#Kemampuan Akhir Yang Direncanakan

- Mahasiswa Mampu membuat Aplikasi dengan berbagai macam layout baik Linear Layout, Relative Layout dan Constraint Layout lengkap dengan elemen elemen user interface yang umum digunakan.

Modul

A. Tujuan

Pada modul ini akan membahas tentang activity, dengan tujuan agar mahasiswa:

- Mengetahui konsep tentang Activity Lifecycle
- Mampu membuat Aplikasi dengan berbagai macam layout baik Linear Layout, Relative Layout, dan Constraint Layout lengkap dengan elemen elemen user interface yang umum digunakan.

B. Dasar Teori

1. Komponen UI (User Interface)

Secara umum Arsitektur user interface (UI) pada aplikasi android adalah user interface yang meliputi Activity dan user interface yang terdiri dari komponen. Semua yang berhubungan dengan user interface pada aplikasi android biasanya berada pada lokasi **res/layout/filename.xml** dimana coding java untuk memanggilnya dikenal dengan **R.layout.filename**.

Komponen-komponennya adalah sebagai berikut:

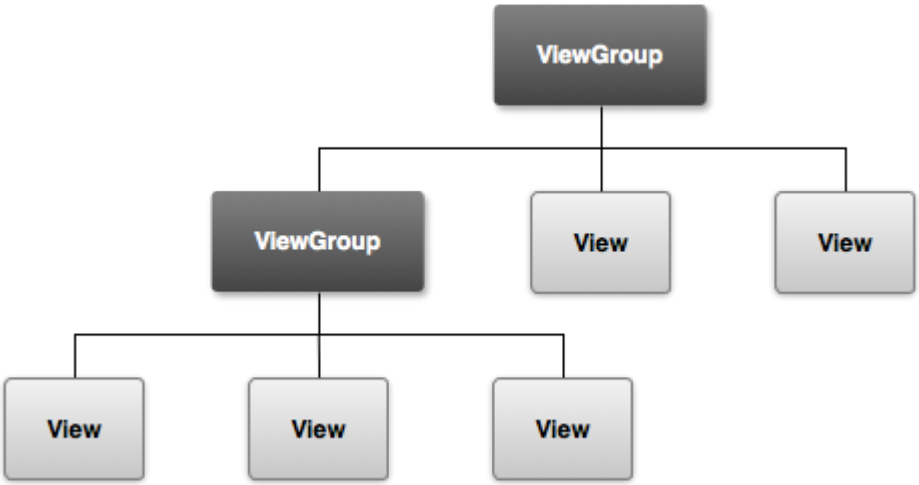
a) <**ViewGroup**>

- Kumpulan view yang dapat menentukan tata letak komponen view secara berbeda, seperti LinearLayout, RelativeLayout, FrameLayout, serta Tabulasi.
- ViewGroup berfungsi sebagai kontainer view (tampilan). Hubungan ViewGroup dengan view adalah *root - child*. ViewGroup yang umum digunakan adalah:
    - **ScrollView**

      ViewGroup yang memungkinkan tampilan layar digulirkan dari atas kebawah dan sebaliknya.

    - **RecyclerView**

      ViewGroup yang berisi list of view (daftar tampilan) atau ViewGroup lainnya yang memungkinkan penggulirannya menambahkan atau menghilangkan tampilan secara dinamis dari layar.
- Contoh gambaran hirarki ViewGroup dengan View adalah berikut



- Beberapa ViewGroup disebut sebagai Layout karena ViewGroup tertentu mengelola *child* dan secara umum digunakan sebagai *root*. Contoh Layout tersebut diantara lain:
    - **LinearLayout**

      ViewGroup yang kontennya sejajar baik horizontal maupun vertikal. Konsep LinearLayout adalah berikut:

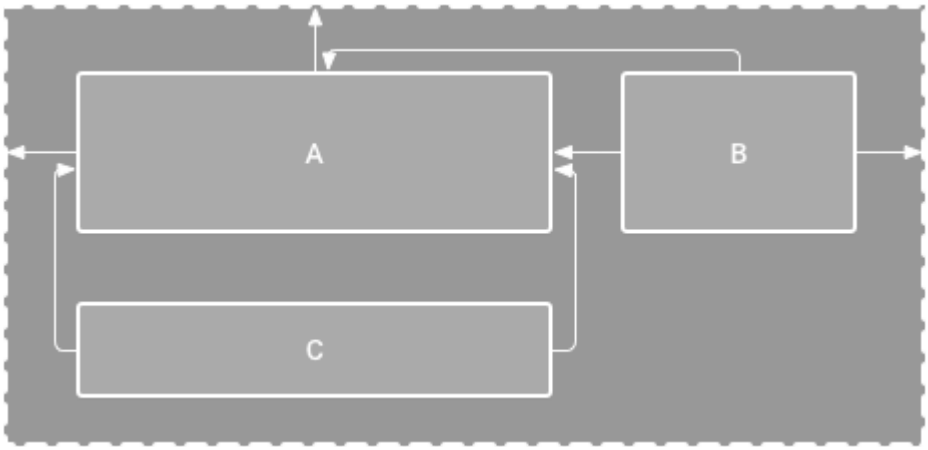
      

    - **RelativeLayout**

      ViewGroup yang kontennya dapat diposisikan pada posisi yang relatif (tidak statis). Setiap View dapat diletakkan di kanan, kiri, bawah, atau atas dari View lainnya. Konsep RelativeLayout adalah berikut:

- **ConstraintLayout**

  ViewGroup yang susunan kontennya sangat fleksibel dan mudah diatur tanpa adanya *nested view group*, lebih mudah dibandingkan dengan RelativeLayout yang konten Viewnya bergantung pada hubungan parent-sibling. Gambaran ConstraintLayout adalah berikut:



- Atribut dari ViewGroup biasanya terdiri dari:
  - **android:id**
  - **resource id**, yang berisi variabel unik dari element tersebut.
  - **android:layout_height**

    dimensi valuenya (height) yang diikutinya dengan opsi “fill_parent” atau “wrap_content”
  - **android:layout_width** dimensi valuenya (width) yang diikutinya dengan opsi “fill_parent” atau “wrap_content”
- Tampilan wrap_content akan membuat layout untuk elemen tersebut selebar dan setinggi tulisan dari elemen tersebut.

- fill_parent akan membuat tulisan mengisi layar secara penuh (layar yang tersedia space untuk elemen tersebut)

b) <**View**>

- Sama seperti <**ViewGroup**> tetapi <**View**> lebih dikenal dengan “individual UI component”, atributnya minimal terdiri dari tiga yang sama persis dengan atribut yang dimiliki oleh <**ViewGroup**>.
- Contoh elemen View antara lain
    - Button
    - TextView
    - EditText

c) <**requestFocus**>

- Element kosong yang bisa didefinisikan didalam <**View**>.

d) <**include**>

- Memasukkan file layout ke dalam layout. Atributnya sama dengan <**ViewGroup**> dan <**View**> tetapi ada satu tambahan atribut yaitu <**resource**> yang berfungsi untuk menentukan file layoutnya.

2. Android Manifest (AndroidManifest.xml)

- File AndroidManifest.xml diperlukan oleh setiap aplikasi android, file ini berada pada folder *root* aplikasi.
- File ini mendeskripsikan variabel global dari paket aplikasi yang digunakan.
- File ini berfungsi untuk mendeskripsikan resource apa saja yang akan digunakan oleh project, seperti koneksi internet, SMS, mengakses GPS, dll.
- Berikut adalah contoh file AndroidManifest.xml:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.hexavara.khairy.exerciseapp">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
```

```
        </activity>
    </application>

</manifest>
```

Beberapa elemen yang terdapat dalam file AndroidManifest.xml ini adalah berikut:

a. <**manifest**>

Root utama dalam AndroidManifest.xml, berisi atribut package aplikasi serta package activity dalam project.

```
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.hexavara.khairy.exerciseapp">
```

b. <**uses-permission**>

Berisi perijinan akses seperti penggunaan akses internet, kamera, sms, dan lain-lain.

```
<uses-permission android:name="android.permission.INTERNET"/>
```

c. <**permission**>

Menjelaskan batasan tentang user permission/security permission.

d. <**instrumentation**>

Mendeklarasikan komponen instrumen yang tersedia untuk menguji fungsionalitas dari paket aplikasi yang digunakan dalam aplikasi android.

e. <**application**>

Elemen root yang berisi deklarasi aplikasi android.

f. <**intent-filter**>

Mendeklarasikan intent yang dibutuhkan oleh aplikasi android yang digunakan. Atribut-atribut bisa diberikan disini untuk mensupply label, icon, data dan informasi yang digunakan dalam aplikasi android.

g. <**action**>

Berisi tentang action type yang didukung oleh komponen-komponen yang berada dalam aplikasi android.

h. <**category**>

Mendeklarasikan kategori-kategori yang didukung oleh aplikasi android.

i. <**data**>

Mendeklarasikan tipe MIME, URL, authority penggunaan URL serta penentuan path yang digunakan dalam URL.

j. <**meta-data**>

Mendeklarasikan meta data yang dibutuhkan sebagai tambahan data yang ada yang digunakan dalam aplikasi android.

k. <**receiver**>

Mendeklarasikan dimana aplikasi diberikan informasi mengenai sesuatu perubahan atau aksi yang terjadi, seperti menerima SMS.

l. <**service**>

Mendeklarasikan komponen yang dapat berjalan sebagai service (berjalan di background).

m. <**provider**>

Mendeklarasikan komponen-komponen yang mengelola data dan mempublikasikannya untuk dikelola/dipakai oleh aplikasi lain.

n. <**uses-sdk uses-sdk android:minSdkVersion = ?**>

Mendeklarasikan versi sdk android minimum yang digunakan.

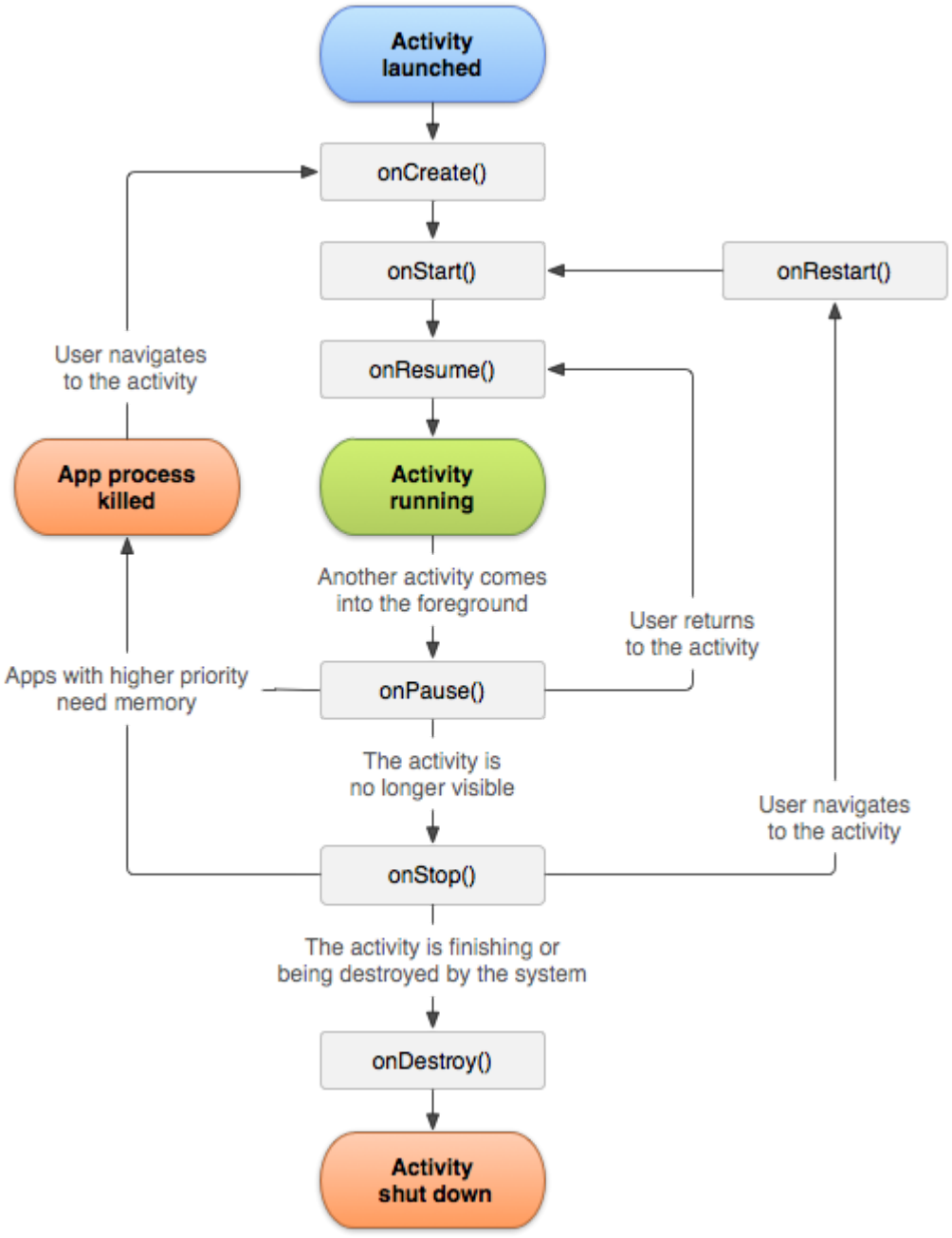
3. Konsep Activity

- Activity merupakan public class dalam aplikasi Android.
- Setiap activity merupakan sesuatu yang unik atau single, yang ditujukan untuk menghandle macam-macam hal yang bisa dilakukan oleh user.
- Activity berhubungan dengan user dimana activity menciptakan windows atau UI yang mana ditampilkan dengan concepts setContenView (View).
- Ada dua method yang pasti dimiliki oleh satu activity, yaitu:
    - onCreate untuk menginisiasi suatu activity. Biasanya dipanggil dengan perintah:
        - setContentView(int) untuk resource yang didefinisikan di layout UI
        - findViewById(int) untuk memanggil widget yang dibutuhkan UI untuk berinteraksi dengan aplikasi.
    - onPause untuk menyatakan ketika user meniggalkan suatu activity.
- Untuk penggunaan dengan Context.startActivity(), semua kelas activity harus sesuai dengan <**activity**> yang dideklarasikan dalam suatu paket di AndroidManifest.xml.

4. Siklus Activity

- Activity aplikasi android dikelola dengan sistem yang dikenal dengan activity stack.
- Ketika activity start, activity diletakkan pada stack yang paling atas dan activity-activity yang sudah jalan berada dibawahnya dan akan terus berada pada posisi atas stack sampai muncul activity yang baru.
- 4 keadaan yang dimiliki activity, yaitu:

    - Active/running Jika activity berada pada posisi atas stack.
    - Pause Jika activity tidak dipakai atau dibutuhkan pada suatu saat tertentu, tetapi activity itu masih ada atau visible, ketika activity baru yang ditangani oleh sistem activity yang lama disebut pause dan masih berada di memory, bisa jadi suatu activity yang sudah keadaan pause tidak ada di memori yang kemungkinan disebabkan oleh keterbatasan memory.
    - Stopped Jika activity sudah tidak dipakai dan digantikan oleh activity lain, activity yang sudah stopped tidak akan pernah dipanggil lagi, dan secara permanen memory pun tidak menyimpan info mengenai activity ini.
    - Restart Jika activity paused atau stopped, sistem dapat mendrop activity ini dari sistem memory, dan ketika user membutuhkan activity tersebut, activity akan kembali ke keadaan awal (restart).
- Berikut ini adalah gambaran mengenai activity lifecycle atau siklus activity di dalam android sistem.



- Terdapat 3 perulangan yang mungkin dialami oleh activity:

    - Entire lifetime Yaitu activity yang terjadi mulai dari onCreate() sampai dengan onDestroy(). Biasanya activity ini akan dibuat setup global ketika mendefinisikannya,
    - Visible lifetime Yaitu activity yang terjadi mulai dari onCreate() sampai dengan onStop().
    - Foreground lifetime Yaitu activity yang terjadi diantara onResume() dan onPause().
- Sehingga sintaks superclass dari gambar diatas adalah sebagai berikut:

```
public class Activity extends ApplicationContext() {
    protected void onCreate(Bundle saveInstanceState);
    protected void onStart();
    protected void onRestart();
    protected void onResume();
    protected void onPause();
    protected void onStop();
    protected void onDestroy();
}
```

Activity Method	Deskripsi	Next
onCreate()	Ketika sebuah activity dibuat, pada method ini dilakukan inisialisasi seperti create view, list data, dll.	onStart()
onRestart()	Ketika sebuah activity dihentikan, dan merupakan prioritas untuk memanggil activity itu kembali.	onStart()
onStart()	Ketika sebuah activity dipanggil sebelum diperlihatkan ke user.	onResume() / onStop()
onResume()	Ketika sebuah activity start/mulai melakukan interaksi dengan user, pada saat ini activity berada pada posisi teratas dari activity stack yang mana user akan melakukan input.	onPause()
onPause()	Ketika sebuah activity lainnya dipanggil/dimulai, method ini digunakan ketika data tidak harus disimpan ke dalam sistem secara permanen.	onResume() / onStop()
onStop()	Ketika sebuah acivity tidak lagi dibutuhkan/tidak terlihat lagi oleh user.	onRestart() / onDestroy()
onDestroy()	Ketika sebuah activity secara permanen tidak lagi dibutuhkan (activity dihancurkan). Dapat juga dilakukan dengan fungsi finish() / isFinishing().	-

Tugas

1. Buat project baru di Android Studio dengan format nama project Week02KelasNama

2. Pada MainActivity.java, ubah AppCompatActivity menjadi Activity
3. Tekan Ctrl+O untuk Override/Implementasi Method. Kemudian Pilih onStart, onPause, onResume, onStop, onRestart, onDestroy, lalu klik OK dengan menekan tombol Shift
4. Lengkapi masing-masing method dengan Toast untuk menampilkan status tiap methodnya
5. Buat tampilan pada project anda dengan mengimplementasikan salah satu dari LinearLayout atau RelativeLayout dengan memasukkan elemen-elemen seperti Button, TextView, dan EditText. Buat dengan konten sekreatif mungkin!