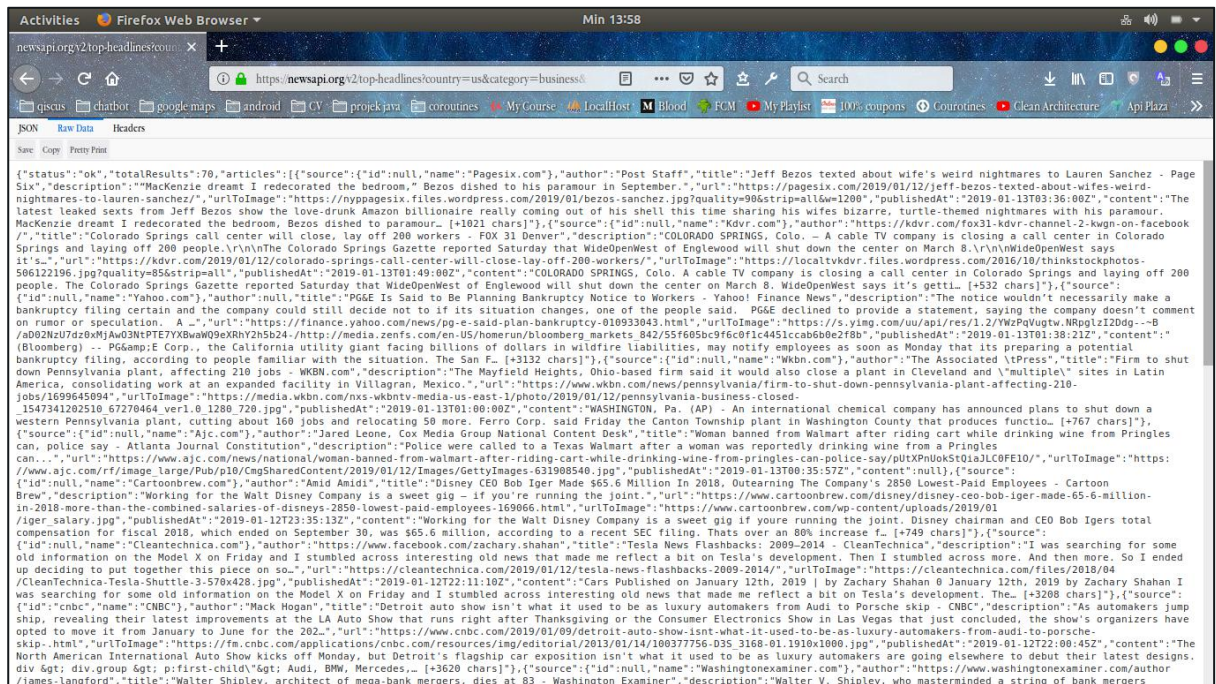
```
implementation 'com.android.support:recyclerview-v7:27.1.0'
implementation 'com.squareup.okhttp3:okhttp:3.10.0'
implementation 'com.squareup.okhttp3:logging-interceptor:3.10.0'
implementation 'com.android.support:cardview-v7:25.3.1'
implementation 'com.squareup.retrofit2:retrofit:2.4.0'
implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
implementation 'com.squareup.picasso:picasso:2.71828'
```


2. Tampilkan permission internet di manifest

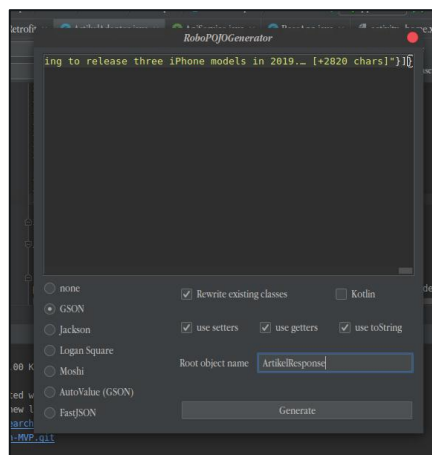
```
<uses-permission android:name="android.permission.INTERNET"/>
```

3. Kunjungi URL :

<https://newsapi.org/v2/top-headlines?country=us&category=business&apiKey=b44949351df04b8289cec4b998ab50b5>. Kemudian pilih raw data seperti gambar dibawah ini dan Copy semua rawdata tersebut. :



4. Buatlah Package Model, kemudian klik kanan pada package tersebut dan pilih new -> Generate POJO from JSON dan copy-paste kan raw data yang telah diambil dari URL tadi seperti berikut, kemudian pilih GSON dan ubah nama menjadi ArtikelResponse, klik generate.



Maka kita akan mendapatkan 3 kelas yaitu : ArtikelResponse, ArticlesItem dan Source

4. Buatlah Activity baru bernama HomeActivity dan ubah XML nya menjadi seperti berikut :

```
<?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">
    <android.support.v7.widget.RecyclerView
        android:id="@+id/rec_artikel"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:listitem="@layout/item_artikel" />
</LinearLayout>
```

6. Setelah membuat activity selanjutnya buatlah layout baru dengan cara klik kanan pada layout->new->Layout Resource File. Beri nama item_artikel dan ubah menjadi seperti berikut :

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    android:elevation="@dimen/cardview_default_elevation"
    app:cardCornerRadius="8dp">
    <RelativeLayout
        android:id="@+id/openEdukasi"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
```

```

<ImageView
    android:id="@+id/img_artikel"
    android:layout_width="match_parent"
    android:layout_height="180dp"
    android:scaleType="centerCrop"
    tools:src="@mipmap/ic_launcher" />
<TextView
    android:id="@+id/tv_judul_artikel"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_above="@+id/tv_tanggal"
    android:layout_margin="8dp"
    android:maxLines="2"
    android:textColor="@color/white"
    android:textSize="24sp"
    android:textStyle="bold"
    tools:text="Judul Artikel" />
<TextView
    android:id="@+id/tv_tanggal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignBottom="@+id/img_artikel"
    android:layout_margin="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:textColor="@color/white"
    android:textSize="14sp"
    tools:text="Tanggal" />
</RelativeLayout>
</android.support.v7.widget.CardView>

```

- 7. Buatlah Package Adapter dan buat class adapter dengan nama ArtikelAdapter. Kemudian tambahkan codingan seperti berikut :**

```

public class ArtikelAdapter extends
RecyclerView.Adapter<ArtikelAdapter.Holder> {
    private List<Artikel> artikelList;
    private Context context;

```

```

public ArtikelAdapter(Context context, List<Artikel> artikelList) {
    this.context = context;
    this.artikelList = artikelList;
}

@NonNull
@Override
public Holder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)
{
    View view = LayoutInflater.from(parent.getContext())
        .inflate(R.layout.item_artikel, parent, false);
    return new Holder(view);
}

@Override
public void onBindViewHolder(ArtikelAdapter.Holder holder, final int
position) {
    holder.bind(position);
}

@Override
public int getItemCount() {
    return artikelList.size();
}

class Holder extends RecyclerView.ViewHolder {
    private RelativeLayout openEdukasi;
    private ImageView imgArtikel;
    private TextView tvJudul;
    private TextView tvTanggal;
    public Holder(View itemView) {
        super(itemView);
        imgArtikel = itemView.findViewById(R.id.img_artikel);
        tvJudul = itemView.findViewById(R.id.tv_judul_artikel);
        tvTanggal = itemView.findViewById(R.id.tv_tanggal);
        openEdukasi = itemView.findViewById(R.id.openEdukasi);
    }
    public void bind(final int position) {
        Picasso.get()
            .load(artikelList.get(position).getUrlToImage())
            .placeholder(R.mipmap.ic_launcher)

```

```

        .error(R.mipmap.ic_launcher)
        .into(imgArtikel);

tvJudul.setText(artikelList.get(position).getTitle());
tvTanggal.setText(artikelList.get(position).getPublishedAt());
openEdukasi.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(context, "clicked : " +
tvJudul.getText().toString() + "\n Position : " + String.valueOf(position),
Toast.LENGTH_LONG).show();
    }
});
}
}
}
}

```

Class adapter akan handle 4 bagian yaitu :

1.Item

2.Binding data

3.Banyak Data

4.ViewHolder

8. Buatlah Package Network dan Buat class BaseApp dan interface ApiService seperti berikut:

```

public class BaseApp extends Application {
    public static ApiService service;
    private String url = "https://newsapi.org/";
    @Override
    public void onCreate() {
        super.onCreate();
        service = getRetrofit().create(ApiService.class);
    }
    private Retrofit getRetrofit() {
        return new Retrofit.Builder()
            .baseUrl(url)
            .addConverterFactory(GsonConverterFactory.create())
            .client(getHttpClient())

```

```

        .build();
    }

    private OkHttpClient getHttpClient() {
        return new OkHttpClient.Builder()
            .addInterceptor(getHttpLogInterceptor())
            .build();
    }

    private Interceptor getHttpLogInterceptor() {
        HttpLoggingInterceptor loggingInterceptor = new
        HttpLoggingInterceptor();

        HttpLoggingInterceptor.Level level;
        if (BuildConfig.DEBUG) {
            level = HttpLoggingInterceptor.Level.BODY;
        } else {
            level = HttpLoggingInterceptor.Level.NONE;
        }

        loggingInterceptor.setLevel(level);
        return loggingInterceptor;
    }
}

```

Level logging bertanggung jawab atas informasi yang ingin kita cetak di log kita. BODY mencetak semuanya. HEADERS mencetak headers response tidak termasuk response body.

```

public interface ApiService {
    @GET("v2/top-headlines")
    Call<ArtikelResponse> getArtikel(@Query("country") String country,
                                     @Query("category") String category,
                                     @Query("apiKey") String apiKey);
}

```

class ApiService merupakan tempat untuk membuat endpoint dari API.

1. Get => artinya kita akan menggunakan method GET untuk mengambil data.

2. @Query => menentukan nama kunci query dengan nilai parameter berannotasi.

Argument lain yang biasa digunakan :

1. @Path - substitusi variabel untuk titik akhir API. Misalnya id film akan ditukar dengan {id} di titik akhir URL.

2. @Query - menentukan nama kunci query dengan nilai parameter berannotasi.

3. **@Body** - payload untuk panggilan POST
4. **@Header** - menentukan header dengan nilai dari parameternya
9. Buat Package baru dan berinama presenter, kemudian buat interface **HomeInterface** dan tambahkan codingan dibawah ini :

```
public interface HomeInterface {  
    void loadArtikel(String country, String category, String apiKey);  
}
```

Class ini akan digunakan dalam pemanggilan presenter di **HomeActivity**.

10. Buatlah class interface baru di package presenter dengan nama **HomeView** dan tambahkan codingan dibawah ini :

```
public interface HomeView {  
    void onSuccess(List<Artikel> artikelList);  
    void onError(String errorMessage);  
    void onFailure(String failureMessage);  
}
```

Class ini bertugas sebagai media komunikasi antara presenter dengan view.

11. Buatlah class **HomePresenter** yang implements ke **HomeInterface** dan tambahkan codingan seperti dibawah ini :

```
public class HomePresenter implements HomeInterface {  
    private Context context;  
    private HomeView homeView;  
    public HomePresenter(Context context, HomeView homeView) {  
        this.context = context;  
        this.homeView = homeView;  
    }  
    @Override  
    public void loadArtikel(String country, String category, String apiKey) {  
        BaseApp.service.getArtikel(country, category, apiKey).enqueue(new  
            Callback<ArtikelResponse>() {  
                @Override  
                public void onResponse(Call<ArtikelResponse> call,  
                    Response<ArtikelResponse> response) {  
                    if (response.isSuccessful()) {
```

```

        homeView.onSucces(response.body().getArticles());
    } else {
        homeView.onError(response.message());
    }
}

@Override
public void onFailure(Call<ArtikelResponse> call, Throwable t) {
    homeView.onFailure(t.getMessage());
}
});
}
}

```

Class presenter bertanggung jawab untuk mendapatkan data dari API, kemudian jika request berhasil maka response akan masuk ke onResponse. Jika dalam onResponse berhasil maka presenter akan mengirim data ke fungsi onSucces melalui interface HomeView, jika response tidak berhasil maka akan mengirim data berupa message ke onError melalui interface HomeView.

Selain itu, jika terjadi kesalahan dalam request maka akan masuk ke onFailure yang kemudian onFailure akan mengirim message ke fungsi onFailure melalui interface HomeView.

12.Terkahir Buka HomeActivity yang meng-implements HomeView dan ubah codingan menjadi seperti dibawah ini :

```

public class HomeActivity extends AppCompatActivity implements HomeView {
    private RecyclerView recyclerViewArtikel;
    private ArtikelAdapter artikelAdapter;
    private List<Artikel> artikelList;
    private LinearLayoutManager layoutManager;
    private HomePresenter homePresenter;
    private String API_KEY = "b44949351df04b8289cec4b998ab50b5";
    private String category = "business";
    private String country = "us";

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

```



```

    }

    private void initView() {
        artikelList = new ArrayList<>();
        artikelAdapter = new ArtikelAdapter(getApplicationContext(),
artikelList);

        linearLayoutManager = new LinearLayoutManager(this);
        recyclerViewArtikel = findViewById(R.id.rec_artikel);
        recyclerViewArtikel.setLayoutManager(linearLayoutManager);
        recyclerViewArtikel.setAdapter(artikelAdapter);

        homePresenter = new HomePresenter(getApplicationContext(), this);
        homePresenter.loadArtikel(country, category, API_KEY);
    }

    @Override
    public void onSuccess(List<Artikel> artikelList) {
        this.artikelList.addAll(artikelList);
        artikelAdapter.notifyDataSetChanged();
    }

    @Override
    public void onError(String errorMessage) {
        Toast.makeText(this, errorMessage, Toast.LENGTH_SHORT).show();
    }

    @Override
    public void onFailure(String failureMessage) {
        Toast.makeText(this, failureMessage, Toast.LENGTH_SHORT).show();
    }
}

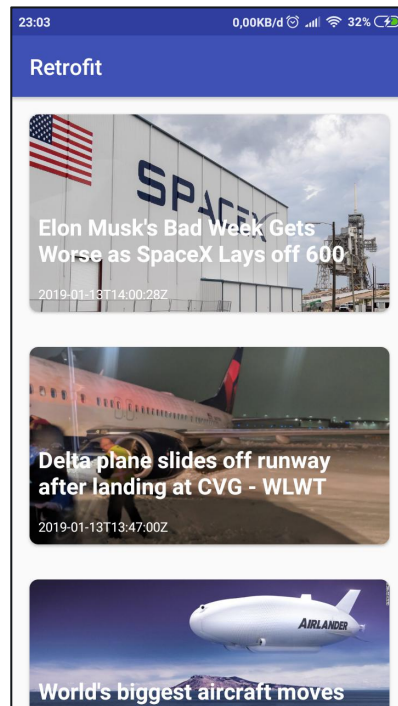
```

Pada fungsi `initView` kita bisa menginisialisasikan terlebih dahulu apa saja yang akan di handle pada `HomeActivity` ini. Pertama inisialisasikan terlebih dahulu adapter dari `recyclerview`, kemudian set `LayoutManager` dari `recyclerview` apakah data yang akan ditampilkan nantinya akan berbentuk `Linear`, `Grid` atau yang lainnya. Setelah itu set adapter `Recyclerview` tadi dengan class adapter dan lakukan request ke API melalui presenter.

Untuk melakukan request ke API, dibutuhkan 3 parameter yaitu `country`, `category` dan `apiKey`. Jika response berhasil maka program akan berjalan masuk ke fungsi `onSuccess`, fungsi dari `notifyDataSetChanged` adalah berguna untuk melakukan refresh data jika ada perubahan dalam adapter dan adapter akan secara otomatis terupdate sesuai dengan data yang telah didapat dari API.

Jika Response ada kegagalan maka program akan berjalan masuk ke `onError` atau `onFailure` tergantung dengan kesalahan yang terjadi dan kemudian aplikasi android akan menampilkan pesan berupa Toast yang telah ditentukan.

13. Selesai, maka tampilan akhir dari aplikasi tersebut adalah sebagai berikut :



C. TUGAS PRATIUM

1. Buatlah Aplikasi Android untuk mengirim data yang telah didapatkan dan ditampilkan ke activity detail (gambar, judul, dll).
2. Pelajari lagi struktur MVP(Model, View, Presenter) pada Android dan berikan penjelasan tentang alur dari MVP tersebut!

MODUL VIII

MEMANFAATKAN *WEBSERVICE II*

A. TUJUAN PRAKTIKUM

- Mengetahui dan memahami bagaimana cara mengakses database lokal di Android

B. DASAR TEORI

1. REST (Representational State Transfer)

REST (*REpresentational State Transfer*) merupakan stkitar arsitektur komunikasi berbasis web yang sering diterapkan dalam pengembangan layanan berbasis web. Umumnya menggunakan HTTP (*Hypertext Transfer Protocol*) sebagai protocol untuk komunikasi data. REST pertama kali diperkenalkan oleh Roy Fielding pada tahun 2000.

Pada arsitektur REST, REST server menyediakan *resources*(sumber daya/data) dan REST client mengakses dan menampilkan *resource* tersebut untuk penggunaan selanjutnya. Setiap *resource* diidentifikasi oleh URIs (*Universal Resource Identifiers*) atau global ID. *Resource* tersebut direpresentasikan dalam bentuk format teks, JSON atau XML. Pada umumnya formatnya menggunakan JSON dan XML.

Keuntungan REST

- bahasa dan platform agnostic
- lebih sederhana/simpel untuk dikembangkan ketimbang SOAP
- mudah dipelajari, tidak bergantung pada tools
- ringkas, tidak membutuhkan layer pertukaran pesan (messaging) tambahan
- secara desain dan filosofi lebih dekat dengan web

Kelemahan REST

- Mengasumsi model point-to-point komunikasi - tidak dapat digunakan untuk lingkungan komputasi terdistribusi di mana pesan akan melalui satu atau lebih perantara
- Kurangnya dukungan stkitar untuk keamanan, kebijakan, keiktalan pesan, dll, sehingga layanan yang mempunyai persyaratan lebih canggih lebih sulit untuk dikembangkan ("dipecahkan sendiri")
- Berkaitan dengan model transport HTTP

Berikut metode HTTP yang umum digunakan dalam arsitektur berbasis REST.

- **GET**, menyediakan hanya akses baca pada *resource*
- **PUT**, digunakan untuk menciptakan *resource* baru
- **DELETE**, digunakan untuk menghapus *resource*
- **POST**, digunakan untuk memperbarui *resource* yang ada atau membuat *resource* baru
- **OPTIONS**, digunakan untuk mendapatkan operasi yang disupport pada *resource*

2. Cara Kerja RESTful web service

Sebuah client mengirimkan sebuah data atau request melalui **HTTP Request** dan kemudian server merespon melalui **HTTP Response**. Komponen dari **http request** :

- Verb, HTTP method yang digunakan misalnya GET, POST, DELETE, PUT dll.
- *Uniform Resource Identifier* (URI) untuk mengidentifikasi lokasi resource pada server.
- HTTP Version, menunjukkan versi dari HTTP yang digunakan, contoh HTTP v1.1.
- Request Header, berisi metadata untuk HTTP Request. Contoh, type client/browser, format yang didukung oleh client, format dari body pesan, seting cache dll.
- Request Body, konten dari data.

Sedangkan komponen dari **http response** :

- Status/Response Code, mengindikasikan status server terhadap resource yang direquest. misal : 404, artinya resource tidak ditemukan dan 200 response OK.
- HTTP Version, menunjukkan versi dari HTTP yang digunakan, contoh HTTP v1.1.
- Response Header, berisi metadata untuk HTTP Response. Contoh, type server, panjang content, tipe content, waktu response, dll
- Response Body, konten dari data yang diberikan.

Latihan Membuat aplikasi CRUD menggunakan webservice :

Kali ini kita akan belajar bagaimana cara mengakses webservice di Android. Materi ini meliputi bagaimana cara pengambilan data, input data, delete dan update data melalui webservice. Kita akan menggunakan library “Retrofit dan OkHttp”. Contoh program ini dapat diunduh di <https://github.com/ddiffa/Modul8.git>

Dokumentasi API bisa dilihat di <https://apibarang.docs.apiary.io/> dan untuk source code pembuatan webservice bisa dilihat di <https://github.com/riskimidiw/api-barang>.

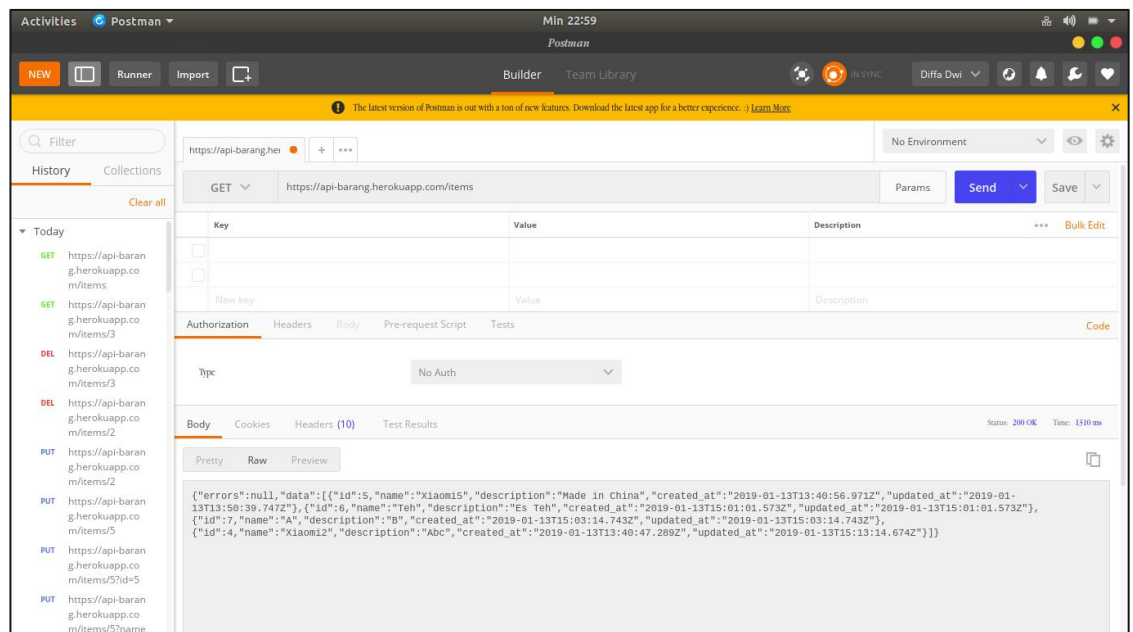
1. Buatlah Project baru, kemudian bernama Modul8 dan pilih Empty Activity. Setelah itu tambahkan beberapa library pada build.gradle (app) seperti dibawah ini kemudian sinkronkan :

```
implementation 'com.android.support:design:28.0.0'
implementation 'com.android.support:support-v4:28.0.0'
implementation 'com.squareup.retrofit2:retrofit:2.5.0'
implementation 'com.squareup.retrofit2:converter-gson:2.5.0'
implementation 'com.android.support:recyclerview-v7:28.0.0'
implementation 'com.squareup.okhttp3:okhttp:3.12.0'
implementation 'com.squareup.okhttp3:logging-interceptor:3.12.0'
implementation 'com.android.support:cardview-v7:28.0.0'
```

2. Tambahkan permission internet di AndroidManifest :

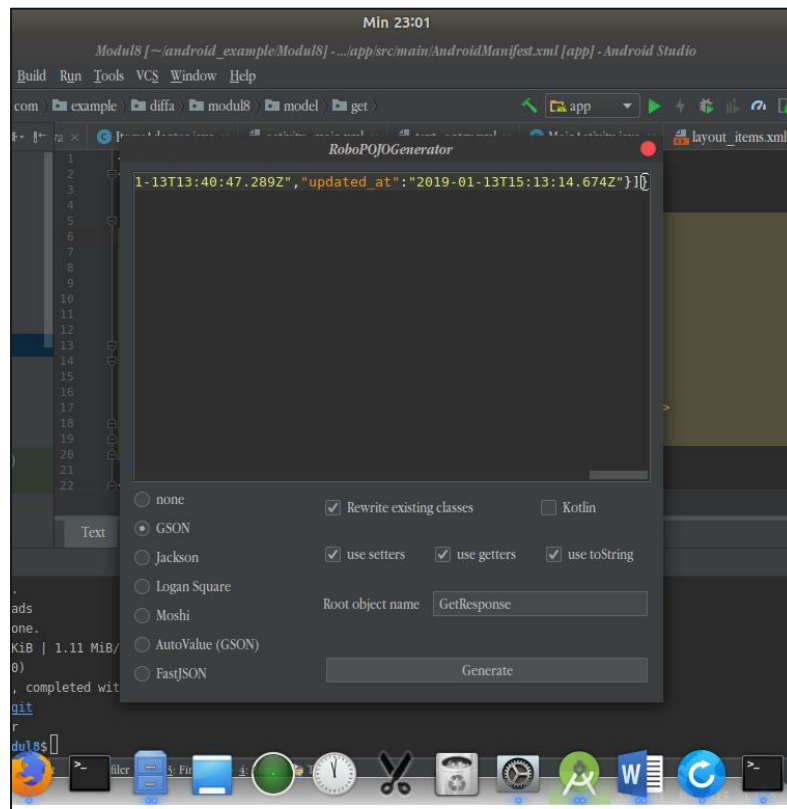
```
<uses-permission android:name="android.permission.INTERNET"/>
```

3. Lihat dokumentasi API di <https://apibarang.docs.apiary.io/> kemudian buka postman untuk mendapatkan response dari tiap perintah yang akan digunakan untuk membuat POJOs dari JSON, gunakan BASE URL ini <https://api-barang.herokuapp.com> contoh :



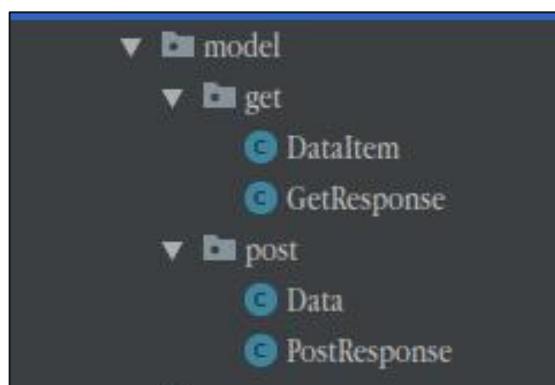
Gambar diatas merupakan penggunaan postman dalam menggunakan GET dari BASE URL yang telah ditentukan.

4. Buatlah package model, kemudian buat package lagi dan bernama get. setelah itu, klik → generate POJO from JSON dan copy-paste rawdata yang kita dapatkan dari postman tadi, seperti gambar dibawah ini :



Ganti none menjadi GSON dan nama menjadi GetResponse. Kemudian Generate.

5. Buatlah hal serupa dalam penggunaan POST, kemudian buatlah package baru bernama post di package model, sehingga menghasilkan seperti gambar dibawah ini :



6. Buat Package baru bernama remote, kemudian buat class interface ApiService seperti dibawah ini :

```
public interface ApiService {  
    @POST("/items")
```

```

    Call<PostResponse> createItems (@Query("name") String name,
                                   @Query("description") String description);

    @GET("/items")
    Call<GetResponse> getAllItems();

    @FormUrlEncoded
    @PUT("/items/{id}")
    Call<JsonObject> updateDataItems (@Path("id") String id,
                                      @Field("name") String name,
                                      @Field("description") String description);

    @DELETE("/items/{id}")
    Call<JsonObject> deleteDataItems (@Path("id") String id);
}

```

7. Buat class baru bernama BaseApp di package remote dan tambahkan code seperti dibawah ini :

```

public class BaseApp extends Application {
    public static ApiService service;
    private String url = "https://api-barang.herokuapp.com";
    @Override
    public void onCreate() {
        super.onCreate();
        service = getRetrofit().create(ApiService.class);
    }
    private Retrofit getRetrofit() {
        return new Retrofit.Builder()
            .baseUrl(url)
            .addConverterFactory(GsonConverterFactory.create())
            .client(getHttpClient())
            .build();
    }
    private OkHttpClient getHttpClient() {
        return new OkHttpClient.Builder()
            .addInterceptor(getHttpLogInterceptor())
            .build();
    }
    private Interceptor getHttpLogInterceptor() {

```

```

        HttpLoggingInterceptor loggingInterceptor = new
        HttpLoggingInterceptor();

        HttpLoggingInterceptor.Level level;
        if (BuildConfig.DEBUG) {
            level = HttpLoggingInterceptor.Level.BODY;
        } else {
            level = HttpLoggingInterceptor.Level.NONE;
        }

        loggingInterceptor.setLevel(level);
        return loggingInterceptor;
    }
}

```

8. Tambahkan satu baris code ini di AndroidManifest dalam application

```
android:name=".remote.BaseApp"
```

9. Buatlah MainActivity dan ubah file XML menjadi seperti dibawah ini :

```

<RelativeLayout 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=".MainActivity">
    <android.support.v7.widget.RecyclerView
        android:id="@+id/rv_items"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
    <android.support.design.widget.FloatingActionButton
        android:id="@+id/fb_items"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignParentBottom="true"
        android:layout_margin="16dp"
        android:src="@drawable/ic_add_circle_white_24dp" />
</RelativeLayout>

```

Perlu diketahui, untuk menambahkan gambar pada FloatingActionButton, kita perlu menambahkan gambarnya terlebih dahulu di resource drawable. Caranya klik

kanan pada drawable → Vector Asset. Kemudian pilih gambar sesuai yang ingin digunakan.

10. Buatlah layout lagi untuk layoutItem dari recyclerview, caranya klik kanan pada layout → new → Layout Resource File. Beri nama layout_items kemudian ok. Setelah itu tambahkan beberapa baris code seperti dibawah ini :

```
<?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="wrap_content"
    android:layout_marginTop="4dp"
    android:layout_marginLeft="16dp"
    android:layout_marginRight="16dp"
    android:layout_marginBottom="4dp"
    android:orientation="vertical">

    <TextView
        android:id="@+id/tv_id"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <TextView
        android:id="@+id/tv_name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textStyle="bold"/>
    <TextView
        android:id="@+id/tv_description"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="end"
        android:orientation="horizontal">
        <Button
            android:id="@+id/btn_edit"
            android:layout_width="wrap_content"
```

```

        android:layout_height="wrap_content"
        android:text="Edit" />
    <Button
        android:id="@+id/btn_delete"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Delete" />
</LinearLayout>
<View
    android:background="#888888"
    android:layout_width="match_parent"
    android:layout_height="2dp" />
</LinearLayout>

```

11. Buat package baru bernama adapter, kemudian buat kelas baru dalam package tersebut dan bernama ItemsAdapter. Setelah itu, tambahkan codingan seperti dibawah ini :

```

public class ItemsAdapter extends RecyclerView.Adapter<ItemsAdapter.Holder>
{
    private Context context;
    private List<DataItem> list;
    private OnItemClickListener listener;

    public ItemsAdapter(Context context, List<DataItem> list,
OnItemClickListener listener) {
        this.context = context;
        this.list = list;
        this.listener = listener;
    }

    @Override
    public Holder onCreateViewHolder(ViewGroup parent, int position) {
        View view = LayoutInflater.from(parent.getContext())
            .inflate(R.layout.layout_items, parent, false);
        return new Holder(view);
    }

    @Override
    public void onBindViewHolder(Holder holder, int position) {

```

```

        holder.bind(position, listener);
    }

    @Override
    public int getItemCount() {
        return list.size();
    }

    public class Holder extends RecyclerView.ViewHolder {
        private TextView tvName, tvDescription, tvId;
        private Button btnEdit, btnDelete;

        public Holder(View itemView) {
            super(itemView);

            tvId = itemView.findViewById(R.id.tv_id);
            tvName = itemView.findViewById(R.id.tv_name);
            tvDescription = itemView.findViewById(R.id.tv_description);
            btnDelete = itemView.findViewById(R.id.btn_delete);
            btnEdit = itemView.findViewById(R.id.btn_edit);
        }

        public void bind(final int position, final OnAdapterClickListener listener) {
            tvId.setText(String.valueOf(list.get(position).getId()));
            tvName.setText(list.get(position).getName());
            tvDescription.setText(list.get(position).getDescription());
            btnEdit.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    listener.onClicked(String.valueOf(list.get(position).getId()),
list.get(position).getName(), list.get(position).getDescription(), "edit");
                }
            });
            btnDelete.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    listener.onClicked(String.valueOf(list.get(position).getId()),
list.get(position).getName(), list.get(position).getDescription(),
"delete");
                }
            });
        }
    }
}

```

```

        });

    }

}

public interface OnAdapterClickListener {

    void onClicked(String id, String name, String description, String key);

}

}

```

Interface OnAdapterClickListener digunakan sebagai media pengiriman data melalui interface, dimana class MainActivity nantinya akan mengimplements OnAdapterClickListener untuk mendapatkan nilai yang dikirim dari adapter.

12. Buat package baru bernama presenter dan tambahkan class interface bernama MainInterface, kemudian tambahkan beberapa baris code seperti dibawah ini :

```

public interface MainInterface {

    void getAllItems();

    void updateItems(String id, String name, String description);

    void deleteItems(String id);

    void createItems(String name, String description);

}

```

13. Buat class interface baru di package presenter dan beri nama MainView kemudian tambahkan beberapa baris code seperti dibawah ini :

```

public interface MainView {

    void getSucces(GetResponse list);

    void setToast(String message);

    void onError(String errorMessage);

    void onFailure(String failureMessage);

}

```

14. Setelah itu, buat class baru lagi di package tersebut dan beri nama MainPresenter yang mengimplements ke class interface MainInterface, kemudian tambahkan beberapa baris code seperti dibawah ini :

```

public class MainPresenter implements MainInterface {

    private MainView mainView;

    public MainPresenter(MainView mainView) {

        this.mainView = mainView;

    }

}

```

```

@Override

public void getAllItems() {
    BaseApp.service.getAllItems().enqueue(new Callback<GetResponse>() {
        @Override
        public void onResponse(Call<GetResponse> call,
Response<GetResponse> response) {
            if (response.isSuccessful())
                mainView.getSucces(response.body());
            else
                mainView.onError(response.message());
        }
        @Override
        public void onFailure(Call<GetResponse> call, Throwable t) {
            mainView.onFailure(t.getMessage());
        }
    });
}

@Override

public void updateItems(String id, String name, String description) {
    BaseApp.service.updateDataItems(id,name,description).enqueue(new
Callback<JsonObject>() {
        @Override
        public void onResponse(Call<JsonObject> call, Response<JsonObject>
response) {
            if (response.isSuccessful())
                mainView.setToast(response.message());
            else
                mainView.onError(response.message());
        }
        @Override
        public void onFailure(Call<JsonObject> call, Throwable t) {
            mainView.onFailure(t.getMessage());
        }
    });
}

@Override

public void deleteItems(String id) {

```

```

        BaseApp.service.deleteDataItems(id).enqueue(new
Callback<JsonObject>() {
            @Override
            public void onResponse(Call<JsonObject> call, Response<JsonObject>
response) {
                if (response.isSuccessful())
                    mainView.setToast(response.message());
                else
                    mainView.onError(response.message());
            }
            @Override
            public void onFailure(Call<JsonObject> call, Throwable t) {
                mainView.onFailure(t.getMessage());
            }
        });
    }
    @Override
    public void createItems(String name, String description) {
        BaseApp.service.createItems(name,description).enqueue(new
Callback<PostResponse>() {
            @Override
            public void onResponse(Call<PostResponse> call,
Response<PostResponse> response) {
                if (response.isSuccessful())
                    mainView.setToast(response.message());
                else
                    mainView.onError(response.message());
            }
            @Override
            public void onFailure(Call<PostResponse> call, Throwable t) {
                mainView.onFailure(t.getMessage());
            }
        });
    }
}
}

```

15. Tambahkan satu layout lagi dan beri nama text_entry, kemudian tambahkan codingan seperti berikut :

```
<?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">
    <EditText
        android:id="@+id/edt_name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <EditText
        android:id="@+id/edt_description"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

Layout ini nantinya akan digunakan untuk membuat sebuah Alert Dialog yang akan dipakai di MainActivity untuk menambahkan barang dan mengupdate barang.

16. Setelah semuanya selesai, selanjutnya tambahkan codingan dibawah ini pada class MainActivity :

```
public class MainActivity extends AppCompatActivity implements MainView,
ItemsAdapter.OnItemClickListener {
    private RecyclerView recyclerView;
    private ItemsAdapter itemsAdapter;
    private MainPresenter presenter;
    private List<DataItem> list;
    private FloatingActionButton floatingActionButton;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        list = new ArrayList<>();
        recyclerView = findViewById(R.id.rv_items);
        floatingActionButton = findViewById(R.id.fb_items);
        floatingActionButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
```

```

        newItemsDialog();

    }

});

itemsAdapter = new ItemsAdapter(this, list, this);
recyclerView.setLayoutManager(new LinearLayoutManager(this));
recyclerView.setAdapter(itemsAdapter);
presenter = new MainPresenter(this);
presenter.getAllItems();
}

private void newItemsDialog() {
    LayoutInflater factory = LayoutInflater.from(this);
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Tambah Barang");
    final View textEntryView = factory.inflate(R.layout.text_entry, null);
    final EditText name = (EditText)
textEntryView.findViewById(R.id.edt_name);
    final EditText description = (EditText)
textEntryView.findViewById(R.id.edt_description);
    name.setHint("Nama Barang");
    description.setHint("Deskripsi");
    name.setText("", TextView.BufferType.EDITABLE);
    description.setText("", TextView.BufferType.EDITABLE);
    builder.setView(textEntryView);
    builder.setPositiveButton("Ya", new DialogInterface.OnClickListener()
{
    @Override
    public void onClick(DialogInterface dialog, int which) {
        if (!name.getText().toString().equals("")) {
            presenter.createItems(name.getText().toString(),
description.getText().toString());
        } else {
            Toast.makeText(MainActivity.this, "Masukkan Nama Barang",
Toast.LENGTH_SHORT).show();
        }
    }
});

    builder.setNegativeButton("Batal", new
DialogInterface.OnClickListener() {

```



```

        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
        }
    });
    builder.show();
}

private void deleteDialog(final String id) {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Apakah kita Benar Akan Menghapus Item ini?");
    builder.setPositiveButton("Ya", new DialogInterface.OnClickListener()
    {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            presenter.deleteItems(id);
        }
    });
    builder.setNegativeButton("Tidak", new
    DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
        }
    });
    builder.show();
}

private void editDialog(final String id, final String name, final String
description) {
    LayoutInflater factory = LayoutInflater.from(this);
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Tambah Barang");
    final View textEntryView = factory.inflate(R.layout.text_entry, null);
    final EditText edtName = (EditText)
textEntryView.findViewById(R.id.edt_name);
    final EditText edtDescription = (EditText)
textEntryView.findViewById(R.id.edt_description);
    edtName.setText(name, TextView.BufferType.EDITABLE);
    edtDescription.setText(description, TextView.BufferType.EDITABLE);
}

```

```

        builder.setView(textEntryView);
        builder.setTitle("Update Barang");
        builder.setPositiveButton("Ya", new DialogInterface.OnClickListener()
{
            @Override
            public void onClick(DialogInterface dialog, int which) {
                presenter.updateItems(id,          edtName.getText().toString(),
edtDescription.getText().toString());
            }
        });
        builder.setNegativeButton("Tidak",          new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                dialog.cancel();
            }
        });
        builder.show();
    }
    @Override
    protected void onResume() {
        super.onResume();
        presenter.getAllItems();
    }
    @Override
    public void getSuccess(GetResponse list) {
        this.list.clear();
        this.list.addAll(list.getData());
        itemsAdapter.notifyDataSetChanged();
    }
    @Override
    public void showToast(String message) {
        Toast.makeText(this, message, Toast.LENGTH_LONG).show();
        presenter.getAllItems();
    }
    @Override
    public void onError(String errorMessage) {

```

```

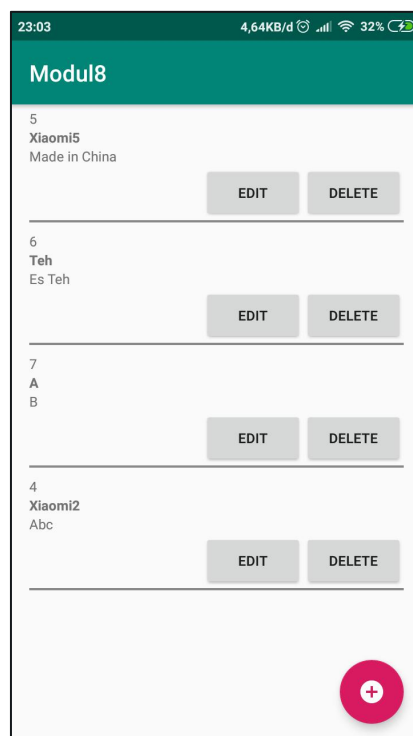
        Toast.makeText(this, errorMessage, Toast.LENGTH_LONG).show();
    }

    @Override
    public void onFailure(String failureMessage) {
        Toast.makeText(this, failureMessage, Toast.LENGTH_LONG).show();
    }

    @Override
    public void onClicked(String id, String name, String description, String
key) {
        if (key.equalsIgnoreCase("edit")) {
            editDialog(id, name, description);
        } else {
            deleteDialog(id);
        }
    }
}
}

```

17. Selesai, Aplikasi akan terlihat seperti ini :



C. TUGAS PRATIKUM

1. Buatlah Aplikasi Android untuk mencari barang sesuai id, menggunakan URL GET <https://api-barang.herokuapp.com/items/{id}>.