Mobile Programming

Catatan Kuliah #7

Alauddin Maulana Hirzan, M. Kom
0607069401

Layout dan **View Mode**

Layout dan **View Mode**

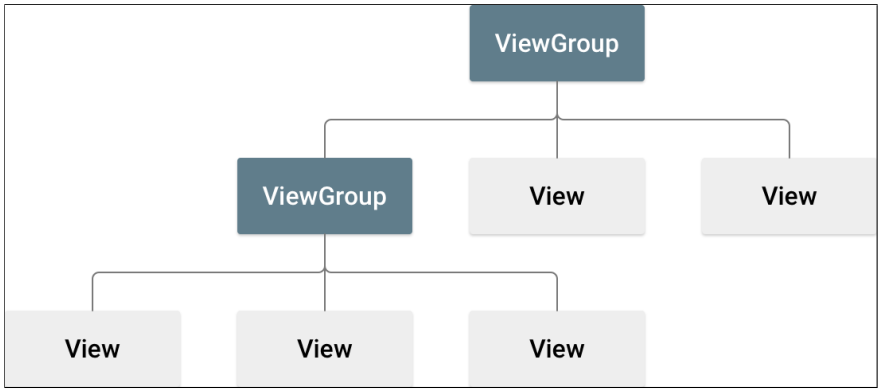
Apa itu **Layout** #1

Menurut **developers.android.com**:

> *Tata letak / **Layout** mendefinisikan struktur untuk antarmuka pengguna di aplikasi, seperti dalam aktivitas. Semua elemen dalam tata letak dibuat menggunakan hierarki objek **View** dan **ViewGroup**. Sebuah objek **View** biasanya menggambar sesuatu yang dapat dilihat dan berinteraksi dengan pengguna. Sedangkan **ViewGroup** adalah wadah tak terlihat yang mendefinisikan struktur tata letak untuk **View** dan objek **ViewGroup** lainnya*

Layout dan **View Mode**

Apa itu **Layout** #2



Layout dan **View Mode**

Apa itu **Layout** #3

Objek **View** biasanya disebut "Widget" dan dapat berupa salah satu dari banyak subkelas, seperti **Button** atau **TextView**.

Objek **ViewGroup** biasanya disebut "Layout" dapat berupa salah satu dari banyak jenis yang menyediakan struktur tata letak yang berbeda, seperti **LinearLayout** atau **ConstraintLayout**.

Layout dan **View Mode**

Apa itu **View** #1

Objek ini mewakili blok bangunan dasar untuk komponen antarmuka pengguna. **View** menempati area persegi panjang di layar dan bertanggung jawab untuk menggambar dan penanganan peristiwa. **View** adalah kelas dasar untuk widget, yang digunakan untuk membuat komponen UI interaktif (tombol, bidang teks, dll.).

Semua objek **View** dalam sebuah jendela diatur dalam satu pohon. Anda dapat menambahkan tampilan baik dari kode atau dengan menentukan pohon tampilan dalam satu atau lebih file tata letak XML.

Layout dan **View Mode**

Apa itu **View** #2

Setelah programmer membuat pohon tampilan, biasanya ada beberapa jenis operasi umum yang ingin dilakukan:

1. Set properties/attributes
2. Set focus
3. Set up listeners
4. Set visibility

Layout dan **View Mode**

Attributes dari View #1

ID
View mungkin memiliki **id** yang terkait dengannya. ID ini biasanya ditetapkan dalam file XML layout, dan digunakan untuk membedakan **WIdget** yang ada di dalam **Component Tree** **View ID** tidak harus unik di seluruh pohon, tetapi merupakan praktik yang baik untuk memastikan bahwa ID tersebut setidaknya unik di dalam bagian pohon

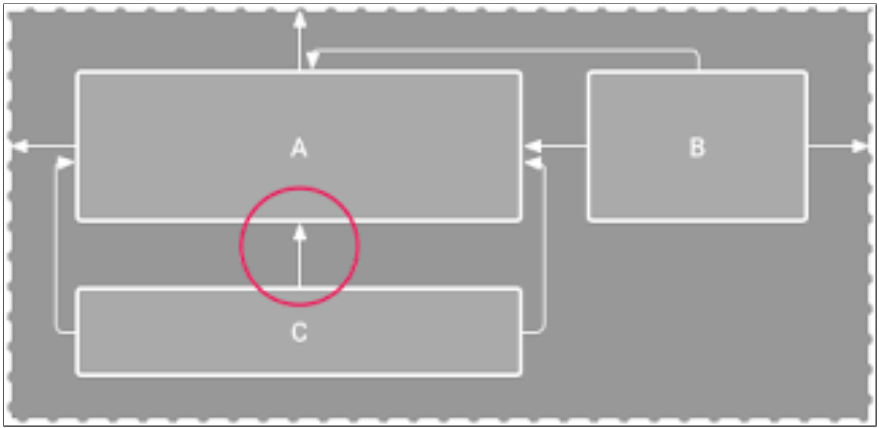
Layout dan **View Mode**

Attributes dari View #2

Posisi
Dalam peletakan **Widget**, Android Studio menggunakan metode **Koordinat** atau **Constraint / Batasan**. Secara **Default**, Android Studio akan memiliki **Constraint** karena cukup mengukur batas widget dengan widget acuan

Layout dan **View Mode**

Attributes dari View #3



Layout dan **View Mode**

Attributes dari View #4

Ukuran
Ukuran suatu **View** dinyatakan dengan lebar dan tinggi. Sebuah tampilan sebenarnya memiliki dua pasang nilai lebar dan tinggi. Ukuran ini dapat diatur dengan menggunakan attribute **layout_width** dan **layout_height**

Ukuran ini dapat diatur dengan cara otomatis maupun manual. Jika dilakukan secara **otomatis**, layout bisa diatur untuk **wrap_content** atau menyesuaikan isi atau **match_constraint** atau sesuai dengan batasan yang sudah diset. Jika dilakukan secara manual cukup dengan mengisikan ukuran dengan unit **dot pixel / dp**

Layout dan **View Mode**

Event Listener dari View #1

Event adalah cara yang berguna untuk mengumpulkan data tentang interaksi pengguna dengan komponen interaktif Aplikasi. Seperti penekanan tombol atau sentuhan layar, dll.

Kerangka kerja Android mempertahankan antrean peristiwa sebagai basis masuk pertama, keluar pertama (FIFO). Programmer dapat menangkap peristiwa ini dalam program dan mengambil tindakan yang tepat sesuai kebutuhan.

Layout dan **View Mode**

Event Listener dari View #2

Ada tiga konsep berikut yang terkait dengan Manajemen Acara Android

- **Event Listener** adalah antarmuka dalam kelas View yang berisi metode callback tunggal.
- **Event Listener Register** adalah proses di mana Penangan Peristiwa didaftarkan dengan Pendengar Peristiwa sehingga penanganannya dipanggil
- **Event Handler** Ketika sebuah peristiwa terjadi dan kita telah mendaftarkan pendengar peristiwa untuk peristiwa tersebut, pendengar peristiwa memanggil Penangan Peristiwa

Layout dan **View Mode**

Jenis-jenis **View**

- TextView
- EditText
- Button
- Image Button
- Date Picker
- RadioButton
- CheckBox buttons
- Image View

Layout dan **View Mode**

Apa itu **ViewGroup**? #1

Sesuai dengan namanya, yaitu **View** dan **Group**. **ViewGroup** memiliki fungsi untuk mengelompokkan berbagai macam **View** / Widget di dalam satu kelompok layout. Sehingga objek-objek yang ada di dalam ini dapat ditata sesuai dari jenis layout yang digunakan.

Di Android, ViewGroup adalah jenis View khusus yang berfungsi sebagai wadah untuk View lain, seperti Tombol, TextView, atau ViewGroup lain. ViewGroup digunakan untuk mengatur dan mengelola tata letak anak View.

Layout dan **View Mode**

Apa itu **ViewGroup**? #2

Dalam struktur bersarang ini, programmer dapat menggunakan parameter tata letak setiap **ViewGroup** untuk mengontrol posisi, ukuran, dan perataan **View** anak. Sebagai contoh, programmer dapat menggunakan LinearLayout sebagai **ViewGroup** induk dan menambahkan dua ViewGroup anak sebagai subkelompok, masing-masing berisi satu atau lebih **View** anak. Programmer kemudian dapat mengatur orientasi **LinearLayout** induk menjadi horizontal atau vertikal untuk mengontrol bagaimana **ViewGroup** anak diatur.

Secara **Default** Android Studio sudah mengimplementasikan satu layout di dalam projek aplikasi yang menjadi **VIewGroup** utama aplikasi.

Layout dan **View Mode**

Jenis **ViewGroup**

- LinearLayout: Sebuah ViewGroup yang mengatur Tampilan anak dalam satu garis horizontal atau vertikal.
- RelativeLayout: Sebuah ViewGroup yang memposisikan anak View relatif terhadap satu sama lain atau terhadap ViewGroup induk.
- FrameLayout: ViewGroup yang menampilkan satu View dalam satu waktu, dengan setiap View baru menggantikan View sebelumnya.
- ConstraintLayout: ViewGroup yang menggunakan batasan untuk memposisikan dan mengukur ukuran anak View relatif terhadap satu sama lain.
- GridLayout: ViewGroup yang mengatur Tampilan anak dalam pola seperti kisi.

Layout dan **View Mode**

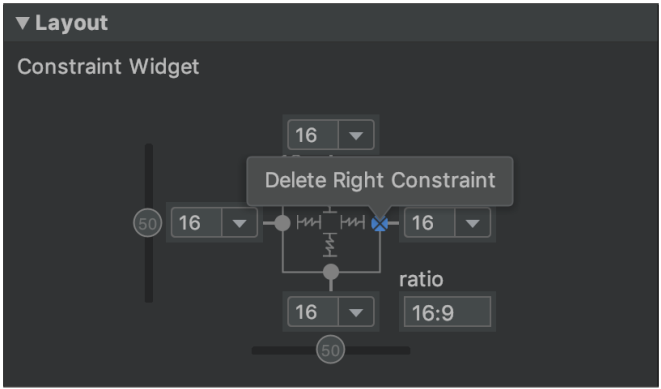
ConstraintLayout Android Studio

Layout jenis ini merupakan layout **default** oleh Android Studio dan memiliki kelebihan-kelebihan dibandingkan layout jenis lainnya. Selain memiliki fleksibilitas dalam menentukan ukuran, letak widget.

Tidak mengatur **Constraint** di masing-masing benda **VIew** / Widget, dapat berakibat larinya benda-benda tersebut ke ujung kanvas **Activity** dan mengakibatkan objek tidak bisa digunakan.

Layout dan **View Mode**

Pengaturan **ConstraintLayout**



Layout dan **View Mode**

Proses *Loading* Layout

Untuk memasukkan data layout ke bahasa pemrograman, dapat dideklarasikan dengan kode. Fungsi **setContentView** akan memanggil file **Resource Layout** dengan nama **main_layout**

Kode

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
}
```

Layout dan **View Mode**

Apa itu **Adapter Layout**?

Jika **Layout** yang dijelaskan sebelumnya mengatur **tata letak objek** yang ada di suatu kanvas **Activity**, maka **Adapter Layout** memiliki tugas untuk mengatur **tata letak data** seperti **Kontak, Pesan**, maupun **Gambar** yang ada di dalam **Gallery**.

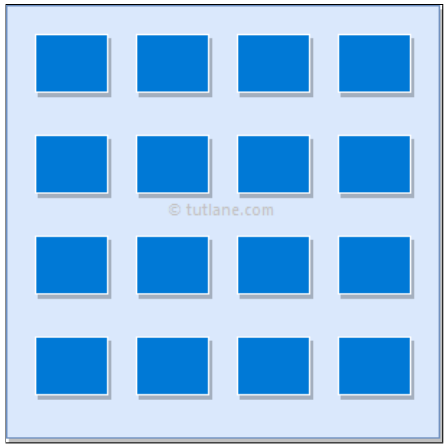
Layout jenis ini hanya memiliki dua jenis saja:

- **Grid Layout** : Menata data dalam bentuk **Grid**
- **List Layout** : Menata dalam bentuk daftar **List**

Layout dan **View Mode**

Grid Layout

Data diatur dengan tata letak a x b sehingga terlihat rapi dan cocok untuk menampilkan **thumbnail** suatu gambar maupun video



Layout dan **View Mode**

List Layout

Data diatur dengan tata letak per baris sehingga cocok untuk menampilkan pesan maupun data kontak


THANK YOU