



Kampus
Merdeka
INDONESIA JAYA

MODUL PRAKTIKUM PEMOGRAMAN BERGERAK



PEMBELAJARAN
DARING KOLABORATIF
(PDK)



TIM Penyusun :

**KRISTINA ANNETASIA BR SITEPU
HERU SAPUTRA**

PERTEMUAN 9 FRAGMENT

9.1 TUJUAN PEMBELAJARAN :

- a. Mahasiswa dapat membuat aplikasi dengan menggunakan fragment.

9.2 ALAT DAN BAHAN :

- a. Laptop/PC
- b. Android Studio
- c. Sistem Operasi Windows

9.3 MATERI

1. Pengertian Fragment

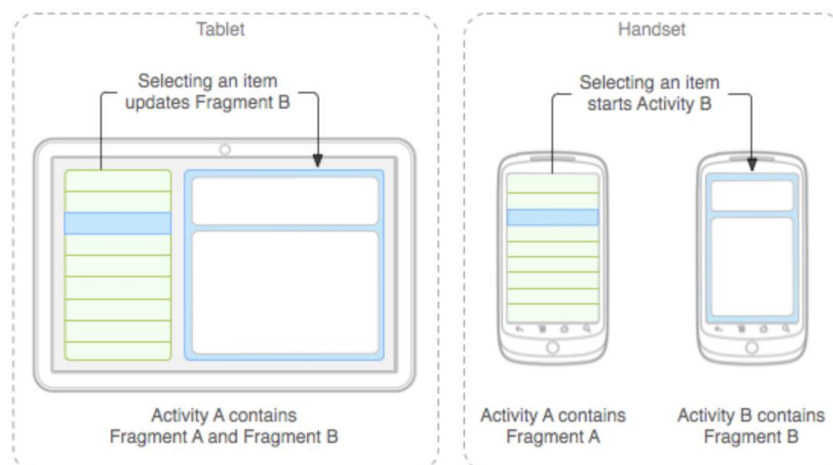
Android Fragment adalah bagian yang dapat digunakan kembali dari antarmuka pengguna android activity yang digunakan untuk membuat UI yang dinamis dan fleksibel.

Di aplikasi Android kita dapat menggunakan beberapa fragmen dalam satu aktivitas untuk membuat UI Multi-Pane dan juga dapat menggunakan satu fragmen dalam beberapa aktivitas.

Fragment Manager bertanggung jawab untuk menambah / menghapus atau mengganti fragmen pada waktu berjalan di mana pun aktivitas.

Bila Kita menambahkan fragmen sebagai bagian dari layout aktivitas, fragmen itu akan berada dalam ViewGroup di hierarki tampilan aktivitas tersebut dan fragmen mendefinisikan layout tampilannya sendiri.

Kita bisa menyisipkan fragmen ke dalam layout aktivitas dengan mendeklarasikan fragmen dalam file layout aktivitas, sebagai elemen `<fragment>`, atau dari kode aplikasi dengan menambahkannya ke ViewGroup yang ada.



Pada gambar di atas, dicontohkan bagaimana dua modul UI yang didefinisikan oleh fragmen bisa digabungkan ke dalam satu activity untuk desain tablet namun dipisahkan untuk desain handset.

2. Membuat Fragment

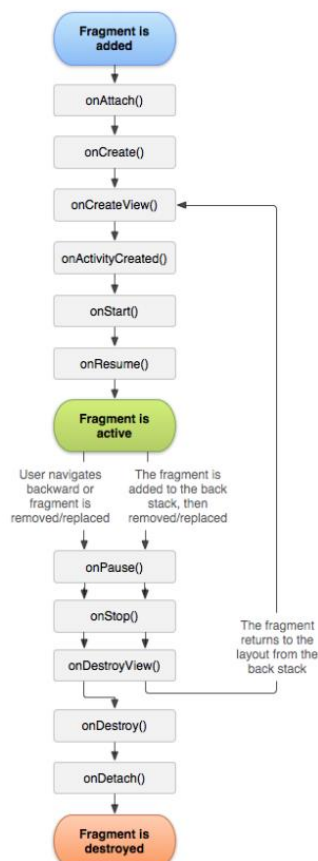
Untuk membuat fragment, kita membuat subkelas fragment (atau subkelas yang ada). kelas fragment memiliki kode yang mirip seperti Activity. Kelas ini memiliki metode callback yang serupa dengan activity seperti onCreate(), onStart(), onPause(), dan onStop().

Kode Fragment Dasar Dalam XML

```
<fragment
android:id="@+id/fragments"
android:layout_width="match_parent"
android:layout_height="match_parent" />
```

3. Siklus Hidup Fragment

Fragment memiliki siklus hidup yang mirip dengan aktivitas, tetapi lebih terpisah. Siklus hidup fragment terdiri dari beberapa metode, di antaranya:



- **onAttach()** metode ini panggil pertama kali bahkan sebelum onCreate() callback dan setelah fragmen telah dipasangkan ke activity
- **onCreate()** Sistem akan memanggilnya saat membuat fragmen. Dalam implementasi, kita harus melakukan inisialisasi komponen penting dari fragmen yang ingin dipertahankan saat fragmen dihentikan sementara atau dihentikan, kemudian dilanjutkan.
- **onCreateView()** Sistem akan memanggilnya saat fragmen menggambar antarmuka pengguna (UI) untuk yang pertama kali. Untuk menggambar UI fragmen, Anda harus mengembalikan View dari metode ini yang menjadi akar layout fragmen. Hasil yang dikembalikan bisa berupa null jika fragmen tidak menyediakan UI.

- **onActivityCreated()** Metode ini dipanggil setelah Activity onCreate() Callback telah menyelesaikan eksekusi. Metode ini merupakan indikasi untuk activity tersebut telah menyelesaikan eksekusi sebelum kita mencoba mengakses dan memodifikasi elemen UI dari activity secara bebas
- **onStart()** metode yang dipanggil setelah fragmen terlihat pada activity
- **onResume()** metode ini dipanggil ketika pengguna berinteraksi dengan fragmen dalam activity setelah Activity onResume() callback. karena sebuah fragment tidak lagi digunakan, maka ia akan melewati serangkaian *reverse callback*
- **onPause()** metode ini dipanggil ketika fragmen tidak lagi berinteraksi dengan pengguna baik karena aktivitasnya sedang ditunda atau operasi fragmen mengubahnya dalam activity.
- **onStop()** metode ini dipanggil ketika fragmen tidak lagi berinteraksi dengan pengguna baik karena aktivitasnya dihentikan atau operasi fragmen mengubahnya dalam activity
- **onDestroy()** metode ini dipanggil untuk memungkinkan fragmen membersihkan resources yang terkait dengan view yang ada pada activity
- **onDestroyView()** metode ini dipanggil untuk melakukan pembersihan akhir dari status fragmen
- **onDetach()** metode ini dipanggil ke fragmen yang tidak lagi dikaitkan dengan aktivitasnya. Biasanya kita harus mengimplementasikan setidaknya metode alur onCreate(), onCreateView(), dan onPause().
- **onSaveInstanceState()** callback ini disebut dimana kita diizinkan untuk menyimpan beberapa data mengenai peristiwa fragmen tepat sebelum aplikasi di pause sehingga pengguna kembali ke aplikasi merekadengan mendapatkan data yang disimpan. Disini dibutuhkan Bundle sehingga kita dapat menyimpan data sebagai key atau nilai.

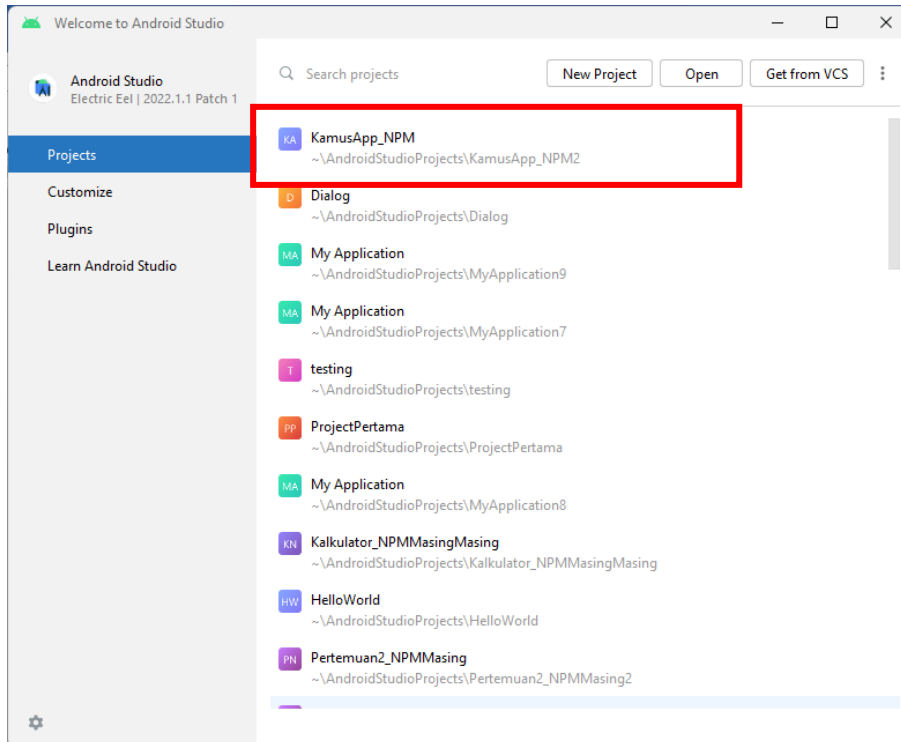
4. Kegunaan Fragmen Di Android

Sebelum Fragment diperkenalkan, kita hanya dapat menampilkan satu activity di layar pada satu waktu tertentu sehingga tidak dapat membagi layar dan mengontrol bagian yang berbeda secara terpisah. Dengan bantuan Fragment, kita dapat membagi layar di berbagai bagian dan mengontrol bagian-bagian yang berbeda secara terpisah.

Dengan menggunakan Fragment, kita dapat membentuk beberapa Fragment dalam satu activity. Fragment memiliki event, layout, dan status mereka sendiri. Hal ini memberikan fleksibilitas dan juga meniadakan limitasi dalam activity tunggal di layar pada suatu waktu.

9.4 PRAKTIKUM

- 1) Gunakan aplikasi Kamus yang telah dibuat sebelumnya pada pertemuan 6 (enam) yaitu KamusApp.



- 2) Modifikasi file XML untuk activity_main.xml hingga menjadi 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"
    android:padding="16dp">

    <Button
        android:id="@+id/btn_eng_ind1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Inggris ke Indonesia" />

    <Button
        android:id="@+id/btn_ind_eng1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Indonesia ke Inggris" />
```

```

<Button
    android:id="@+id/btn_ind_kor1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Indonesia ke Korea" />

<Button
    android:id="@+id/btn_infol"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Info Aplikasi" />

</LinearLayout>

```

3) Modifikasi MainActivity.java

```

package com.example.kamusapp_npm;

import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

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

        Button btnEngInd = findViewById(R.id.btn_eng_ind1);
        Button btnIndEng = findViewById(R.id.btn_ind_eng1);
        Button btnIndKor = findViewById(R.id.btn_ind_kor1);
        Button btnInfo = findViewById(R.id.btn_infol);

        // Navigasi ke activity translate dengan mengirim
        intent data
        btnEngInd.setOnClickListener(v ->
startTranslateActivity("engInd"));
        btnIndEng.setOnClickListener(v ->
startTranslateActivity("indEng"));
        btnIndKor.setOnClickListener(v ->
startTranslateActivity("indKor"));

        // Menampilkan dialog info aplikasi
        btnInfo.setOnClickListener(v -> showInfoDialog());
    }

    private void startTranslateActivity(String languagePair) {

```

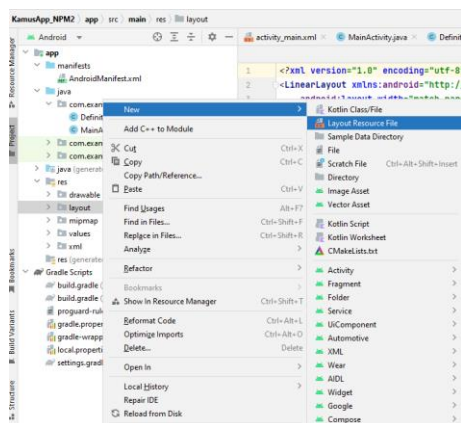
```

        Intent intent = new Intent(MainActivity.this,
TranslateActivity.class);
        intent.putExtra("LANGUAGE_PAIR", languagePair);
        startActivity(intent);
    }

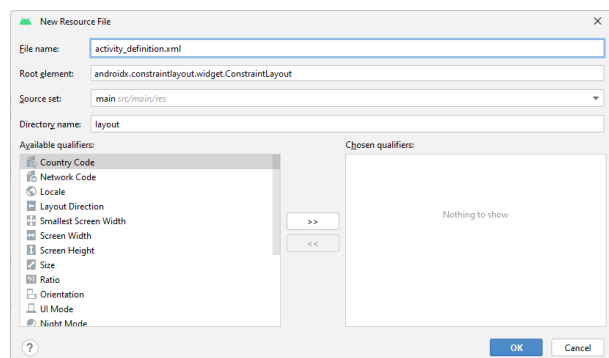
    private void showInfoDialog() {
        AlertDialog.Builder builder = new
AlertDialog.Builder(this);
        builder.setTitle("Tentang Aplikasi")
            .setMessage("Ini adalah aplikasi kamus 3 bahasa
dengan fitur fragment, dialog, dan intent.")
            .setPositiveButton("OK", (dialog, which) ->
dialog.dismiss())
            .create().show();
    }
}

```

- 4) Buat file XML bernama activity_translate.xml, dengan cara klik kanan pada layout → New → Layout Resource File. Kemudian akan muncul kotak dialog New Resource File (1). Pada bagian File name isikan nama file (2).



(1)



(2)

- 5) Kemudian ketikkan koding berikut pada file activity_translate.xml

```

<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp">

    <EditText
        android:id="@+id/edit_text_input"
        android:layout_width="match_parent"

```

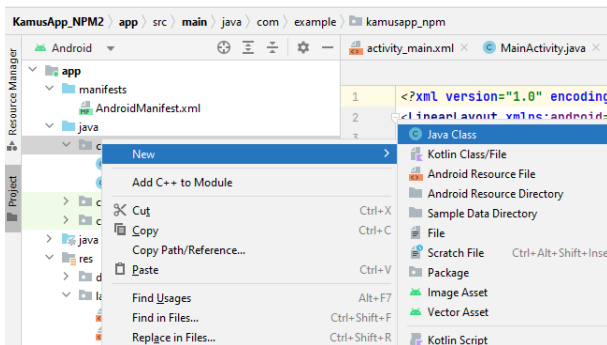
```
        android:layout_height="wrap_content"
        android:hint="Masukkan kata" />

    <FrameLayout
        android:id="@+id/fragment_container"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

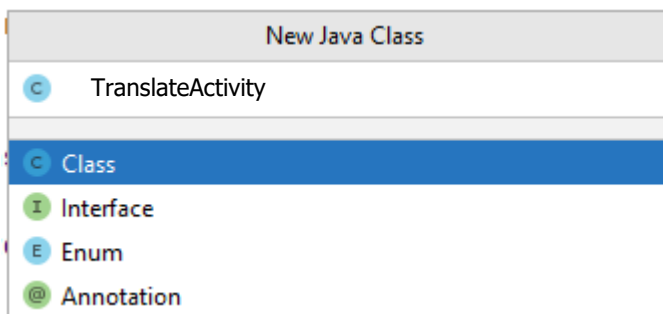
</LinearLayout>
```

6) Kemudian buat file java dengan nama TranslateActivity.java. Di file ini, kita akan mendapatkan kata dan bahasa yang dipilih dan menampilkan definisinya. Adapun cara menambahkan file tersebut yaitu :

a. Klik kanan pada folder java → New → Java Class



b. Ketikkan nama file kemudian tekan Enter



c. Ketikkan kode program berikut ke file tersebut

```
import android.os.Bundle;
import android.widget.EditText;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.FragmentTransaction;

public class TranslateActivity extends AppCompatActivity {

    private String languagePair;
```



```

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

    languagePair =
getIntent().getStringExtra("LANGUAGE_PAIR");
    EditText editTextInput =
findViewById(R.id.edit_text_input);

    // Load fragment saat input berubah
    editTextInput.setOnKeyListener((v, keyCode, event) -> {
        String input = editTextInput.getText().toString();
        loadTranslationFragment(input);
        return false;
    });
}

private void loadTranslationFragment(String input) {
    FragmentTranslate fragment =
FragmentTranslate.newInstance(input, languagePair);
    FragmentTransaction transaction =
getSupportFragmentManager().beginTransaction();
    transaction.replace(R.id.fragment_container, fragment);
    transaction.commit();
}
}

```

7) Buat file FragmentTranslate.java kemudian ketikkan kode program berikut :

```

package com.example.kamusapp_npm;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import androidx.fragment.app.Fragment;
import java.util.HashMap;

public class FragmentTranslate extends Fragment {

    private static final String ARG_INPUT = "input";
    private static final String ARG_LANGUAGE_PAIR =
"language_pair";

    private String input;
    private String languagePair;

    public FragmentTranslate() {
        // Required empty public constructor
    }
}

```

```

    }

    public static FragmentTranslate newInstance(String input,
String languagePair) {
        FragmentTranslate fragment = new FragmentTranslate();
        Bundle args = new Bundle();
        args.putString(ARG_INPUT, input);
        args.putString(ARG_LANGUAGE_PAIR, languagePair);
        fragment.setArguments(args);
        return fragment;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        if (getArguments() != null) {
            input = getArguments().getString(ARG_INPUT);
            languagePair =
getArguments().getString(ARG_LANGUAGE_PAIR);
        }
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                                Bundle savedInstanceState) {
        View view =
inflater.inflate(R.layout.fragment_translate, container,
false);
        TextView textViewOutput =
view.findViewById(R.id.text_view_output);

        HashMap<String, String> dictionary = initDictionary();
        String translatedText = getTranslation(input,
dictionary);
        textViewOutput.setText(translatedText != null ?
translatedText : "Tidak ada terjemahan");

        return view;
    }

    private HashMap<String, String> initDictionary() {
        HashMap<String, String> dictionary = new HashMap<>();
        // Kosakata contoh - Isi dengan lebih banyak kata
        dictionary.put("hello", "halo"); // Inggris ke
Indonesia
        dictionary.put("halo", "hello"); // Indonesia ke
Inggris
        dictionary.put("안녕하세요", "hello"); // Korea ke
Inggris
        dictionary.put("selamat", "안녕하세요"); // Indonesia ke
Korea
    }

```

```

        return dictionary;
    }

    private String getTranslation(String input, HashMap<String,
String> dictionary) {
        switch (languagePair) {
            case "engInd":
                return dictionary.get(input);
            case "indEng":
                return dictionary.get(input);
            case "indKor":
                return dictionary.get(input);
            // Tambahkan pasangan bahasa lainnya sesuai
kebutuhan
            default:
                return null;
        }
    }
}

```

8) Ketikkan kode berikut pada file fragment_translate.xml :

```

<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="16dp">

    <TextView
        android:id="@+id/text_view_output"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Hasil terjemahan akan tampil di sini" />
</LinearLayout>

```

9) Modifikasi kode program pada bagian AndroidManifest.xml

```

14 | <activity
15 |     android:name=".MainActivity"
16 |     android:exported="true">
17 |     <intent-filter>
18 |         <action android:name="android.intent.action.MAIN" />
19 |
20 |         <category android:name="android.intent.category.LAUNCHER" />
21 |     </intent-filter>
22 | </activity>
23 |
24 | <activity
25 |     android:name=".TranslateActivity"
26 |     android:exported="true">
27 |     <intent-filter>
28 |         <action android:name="android.intent.action.MAIN" />
29 |
30 |         <category android:name="android.intent.category.LAUNCHER" />
31 |     </intent-filter>
32 | </activity>

```

10) Penjelasan Kode

a) MainActivity.java:

- Menampilkan tombol untuk memilih pasangan bahasa dan Intent digunakan untuk berpindah ke TranslateActivity.
- Dialog ditampilkan ketika pengguna ingin melihat informasi aplikasi.

b) TranslateActivity.java:

- Mengelola input kata dan menampilkan hasil terjemahan menggunakan FragmentTranslate. Setiap kali pengguna mengetikkan sesuatu, fragment akan diperbarui dengan hasil terjemahan.

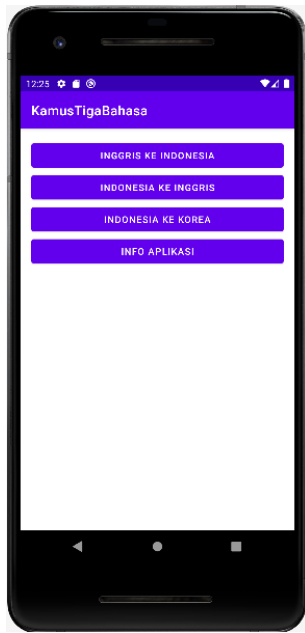
c) FragmentTranslate.java:

- Fragment ini menampilkan hasil terjemahan berdasarkan input pengguna dan bahasa yang dipilih. Kosakata bisa dikembangkan lebih lanjut di sini.

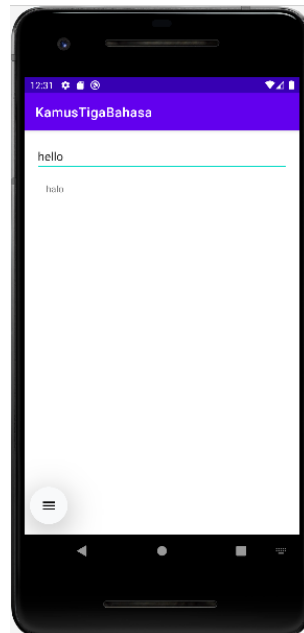
d) Dialog: Menampilkan dialog informasi sederhana tentang aplikasi.

11) Menjalankan Aplikasi

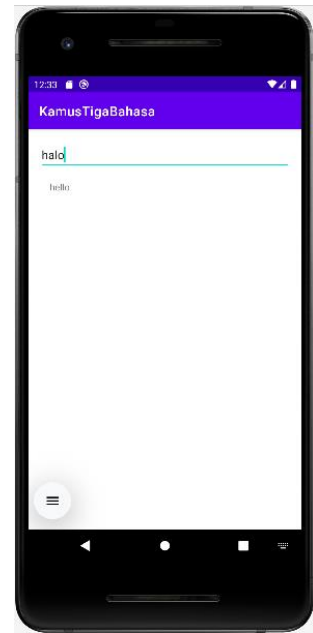
- Hubungkan perangkat Android atau buka Android Emulator.
- Klik tombol Run di Android Studio.
- Pilih perangkat yang ingin digunakan untuk menjalankan aplikasi.
- Aplikasi *KamusApp* akan ditampilkan.



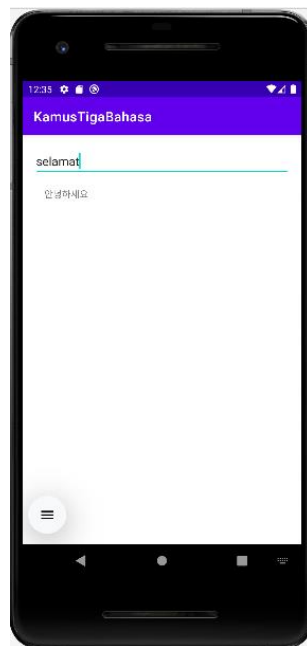
Tampilan saat aplikasi pertama kali dirunning



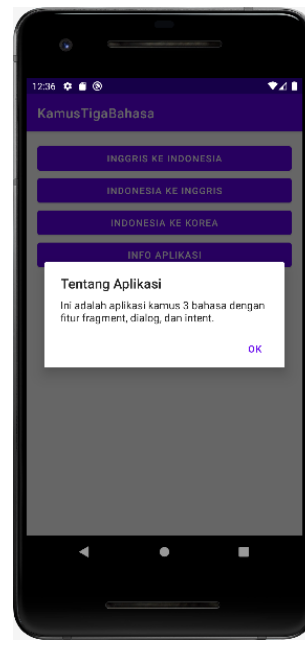
Tampilan translate Inggris ke Indonesia



Tampilan translate Indonesia ke Inggris



Tampilan translate Indonesia ke Korea



Tampilan Info Aplikasi

9.5 TUGAS

1. Demokan project tersebut hingga muncul di emulator!
2. Tambahkan minimal 10 kosakata baru ke dalam kamus yang sudah ada. Pastikan kosakata tersebut mencakup terjemahan untuk setiap kombinasi bahasa (Inggris-Indonesia, Indonesia-Korea, dll.).
3. Tugas dikumpul dalam format pdf dengan nama file : NPM_Tugaske?

4. Didalam file tersebut dituliskan :

- NPM
- Nama Lengkap
- Program Studi
- Kelas