

## MODUL 4

### LINEAR LAYOUT DAN CONSTRAINT LAYOUT



#### CAPAIAN PEMBELAJARAN

---

1. Mahasiswa mampu memahami dan mengetahui tentang Linear Layout dan Constraint Layout
2. Mahasiswa mampu menggunakan komponen view dasar (TextView, EditText, Button) dalam aplikasi



#### KEBUTUHAN ALAT/BAHAN/SOFTWARE

---

1. Android Studio 3.5
2. Handphone Android versi 7.0 (Nougat)
3. Kabel data USB.
4. Driver ADB.



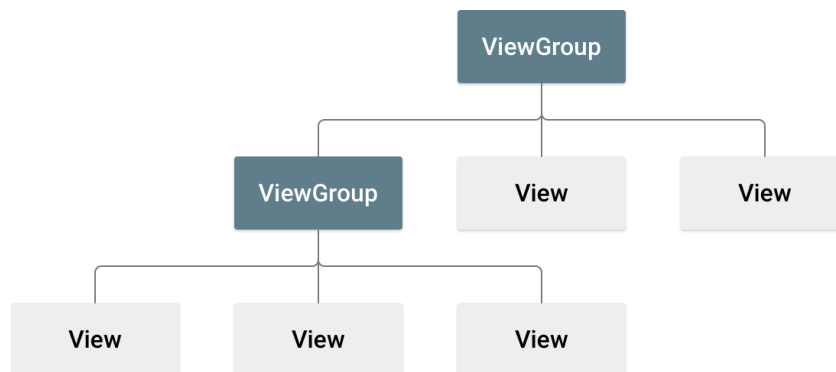
#### DASAR TEORI

---

##### Layout

Layout menentukan struktur untuk antarmuka pengguna di aplikasi Anda, seperti di dalam activity. Semua elemen pada layout dibuat menggunakan hierarki objek `View` dan `ViewGroup`. `View` biasanya menggambar sesuatu yang pengguna bisa

melihat dan berinteraksi dengannya. Sedangkan `ViewGroup` adalah container yang tidak terlihat yang menentukan struktur layout untuk `View` dan objek `ViewGroup` lainnya, seperti yang ditampilkan pada Gambar berikut.



Gambar 1. Ilustrasi dari hierarki tampilan, yang mendefinisikan layout UI

Pada dasarnya semua elemen antar pengguna di aplikasi Android dibangun menggunakan dua buah komponen inti, yaitu `View` dan `viewgroup`.

Sebuah `View` adalah obyek yang menggambar komponen tampilan ke layar yang mana pengguna dapat melihat dan berinteraksi langsung.

Contoh komponen turunan dari **View** seperti :

- **TextView**, komponen yang berguna untuk menampilkan teks ke layar.
- **Button**, komponen yang membuat pengguna dapat berinteraksi dengan cara ditekan untuk melakukan sesuatu.
- **ImageView**, Komponen untuk menampilkan gambar.
- **ListView**, komponen untuk menampilkan informasi dalam bentuk list.
- **GridView**, komponen untuk menampilkan informasi dalam bentuk grid.
- **RadioButton**, komponen yang memungkinkan pengguna dapat memilih satu pilihan dari berbagai pilihan yang disediakan.
- **Checkbox**, komponen yang memungkinkan pengguna dapat memilih lebih dari satu dari pilihan yang ada.

Sedangkan **ViewGroup** adalah sebuah objek yang mewadahi objek-objek **View** dan **ViewGroup** itu sendiri sehingga membentuk satu kesatuan tampilan aplikasi yang utuh. Contoh komponen **ViewGroup** adalah:

- **LinearLayout**
- **FrameLayout**
- **RelativeLayout**
- **TableLayout**

## **Linear Layout**

**LinearLayout** adalah sekelompok tampilan yang menyejajarkan semua anak dalam satu arah, secara vertikal atau horizontal. Anda bisa menetapkan arah layout dengan atribut `android:orientation`.



`android:orientation="vertical"`   `android:orientation="horizontal"`

## **Constraint Layout**

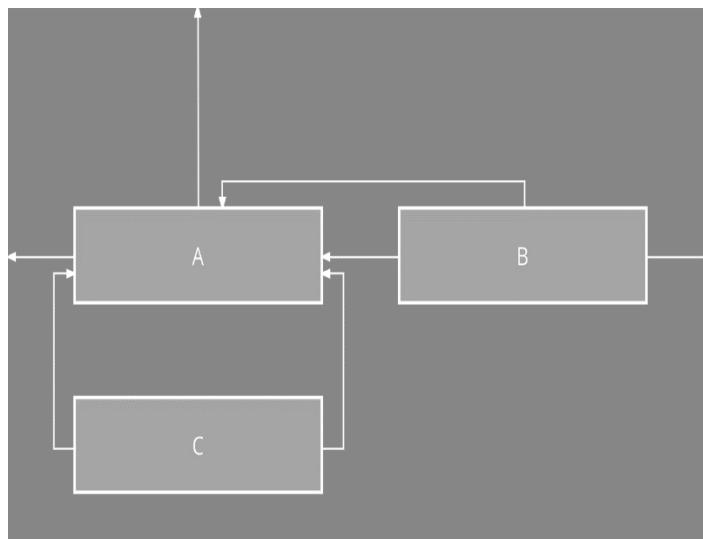
**ConstraintLayout** memungkinkan Anda membuat tata letak yang kompleks dan besar dengan hierarki tampilan datar (tidak ada grup tampilan tersarang). **ConstraintLayout** mirip dengan **RelativeLayout** yang semua tampilannya diletakkan sesuai dengan hubungan antara tampilan yang setara dan tata letak induk, tetapi lebih fleksibel dari **RelativeLayout** dan lebih mudah digunakan dengan Layout Editor Android Studio.

Semua kecanggihan **ConstraintLayout** tersedia langsung dari fitur visual Layout Editor, karena API tata letak dan Layout Editor dibuat khusus untuk digunakan dengan satu sama lain. Jadi, Anda dapat membuat tata letak menggunakan **ConstraintLayout** sepenuhnya dengan menarik lalu melepas (*drag and drop*), dan bukan dengan mengedit XML.

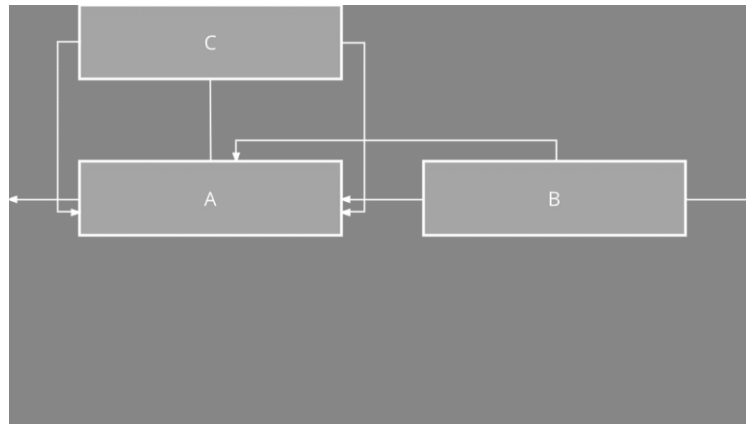
Kita perlu menentukan posisi dari view atau bagaimana agar view tersebut terhubung dengan parent layout atau view lainnya. Karena setelah ditambahkan, view tersebut tidak memiliki constraint yang menghubungkannya dengan parent layout atau view lainnya. Sehingga ketika dijalankan, posisi dari view tersebut akan berada di bagian atas sebelah kiri.

Kita akan menggunakan constraint sebagai dasar dalam menentukan posisi agar sebuah view dapat terhubung dengan view lainnya sesuai harapan kita.

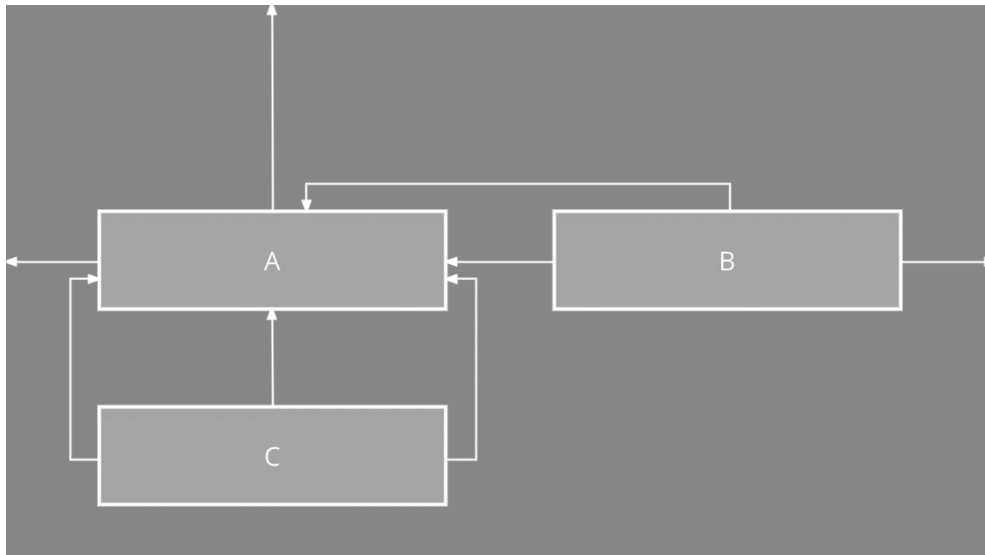
Setiap view setidaknya memiliki satu **vertical** dan **horizontal constraint**. Misal kita memiliki sebuah layout dengan tampilan pada Layout Editor seperti berikut:



Susunan tampilan di atas akan terlihat normal. Tidak ada yang salah di Layout Editor. Tapi jika kita perhatikan seksama, **view C** di atas hanya memiliki **horizontal constraint** yang diatur sejajar dengan **view A**. Sehingga ketika jika kita coba menjalankannya, sama seperti yang disebutkan di atas, maka posisi dari **view C** akan berada di posisi atas seperti berikut:



Berbeda jika kita menambahkan **vertikal constraint** pada **view C** yang diatur terikat dengan **view A** seperti berikut:



Ketika dijalankan, yang tampil akan sesuai dengan apa yang terlihat di Layout Editor.

Untuk menggunakan `ConstraintLayout` di project Anda, ikuti langkah-langkah berikut:

1. Pastikan Anda memiliki repositori `maven.google.com` yang dinyatakan di file `build.gradle` tingkat modul:

```
repositories {
    google()
}
```

2. Tambahkan library sebagai dependensi di file build.gradle yang sama, seperti yang ditunjukkan di contoh berikut. Perlu diperhatikan bahwa versi terbaru mungkin berbeda dengan yang ditunjukkan di contoh:

```
dependencies {  
    implementation 'com.android.support.constraint:constraint-layout:1.1.2'  
}
```

3. Di notifikasi sinkronisasi atau toolbar, klik Sync Project with Gradle Files.

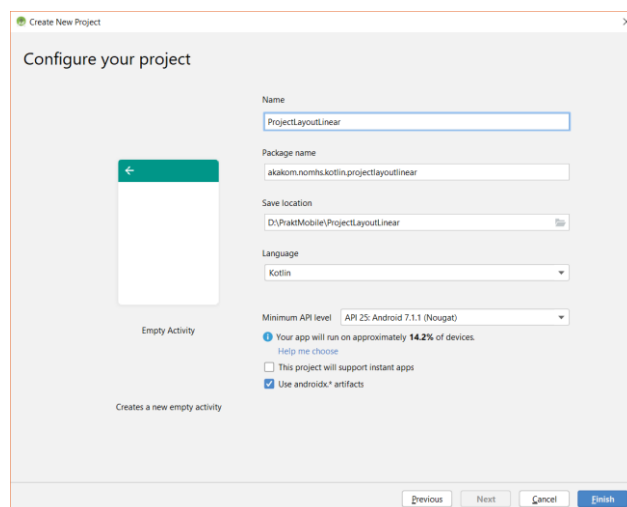
Sekarang Anda sudah siap untuk membuat tata letak menggunakan ConstraintLayout.



## PRAKTIK

### Praktik 1. Proyek LinearLayout

1. Buatlah project dengan nama **LayoutLinear** dengan cara **klik** menu **File** → **New** → **New Project ...**
2. Kemudian pilih **Empty Activity**, lalu **klik** button **Next**
3. Beri project anda yang baru dengan nama **LinearLayout**, kemudian **klik** button **Finish**.



- Langkah selanjutnya, buka file **activity\_main.xml** yang terdapat pada **app** → **res** → **layout**. Secara default pada saat membuat project baru maka akan muncul teks **Hello World**.
- Buka file **activity\_main.xml** kemudian pilih tab **Text**, akan terlihat kode seperti berikut.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>

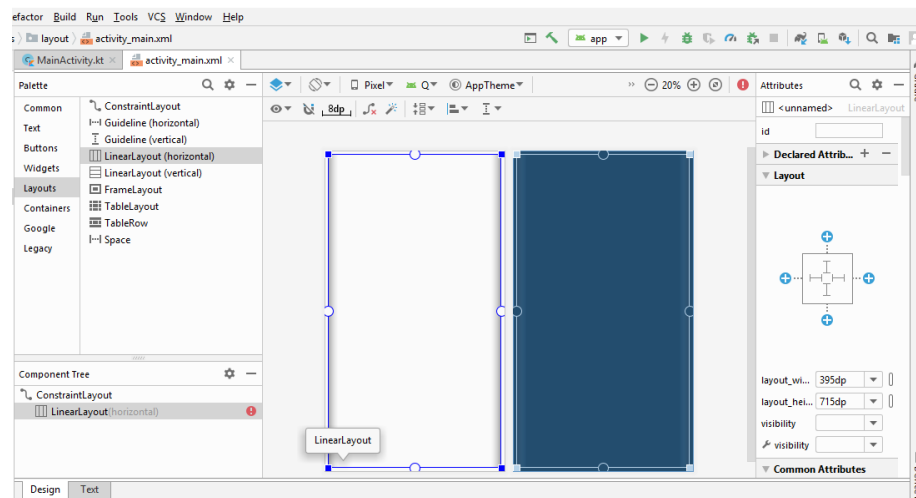
</androidx.constraintlayout.widget.ConstraintLayout>
```

- Standar layout pertama adalah Constraint Layout. Ubahlah menjadi Linear Layout dengan kode berikut. Tambahkan atribut **android:orientation** dengan nilai **vertical**.

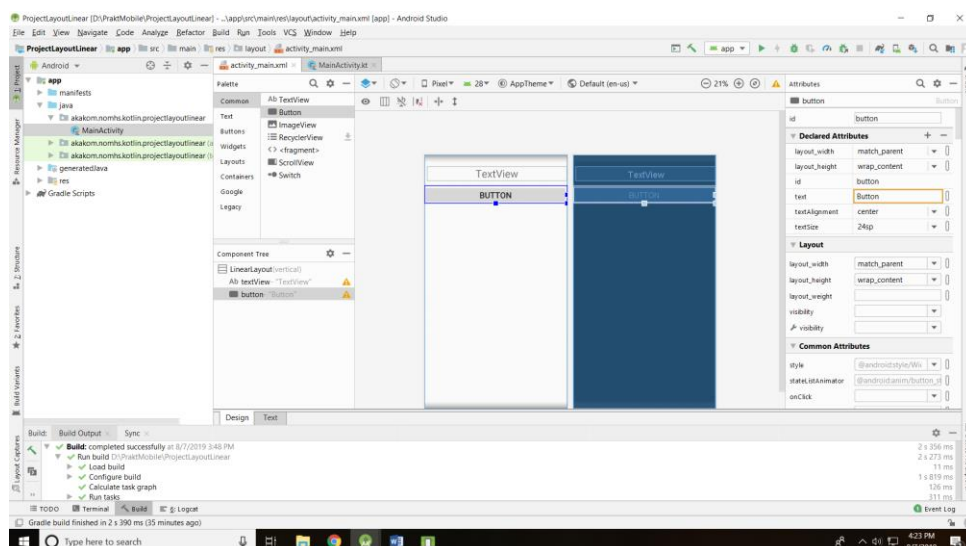
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

</LinearLayout>
```

- Maka hasilnya akan terlihat sebuah linear layout berhasil ditempatkan di bagian area atau canvas.



8. Tambahkan komponen TextView dan Button, sehingga menjadi sebagai berikut.



9. Jalankan dan amati hasilnya.
10. Ubah atribut-atribut yang ada dan jalankan lagi, amati perubahannya.

## Praktik 2. Proyek Constraint Layout

1. Buat project baru dengan nama **Constraint Layout**.
2. Buka file **activity\_main.xml** dan klik tab **Desain**.
3. Anda akan menambahkan constraint (batasan) secara manual, maka koneksi otomatis kita matikan. Di toolbar, temukan **Turn Off/On Autoconnect** toggle button, yang ditunjukkan di bawah ini. (Jika Anda tidak dapat melihat toolbar,



klik di dalam area editor desain dari Layout Editor.) Pastikan autoconnect tidak aktif.

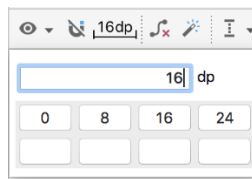


→ Autoconnect is ON





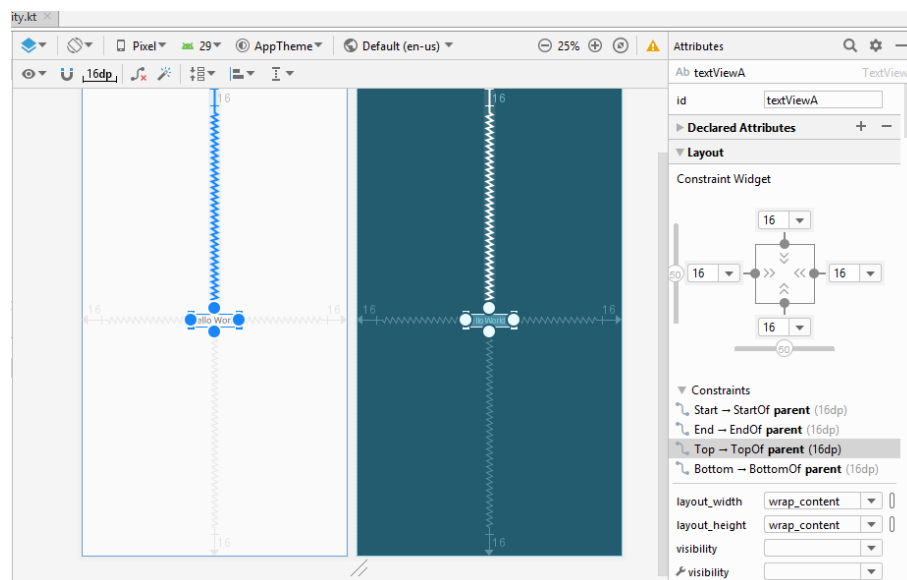
→ Autoconnect is OFF

- Gunakan toolbar untuk mengatur default margins ke 16dp.

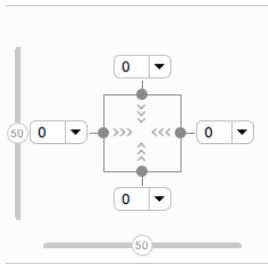


Ketika Anda mengatur margin default ke 16dp, constraint baru dibuat dengan margin ini, jadi Anda tidak perlu menambahkan margin setiap kali Anda menambahkan constraint.

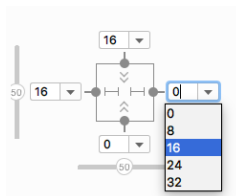
- Perbesar menggunakan ikon +  33%  di sebelah kanan toolbar, hingga teks Hello World terlihat di dalam tampilan teksnya.
- Klik pada tampilan teks **Hello World** untuk membuka panel **Attributes**.



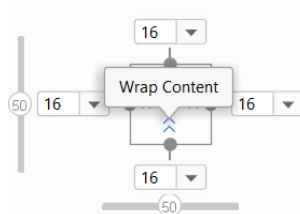
- View Inspector, yang ditunjukkan pada gambar di bawah, adalah bagian dari panel Atribut. View Inspector mencakup kontrol untuk atribut layout seperti constraint, constraint types, constraint bias, and view margins.



8. Constraint bias menempatkan elemen tampilan di sepanjang sumbu horizontal dan vertikal. Secara default, tampilan dipusatkan di antara dua constraint dengan bias 50%. Untuk menyesuaikan bias, Anda dapat menarik slider bias di view inspector. Menarik slider bias mengubah posisi tampilan sepanjang sumbu.
9. Tambahkan margin untuk TextView Hello World. Perhatikan bahwa dalam view inspector, margin kiri, kanan, atas, dan bawah untuk tampilan teks adalah 0. Margin default tidak ditambahkan secara otomatis, karena tampilan ini dibuat sebelum Anda mengubah margin default. Untuk margin kiri, kanan, dan atas, pilih 16dp dari menu drop-down di inspektur tampilan. Misalnya, dalam tangkapan layar berikut ini Anda menambahkan `layout_marginEnd` (`layout_marginRight`).



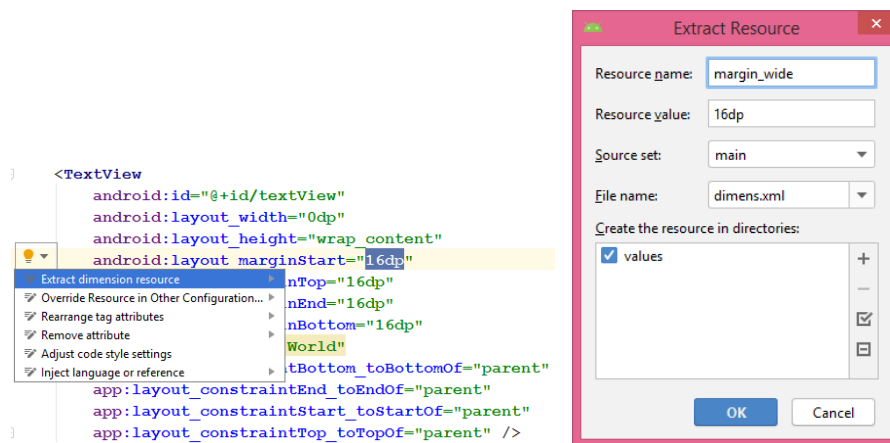
10. Sesuaikan batasan dan margin untuk TextView. Di view inspector, panah `>>>` di dalam kotak mewakili tipe constrain:
  - a. `>>>` wrap content: Tampilan hanya selebar kontennya.



- b. `|` fixed: Anda dapat menentukan dimensi sebagai margin tampilan di kotak teks di sebelah panah constrain tetap.
- c. `||||` match constraint: Tampilan melebar sebanyak mungkin untuk memenuhi constraint di setiap sisi, setelah memperhitungkan margin

tampilan sendiri. Constrain ini sangat fleksibel, karena memungkinkan layout untuk beradaptasi dengan berbagai ukuran dan orientasi layar. Dengan membiarkan tampilan sesuai dengan constraint, Anda membutuhkan layout yang lebih sedikit untuk aplikasi yang Anda buat.

11. Di view inspector, ubah constraint kiri dan kanan ke Match Constraints. (Klik simbol panah untuk beralih di antara jenis constraint.)
12. Pindah ke tab **Text**, Ekstrak resource dimensi untuk layout\_marginStart, dan atur nama Resource ke margin\_wide. (blok pada bagian isian dari layout\_marginStart, pilih **extract dimensions resource**).



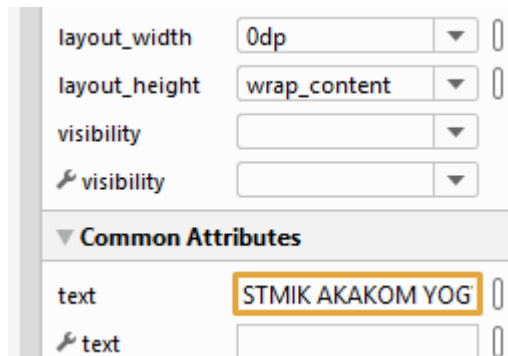
Maka hasilnya akan menjadi seperti berikut.

```
<TextView
    android:id="@+id/textView"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="@dimen/margin_wide"
    android:layout_marginTop="16dp"
    android:layout_marginEnd="16dp"
```

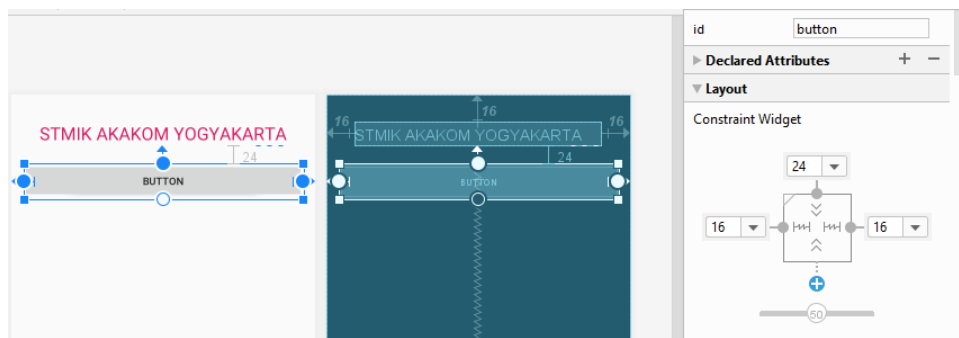
13. Kerjakan untuk layout\_marginEnd dan layout\_marginTop. Hasilnya adalah sebagai berikut

```
android:layout_marginTop="@dimen/margin_wide"
android:layout_marginStart="@dimen/margin_wide"
android:layout_marginEnd="@dimen/margin_wide"
```

14. Gantilah isi atribut text dari Hello World menjadi STMIK AKAKOM YOGYAKARTA.



15. Kemudian, tambahkan satu button, atur constraintnya. Hasilnya sebagai berikut.



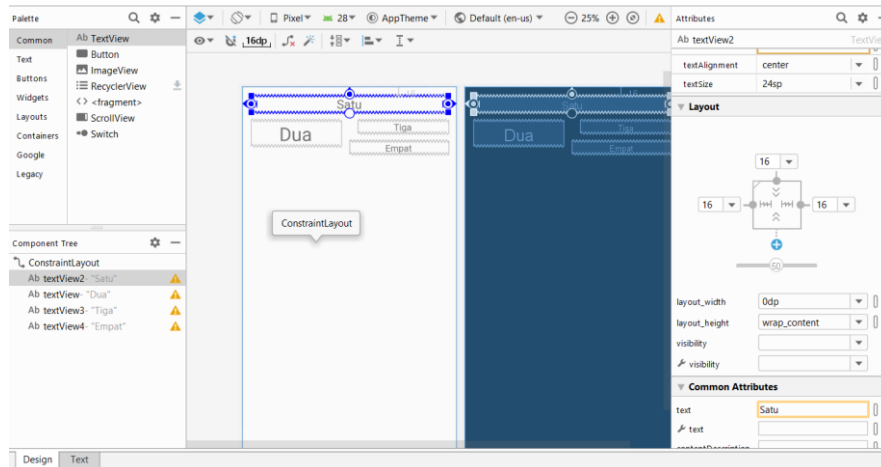
16. Jalankan dan amati hasilnya.

17. Ubah atribut-atribut yang ada dan jalankan lagi, amati perubahannya.



## LATIHAN

1. Buat project baru dengan desain sebagai berikut.



2. Buat project baru dengan menggunakan Linear Layout dengan minimal 3 komponen (TextView/Button) yang ditambahkan dan eksplorasilah atribut-atribut yang ada.
3. Buat project baru dengan menggunakan Constrains Layout dengan minimal 3 komponen (TextView/Button) yang ditambahkan dan eksplorasilah atribut-atribut yang ada.



## TUGAS

---

1. Buat Project pada perangkat komputer anda untuk mengimplementasikan layout dengan berbagai bentuk tampilan.



## REFERENSI

---

1. <https://developer.android.com/guide/topics/ui/layout/linear>

2. <https://developer.android.com/training/constraint-layout>
3. <https://developer.android.com/guide/topics/ui/notifiers/toasts>
4. <https://codelabs.developers.google.com/codelabs/kotlin-android-training-get-started/#0>
5. <https://developer.android.com/studio/run/device>