Modul Hari Pertama

Basic Android Application Development

Hanya untuk penggunaan pribadi dan proses pembelajaran di kelas Basic Android Application Development Codepolitan.com

1. Membuat Layout Aplikasi Android

Layout merupakan salah satu elemen utama sebuah aplikasi Android. Modul ini akan membahas bagaimana cara membuat layout sebuah aplikasi Android di Android Studio.

Pembuatan layout aplikasi Android dapat dilakukan dengan dua cara yaitu melalui XML dan Java. Pembuatan layout dengan XML dibuat dengan menuliskan komponen-komponen yang sebelumnya sudah dibuat oleh tim Android Google sehingga bisa kita langsung pakai. Selain menggunakan XML kita juga bisa membuat layout langsung menggunakan Java.

2. Membuat Layout dengan XML

Di kelas Basic Android Application Development ini kita hanya akan mempelajari cara pembuatan layout menggunakan XML. Komponen utama pembuatan layout menggunakan XML adalah View dan ViewGroup.

2.1. View

View merupakan komponen yang menentukan objek apa yang dilihat oleh pengguna. Contoh View adalah `Button`, `TextView`, dan `EditText`.

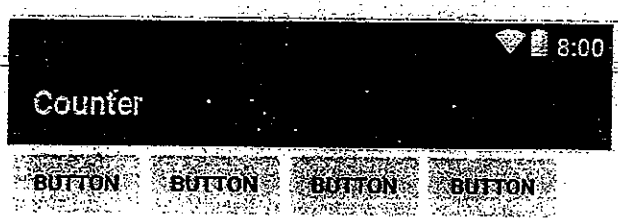
2.2. ViewGroup

ViewGroup merupakan komponen yang dapat memiliki beberapa View dan memposisikannya sesuai kebutuhan. Ada banyak jenis ViewGroup namun yang akan dibahas secara mendalam dikelas ini adalah LinearLayout dan RelativeLayout.

LinearLayout

LinearLayout merupakan sebuah ViewGroup yang hanya dapat memposisikan komponen didalamnya dengan cara menyusun kesamping (*horizontal*) atau menyusun ke bawah (*vertical*).

Horizontal LinearLayout





Mari kita belajar membuat LinearLayout yang memposisikan komponen-komponen di dalamnya dalam bentuk menyusun ke samping.

Buka file `activity_main.xml` lalu ubah ViewGroup utamanya menjadi LinearLayout:

```
<?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="com.codepolitan.counter.MainActivity"
    android:orientation="horizontal"
    >

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>
```

Baris pertama adalah kode yang wajib selalu ada di setiap file `.xml` untuk layout.

Lalu di bawahnya kita mendeklarasi sebuah LinearLayout. Setiap ViewGroup paling luar wajib menambahkan kode:

```
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"
```

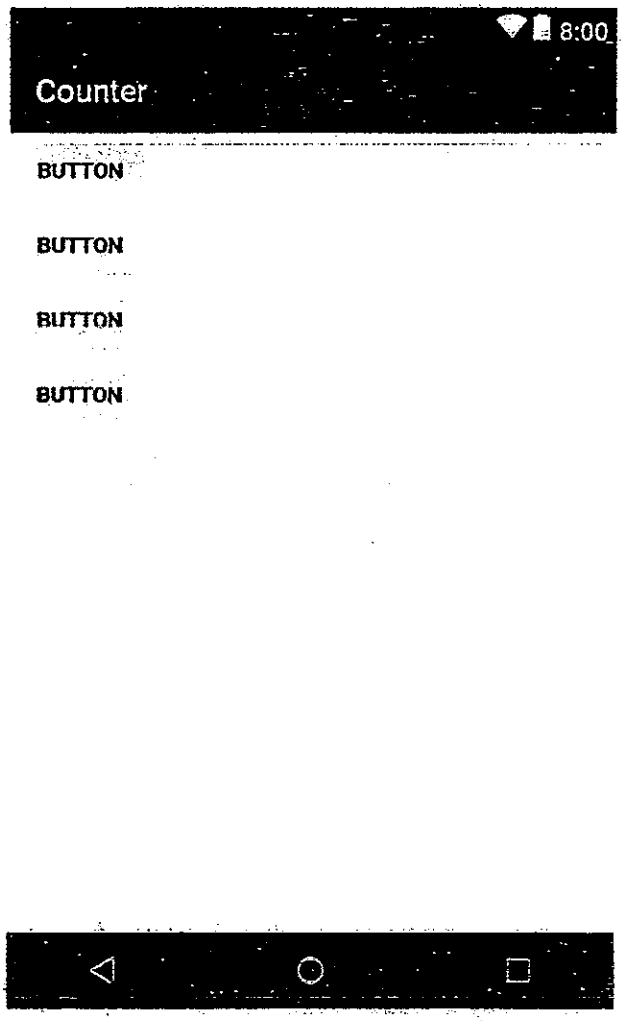
Atribut `android:layout_width` dan `android:layout_height` menentukan tinggi dan lebarnya. Karena di sini kita menentukannya sebagai `match_parent`, maka LinearLayout ini akan setinggi dan selebar layar.

Di dalam LinearLayout kita menambahkan empat View berjenis `Button`. Perhatikan di sini bahwa apabila kita ingin menambahkan suatu komponen di dalam ViewGroup lain, maka kita harus menuliskannya seperti ini:

```
<LinearLayout ...> // Tag pembuka
    <Button /> // komponen isi, karena tidak ada komponen lain di dalam
  Button
    <Button /> // maka kita langsung menutupnya dengan tanda /> di akhir
  penulisan.
</LinearLayout> // Tag penutup, perhatikan tanda "/"
```

Sebuah LinearLayout jika tidak memiliki atribut `android:orientation` maka secara otomatis akan memposisikan isinya menyusun ke samping.

Vertical LinearLayout



Mari kita belajar membuat LinearLayout yang memposisikan komponen-komponen di dalamnya dalam bentuk menyusun ke bawah.

Buka file `activity_main.xml` lalu ubah ViewGroup utamanya menjadi LinearLayout:

```
<?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="com.codepolitan.counter.MainActivity"
    android:orientation="vertical"
    >

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>
```

Perhatikan bahwa kode di atas memiliki konten yang sama kecuali pada `android:orientation` dimana kita menuliskan `vertical`.

Layout Weight

Ada sebuah fitur unik yang hanya ada di LinearLayout yaitu Layout Weight. Dengan menambahkan `android:layout:weight` ke sebuah komponen, kita bisa memberikan sisa ruang yang ada kepada komponen-komponen tersebut. Belum terbayang? Mari kita coba.

Buka `activity_main.xml` yang didalamnya telah ada LinearLayout vertikal. Berikan atribut `android:layout_weight="1"` ke Button pertama.

```
<?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="com.codepolitan.counter.MainActivity"
    android:orientation="vertical"
    >

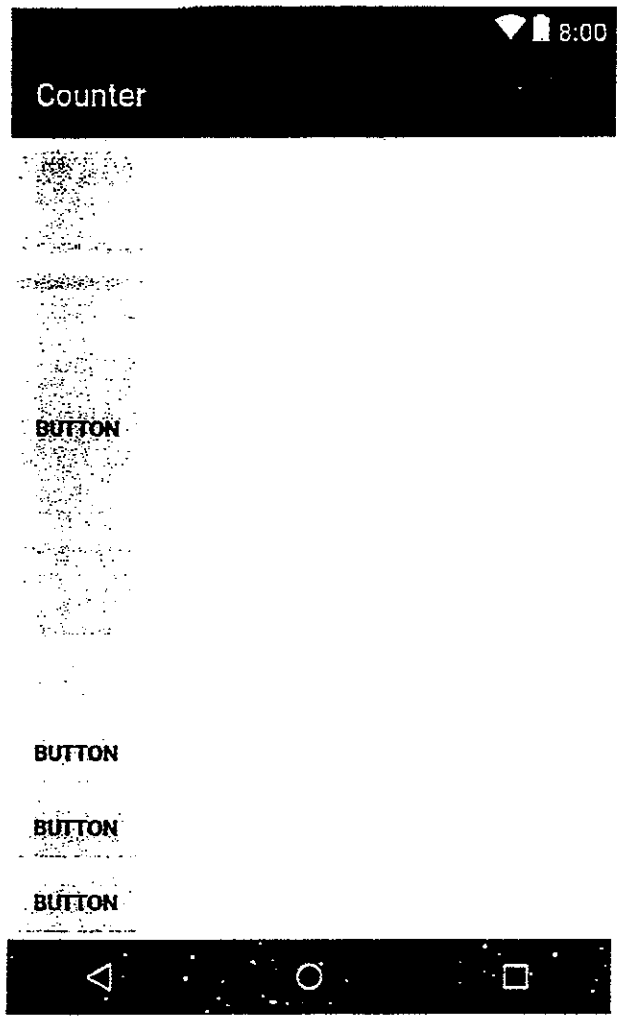
    <Button
        android:layout_weight="1"
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>
```



Karena LinearLayout kita vertical, maka layout_weight akan memberikan sisa ruang ke Button pertama untuk tingginya. Apabila LinearLayout menggunakan orientasi horizontal maka layout_weight akan memberikan sisa ruang ke lebarnya.

Bagaimana jika kita memberikan layout_weight ke lebih dari satu komponen?

```
<?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="com.codepolitan.counter.MainActivity"
    android:orientation="vertical"
    >
```

```
    <Button
        android:layout_weight="1"
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

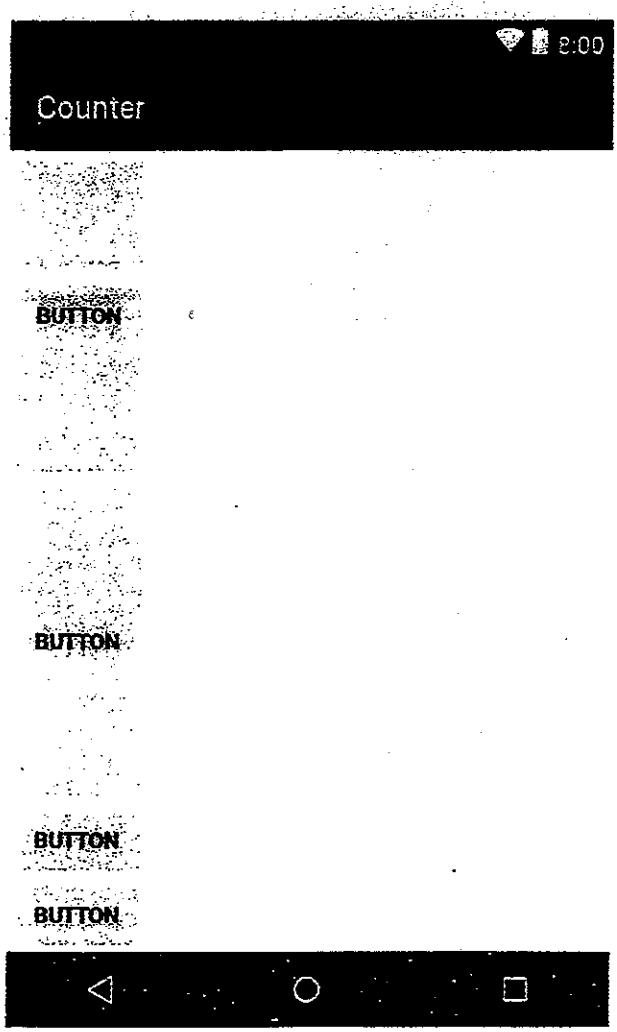
    <Button
        android:layout_weight="1"
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:text="Button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</LinearLayout>
```

Jika ada lebih dari satu komponen yang memiliki layout_weight dengan angka yang sama (pada contoh di atas adalah 1), maka sisa ruang akan dibagi dua.



Sekarang coba lakukan latihan berikut:

- Berikan android:layout_weight="1" ke semua Button. Apa yang terjadi?
- Berikan android:layout_weight="4" ke salah salah Button. Apa yang terjadi?

RelativeLayout - mengatur posisi layout

RelativeLayout merupakan salah satu jenis layout yang menampilkan kontennya menggunakan posisi relatif. Relatif yang seperti apa? Tuliskan dulu kode XML berikut:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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.codepolitan.counter.MainActivity"
    >

    <Button
        android:layout_alignParentLeft="true"
        android:layout_centerVertical="true"
        android:id="@+id/btn1"
        android:layout_weight="1"
        android:text="Button 1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:layout_centerInParent="true"
        android:id="@+id/btn2"
        android:layout_weight="1"
        android:text="Button 2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:layout_alignParentRight="true"
        android:layout_centerVertical="true"
        android:id="@+id/btn3"
        android:layout_weight="1"
        android:text="Button 3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:layout_centerHorizontal="true"
        android:id="@+id/btn4"
        android:layout_weight="4"
        android:text="Button 4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

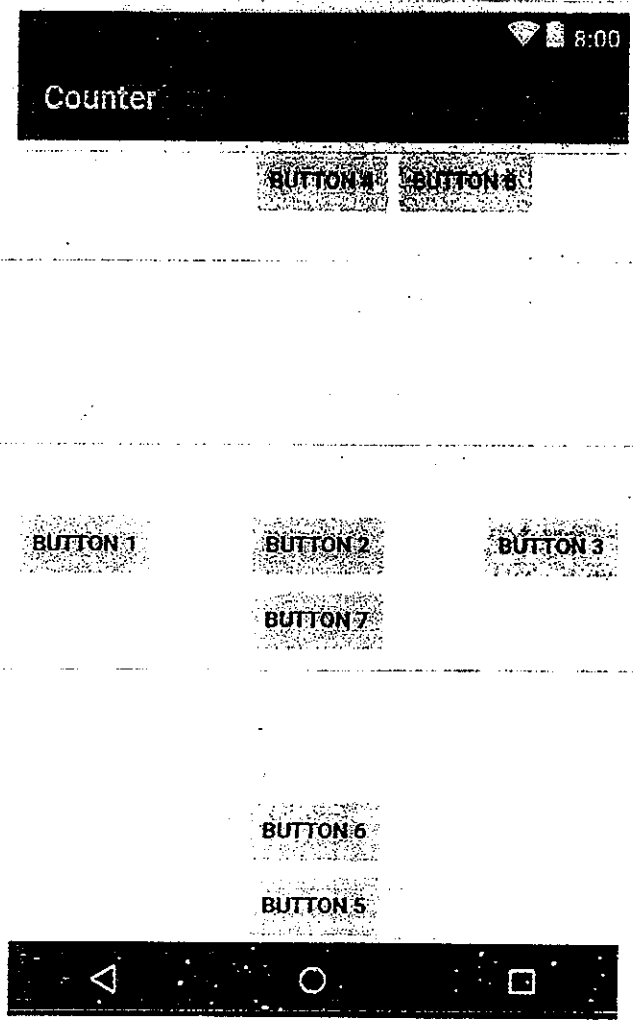
    <Button
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:id="@+id/btn5"
        android:layout_weight="4"
        android:text="Button 5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
```

```
    <Button
        android:layout_above="@id/btn5"
        android:layout_centerHorizontal="true"
        android:id="@+id/btn6"
        android:layout_weight="4"
        android:text="Button 6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:layout_below="@id/btn2"
        android:layout_centerHorizontal="true"
        android:id="@+id/btn7"
        android:layout_weight="4"
        android:text="Button 7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:layout_toRightOf="@id/btn4"
        android:id="@+id/btn8"
        android:layout_weight="4"
        android:text="Button 8"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```



Pada gambar di atas kita menentukan bahwa:

- Button 1 berada di tengah layar secara vertikal
- Button 2 berada di tengah layar secara vertikal dan horizontal
- Button 3 berada di tengah layar sercara vertikal dan berada di sebelah kanan layar
- Button 4 berada di tengah layar secara horizontal
- Button 5 berada di tengah layar secara horizontal dan berada di layar bagian bawah
- Button 6 berada di atas Button 5 dan ditengah layar secara horizontal
- Button 7 berada tengah layar secara horizontal dan berada di bawah Button 2
- Button 8 di sebelah kanan Button 4



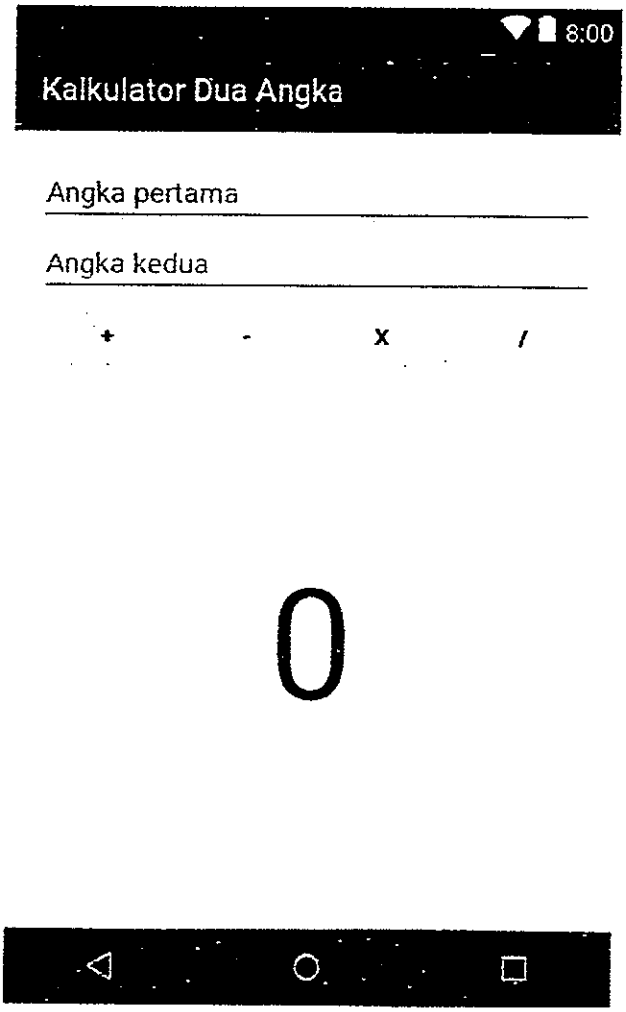
Opsi layout lain untuk RelativeLayout dapat dilihat di sini[1].



Perhatikan saat kita ingin mengatakan suatu komponen berada di sebelah kiri, kanan, atas, bawah komponen yang lain, maka kita harus memberikan id dari komponen yang satunya.

3. Membuat Aplikasi Android Interaktif Sederhana

Sekarang kita akan belajar bagaimana membuat aplikasi Android yang interaktif. Aplikasi yang akan kita buat adalah kalkulator dua angka sederhana.



[1] https://developer.android.com/reference/android/widget/RelativeLayout.html

Pertama, buat sebuah *project* baru dengan pengaturan sebagai berikut:

- Application name: Kalkulator Dua Angka
- Company domain: codepolitan.com (atau domain sendiri jika ada)
- Project location: (masukkan folder tempat ingin menyimpan project ini)
- Minimum SDK: API 16
- Template: Empty activity

3.1. Mengatur Layout Aplikasi

Setelah project jadi, sekarang waktunya untuk mendesain tampilan aplikasi kita. Berdasarkan gambar di atas, bisakah kamu menyusun sendiri layoutnya?

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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.codepolitan.kalkulatorduaangka.MainActivity"
    android:layout_margin="16dp"
    >

    <EditText
        android:hint="Angka pertama"
        android:id="@+id/etAngka1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <EditText
        android:hint="Angka kedua"
        android:layout_below="@id/etAngka1"
        android:id="@+id/etAngka2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <LinearLayout
        android:id="@+id/btnOperasi"
        android:layout_below="@id/etAngka2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
```

```
        <Button
            android:text="+"
            android:layout_weight="1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        <Button
            android:text="-"
            android:layout_weight="1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        <Button
            android:text="x"
            android:layout_weight="1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        <Button
            android:text="/"
            android:layout_weight="1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
    </LinearLayout>

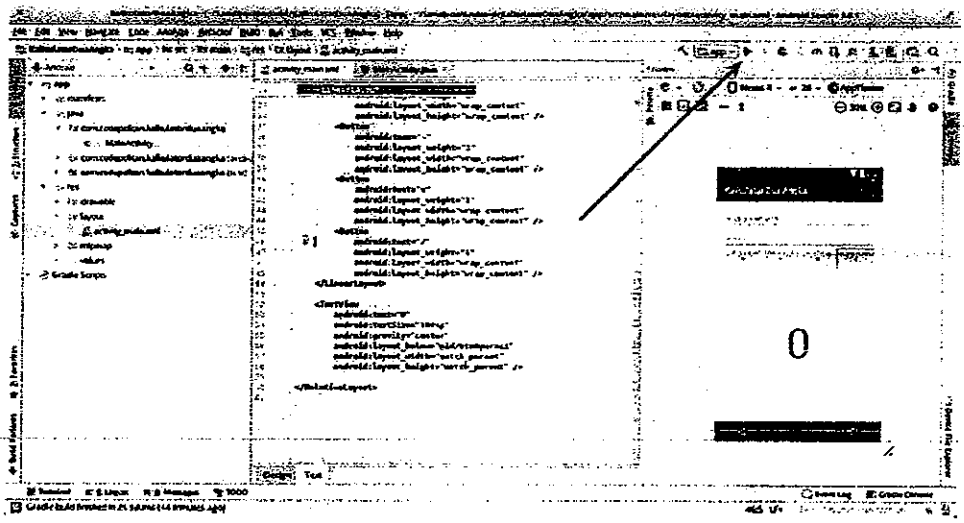
    <TextView
        android:id="@+id/tvHasil"
        android:text="0"
        android:textSize="100sp"
        android:gravity="center"
        android:layout_below="@id/btnOperasi"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

</RelativeLayout>
```

Kode yang baru saja kita tulis menggunakan RelativeLayout sebagai parent utamanya. Lalu kita menambahkan komponen EditText, Button, dan TextView.

Karena kita ingin agar keempat tombol memiliki lebar yang sama, maka kita bisa memanfaatkan LinearLayout untuk menggunakan layout_weight sehingga lebarnya bisa disamaratakan.

Sekarang coba jalankan aplikasi ini di emulator atau device melalui tombol **RUN**.



3.2. Menghubungkan XML dengan Java

Meski sudah dapat dijalankan, namun aplikasi yang kita buat belum selesai karena setelah angka sudah dimasukkan dan tombol operasi dipilih, tidak ada yang terjadi. Hal ini karena kita belum menghubungkan XML dengan Java.

Mari kita hubungkan komponen yang ada di XML dengan Java.

Buka file `MainActivity.java` lalu tambahkan kode berikut:

```
public class MainActivity extends AppCompatActivity {

    EditText etAngka1,etAngka2;
    TextView tvHasil;

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

Perhatikan bahwa kita membuat tiga buah variabel. Dua variabel bertipe EditText dan sebuah variabel bertipe TextView. Mengapa kita melakukan hal ini?

Hal ini karena komponen XML tidak dapat memperbarui dirinya sendiri sehingga harus kita hubungkan dengan Java (sekarang sudah ada Data Binding Library[2]

[2] https://developer.android.com/topic/libraries/data-binding/index.html

yang dapat melakukannya namun tidak akan kita bahas di kelas ini). Untuk memperbarui konten komponen yang ada di XML, kita cukup membuat variabel dengan tipe yang sama.

Setelah membuat variabel, selanjutnya kita harus mengarahkan variabel ini ke komponen yang diinginkan di XML. Pengarahan ini harus dilakukan di dalam method `onCreate()`.

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

    etAngka1 = findViewById(R.id.etAngka1);
    etAngka2 = findViewById(R.id.etAngka2);
    tvHasil = findViewById(R.id.tvHasil);
}
```

Perhatikan di dalam argumen pemanggilan method `findViewById` kita mengirimkan `R.id.etAngka1`, dst. Isi dari `findViewById` adalah atribut `android:id` yang kita tuliskan di layout XML. Periksa kembali file `activity_main.xml` untuk memastikan mereka sudah ditulis.



Nama variabel Java tidak harus sama dengan nama id komponen XML meskipun pada contoh di atas kebetulan nama variabelnya sama.

Apapun nama variabelnya asalkan diarahkan ke komponen yang seharusnya maka ia juga akan tetap bekerja dengan baik.

3.3. Membuat Click Listener

Selanjutnya kita akan membuat sebuah listener yang melakukan suatu operasi saat sebuah tombol disentuh.

Buka file `activity_main.xml` dan tambahkan atribut `onClick` ke tiap Button:

```
// kode lain disembunyikan agar lebih singkat

    <LinearLayout
```

```
        android:id="@+id/btnOperasi"
        android:layout_below="@id/etAngka2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <Button
            android:onClick="tambah"
            android:text="+"
            android:layout_weight="1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        <Button
            android:onClick="kurang"
            android:text="-"
            android:layout_weight="1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        <Button
            android:onClick="kali"
            android:text="x"
            android:layout_weight="1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        <Button
            android:onClick="bagi"
            android:text="/"
            android:layout_weight="1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
    </LinearLayout>

// kode lain disembunyikan agar lebih singkat
```

Perhatikan teks yang ada di dalam atribut `android:onClick`. Kita harus membuat sebuah method dengan nama yang sama di MainActivity.java. Misalnya, karena di Button + kita memberikan `onClick="tambah"` maka method yang kita buat adalah:

```
public class MainActivity extends AppCompatActivity {

    EditText etAngka1,etAngka2;
    TextView tvHasil;

    @Override
```

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

        etAngka1 = findViewById(R.id.etAngka1);
        etAngka2 = findViewById(R.id.etAngka2);
        tvHasil = findViewById(R.id.tvHasil);
    }


    public void tambah(View view) {
    }

}
```

Silahkan deklarasi tiga method untuk tiga Button sisanya.

3.4. Membaca dan Mengubah Konten XML dari Java

Mari kita selesaikan proses penjumlahan. Didalam method tambah, kita tambahkan kode berikut:

```
public void tambah(View view) {
    String angka1 = etAngka1.getText().toString();
    String angka2 = etAngka2.getText().toString();

    if (TextUtils.isEmpty(angka1) && TextUtils.isEmpty(angka2)) {
        Toast.makeText(MainActivity.this, "Masukan kedua angka terlebih
 dahulu", Toast.LENGTH_SHORT).show();
    }
}
```

Yang kita lakukan pertama kali adalah membaca isi dari EditText dan memasukkannya ke dalam variabel angka1 dan angka2 yang bertipe String. Di bawahnya kita periksa apakah pengguna sudah mengisi EditText sebelum memilih operasi penjumlahan.

Mari kita lanjutkan dengan menjumlahkan kedua angka dan menampilkan hasilnya:

```
public void tambah(View view) {
    // sama dengan kode di atas
```

```
    Double hasil = Double.valueOf(angka1) + Double.valueOf(angka2);
    tvHasil.setText(String.valueOf(hasil));
}
```

Karena data yang dibaca dari EditText masih berupa String, kita harus ubah dulu ke dalam bentuk angka (Double, Long, int, float, dll) menggunakan `NamaKelas.valueOf()` (jika menggunakan int panggil Integer.valueOf()).

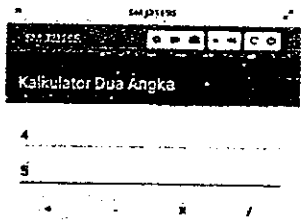
Terakhir kita tampilkan isi dari variabel hasil yang sudah memiliki hasil penjumlahan angka1 dengan angka2.



Method `setText()` hanya menerima data bertipe String. Karena hasil merupakan Double, maka harus kita ubah menjadi string dengan memanfaatkan method `String.valueOf()`.

Trik lainnya adalah menggabungkan variabel hasil dengan String kosong, misalnya `tvHasil.setText(""+hasil)`

Jalankan dan pastikan proses penjumlahan telah berjalan dengan sempurna.



9.0

Apakah kamu bisa melanjutkan program ini sendiri?

```
public class MainActivity extends AppCompatActivity {

    EditText etAngka1,etAngka2;
```

```
    TextView tvHasil;

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

        etAngka1 = findViewById(R.id.etAngka1);
        etAngka2 = findViewById(R.id.etAngka2);
        tvHasil = findViewById(R.id.tvHasil);
    }


    public void tambah(View view) {
        String angka1 = etAngka1.getText().toString();
        String angka2 = etAngka2.getText().toString();

        if (TextUtils.isEmpty(angka1) && TextUtils.isEmpty(angka2)) {
            Toast.makeText(MainActivity.this, "Masukan kedua angka
terlebih dahulu", Toast.LENGTH_SHORT).show();
        }

        Double hasil = Double.valueOf(angka1) + Double.valueOf(angka2);
        tvHasil.setText(String.valueOf(hasil));
    }

    public void kurang(View view) {
        String angka1 = etAngka1.getText().toString();
        String angka2 = etAngka2.getText().toString();

        if (TextUtils.isEmpty(angka1) && TextUtils.isEmpty(angka2)) {
            Toast.makeText(MainActivity.this, "Masukan kedua angka
terlebih dahulu", Toast.LENGTH_SHORT).show();
        }

        Double hasil = Double.valueOf(angka1) + Double.valueOf(angka2);
        tvHasil.setText(String.valueOf(hasil));
    }

    public void kali(View view) {
        String angka1 = etAngka1.getText().toString();
        String angka2 = etAngka2.getText().toString();

        if (TextUtils.isEmpty(angka1) && TextUtils.isEmpty(angka2)) {
```

```
            Toast.makeText(MainActivity.this, "Masukan kedua angka
 terlebih dahulu", Toast.LENGTH_SHORT).show();
        }

        Double hasil = Double.valueOf(angka1) + Double.valueOf(angka2);
        tvHasil.setText(String.valueOf(hasil));
    }

    public void bagi(View view) {
        String angka1 = etAngka1.getText().toString();
        String angka2 = etAngka2.getText().toString();

        if (TextUtils.isEmpty(angka1) && TextUtils.isEmpty(angka2)) {
            Toast.makeText(MainActivity.this, "Masukan kedua angka
terlebih dahulu", Toast.LENGTH_SHORT).show();
        }

        Double hasil = Double.valueOf(angka1) + Double.valueOf(angka2);
        tvHasil.setText(String.valueOf(hasil));
    }
}
```

Yeay \o/, kamu telah melalui hari pertama dengan selamat.