# MODUL 4

# (INTENT, ACTIVITY, FRAGMENT dan PASSING DATA ANTAR INTENT)

## DESKRIPSI TEMA

- Intent dan passing data antar Intent
- Activity dan lifecyclenya
- Fragment dan penggunaannya

## CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

Mahasiswa mampu menggunakan Activity dan Fragment serta melakukan pengiriman dan penerimaan data antar Activity dan/atau Fragment menggunakan class Intent.

## PENUNJANG PRAKTIKUM

1. Software Android Studio

## LANGKAH-LANGKAH PRAKTIKUM

**Tutorial: Intent dan Passing Data antar Intent.**

Secara sederhana, Intent merupakan cara pada Android untuk berpindah halaman (Activity / Fragment). Pada tutorial ini Anda akan mempelajari cara berpindah halaman dengan menggunakan Intent (baik explicit maupun implicit) dan mem-*passing* data dari satu Activity ke Activity lainnya.

Bukalah Android Studio dan pilih **Start a new Android Studio Project**

1. Buatlah project dengan format:

   Name : **Week [xxA]_[NIM].** Contoh: **Week 04A_12001**.

   Company Domain : **umn.ac.id**.

   Project Location : **D:\FTI\IF634_IS534_MOBILE\workspaces**

   Target Android : (Phone and Tablet) **API 25: Android 7.1.1 (Nougat)**

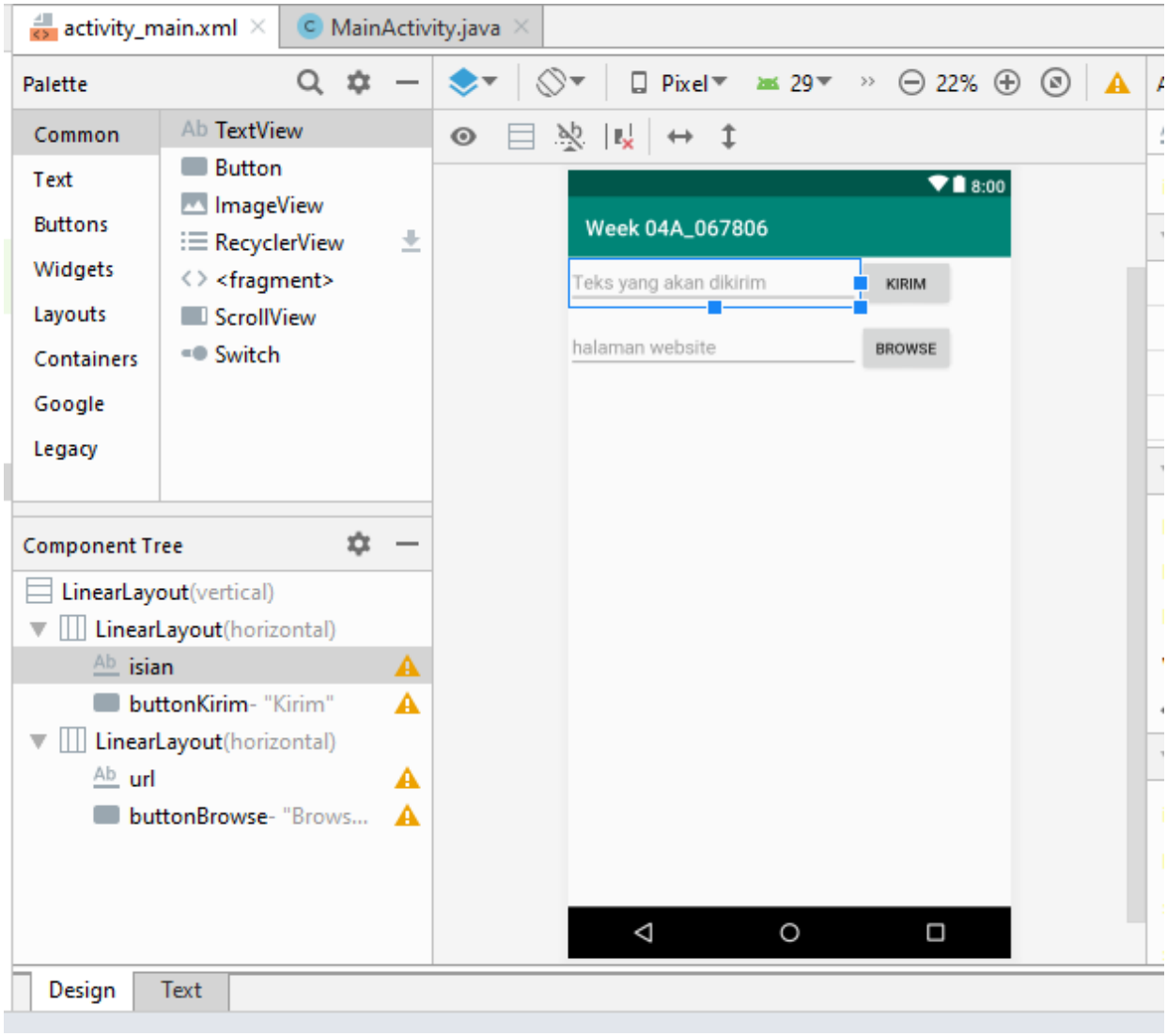
   Activity Template : **Empty Activity**

   Activity Name : **MainActivity**

   Layout Name : **activity_main**

2. Buka dan edit file layout **app > res > layout activity_main.xml**. Gunakan **LinearLayout** (Vertical) sebagai root layout manager, serta **LinearLayout** (Horizontal) sebagai sub layout yang menampilkan

**EditText** dan **Button**. Baris pertama akan digunakan untuk memanfaatkan passing data menggunakan **Intent Explicit** yang akan membuka Activity berikutnya dan baris kedua akan digunakan untuk memanfaatkan passing data menggunakan **Intent Implicit** yang akan memanggil browser untuk menampilkan halaman yang diinputkan user via **EditText**. Gunakan **"Desain"** mode sehingga tampilan seperti pada gambar berikut:




Atau gunakan **"Text"** mode sehingga file **activity_main.xml** seperti terlihat pada kode sebagai 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"
    android:orientation="vertical"
    tools:context=".MainActivity">
```

```
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="60dp">
        <EditText
            android:id="@+id/isian"
            android:layout_width="270dp"
            android:layout_height="wrap_content"
            android:hint="Teks yang akan dikirim" />
        <Button
            android:id="@+id/buttonKirim"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Kirim" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="60dp"
        android:orientation="horizontal">
        <EditText
            android:id="@+id/url"
            android:layout_width="270dp"
            android:layout_height="wrap_content"
            android:hint="halaman website" />
        <Button
            android:id="@+id/buttonBrowse"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Browse" />
    </LinearLayout>
</LinearLayout>
```

3. Buka dan edit file **MainActivity.java** untuk dapat berinteraksi dengan object-object yang ada pada layout di atas. Deklarasikan dua **EditText** object dengan nama **etIsian** dan **etUrl** dan dua Button object dengan nama **btnKirim** dan **btnBrowse**. Tambahkan juga pada **onCreate(...)** method untuk tersambung dengan object pada layout object untuk masing-masing View object di atas. Kode pada **MainActivity.java** akan terlihat seperti berikut:

```
public class MainActivity extends AppCompatActivity {
    private EditText    etIsian, etUrl;
    private Button      btnKirim, btnBrowse;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        etIsian = findViewById(R.id.isian);
        etUrl = findViewById(R.id.url);
        btnBrowse = findViewById(R.id.buttonBrowse);
        btnKirim = findViewById(R.id.buttonKirim);
    }
}
```

4. Selanjutnya kita akan eksperimen dengan passing data menggunakan **Implicit Intent,** di mana ketika user klik tombol **Browse,** maka app anda akan mencari dan membuka web browser yang telah didefinisikan oleh Adroid Runtime. Jika isian Halaman website kosong, app akan tetap membuka web browser dengan default website UMN (http://www.umn.ac.id). Tambahkan kode berikut pada method onCreate(...) melanjutkan langkah di atas:

```
btnBrowse.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String urlText = etUrl.getText().toString();
        if(urlText.isEmpty()){
            urlText = "http://www.umn.ac.id/";
        }
        Intent browseIntent = new Intent(Intent.ACTION_VIEW);
        browseIntent.setData(Uri.parse(urlText));
        if(browseIntent.resolveActivity(getPackageManager()) != null){
            startActivity(browseIntent);
        }
    }
});
```

5. Sementara untuk eksperimen dengan **Explicit Intent,** kita harus siapkan **Activity** yang akan menerima request dari MainActivity ketika user meng-klik tombol **"Kirim"**. Buat Activity baru dengan nama **ActivityDua,** dengan cara klik kanan pada folder **app > java > [domain].week04a_[NIM]** lalu pilih **New > Activity > Empty Activity**. Setelah dialog isian muncul, isi nama dengan **ActivityDua** dan nama layout dengan **activity_dua** lalu klik **"Finish"**.

6. Buka dan edit file **AndroidManifest.xml** pada folder **app > manifests**. Temukan element **<activity>** yang baru saja dibuat oleh Android Studio untuk Activity kedua.

```
<activity android:name=".ActivityDua"></activity>
```

Ganti skrip tersebut dengan:

```
<activity android:name=".ActivityDua"
    android:label="Activity Kedua"
    android:parentActivityName=".MainActivity">
    <meta-data
        android:name="android.support.PARENT_ACTIVITY"
        android:value=
          "id.ac.umn.khaeruzzaman.yaman.week04a_067806.MainActivity" />
</activity>
```

Dengan menggunakan attribute parentActivityName, kita mengindikasikan bahwa **MainActivity** adalah parent dari **ActivityDua**. Hubungan ini dapat digunakan untuk Up Navigation pada app kita.

App bar pada **ActivityDua** akan memiliki panah kiri ( ← ) sehingga user dapat melakukan navigasi **"upward"** ke **MainActivity**.

7. Buka dan edit layout file **activity_dua.xml** pada folder **app > res > layout**. Untuk membuat sebuah TextView yang akan menampilkan pesan atau teks yang diterima dari **MainActivity,** sebuah **EditText** untuk mengisikan pesan balasan yang dikirim balik ke **MainActivity,** dan sebuah button untuk mengirim balasan tersebut. Sekilas tampilan Activity ke dua ini mirip dengan ActivityMain dengan hanya memiliki satu Button saja. Berikut skrip lengkap dari **activity_dua.xml**:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".ActivityDua">
    <TextView
        android:id="@+id/pesanDiterima"
        android:layout_width="match_parent"
        android:layout_height="60dp"
        android:text="" />
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="60dp"
        android:orientation="horizontal">
        <EditText
            android:id="@+id/pesanBalik"
            android:layout_width="270dp"
            android:layout_height="wrap_content"
            android:hint="Jawaban" />
        <Button
            android:id="@+id/kirimBalik"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:onClick="kirimBalik"
            android:text="Balas" />
    </LinearLayout>
</LinearLayout>
```

Catatan: layout file ini memiliki sebuah error pada object Button **kirimBalik** untuk attribute **android:onClick="kirimBalik"** karena pada activity **ActivityDua.java** belum ada implementasi method dengan nama dan signature **public void kirimBalik(View view)** yang akan menangani klik button pada button **kirimBalik**.

8. Buka file **ActivityDua.java** untuk membuat sebuah TextView object yang menerima pesan dari MainActivity, EditText object yang dapat diisi jawaban yang akan dikirim balik ke MainActivity,

sebuah Button yang akan menangani interaksi klik dari user, serta beberapa method yang harus diimplementasikan agar proses passing data dari dan ke MainActivity berjalan dengan baik. Berikut kode yang harus ditambahkan pada **ActivityDua.java**:

```
public class ActivityDua extends AppCompatActivity {
    private TextView tvPesanDiterima;
    private EditText etJawaban;
    private Button btnBalasKirim;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_dua);
        tvPesanDiterima = findViewById(R.id.pesanDiterima);
        etJawaban = findViewById(R.id.pesanBalik);
        btnBalasKirim = findViewById(R.id.kirimBalik);
        Intent mainIntent = getIntent();
        String pesanDiterima =
               mainIntent.getStringExtra("PesanDariMain");
        tvPesanDiterima.setText(pesanDiterima);
    }
    public void kirimBalik(View view){
        String jawaban = etJawaban.getText().toString();
        Intent balasIntent = new Intent();
        balasIntent.putExtra("Jawaban", jawaban);
        setResult(RESULT_OK, balasIntent);
        finish();
    }
}
```

Object **mainIntent** adalah object yang merepresentasikan Intent yang memanggil **ActivityDua** yaitu **MainActivity**. Method **getStringExtra("PesanDariMain")** mengambil data yang dibawa / dikirim oleh **mainIntent** tersebut dengan nama **"PesanDariMain"** yang selanjutnya dutampilkan pada oobject TextView **tvPesanDiterima**. Method **kirimBalik(...)** wajib ada pada **ActivityDua** karena object Button **kirimBalik** yang ada pada layout file activity_dua.xml memiliki atribut **android:onClick="kirimBalik"**. Pada Method ini, untuk kembali ke MainActivity dengan mengirimkan hasil (atau pesan) tidak menggunakan method **startActivity(...)** tetapi menggunakan method **setResult(...)** karena **ActivityDua** ini aktif karena dipanggil oleh **MainActivity** dengan menggunakan s**tartActivityForResult()** yang akan menunggu feedback atau balasan dari activity yang dipanggilnya, dalam eksperimen ini method **setResult(...)** sebagai bentuk feedback dari activity yang dipanggil. Method **setResult(...)** ini akan berkomunikasi dengan callback method **onActivityResult(...)** yang diimplemetasi pada activity pemanggil yaitu **MainActivity**.

9. Buka dan tambahkan TextView pada **activity_main.xml** untuk dapat menampun pesan jawaban yang diberikan oleh **ActivityDua**. Letakkan penambahan TextView di bawah ini setelah inner

```
<TextView
    android:id="@+id/jawaban"
    android:layout_width="match_parent"
    android:layout_height="60dp" />
```

LinearLayout yang kedua:

10. Buka dan lengkapi **MainActivity.java** untuk mendeklarasikan variable TextView **tvJawaban** yang diletakkan didalam class MainActivity sebelum **onCreate(...)** method sebagai berikut:

Selanjutnya lakukan koneksi variable **tvJawaban** dengan object TextView **@+id/jawaban** yang ada pada layout file **activity_main.xml** di dalam method **onCreate(...),** seperti di bawah ini:

```
tvJawaban = findViewById(R.id.jawaban);
```

Masih di dalam method **onCreate(...)** implementasikan penanganan klik pada button **btnKirim** menggunakan method **setOnClickListener(...)** yang akan mengirim sebuah pesan dengan nama **"PesanDariMain"** yang dibawa oleh Intent yang merepersentasikan activity yang dipanggil yaitu

```
btnKirim.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intentDua = new Intent(MainActivity.this,
                                ActivityDua.class);
        String isian = etIsian.getText().toString();
        intentDua.putExtra("PesanDariMain",isian);
        startActivityForResult(intentDua, 1);
    }
});
```

**ActivityDua.class** dengan menggunakan method **startActivityForResult(...)**. Berikut kode implementasi dari penanganan button tersebut:

11. Untuk dapat menerima kembali jawaban dari child activity dan / atau dari activity yang dipanggil menggunakan **startActivityForResult(...),** kita harus mengimplementasikan call-back method **onActivityResult(...)** pada **MainActivity.** Sebagai overriding method, kode awal yang dilakukan alah memanggil method yang pada parent class dengan menggunakan keyword **super,** yaitu **super.onActivityResult(...).** Kemudian yakinkan bahwa nilai **requestCode** yang dikirim oleh method **startActivityForResult(...)** dengan yang diterima harus sama. Untuk kasus eksperimen ini

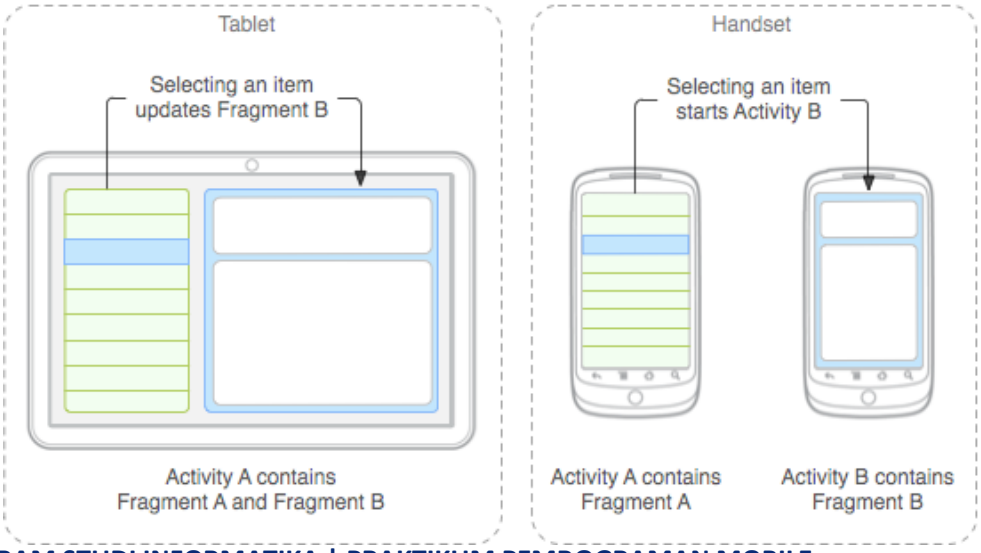
nilainya adalah **1 (satu)**. Begitu juga dengan resultCode, harus sama antara yang dikirim oleh method **setResult(...)** yang ada pada activity yang dipanggil (**ActivityDua**) dengan yang diterima oleh method ini. Untuk eksperimen ini nilainya adalah **RESULT_OK**. Berikut kode pada method tersebut:

```
@Override
public void onActivityResult(int requestCode,
                             int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == 1) {
        if (resultCode == RESULT_OK) {
            String jawaban = data.getStringExtra("Jawaban");
            tvJawaban.setText(jawaban);
        }
    }
```

12. Demikian Tutorial pengembangan sebuah Android app dengan memanfaatkan Activity, Intent (implicit dan excplicit), serta passing data antar Activity melelui Intent.

**Tutorial: Fragment (via Layout)**

Fragment mewakili perilaku atau bagian dari antarmuka pengguna dalam Activity. Anda bisa mengombinasikan beberapa fragmen dalam satu Activity untuk membangun UI multipanel dan menggunakan kembali sebuah fragmen dalam beberapa Activity. Anda bisa menganggap fragmen sebagai bagian modular dari Activities, yang memiliki daur hidup sendiri, menerima kejadian masukan sendiri, dan yang bisa Anda tambahkan atau hapus saat aktivitas berjalan (semacam "sub aktivitas" yang bisa digunakan kembali dalam aktivitas berbeda). Salah satu kegunaan yang praktis dari penggunaan fragment adalah ketika harus membuat 2 buah antarmuka yang mirip antara Tablet dan Smartphone yang dapat dilihat pada gambar berikut:



Fragment memiliki Lifecycle yang berbeda dari Activity, sehingga Anda harus mempelajarinya lebih lanjut. Lifecycle pada Fragment dapat dilihat pada tautan:

https://developer.android.com/guide/components/fragments?hl=id.

Pada tutorial ini Anda akan belajar untuk membuat Fragment sederhana dengan menggunakan metode layout.

1. Bukalah app\res\layout\activity_first.xml
2. Tambahkan 2 buah button tepat di bawah EditText pada **app\java\[namapackage]\MainActivity**, dengan id "main_button_change_1" dan "main_button_change_2", width "match_parent", text "Halaman 1" dan "Halaman 2"

   Layout tetap menggunakan ConstraintLayout, Anda boleh mempergunakan Tab **Design** / Drag-Drop-UI, tapi tetap disarankan tetap menggunakan Tab **Text**

   Buatlah tampilan semirip mungkin dengan gambar di bawah ini.


3. Buatlah dua buah Activity baru (Empty Activity) dengan nama **SecondActivity** dan **ThirdActivity**
4. Tambahkan pada **app\java\[namapackage]\MainActivity** kode untuk berpindah Activity dari

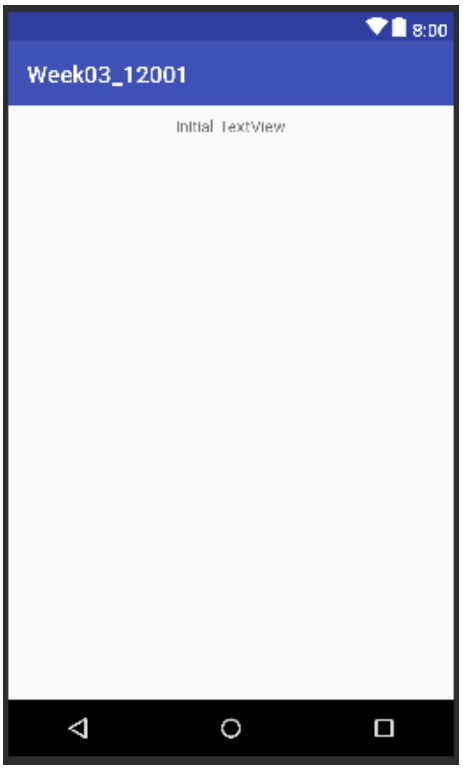
**MainActivity** ke **SecondActivity** dan dari **MainActivity** ke **ThirdActivity** pada saat Button main_button_change_1 dan main_button_change_2 pada **app\res\layout\activity_main.xml** ditekan.

5. Tambahkan dua buah layout, dengan Root Element-nya berupa LinearLayout dengan orientation vertical, masing-masing berikan nama fragment_first.xml dan fragment_second.xml.

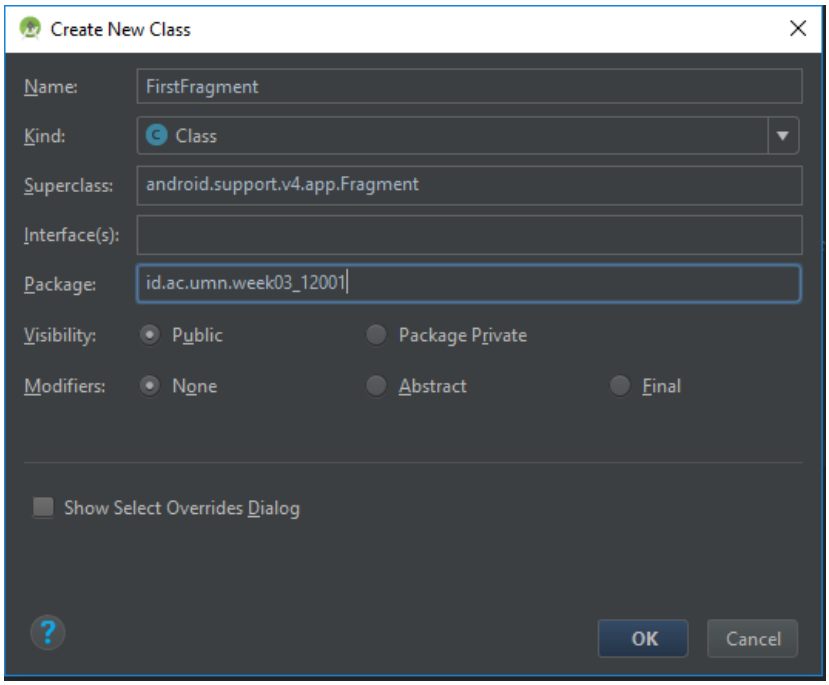
6. Pada fragment_first.xml, tambahkan sebuah EditText dengan id "fragment_first_edittext_tulisan", width "match_parent", hint "Tulisan" dan sebuah Button dengan id "fragment_first_button_berubah" dan text "Ganti Tulisan"

   Hasil Akhir pada **fragment_first.xml** dapat dilihat pada gambar di bawah.


7. Pada **fragment_second.xml**, tambahkan sebuahTextView dengan id "fragment_second_textview_tulisan", width "wrap_content", dan text "Initial TextView". Modifikasilah sehingga hasil akhir pada **fragment_second.xml** dapat dilihat pada gambar di bawah




8. Buatlah sebuah class Java (Klik kanan pada **package → New → Java Class** dengan nama "FirstFragment" dan merupakan **superclass** dari "android.support.v4.app.Fragment".




(Samakan package dengan package yang Anda buat)

Apabila nanti pada komputer di UMN terdapat error pada gradle yang menyatakan bahwa **implementation 'com.android.support:support-v4:27.1.1'** tidak dapat ditemukan, gantilah dengan

**implementation 'com.android.support:support-fragment:27.1.1'** kemudian Sync gradle ulang.

9. Bukalah **FirstFragment** yang sudah dibuat, dan tambahkan / modifikasi kode seperti di bawah ini.

```
public class FirstFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
                             @Nullable ViewGroup container,
                             @Nullable Bundle savedInstanceState) {

        View view = inflater.inflate(R.layout.fragment_first, container,  attachToRoot: false);
        return view;
    }
}
```

Fungsi dari inflater ini adalah sebagai **penempel** layout (xml) terhadap **ViewGroup** yang disediakan pada Fragment / Activity.

10. Buatlah satu class lagi dengan nama **SecondFragment** dengan superclass yang sama seperti **FirstFragment** dan tambahkan / modifikasi kode seperti di bawah ini.

```
public class SecondFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
                             @Nullable ViewGroup container,
                             @Nullable Bundle savedInstanceState) {

        View view = inflater.inflate(R.layout.fragment_second, container,  attachToRoot: false);
        return view;
    }
}
```

11. Bukalah activity_second.xml dan tambahkanlah 2 buah fragment dengan id "second_activity_fragmenT_1" dan "second_activity_fragment_2", width "match_parent" dan height "wrap_content" dan hubungkan fragment 1 dengan **FirstFragment** dan fragment 2 dengan **Second Fragment**.
Kode lengkapnya dapat dilihat di bawah sini namun ...

Cobalah untuk mengerjakannya tanpa melihat kode di bawah ini terlebih dahulu.

```
<?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=".SecondActivity">


    <fragment
        android:id="@+id/second_activity_fragment_1"
        android:name="id.ac.umn.week03_12001.FirstFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <fragment
        android:id="@+id/second_activity_fragment_2"
        android:name="id.ac.umn.week03_12001.SecondFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/second_activity_fragment_1" />
</android.support.constraint.ConstraintLayout>
```

(Samakan name dengan package yang Anda buat)

12. Jalankan aplikasi yang sudah Anda buat, dan tekanlah tombol dengan tulisan **Halaman 2** pada **MainActivity,** seharusnya aplikasi akan memunculkan tampilan sebagai berikut

13. Selamat, Anda sudah berhasil membuat Fragment dengan menggunakan Layout secara manual.

**Tutorial: Fragment (via Programmatic)**

Pada tutorial sebelumnya Anda sudah berhasil membuat fragment dengan menggunakan layout, dengan cara membuat 2 Class Fragment dan 2 Layout Fragment dan hanya menempelkan Fragment tersebut secara langsung pada <fragment> yang terdapat di Activity. Pada tutorial ini Anda akan me-"nempelkan" Fragment terhadap <fragment> pada Activity secara *programmatic*.

Kita akan menggunakan beberapa Class baru yang bertugas untuk mengelola seluruh Fragment yang ada pada aplikasi Android yaitu **FragmentManager** dan **FragmentTransaction**, kedua Class ini sangat berguna apabila Anda akan membuat Fragment yang bersifat dinamis, misalnya dengan menekan satu tombol Anda akan mengubah tampilan dari separuh halaman Activity.

1. Buatlah sebuah Activity baru (Empty Activity) dengan nama **ThirdActivity** dan layout dengan nama **activity_third.xml**
2. Buatlah kode agar pada **MainActivity**, tombol main_button_change_2 dapat berpindah ke **ThirdActivity** pada saat ditekan tombolnya.
3. Bukalah **activity_third.xml** dan tuliskanlah kode di bawah ini.

```
<?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=".ThirdActivity">

    <FrameLayout
        android:id="@+id/third_activity_fragment_1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <FrameLayout
        android:id="@+id/third_activity_fragment_2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintTop_toBottomOf="@id/third_activity_fragment_1"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"/>

</android.support.constraint.ConstraintLayout>
```

Perhatikan bahwa sudah tidak menggunakan tag <fragment> namun sudah diganti menjadi <FrameLayout>

Hal ini dikarenakan untuk membuat Fragment bersifat programmatic dibutuhkan <FrameLayout> bukan <fragment>.

4. Bukalah **ThirdActivity** dan tambahkan kode yang ada di bawah ini.

```
public class ThirdActivity extends AppCompatActivity {

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

        FragmentManager fragmentManager = getSupportFragmentManager();
        FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();

        Fragment firstFragment = new FirstFragment();
        fragmentTransaction.replace(R.id.third_activity_fragment_1, firstFragment);

        Fragment secondFragment = new SecondFragment();
        fragmentTransaction.replace(R.id.third_activity_fragment_2, secondFragment);
        fragmentTransaction.commit();
    }
}
```

Perhatikan pada saat import Fragment, FragmentManager, dan FragmentTransaction. Pada Praktikum ini kita menggunakan android.support.v4.app.xxx untuk ketiganya.

Selamat, Anda sudah berhasil membuat Fragment secara programmatic.

**Latihan, Tugas, dan Eksperimen:**

1. Ikuti langkah-langkah di atas sampai project anda dengan nama **Week04a_[NIM]** dan Experimen dengan Fragment baik yang menggunakan Layout (Manual) maupun yang programmatic sudah dapat dijalankan dan berhasil menampilkan sesuai dengan yang diharapkan. Zip (kompres) file-file berikut dan submit ke isian tugas pada eLearning:
   a. Week04a_[NIM]: (Activity, Intent dan Passing Data antar Intent)
      1) MainActivity.java
      2) ActivityDua.java
      3) activity_main.xml
      4) activity_dua.xml
      5) AndroidManifest.xml

b. Fragment menggunakan Layout
   1) MainActivity.java
   2) SecondActivity.java
   3) ThirdActivity.java
   4) FirstFragment.java
   5) SecondFragment.java
   6) Activity_main.xml
   7) fragment_first.xml
   8) fragment_second.xml
   9) activity_second.xml
   10) activity_third.xml

c. Fragment via Programmatic
   1) MainActivity.java
   2) SecondActivity.java
   3) ThirdActivity.java
   4) FirstFragment.java
   5) SecondFragment.java
   6) Activity_main.xml
   7) fragment_first.xml
   8) fragment_second.xml
   9) activity_second.xml
   10) activity_third.xml

2. Lakukan pengujian terhadap app anda dengan skenario berbagai kemungkinan yang mengirim data dari satu activity ke activity lainnya. Beri catatan dan kesimpulan dari hasil pengujian app anda dan submit ke eLearning portal.