



BINAR
ACADEMY

Fragment & Fragment Lifecycle

Silver - Chapter 3 - Topic 3

**Selamat datang di Chapter 3 Topic 3 online
course Android Developer dari Binar Academy!**



Hi Teman-Teman 🙋

Pada topik sebelumnya, kita udah belajar konsep pemrograman dengan fitur **Activity dan Activity Lifecycle**.

Nah, di **Topik 3** ini kita bakal mendalami fitur lain yang mirip sama mereka, yaitu **Fragment dan Fragment Lifecycle**.

Yuk, lanjut!



Detailnya, kita bakal bahas hal-hal berikut ini:

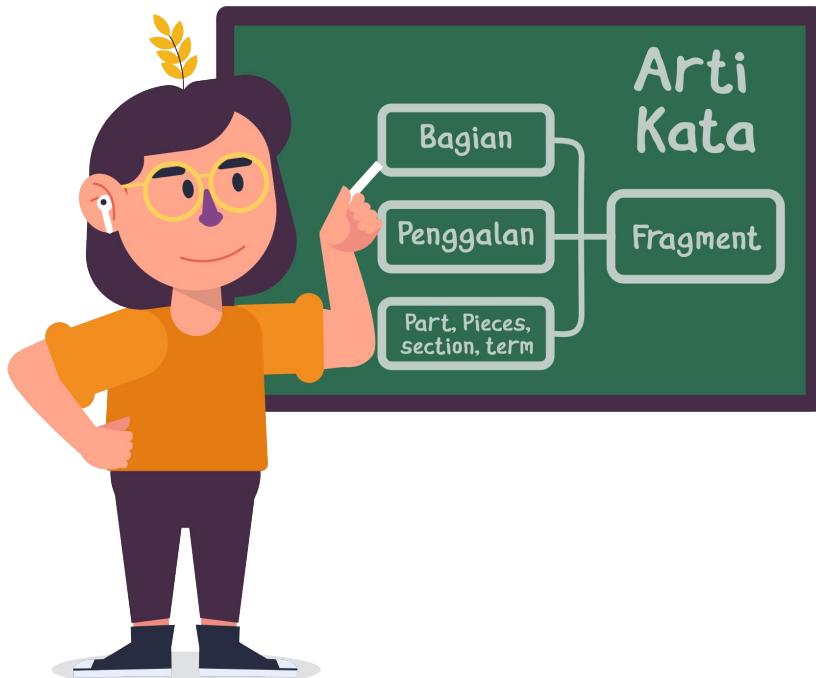
- Konsep Fragment di Android
- Fragment dan Activity
- Fragment Lifecycle
- Cara Membuat Fragment - ViewPager





Satu pertanyaan kita semua saat pertama kali denger kata ini :

Apa sih **Fragment** itu?



Apa itu Fragment?

Fragment itu mewakili **perilaku atau sebagian user interface (UI) di FragmentActivity**.

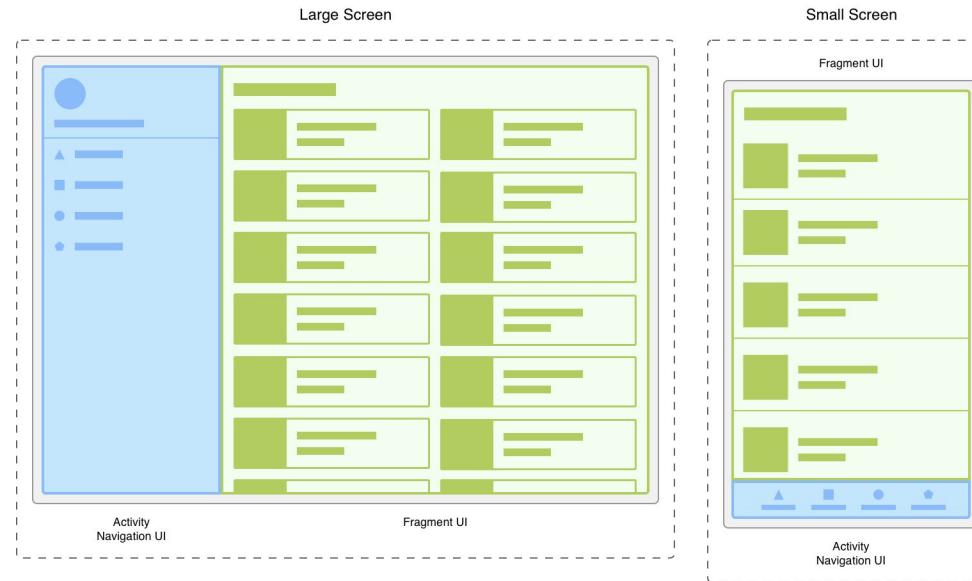
Kita bisa gabungin beberapa Fragment dalam satu Activity untuk membangun UI multi-panel, dan menggunakan kembali Fragment dalam beberapa Activity.



Wujud Fragment, itu kayak gimana sih?

Bisa dilihat di gambar dibawah inih~ Bayangin kamu liat timeline twitter atau liat list profil temen-temen kamu di media sosial. Nah mirip tuh kayak tampilan itu.

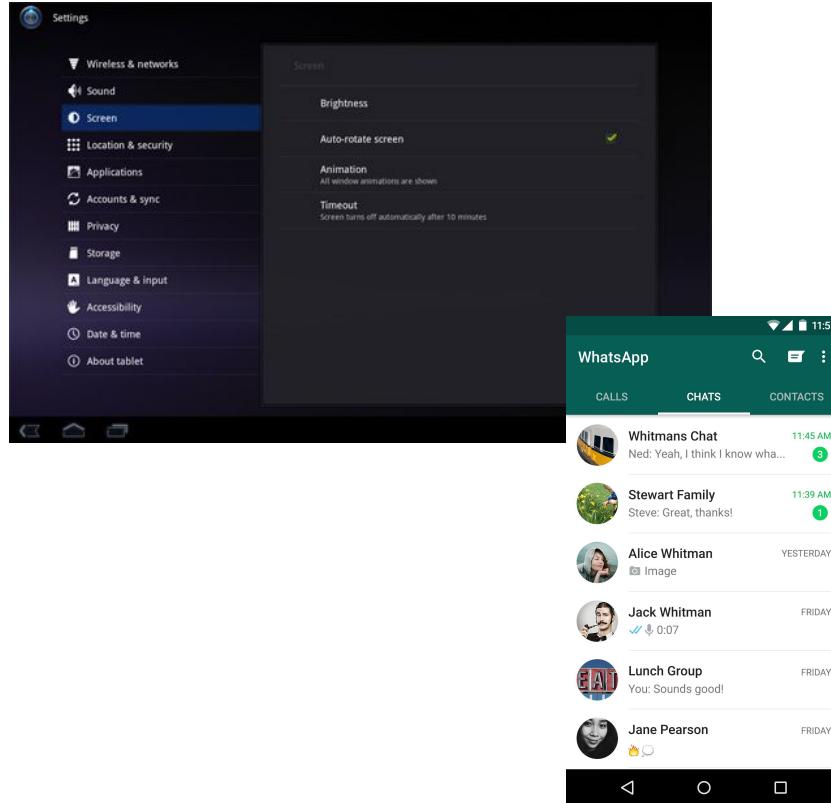
Cuma kalau di HP其实tampilannya bakal kayak gambar sebelah kanan (small screen).





Kita bisa bilang kalo Fragment itu **bagian modular dari suatu Activity**, yang punya karakter kayak gini :

- memiliki siklus hidupnya sendiri,
- menerima inputnya sendiri, dan
- Dapat kita tambahkan atau hapus saat Activity sedang berjalan (seperti "sub Activity" yang bisa kita gunakan kembali dalam berbagai Activity).



Sejarah Fragment

Pertamanya sih, Android memperkenalkan Fragment di Android 3.0 (API level 11). Terutama untuk mendukung desain UI yang lebih dinamis dan fleksibel di layar besar, kayak tablet.

Karena layar tablet jauh lebih besar daripada handphone, otomatis kan jadi lebih banyak ruang tuh untuk menggabungkan dan menukar komponen UI.

Nah debut lah si Fragment ini buat seimbangin layar besar itu.



Biar kebayang, coba pakai analogi, ya!

Masih inget nggak nih sama analogi di topik 2 tentang Activity dan Activity Lifecycle? Fragment hampir mirip tuh sama analogi Activity, beda dikit aja.

Pada Activity, analoginya hanya menggunakan satu foto sebagai tampilan.

Sedangkan pada konsep Fragment, **foto yang digunakan dapat diubah-ubah berdasarkan interaksi pengguna.**



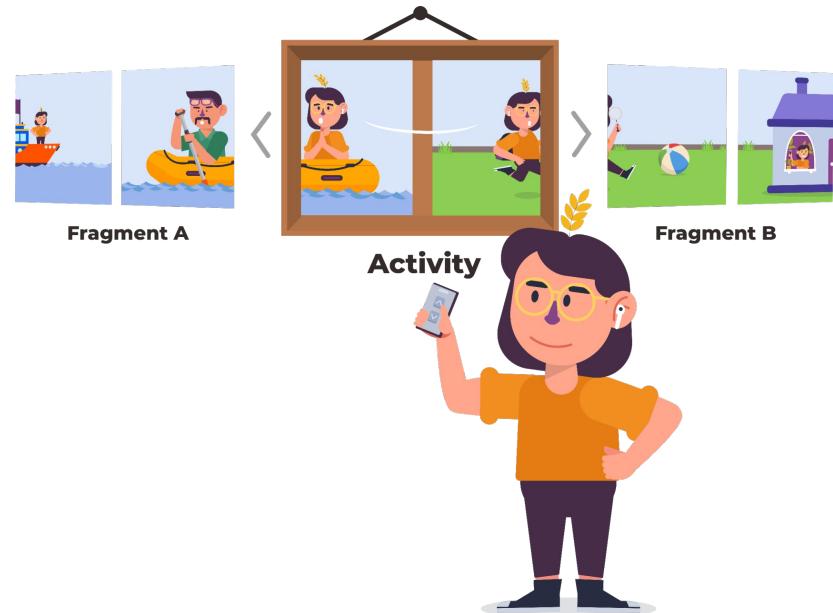


Nah, dari cerita itu, coba kondisinya kamu ganti dengan panduan kayak gini:

Bingkai: Sebagai Activity tempat Fragment tampil.

Foto: Fragment yang dapat diganti dengan Fragment lainnya.

Seperti itu lah Fragment bekerja. Kalo Fragment adalah foto dan Activity itu bingkai fotonya, berarti apakah keduanya harus selalu bersama?





Pasti kamu juga gak asing dengan gambar disamping.

Untuk memunculkan berbagai gambar dalam layar, fragment memberikan developer kebebasan untuk mendesain bentuk view sedemikian rupa.

Tampilan kita pun akan semakin memanjakan mata user.





Dengan analogi bingkai dan foto tadi,
menurut kamu apakah Fragment ini
bisa tampil tanpa Activity?

Kamu keep dulu ya jawabanya, setelah
ini kita bahas faktanya setajam silet.



Kalau tanpa Activity, Fragment bisa jalan nggak?

Sorry to say, nggak bisa...

Fragment nggak bisa tampil tanpa Activity, karena Fragment membutuhkan Activity untuk meng-Attach dirinya sendiri.



Fragment A

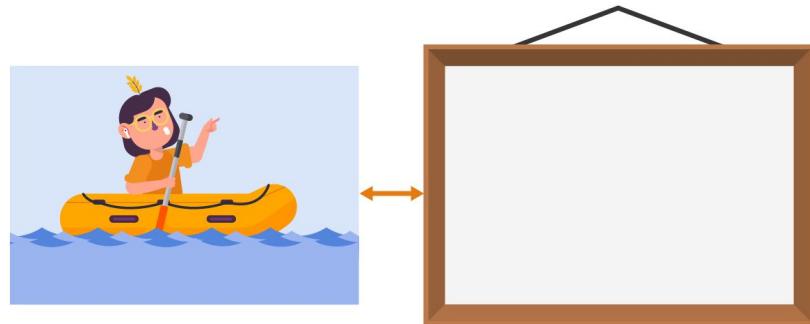
Fragment B

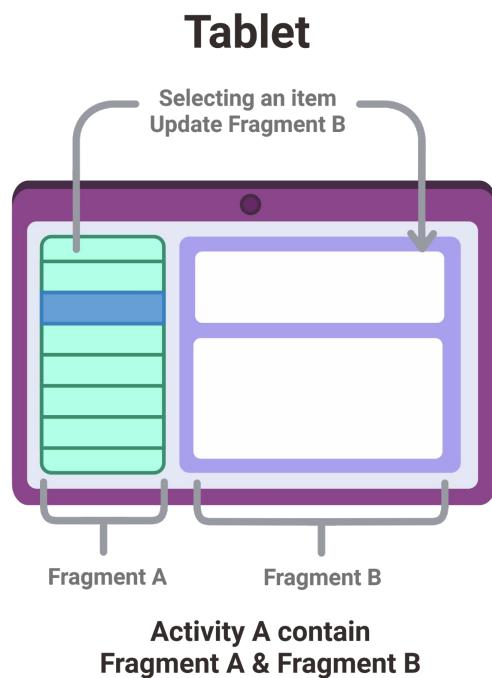




Layaknya analogi sebelumnya yang mengibaratkan bingkai foto sebagai Activity dan lembaran foto sebagai Fragment, kita bisa asumsikan bahwa **lembaran foto tidak bisa tampil di dinding tanpa adanya bantuan bingkai foto.**

Lalu lembaran foto menempelkan (*attach*) dirinya pada bingkai foto.





Contoh penerapan Fragment pada tablet

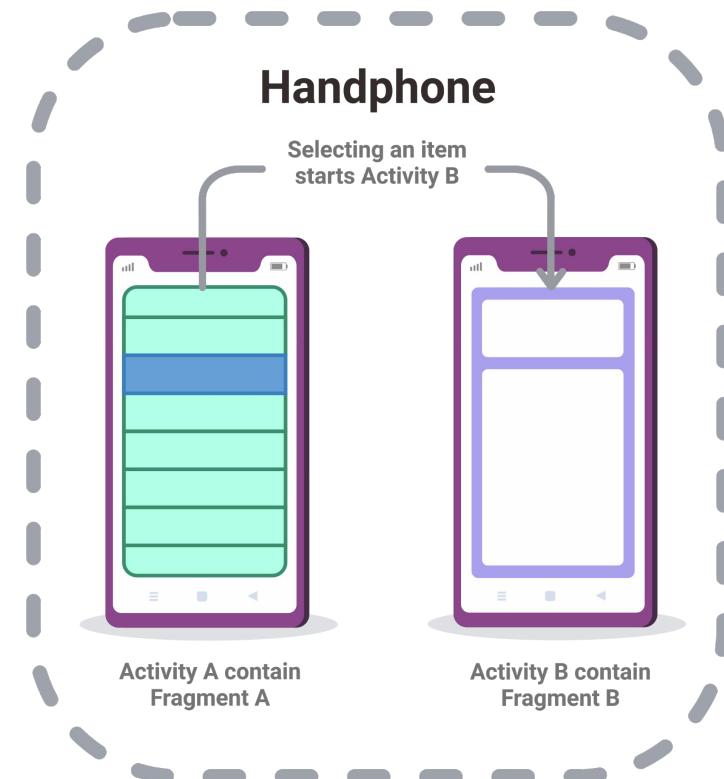
Pada tablet, aplikasi berita bisa menggunakan satu Fragment untuk menampilkan list artikel di sebelah kiri dan Fragment lain untuk menampilkan artikel di sebelah kanan.

Kedua Fragment tersebut muncul dalam satu Activity, berdampingan, dan setiap Fragment memiliki lifecycle method callback masing-masing dan menangani input pengguna mereka sendiri.



Jadi, alih-alih menggunakan satu Activity untuk memilih artikel dan Activity lain untuk membaca artikel, pengguna bisa memilih artikel dan membacanya semuanya dalam Activity yang sama.

Kamu bisa tengok kanan dan kiri dalam satu gelaran yang sama.

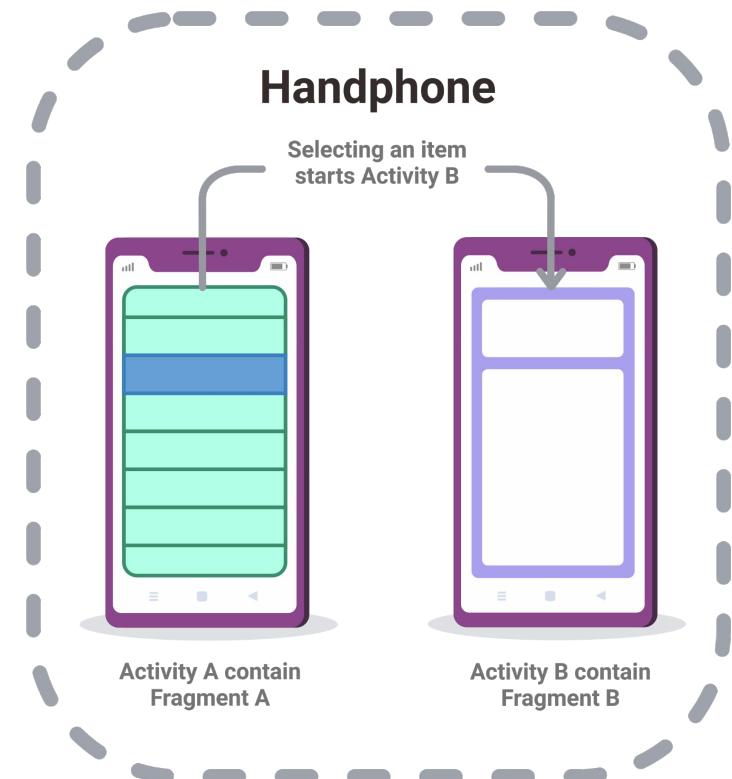


Contoh penerapan Fragment pada Ponsel

Pada contoh aplikasi berita yang sebelumnya, aplikasi dapat menyematkan dua Fragment di Activity A saat dijalankan pada perangkat berukuran tablet.

Hal itu bisa terjadi karena ukuran layar tablet seluas samudera sehingga bisa deh menampilkan dua Fragment yang berbeda pada satu layar.

Namun, pada layar seukuran ponsel, nggak ada ruang yang cukup untuk kedua Fragment.



Jadi, Activity A hanya menyertakan Fragment untuk list artikel. Ketika pengguna memilih artikel, ia baru memulai Activity B yang mencakup Fragment kedua untuk dibaca artikel.

Dengan begitu, aplikasi ini mendukung tablet dan handset dengan menggunakan kembali Fragment dalam kombinasi yang berbeda.



Gimana gengs? cukup kebayang yaa sama **Fragment** dan **Activity** ini.

Terus, kalo gitu apakah **Fragment** juga memiliki **Lifecycle**?



Jawabannya, Tentu aja!

Bahkan Fragment Lifecycle dan Activity Lifecycle sangat berhubungan erat a.k.a bestfriend forever, lho!



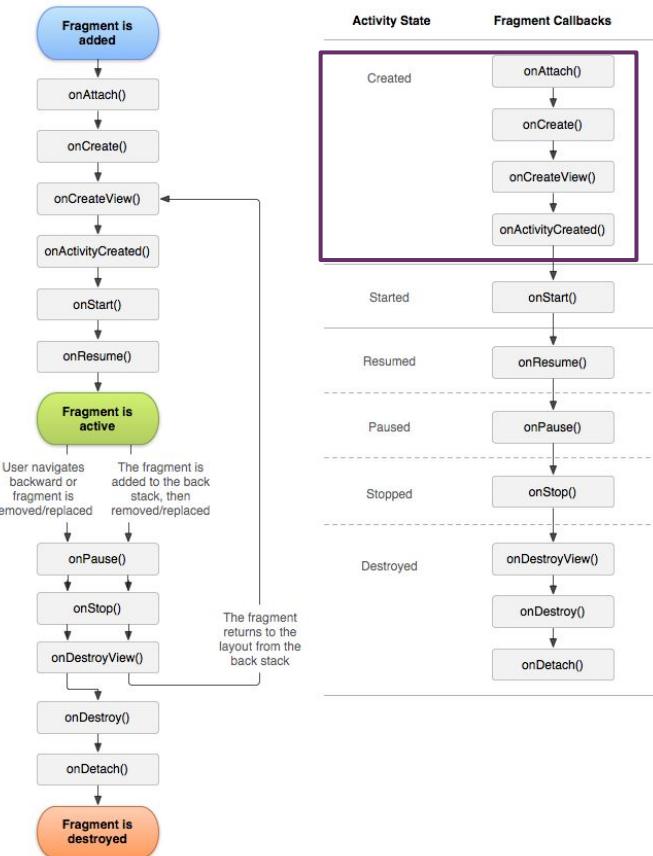


Seperi apa sih hubungannya?

Mari kita lihat hubungannya pada gambar di samping!

Ternyata callback atau state dari Fragment Lifecycle terangkum pada masing-masing Activity state.

Coba deh kamu liat kotak ungu, state **onAttach**, **onCreate**, **onCreateView** dan **onActivityCreated** pada Fragment Lifecycle akan dijalankan setelah state **onCreate** pada Activity Lifecycle.

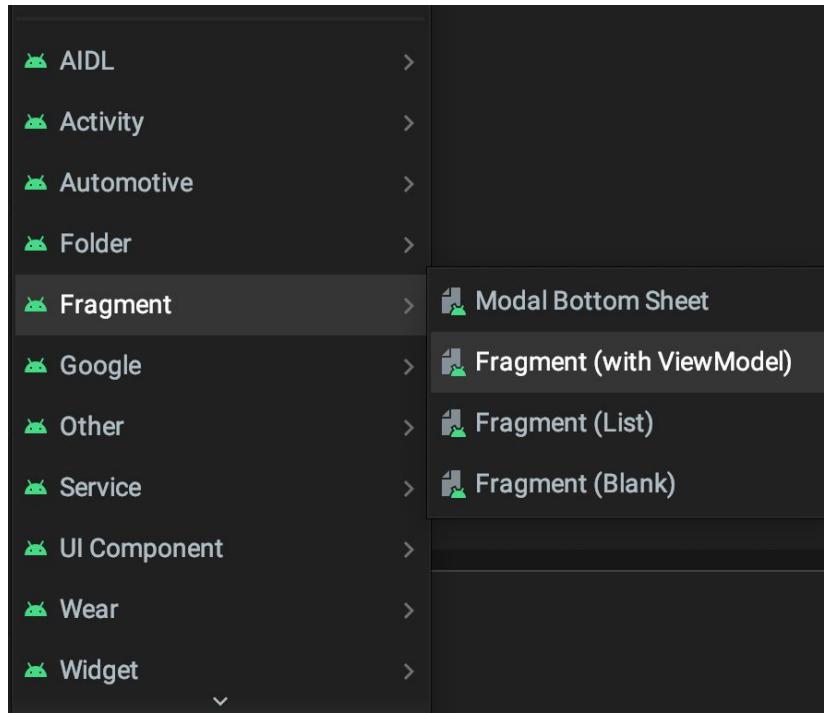




Sekarang coba kita recall, sama siapa Fragment bucin forever? dan sama siapa Fragment cuma bestfriend aja?

Kalo kamu bisa menjawab itu berarti kamu siap belajar to the next level.

Lanjut, gimana caranya membuat Fragment?



Ternyata membuatnya gampang!

Bisa sambil merem deh~ Layaknya membuat Activity, Android Studio udah siapin wizard untuk menjamu Fragment. Kamu hanya perlu :

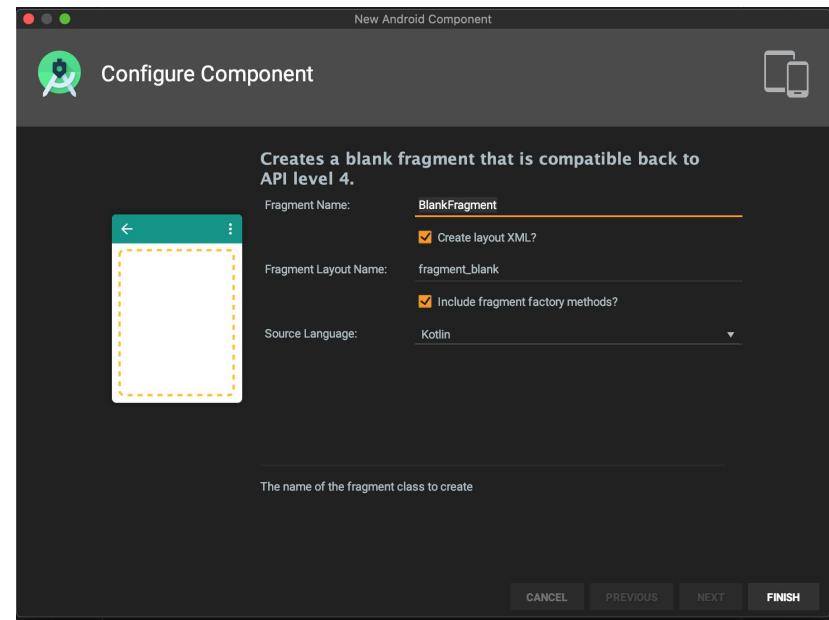
- Klik kanan pada *package* tempat Fragment baru akan dibuat
- Navigasi ke New
- Navigasi ke Fragment

Setelah itu bakal muncul berbagai macam jenis *template* Fragment yang sudah disediakan oleh Android Studio.



Layaknya sebuah Activity, Fragment juga terdiri dari :

- **File Class** dan
- **File Layout XML.**





Yuks, kita lihat code Fragment!

Ketika pertama kali membuat Fragment menggunakan wizard dari Android Studio, maka perlu ditambahin beberapa code template pada Kotlin Class dan File Layout XML Fragment kita.

Masih ingat `setContentView()` pada topic sebelumnya? Yap, method itu digunakan untuk mengaplikasikan Layout XML terhadap file Class Activity.

Bedanya sama Activity, pada Fragment terdapat method `onCreateView` untuk meng-inflate view yang akan digunakan pada Fragment.

```
● ● ●

class BlankFragment : Fragment() {

    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View?
    {
        // Inflate the layout for this Fragment
        return inflater.inflate(R.layout.Fragment_blank, container, false)
    }
}
```



Method Inflate

Method **inflate()** mengambil tiga argumen:

- ID resource dari layout XML yang ingin kita *inflate*.
- ViewGroup menjadi induk dari *inflate layout*. Diisi oleh *container* yang ada pada parameter **onCreateView**.
- Boolean menunjukkan apakah *inflate layout* harus dilampirkan ke **ViewGroup** (parameter kedua) selama proses *inflate*.

```
● ● ●

class BlankFragment : Fragment() {

    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View?
    {
        // Inflate the layout for this Fragment
        return inflater.inflate(R.layout.Fragment_blank, container, false)
    }
}
```



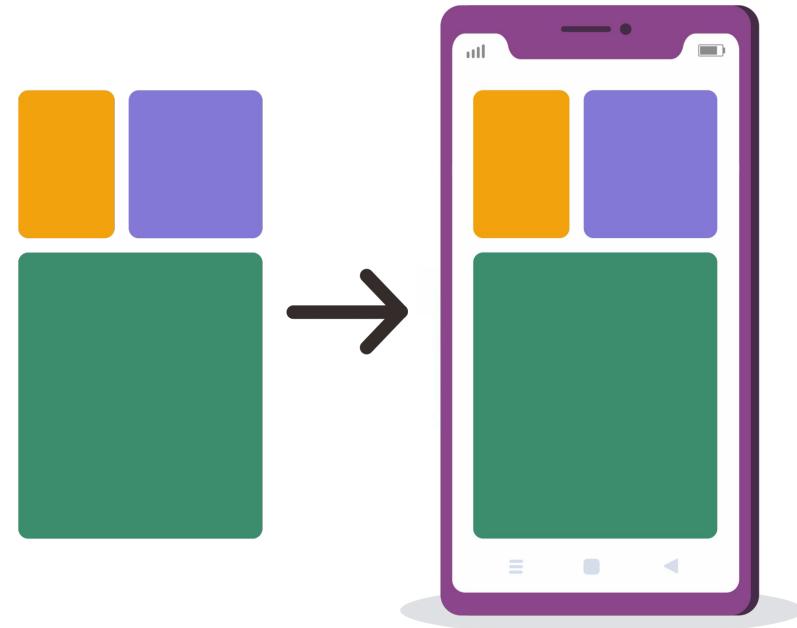
Ciyee.. udah belajar code membuat Fragment.

Next, biar makin membanggakan Desa Android RT 01 RW 03, gimana cara menggunakan Fragment?



Menambahkan Fragment ke Activity

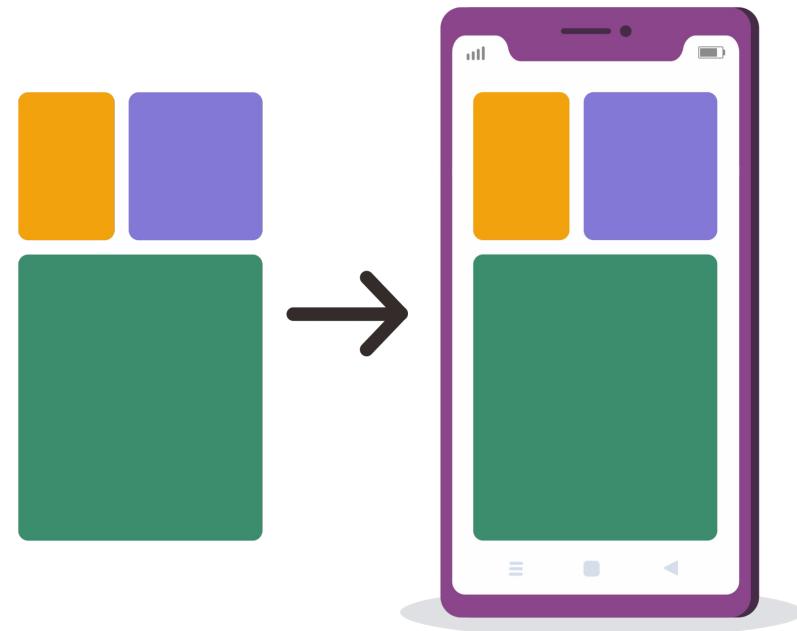
Biasanya, sebuah Fragment menyumbangkan sebagian UI untuk Activity host yang tertanam sebagai bagian dari hierarki tampilan keseluruhan Activity.





Ada dua cara yang bisa kita lakukan untuk menambahkan Fragment ke layout Activity:

1. Deklarasi Fragment di dalam file layout Activity.
2. Secara terprogram tambahkan Fragment ke ViewGroup yang ada.



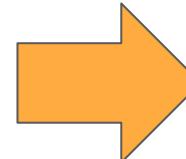


1. Deklarasi fragment di dalam file layout activity

Kita cukup mengubah Layout XML dari Activity kita seperti ini:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Fragment android:name="com.binarchapter5.BlankFragment"
        android:id="@+id/list"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="match_parent" />
    <Fragment android:name="com.binarchapter5.BlankFragment2"
        android:id="@+id/viewer"
        android:layout_weight="2"
        android:layout_width="0dp"
        android:layout_height="match_parent" />
</LinearLayout>
```

Hasilnya :





2. Secara terprogram, tambahkan Fragment ke ViewGroup yang ada

Kapan pun Activity kita lagi berjalan, kita bisa menambahkan Fragment ke layout Activity kita.

Kita hanya perlu menentukan **viewGroup** untuk meletakkan Fragment.



```
val FragmentManager = supportFragmentManager  
val FragmentTransaction = FragmentManager.beginTransaction()
```



Menggunakan Fragment

Untuk melakukan transaksi Fragment dalam Activity, (seperti menambah, menghapus, atau mengganti Fragment), kita **harus menggunakan API dari FragmentTransaction**.

Kita juga bisa mendapatkan instance FragmentTransaction dari FragmentActivity kita seperti contoh disamping 👉



```
val FragmentManager = supportFragmentManager  
val FragmentTransaction = FragmentManager.beginTransaction()
```



Kemudian kita juga bisa menambahkan Fragment menggunakan method `add()`, menentukan Fragment yang akan ditambahkan dan tampilan untuk memasukkannya.

Contohnya bisa kamu lihat di slide setelah ini!



```
val Fragment = ExampleFragment()  
FragmentTransaction.add(R.id.Fragment_container, Fragment)  
FragmentTransaction.commit()
```



Fragment bisa diterapkan menggunakan ViewPager

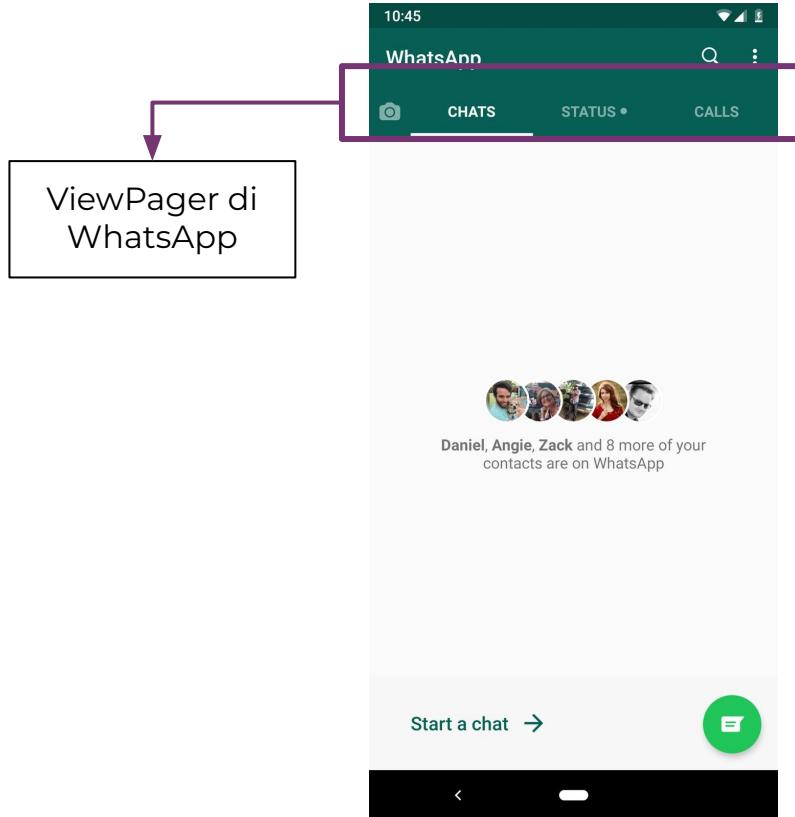
Fragment dengan ViewPager bisa disebut juga dengan screen slide.

Screen slide adalah transisi antara satu layar penuh ke yang lain dan umumnya dengan UI seperti setup wizards atau slideshows.

Contohnya tuh kayak pas kita pake WhatsApp. Coba deh kamu lihat gambar disamping, kerasa familiar nggak sih?

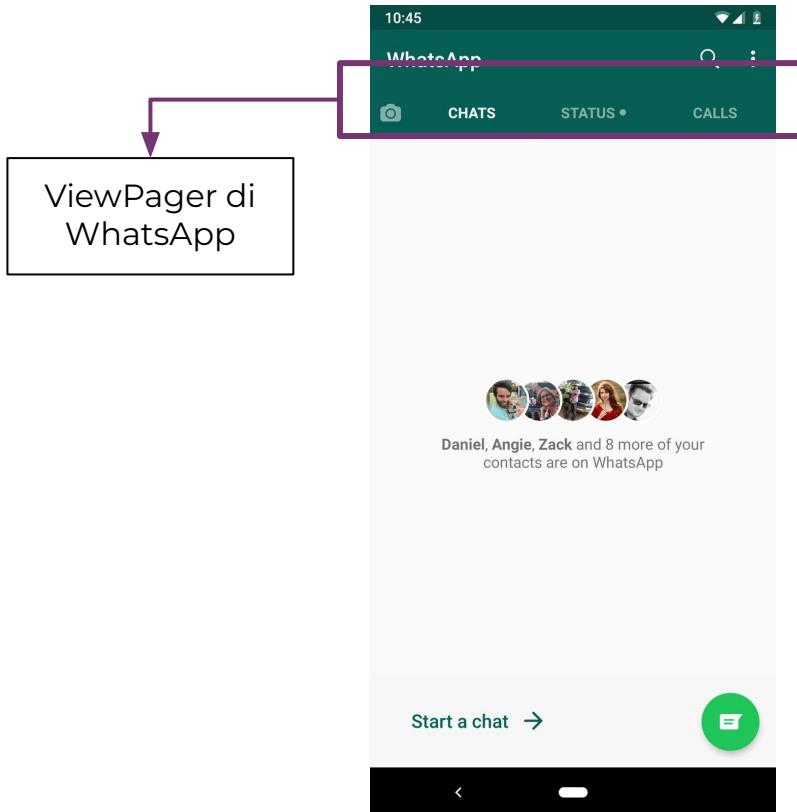


Menggunakan Fragment



Kemampuan Fragment memungkinkan kita melakukan screen slides dengan menggunakan ViewPager yang disediakan oleh library.

Nah, ViewPager sendiri dapat membuat animasi screen slide secara otomatis. Jadi kita nggak usah pusing mikirin animasi transisi slide-nya.



Aplikasi yang cukup terkenal yang menerapkan Fragment dengan ViewPager adalah WhatsApp. Kita bisa memindahkan screen antar layar chat, status, dengan log panggilan hanya dengan geser-geser.



“Lalu gimana nih membuat **Fragment yang bisa menerapkan **ViewPager**? ”**

Tenang~ abis ini kamu siap-siap, kita langsung berangkat buat belajar langkah-langkahnya!

Fragment yang diterapkan menggunakan **ViewPager** bisa dibuat dengan beberapa langkah :

1. **Membuat Fragment**
2. **Tambahkan ViewPager ke Layout**
3. **Membuat Adapter untuk ViewPager**





1. Membuat Fragment

Pertama, buat dulu class Fragment yang mengembalikan layout yang baru saja kita buat di method `onCreateView()`.

Kemudian kita dapat membuat *instance* Fragment ini dalam Activity *parent* kapan pun kita membutuhkan halaman baru untuk ditampilkan kepada pengguna.

Seperti yang ada pada gambar.

```
class ScreenSlidePageFragment : Fragment() {
    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View = inflater.inflate(R.layout.Fragment_screen_slide_page, container, false)
```



2. Tambahkan ViewPager ke Layout

Object ViewPager memiliki fungsi *swipe* bawaan untuk melakukan transisi melalui halaman-halaman dan mereka akan menampilkan animasi *screen slide* secara *default*, jadi kita nggak perlu membuat animasi sendiri.

ViewPager menggunakan objek **PagerAdapter** sebagai fungsi untuk halaman baru ketika ditampilkan, sehingga **PagerAdapter** akan menggunakan Class Fragment yang kita buat sebelumnya.



```
<!-- Activity_screen_slide.xml -->
<android.support.v4.view.ViewPager
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/pager"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```



3. Membuat Adapter untuk ViewPager

Nah, Class inilah yang bertugas untuk membuat dan mengatur ViewPager untuk menampilkan apa yang kita inginkan.

Dalam kasus ini, kita kan menampilkan Fragment dalam ViewPager, jadi ViewPager inilah yang akan menampung Fragment-Fragment yang akan ditampilkan.



```
// Pager adapter akan menyediakan halaman ke view pager widget.  
  
val pagerAdapter = ScreenSlidePagerAdapter(supportFragmentManager)  
mPager.adapter = pagerAdapter  
  
private inner class ScreenSlidePagerAdapter(fm: FragmentManager) : FragmentStatePagerAdapter(fm) {  
    override fun getCount(): Int = NUM_PAGES  
    override fun getItem(position: Int): Fragment = ScreenSlidePageFragment()  
}
```



Biar paham, Yuk ikut Challenge Hari ini~

Buatlah suatu Aplikasi yang menggunakan beberapa Fragment, bisa menggunakan 2 Fragment atau lebih dengan menggunakan variasi selain menerapkan Fragment dengan ViewPager.

Beberapa pilihannya antara lain (Pilih salah satu) :

- Navigation Drawer
- Bottom Navigation

Untuk referensi, bisa cek link berikut yah:

- [Navigation Drawer in Android - GeeksforGeeks](#)
- [Bottom Navigation Bar in Android - GeeksforGeeks](#)

Saatnya kita Quiz!





1. Method apa yang berfungsi untuk menerapkan suatu Layout XML pada suatu Fragment?

- A. inflate()
- B. setContentView()
- C. onActivityCreated()



1. Method apa yang berfungsi untuk menerapkan suatu Layout XML pada suatu Fragment?

- A. inflate()
- B. setContentView()
- C. onActivityCreated()

Method inflate() dipanggil dalam method onCreateView(). Method ini digunakan untuk binding atau “meniupkan” Layout XML dengan Fragment.



2. Apakah Fragment bisa berdiri sendiri tanpa Activity?

- A. Bisa
- B. Tidak bisa
- C. Bisa dengan syarat



2. Apakah Fragment bisa berdiri sendiri tanpa Activity?

- A. Bisa
- B. Tidak bisa
- C. Bisa dengan syarat

Fragment membutuhkan Activity untuk “disinggahi” atau di-attach jadi nggak bisa jalan sendiri tanpa Activity.



3. Untuk melakukan navigasi Fragment seperti menambah, menghapus, mengubah Fragment bisa menggunakan?

- A. Intent
- B. startActivityForResult
- C. FragmentTransaction



3. Untuk melakukan navigasi Fragment seperti menambah, menghapus, mengubah Fragment bisa menggunakan?

- A. Intent
- B. startActivityForResult
- C. FragmentTransaction

Mengolah Fragment secara dinamis menggunakan code memerlukan `FragmentTransaction`.
`FragmentTransaction` didapatkan dari `FragmentManager.beginTransaction()` .
Sedangkan `FragmentManager` didapatkan dari `supportFragmentManager`.



4. Untuk membuat tampilan “ViewPager”, dibutuhkan sintaks berikut, dimana fungsinya adalah untuk

- A. Membuat Fragment
- B. Menambahkan ViewPager dalam Layout
- C. Membuat Adapter untuk ViewPager



```
<!-- Activity_screen_slide.xml -->
<android.support.v4.view.ViewPager
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/pager"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```



4. Untuk membuat tampilan “ViewPager”, dibutuhkan sintaks berikut, dimana fungsinya adalah untuk

- A. Membuat Fragment
- B. Menambahkan ViewPager dalam Layout
- C. Membuat Adapter untuk ViewPager



```
<!-- Activity_screen_slide.xml -->
<android.support.v4.view.ViewPager
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/pager"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

ViewPager memungkinkan memungkinkan mengubah tampilan hanya dengan slide. Tidak hanya layar, ViewPager juga bisa digunakan untuk membuat carousell atau slideshow di layar Home Page kita, lho!



5. Manakah method yang hanya ada pada *fragment*?

- A. onDetach
- B. onStop
- C. onStart



5. Manakah method yang hanya ada pada *fragment*?

- A. onDetach
- B. onStop
- C. onStart

Method `onDetach` adalah method yang hanya ada pada Fragment Lifecycle. Berarti Activity lifecycle tidak memiliki method ini.

Method `onDetach` dijalankan ketika suatu fragment dilepaskan terhadap activity atau container yang menampungnya. Selain itu, `onDetach` juga akan dijalankan jika orientasi layar berubah.

Referensi dan bacaan lebih lanjut~

1. [Fragment | Android Developers](#)
2. [Navigation Drawer in Android - GeeksforGeeks](#)
3. [Bottom Navigation Bar in Android - GeeksforGeeks](#)





Nah, selesai sudah pembahasan kita di **Chapter 3 Topic 3** ini.

Selanjutnya, kita bakal bahas tentang **Intent, Bundle, Serializable, dan Parcelable**.

Penasaran kayak gimana? Cus langsung ke topik selanjutnya~



Terima Kasih!



Next Topic

loading...