PRAKTIKUM 1: Pengenalan Android Studio dan Membuat Aplikasi Sederhana

Tujuan:

Mahasiswa dapat menerapkan prinsip dasar OOP yang telah dipelajari sebelumnya ke dalam platform mobile (Android) dan mampu membuat sebuah aplikasi sederhana menggunakan perangkat lunak Android Studio.

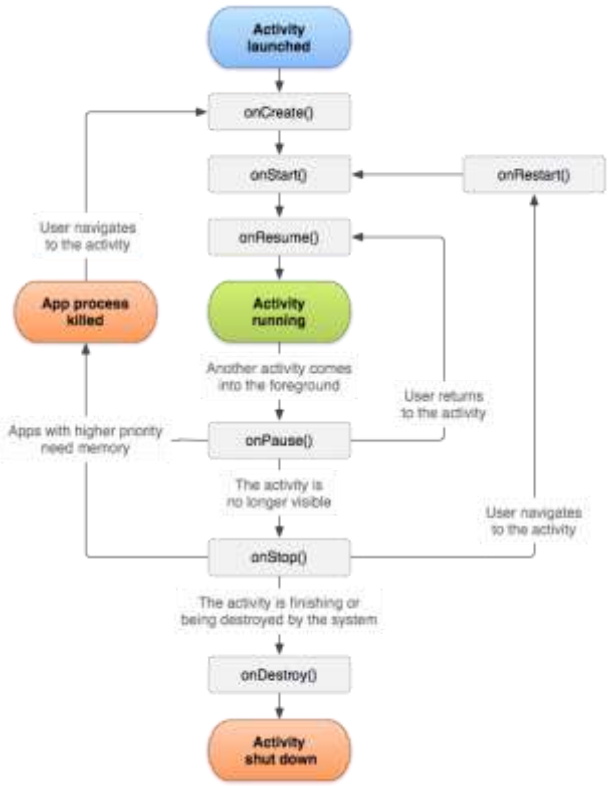
PERSYARATAN HARDWARE

- **Android Studio**
- **Android SDK**
- **Android Emulator, atau Kabel Data + USB Driver Android untuk Smartphone tertentu**

Sebelum terjun ke dalam praktek, akan dibahas sedikit tentang teori dalam pemrograman Android dan siklus hidupnya.

SIKLUS HIDUP APLIKASI ANDROID

Pertama, tidak seperti aplikasi berbasis Desktop yang telah dipelajari sebelumnya dimana suatu aplikasi (dalam bahasa Java) dimulai dari metode static main, di sini Android menerapkan sebuah siklus hidup untuk masing-masing bagian dari aplikasinya, dan di sini akan dibahas tentang siklus hidup tentang salah satu komponen utama dari aplikasi Android yakni Acitivity (yang dapat disamakan oleh sebuah JFrame dalam aplikasi Desktop).



https://developer.android.com/guide/components/images/activity_lifecycle.png

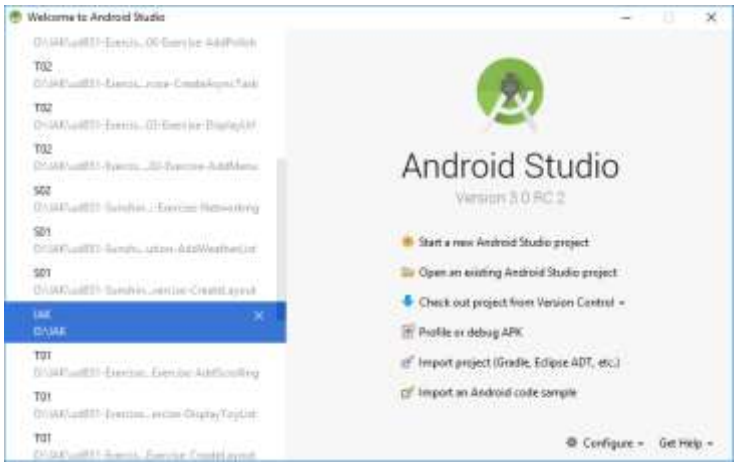
KOMPONEN UTAMA UNTUK MERANCANG APLIKASI SEDERHANA

Terdapat setidaknya 2 komponen yang wajib ada untuk menyusun sebuah Activity dalam aplikasi Android, yakni **layout** (yang dikemas dalam syntax XML) dan **file logic** dalam bahasa Java (Android Studio versi 3 ke atas juga menyediakan pilihan bahasa lain yakni Kotlin). File Layout di sini mewakili tampilan sebuah Activity, alias GUI dari aplikasi, dimana tiap file layout berelasi dengan setidaknya satu file logic yang merupakan sebuah Activity.

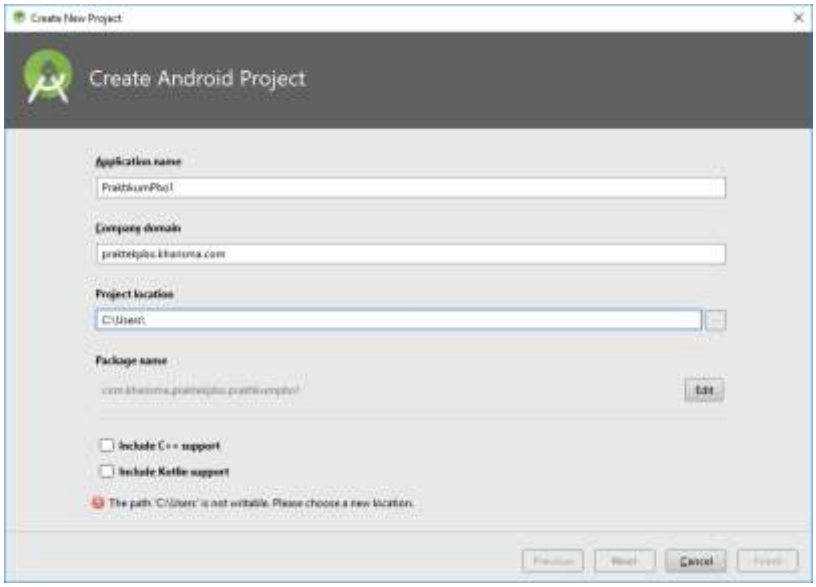
Selanjutnya, terdapat file yang berisi dari logika aplikasi, yakni file berekstensi Java yang nantinya akan berisi logika kerja dari sebuah layout. Dalam file tersebut didefinisikan sebuah kelas yang biasanya mewarisi kelas **Activity** atau **AppCompatActivity**. File logic ini setidaknya wajib melakukan **override** pada metode **onCreate** (perhatikan kembali siklus hidup) dimana metode mewakili **entry point** (titik mulai) dari Activity.

MARI KITA MULAI

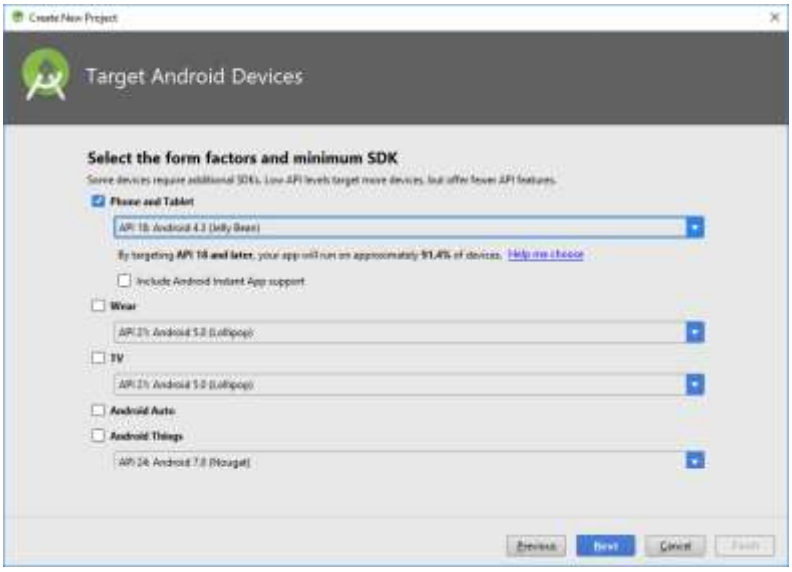
1. Jalankan aplikasi Android Studio Anda.
2. Kemudian buat project baru. (**Start a new Android Studio project**)



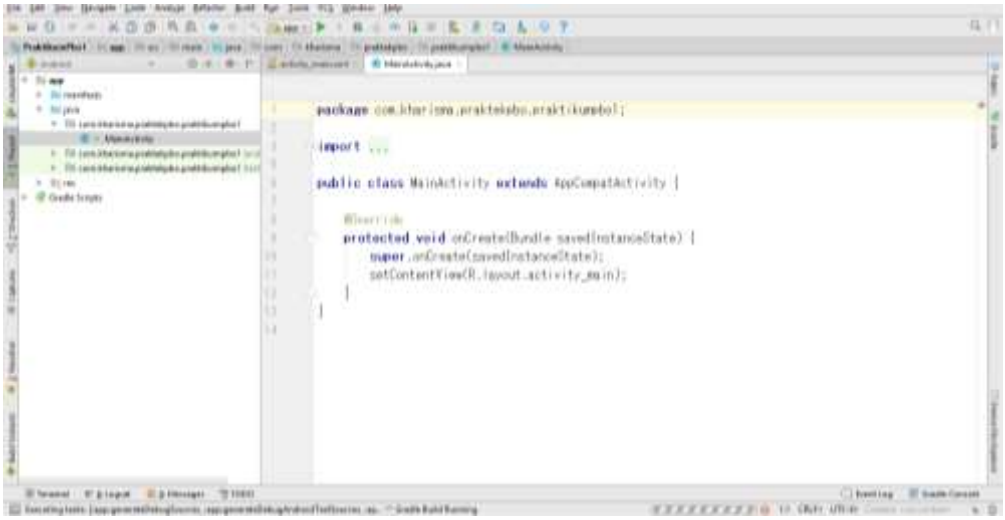
3. Masukkan nama aplikasi, dan nama website (ini berfungsi sebagai nama package untuk project), ubah lokasi project jika perlu, lalu klik Next.



4. Selanjutnya akan dilakukan pemilihan target pembuatan aplikasi (smartphone, TV, dll). Di sini, pastikan hanya **Phone and Tablet** yang dicentang. Selanjutnya, akan dipilih target SDK untuk aplikasi yang dibuat. Berdasarkan perkiraan dan pengamatan saat ini, rata-rata smartphone yang digunakan memiliki OS JellyBean versi 4.1 sebagai versi terendah yang beredar, jadi silakan memilih API 16.



5. Selanjutnya pilih **Empty Activity**, lalu klik Next.
6. Lalu tentukan nama **Activity** pertama yang akan dibuat (umumnya **MainActivity** secara default). Untuk kemudahan, pastikan **Generate Layout File** dan **Backwards Compatibility (AppCompat)** dicentang, lalu klik Finish.
7. Android Studio kemudian akan menyiapkan Project, dan menampilkan Layout (xml) dan Activity (java) pertama yang akan dikerjakan.



8. Setelah proses Gradle Build selesai, silakan buka activity_main.xml (jika menggunakan nama MainActivity saat pembuatan project).

BASIC LAYOUT

Jika terdapat kata **ConstraintLayout** di sana, silakan hapus seluruh isi file.

Dalam sebuah file layout membutuhkan setidaknya:

1. Parent View/ViewGroup (bertipe LinearLayout, RelativeLayout, dsb) yang berfungsi sebagai container yang menampung widget/view.
2. Widget/View, yang merujuk kepada komponen-komponen tampilan seperti TextView, EditText, Button, dll.

Syntax untuk menyusun sebuah layout sederhana:

```
<Tipe_Parent
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"
/>
<!—Jika Widget tidak memiliki Child-->
<!—Tambahkan ID jika dibutuhkan -->
        <Tipe_Widget
        android:id="@+id/id_widget"
        android:layout_width="wrap_content"
         android:layout_height="wrap_content"
        />
```

</Tipe_Parent>

Untuk menentukan ukuran suatu komponen, perhatikan layout_width dan layout_height. Value yang umumnya digunakan adalah **wrap_content** (tinggi dan lebar seadanya), **match_parent** (menyesuaikan dengan komponen induk), maupun diset manual (contohnya: **200dp**, atau **200px**, atau **200in**, dll).

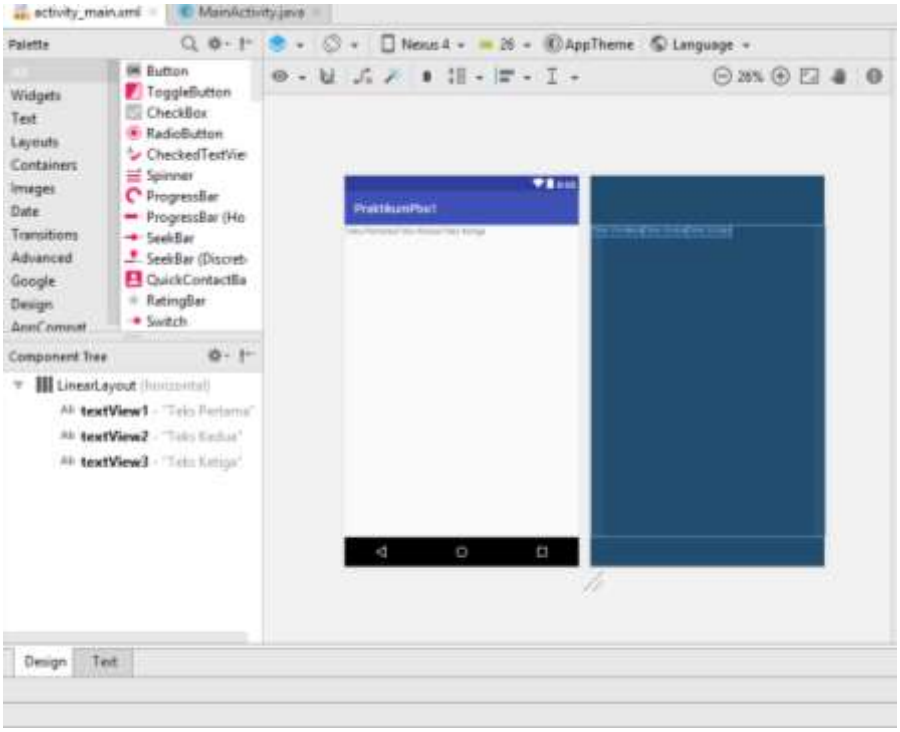
Sebagai percobaan dasar, silakan copy dan paste kode ini ke dalam activity_main.xml.

```
<?xml version="1.0" encoding="utf-8"?>
<!-- TODO: (1) Pilih Orientasi Layout sesuai selera -->
<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="com.kharisma.praktekpbo.praktikumpbo1.MainActivity">

    <!-- TODO: (2) Silakan bereksperimen dengan widget-widget yang ada -->
    <!-- TODO: (3) Tambahkan widget yang dibutuhkan seperlunya -->
    <!-- TODO: (4) Edit properti dari widget sesuai selera -->
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Teks Pertama" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Teks Kedua" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Teks Ketiga" />

</LinearLayout>
```

Dan berikut ini beberapa properti widget yang tidak dicantumkan dalam kode (silakan gunakan fitur IntelliSense, yakni Ctrl+Space untuk menampilkan pilihan value, dan TAB untuk AutoComplete):

- ➔ Untuk LinearLayout **android:orientation=""**
- ➔ Untuk semua jenis View dan Widget: **android:margin=""**, **android:padding=""**
- ➔ Untuk beberapa jenis komponen: **android:background=""**, **android:textColor=""**, **android:onClick=""**

Untuk membantu perancangan layout, Android Studio menyediakan fitur Preview untuk memperlihatkan bagaimana hasil dari layout yang dibuat. Dan jika lebih memilih untuk membuat komponen dengan proses Drag and Drop, terdapat alternatif menyusun Layout yakni mode **Design**, yang berbeda dengan metode **Text**. Tidak hanya itu, di mode Design, juga terdapat **Pallette** yang menampilkan daftar keseluruhan Widget dan View yang dapat digunakan.



PEMROGRAMAN DASAR ANDROID

Sebuah Activity hanya akan bekerja sepenuhnya jika sudah diberikan logika. Dan logika tersebut hanya dapat didefinisikan melalui file Java yang telah meng-extend class Activity atau AppCompatActivity.

Berikut ini adalah isi kode dari sebuah Activity kosong.

```
package com.kharisma.praktekpbo.praktikumpbo1;

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

public class MainActivity extends AppCompatActivity {

    // TODO: (4) Deklarasikan Referensi Object dari Widget yang digunakan
    // TODO: (5) Deklarasikan variabel/atribut seperlunya

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

```
        // TODO: (6) BOILERPLATE: Letakkan inisialisasi object View/Widget di sini
        // TODO: (7) Do something else
    }

    // TODO: (8) Jika ada fungsi onClick, definisikan di sini
    // TODO: (9) Buat fungsi tambahan seperlunya
}
```

Perlu diingat kembali bahwa sebuah Activity memulai aktivitas dari metode onCreate, dimana di situ terdapat metode penting yang bernama **setContentView** yang menerima parameter berupa id layout yang telah dibuat sebelumnya. Metode ini memastikan bahwa Activity akan memuat tampilan berdasarkan isi dari file layout yang ditunjuk oleh parameter.

BERMAIN DENGAN WIDGET

Sebelum sebuah widget siap menerima input dan berinteraksi dengan logika program, maka diperlukan sebuah referensi yang ditampung oleh objek dalam bahasa Java.

(This is called Boilerplate Codes by some references out there)

Contoh kasus:

1. Jika ingin merubah teks dari sebuah TextView dengan nilai dari komponen EditText, maka diperlukan 2 buah objek referensi untuk masing-masing widget.

   // Contoh Referensi Objek

   TextView tvDisplay;

   EditText etInput;

2. Selanjutnya, kedua objek tersebut perlu diinisialisasi dengan nilai referensi yang berupa id dari widget yang ingin dikendalikan. (inisialisasi sebaiknya dilakukan di dalam metode onCreate)

   // Contoh Inisialisasi Objek

   tvDisplay = (TextView) findViewById(R.id.textView1);

   etInput = (EditText) findViewById(R.id.etInput);

   // Metode findViewById mengembalikan objek View, sehingga perlu dilakukan **type-casting** (Gradle versi terbaru umumnya sudah tidak membutuhkan type-casting, terlihat pada pesan Compiler Warning)

3. Dikarenakan objek telah diinisialisasi, maka program java pada Activity siap mengolah data dari widget tersebut

Metode dasar yang dapat digunakan secara umum pada widget TextView dan EditText yakni **getText()** dan **setText()**. (Metode **getText()** mengembalikan **Editable** dan harus dikonversi menggunakan **toString()**).

MENANGANI EVENT ONCLICK

Bagaimana jika aplikasi perlu menangani perintah click ketika user menyentuh widget? Untuk hal ini diperlukan bantuan **onClickListener** (konsepnya mirip dengan **Swing & AWT**). Di dalam Android, terdapat setidaknya 3 cara untuk menangani event onClick. Dalam pembahasan ini hanya akan dibahas 2 saja.

Cara Pertama:

1. Pada file layout (xml), pada widget yang ingin dipasangkan event onClick, silakan tambahkan properti **onClick="namaFungsi"**.
2. Kemudian pada file Activity (java) deklarasikan sebuah fungsi bernilai kembalian **void** dengan nama **namaFungsi** yang menerima satu parameter bertipe **View**.
3. Di dalam fungsi tersebut silakan tambahkan perintah yang akan dikerjakan saat widget tersebut diklik.

Cara Kedua:

1. Pada file Activity, inisialisasi objek dari widget yang akan diberi onClickListener.
2. Lalu pada objek tersebut, panggil metode **setOnClickListener** yang menerima satu parameter yakni **View.OnClickListener**.
3. Setelah itu, implementasi method-methodnya (Tips: Letakkan kursor pada lokasi error, lalu gunakan Alt+Enter dan pilih **Implement methods...**)
4. Akan muncul method **onClick** yang telah di-override, dan di sanalah akan dimasukkan perintah yang dieksekusi saat widget diklik.

APA YANG AKAN DIKERJAKAN?

Pada praktikum kali ini, Anda akan membuat sebuah aplikasi sederhana yang dapat menerima inputan berupa ketikan dari user, dan setidaknya sebuah Button yang akan memulai pemrosesan input lalu menampilkan hasilnya.

Algoritma:

- ➔ Aplikasi dilaunch, Activity menampilkan layout
- ➔ User memasukkan inputan berupa ketikan di beberapa field
- ➔ User menekan Button, lalu inputan diproses
- ➔ Hasil proses kemudian ditampilkan segera

Dalam praktikum ini, Anda diharapkan untuk berkreasi sendiri memilih topik aplikasi sederhana yang akan dibuat. Tipe aplikasi yang pas untuk topik ini seperti:

- ➔ Kalkulator sederhana
- ➔ Konversi mata uang
- ➔ Menghitung luas/volume bangun datar dan bangun ruang
- ➔ Penerapan fungsi matematika/akuntansi/ekonomi
- ➔ Dan lain-lain yang membutuhkan perhitungan

Dan untuk ke depannya, Anda diharapkan agar selalu ingat menerapkan prinsip OOP dengan membuat POJO (Plain Old Java Object) sebagai perwakilan entitas data yang Anda gunakan selama merancang aplikasi Android.

Lupa/Belum tau tentang POJO?

Here's an example:

```
public class Pojo {

    private String id;
    private String name;
    private int value;

    public Pojo() {
        id = "";
        name = "Nonamed";
        value = 0;
    }

    public Pojo(String id, String name, int value) {
        this.id = id;
        this.name = name;
        this.value = value;
    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getValue() {
        return value;
    }

    public void setValue(int value) {
        this.value = value;
    }
}
```

Anda diharapkan untuk selalu memodelkan data ke dalam bentuk seperti ini ke depannya.

BONUS METHODS

```
// Menampilkan pesan notifikasi sementara di bagian bawah layar Device
// Android tidak dapat menggunakan System.out.println
Toast.makeText(this, "Hello World", Toast.LENGTH_LONG).show();


// Menutup/mengakhiri Activity
finish();


// Mengubah Judul Activity
setTitle("Judul");
```