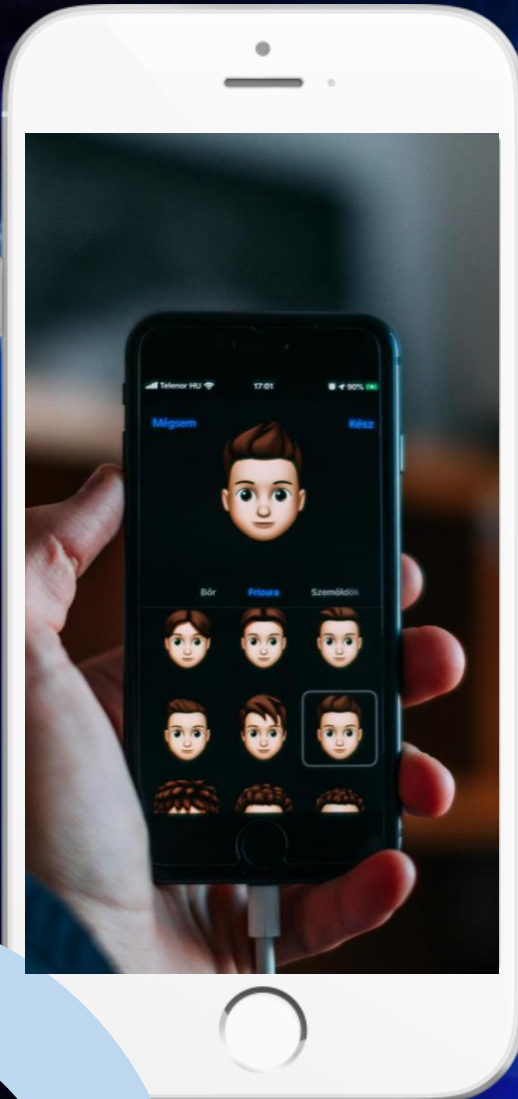




UNIVERSITAS
BUDI LUHUR

Alert Box, Conditional Statement and EditText

Presented By Putri Hayati, S.ST, M.Kom



[Read more](#)



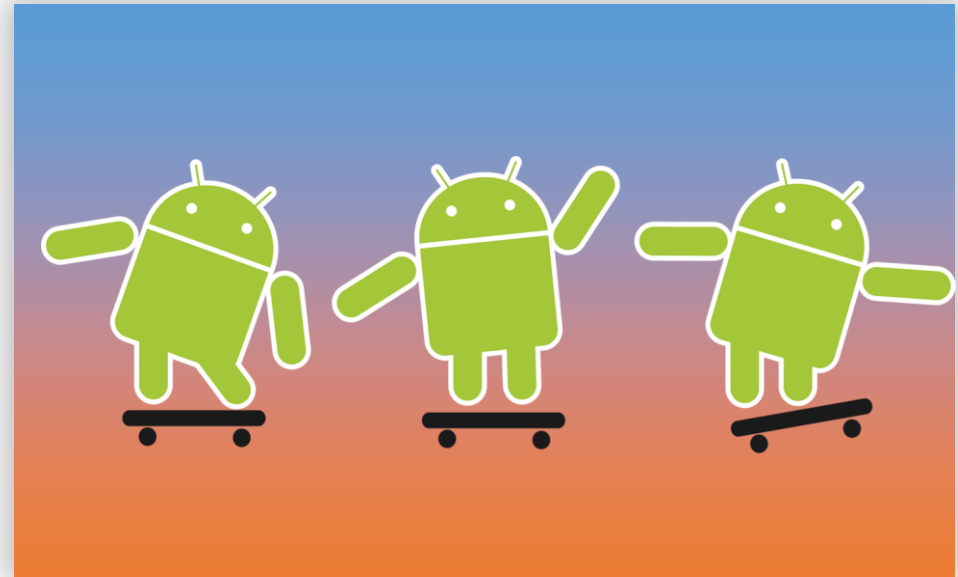
Kategori

6.0 Alert Box

6.1 EditText

6.2 Conditional Statement

6.3 Praktikum

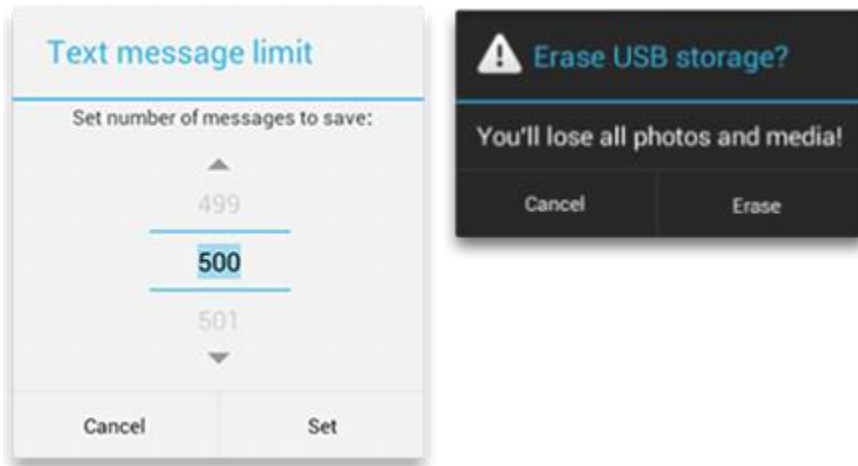




Alert Box



Alert Box - Dialog - Pop Up



Dialog adalah jendela kecil yang meminta pengguna untuk membuat keputusan atau memasukkan informasi tambahan. Dialog tidak memenuhi layar dan biasanya digunakan untuk peristiwa modal yang mengharuskan pengguna untuk melakukan tindakan sebelum bisa melanjutkan

Class Dialog adalah class dasar untuk dialog, tetapi Anda harus menghindari pembuatan instance Dialog secara langsung. Sebagai gantinya, gunakan salah satu dari subclass berikut:

- **AlertDialog**

Dialog yang bisa menampilkan judul, maksimum tiga tombol, daftar item yang dapat dipilih, atau tata letak khusus.

- **DatePickerDialog** atau **TimePickerDialog**

Dialog berisi UI yang sudah didefinisikan dan memungkinkan pengguna memilih tanggal atau waktu.

Membuat Fragmen Dialog

Anda bisa menghasilkan beragam desain dialog, termasuk tata letak kustom dan yang dijelaskan dalam panduan desain Dialog, dengan memperluas `DialogFragment` dan membuat `AlertDialog` dalam metode callback `onCreateDialog()`.

Misalnya, berikut `AlertDialog` dasar yang dikelola dalam `DialogFragment`:

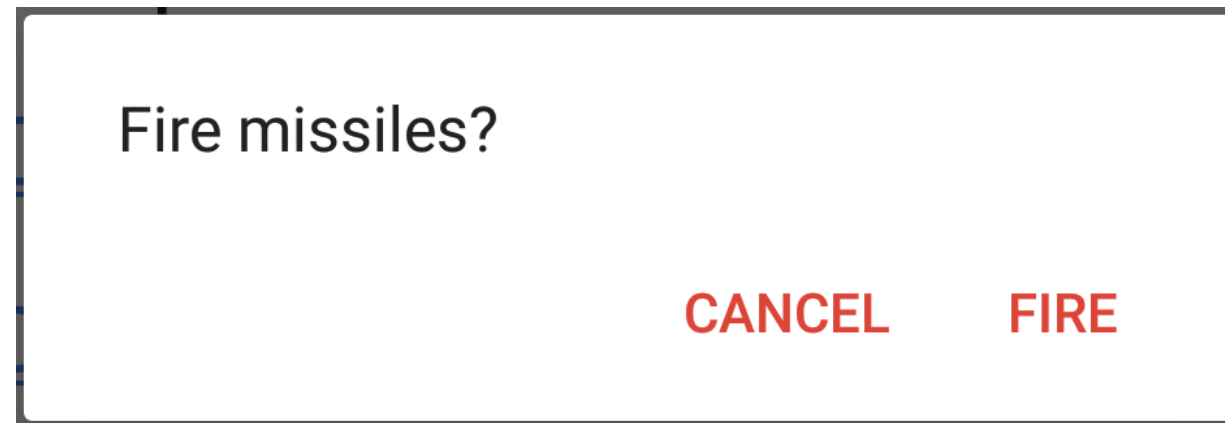
```
public class FireMissilesDialogFragment extends DialogFragment {
    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        // Use the Builder class for convenient dialog construction
        AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
        builder.setMessage(R.string.dialog_fire_missiles)
            .setPositiveButton(R.string.fire, new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    // FIRE ZE MISSILES!
                }
            })
            .setNegativeButton(R.string.cancel, new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    // User cancelled the dialog
                }
            });
        // Create the AlertDialog object and return it
        return builder.create();
    }
}
```

Membuat Fragmen Dialog

Sekarang, bila Anda membuat instance class ini dan memanggil `show()` pada objek itu, dialog akan muncul seperti yang ditampilkan dalam gambar 1.

Bagian berikutnya menjelaskan selengkapnya penggunaan API `AlertDialog.Builder` untuk membuat dialog.

Bergantung pada seberapa rumit dialog tersebut, Anda dapat mengimplementasikan berbagai metode callback lain dalam `DialogFragment`, termasuk semua metode siklus proses fragmen dasar.



Membuat Dialog Peringatan

Class `AlertDialog` memungkinkan Anda membuat berbagai desain dialog dan sering kali satu-satunya class dialog yang akan Anda perlukan. Seperti yang ditampilkan dalam gambar 2, ada tiga area pada dialog peringatan:

- Judul

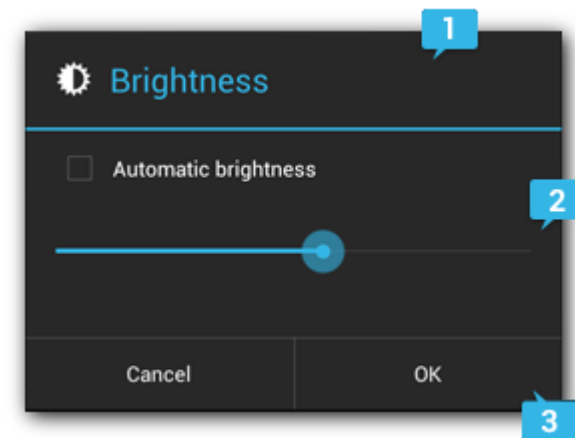
Area ini opsional dan hanya boleh digunakan bila area isi berisi pesan terperinci, daftar, atau tata letak khusus. Jika Anda perlu menyatakan pesan atau pertanyaan sederhana (seperti dialog dalam gambar 1), Anda tidak memerlukan judul.

- Area isi

Area ini dapat menampilkan pesan, daftar, atau tata letak khusus lainnya.

- Tombol tindakan

Tidak boleh ada lebih dari tiga tombol tindakan dalam dialog.



Membuat Dialog Peringatan

Class AlertDialog.Builder menyediakan **API** yang memungkinkan Anda membuat AlertDialog dengan jenis isi ini, termasuk tata letak khusus.

Untuk membuat AlertDialog:

```
// 1. Instantiate an <a href="/reference/android/app/AlertDialog.Builder.html">AlertDialog.Builder</a></code> with its constructor
AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());

// 2. Chain together various setter methods to set the dialog characteristics
builder.setMessage(R.string.dialog_message)
    .setTitle(R.string.dialog_title);

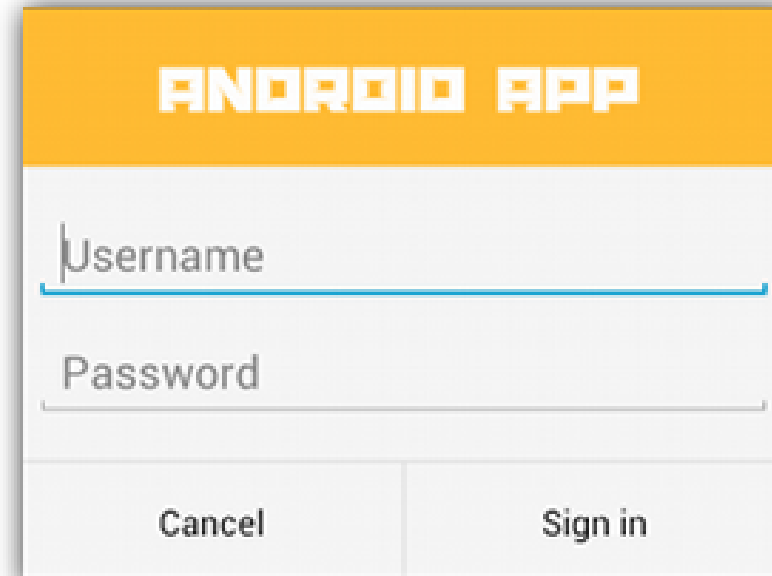
// 3. Get the <a href="/reference/android/app/AlertDialog.html">AlertDialog</a></code> from <a href="/reference/android/app/AlertDialog.Builder.html#create()">create()</a></code>
AlertDialog dialog = builder.create();
```


Membuat Tata Letak Khusus

Jika Anda menginginkan tata letak khusus dalam dialog, buat tata letak dan tambahkan ke AlertDialog dengan memanggil `setView()` pada objek `AlertDialog.Builder`.

Secara default, tata letak khusus akan mengisi jendela dialog, tetapi Anda tetap dapat menggunakan metode `AlertDialog.Builder` untuk menambahkan tombol dan judul.

Misalnya, berikut adalah file tata letak untuk dialog dalam Gambar berikut :



The image shows a screenshot of an Android application's alert dialog. The dialog has a yellow header bar with the text "ANDROID APP" in white. Below the header, there are two text input fields. The first field is labeled "Username" and the second field is labeled "Password". At the bottom of the dialog, there are two buttons: "Cancel" on the left and "Sign in" on the right. The dialog is centered on the screen.

res/layout/dialog_signin.xml

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
    <ImageView
        android:src="@drawable/header_logo"
        android:layout_width="match_parent"
        android:layout_height="64dp"
        android:scaleType="center"
        android:background="#FFFFBB33"

    android:contentDescription="@string/app_name"
    "/>
    <
```

EditText

```
        android:id="@+id/username"
        android:inputType="textEmailAddress"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:layout_marginLeft="4dp"
        android:layout_marginRight="4dp"
        android:layout_marginBottom="4dp"
        android:hint="@string/username" />
```

<EditText

```
        android:id="@+id/password"
        android:inputType="textPassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="4dp"
        android:layout_marginLeft="4dp"
        android:layout_marginRight="4dp"
        android:layout_marginBottom="16dp"
        android:fontFamily="sans-serif"
        android:hint="@string/password"/>
```

```
</LinearLayout>
```



EditText

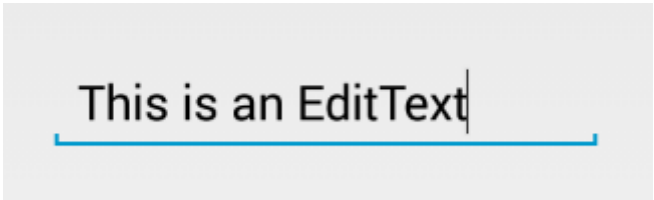


EditText

EditText adalah cara standar untuk memasukkan teks di aplikasi Android. Jika pengguna diminta untuk memasukkan suatu teks, maka View ini lah yang menjadi sarana utama untuk melakukannya.

EditText dapat kita tambahkan ke dalam layout dengan kode XML berikut:

```
<EditText  
    android:id="@+id/plain_text_input"  
    android:layout_height="wrap_content"  
    android:layout_width="match_parent"  
    android:inputType="text"/>
```



This is an EditText

Mengambil Teks dari EditText

Aturan-aturan khusus untuk mendapatkan masukan dari pengguna dengan mengatur atribut android:inputType. Dengan atribut ini kita bisa meminta informasi nomor telepon, password, angka, dll. Berikut input type paling umum:

Tipe	Deskripsi
textUri	Teks yang akan dipakai sebagai URI
textEmailAddress	Teks yang akan dipakai sebagai alamat email
textPersonName	Teks yang akan sebagai nama orang
textPassword	Teks yang akan dipakai sebagai password dan dirahasiakan
number	Pengguna hanya dapat memasukkan angka
phone	Untuk memasukkan nomor telepon
date	Untuk mengisi tanggal
time	Untuk mengisi waktu (jam)
textMultiline	Untuk mengisi data teks yang terdiri dari beberapa baris

Mengambil teks yang dimasukkan lewat **EditText** dapat kita lakukan di Java dengan kode :

```
EditText simpleEditText =  
(EditText)  
findViewById(R.id.et_simple);  
String strValue =  
simpleEditText.getText().toString()  
();
```



Conditional Statement



Conditional Statement

IF ELSE adalah suatu perintah **conditional** atau percabangan, yang digunakan dalam **bahasa pemrograman Java**, untuk memeriksa suatu kondisi, lalu akan menjalankan program sesuai dengan syarat/kondisi yang akan ditampilkan. Jika **(if) kondisi bernilai benar (true)**, maka program akan menjalankan pernyataan **(statement) A**. Namun **jika tidak (else if)**, maka program akan menjalankan pernyataan **(statement) B**. Atau jika kedua kondisi **(if dan else if salah, = else)** tidak ada yang benar, maka program akan menjalankan pernyataan **(statement) C**.

Contoh syntax :

```
if ( syarat kondisi ) {  
    //menjalankan statement A  
}else if ( syarat kondisi ) {  
    //menjalankan statement B  
}else {  
    //menjalankan statement c  
}
```

Contoh Syntax tanpa bracket :

```
if ( syarat kondisi )  
    //menjalankan statement A  
else if ( syarat kondisi )  
    //menjalankan statement B  
else  
    //menjalankan statement c
```


Conditional Statement

Condition is true

```
int number = 5;  
  
if (number > 0) {  
    // code  
}  
else {  
    // code  
}  
// code after if...else
```

Condition is false

```
int number = 5;  
  
if (number < 0) {  
    // code  
}  
else {  
    // code  
}  
// code after if...else
```

STATEMENT IF

Statement If ini digunakan jika pernyataan untuk mengeksekusi beberapa kode hanya jika kondisi tersebut itu adalah benar.

Struktur Logika If

```
if (kondisi Benar);
```

STATEMENT IF ELSE

Statement if else digunakan untuk mengeksekusi beberapa kode jika kondisi benar dan kode lain jika kondisi salah. Biasanya, pilihannya itu hanya ada 2 pilihan.

Struktur Logika If Else

```
if Kondisi Benar;  
else  
kondisi salah;
```

STATEMENT CASE

Secara sederhana, statement percabangan Case mirip seperti struktur If Then Else yang berulang. Jika di dalam If Then Else kita memiliki format penulisan seperti berikut:

```
IF (kondisi1) THEN  
    (kode program 1)  
ELSE IF (kondisi2)  
THEN  
    (kode program 2)  
ELSE IF (kondisi3)  
THEN  
    (kode program 3)
```

Maka di struktur CASE, format penulisannya seperti ini:

```
CASE (expression) OF  
    kondisi 1:  
    (kode program1);  
    kondisi 2:  
    (kode program2);  
    kondisi 3:  
    (kode program3);  
end;
```



Praktikum



Membuat Alert Dialog di Android Studio

1. Pertama - tama silahkan kalian buat sebuah project baru di **Android Studio** atau IDE
2. Project name : **Nim_AlertDialog**
3. Pilih **“Empty Activity”**
4. **Next**
5. **app/res/layout/activity_main.xml**
6. **Dan seterusnya**

XML Layout File

Buka file `activity_main.xml` dalam projek android studio kalian dan tambahkan widget button di dalamnya. Berikut ini adalah kode lengkap untuk file `activity_main.xml`

`app/res/layout/activity_main.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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">

    <Button
        android:onClick="openAlertDialog"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Open Alert Dialog"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Java Activity File

Sekarang buka file **MainActivity** dalam folder java pada projek android studio kalian. Tambahkan kode dibawah ini untuk membuat alert dialog yang menampilkan icon aplikasi, positive dan negative button yang bisa menampilkan toast ketika di klik. Berikut ini adalah kode lengkap untuk file **MainActivity.java**

[app/java/com.androidrion.alertdialog/MainActivity.java](#)



Java Activity File

```
package com.androidrion.alertdialog;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import android.content.DialogInterface;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

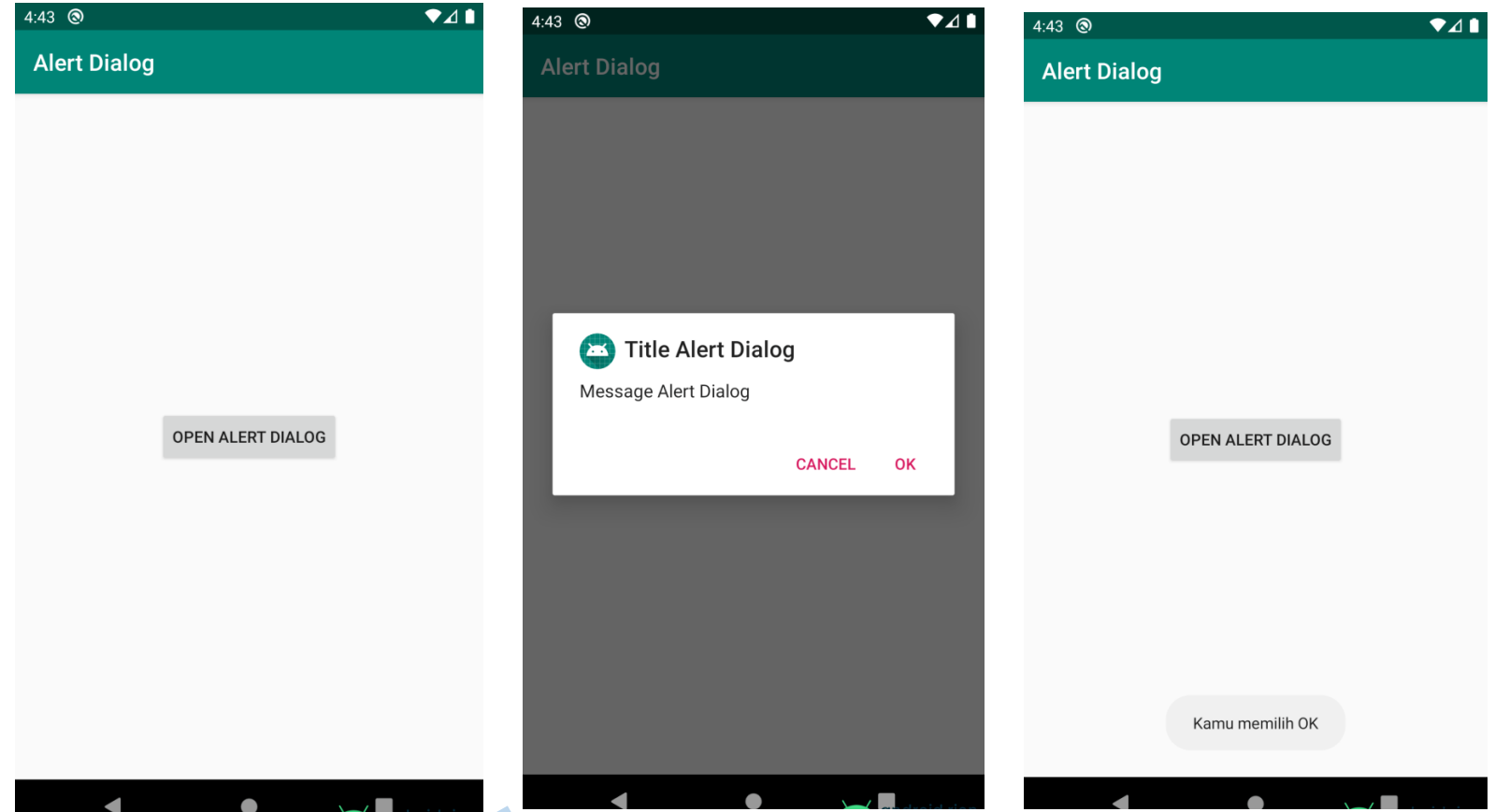
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void openAlertDialog(View view) {
        new AlertDialog.Builder(this)
            .setIcon(R.mipmap.ic_launcher)
            .setTitle("Title Alert Dialog")
            .setMessage("Message Alert Dialog")
            .setPositiveButton("OK", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(MainActivity.this, "Kamu memilih OK",
Toast.LENGTH_SHORT).show();
                }
            })
    }
}
```


Java Activity File

```
.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {  
    @Override  
    public void onClick(DialogInterface dialog, int which) {  
        Toast.makeText(MainActivity.this, "Kamu memilih  
Cancel", Toast.LENGTH_SHORT).show();  
    }  
})  
.show();  
}  
}
```

Run 'app'

Sekarang jalankan proyek android studio kalian. Pada tampilan awal aplikasi kalian bisa melihat tombol yang berada di tengah aplikasi. Jika kalian mengklik tombol tersebut, maka aplikasi akan menampilkan dialog yang memiliki aksi OK dan CANCEL. Jika kode di atas tidak terjadi masalah, maka aplikasi akan terlihat seperti pada gambar dibawah ini.



Any Question ?

