

# Android Development

Riky Ahmad Fathoni



# Riky Ahmad Fathoni

- Telegram : @rikyahmad
- LinkedIn : linkedin.com/in/riky
- Facebook : fb.com/rikyahmadf
- Email : [riky.fathoni@gmail.com](mailto:riky.fathoni@gmail.com)



# Agenda

- Pengenalan Android
- Android Compability
- Struktur Project
- Manifest File
- Resource
- Screen Compability
- Activity
- Fragment
- View
- View Binding
- Layout
- Intent
- Recycler View
- Rest Api
- Debugging
- Testing
- Profiling
- Quiz

---

# Pengenalan Android



# Sejarah Android

- Android adalah sistem operasi berbasis Linux yang dirancang untuk perangkat layar sentuh seperti telepon pintar atau tablet.
- Android awalnya dikembangkan oleh Android Inc, Android Inc didirikan oleh Andy Rubin pada tahun 2003.
- Google mengakuisisi Android Inc pada tahun 2005, dan menjadikan anak perusahaan sepenuhnya yang dimiliki oleh Google.
- Sistem operasi Android dirilis secara resmi pada tahun 2007, dan ponsel Android pertama mulai dijual pada tahun 2008.
- <https://www.android.com/?hl=id>.



# Versi Android

- Layaknya sistem operasi, Android sendiri sudah berkembang sejak pertama kali dikenalkan.
- Ketika materi ini dibuat, Android sudah mendukung versi 13.
- Seperti yang kita tahu, bukan berarti semua perangkat sudah menggunakan Android versi 13, bisa jadi beberapa perangkat tidak mendukung sistem operasi Android versi terbaru.

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.1 Jelly Bean	16	
4.2 Jelly Bean	17	99,9%
4.3 Jelly Bean	18	99,7%
4.4 KitKat	19	99,7%
5.0 Lollipop	21	98,8%
5.1 Lollipop	22	98,4%
6.0 Marshmallow	23	96,2%
7.0 Nougat	24	92,7%
7.1 Nougat	25	90,4%
8.0 Oreo	26	88,2%
8.1 Oreo	27	85,2%
9.0 Pie	28	77,3%
10. Q	29	62,8%
11. R	30	40,5%
12. S	31	13,5%



# Android Development

- Saat pertama kali dikenalkan untuk membuat aplikasi di Android, masih menggunakan bahasa pemrograman Java.
- Namun bukan berarti semua fitur pemrograman Java bisa digunakan, karena Android saat ini hanya mendukung versi maksimal Java 8.
- Android saat ini hanya mendukung bahasa pemrograman Java dan kotlin.
- <https://developer.android.com/?hl=id>.



# Android SDK

- Saat kita membuat aplikasi Android menggunakan Android Studio, maka kita membutuhkan Android SDK (Software Development Kit).
- Android SDK akan terinstall secara otomatis ketika pertama kali kita membuat project, namun. kadang kita ingin menambah fitur atau melakukan update terhadap Android SDK yang kita install.
- Kita bisa menggunakan Android Studio untuk melakukan management Android SDK.





# Android Studio

- Android Studio adalah Integrated Development Environment (text editor) yang disediakan oleh Google untuk membuat aplikasi Android.
- Android Studio sendiri dibangun di atas JetBrains IntelliJ IDEA, sehingga akan sangat familiar untuk programmer yang sebelumnya sudah menggunakan JetBrains IntelliJ IDE.
- Android Studio bisa didapatkan dengan gratis.
- <https://developer.android.com/studio?hl=id>



# Instalasi Android Studio

- Download Android Studio sesuai sistem operasi yang digunakan melalui situs :  
<https://developer.android.com/studio>.
- Instalasi Android Studio sesuai sistem operasi dapat dilihat melalui situs :  
<https://developer.android.com/studio/install>.

---

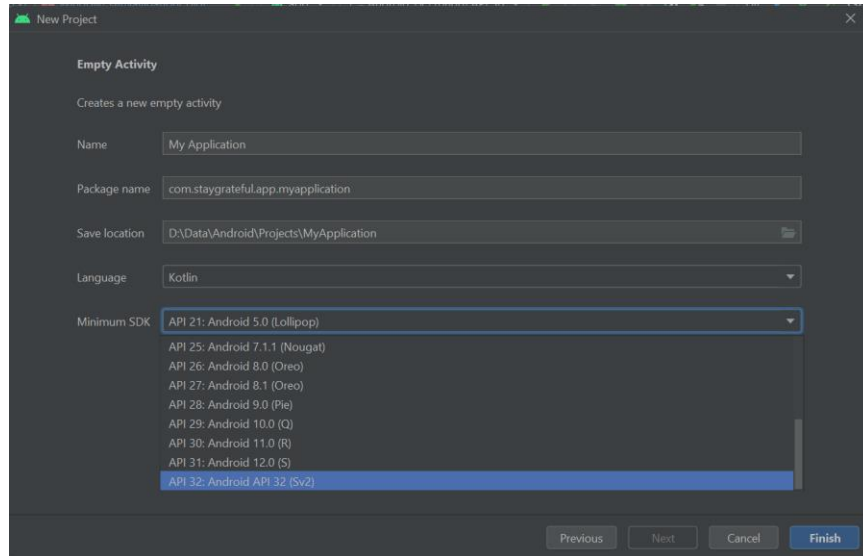
# Android Compatibility



# Android Compatibility

- Karena Android adalah sistem operasi yang Open Source, maka banyak sekali vendor yang membuat device yang menggunakan sistem operasi Android
- Dari mulai smartphone, tablet, smart tv sampai dashboard untuk mobil
- Oleh karena itu, kita perlu berhati-hati untuk memastikan kode program kita bisa berjalan di device yang berbeda
- Salah satu yang paling penting adalah, memilih Android API Level yang ingin kita gunakan
- Saat kita membuat aplikasi Android, kita perlu menentukan API Level minimal yang akan kita gunakan, hal ini dilakukan untuk memastikan aplikasi kita bisa berjalan dengan baik pada sistem operasi Android yang menggunakan API Level tersebut

# Gambar API Level





# Menentukan API Level

- Semakin tinggi API Level yang kita pilih, semakin banyak fitur yang bisa digunakan, tapi semakin sedikit juga device yang sudah mendukung.
- Oleh karena itu, kita perlu hati-hati menentukan API Level minimal yang akan kita gunakan untuk membuat aplikasi Android.
- Salah satu yang paling mudah, kita bisa melihat statistic pengguna device Android berdasarkan API Level nya.



# Android Release Note

- Untuk melihat daftar fitur apa saja yang terdapat di versi Android tertentu, kita bisa melihat detailnya di release note Android nya
- <https://developer.android.com/about/versions?hl=id>

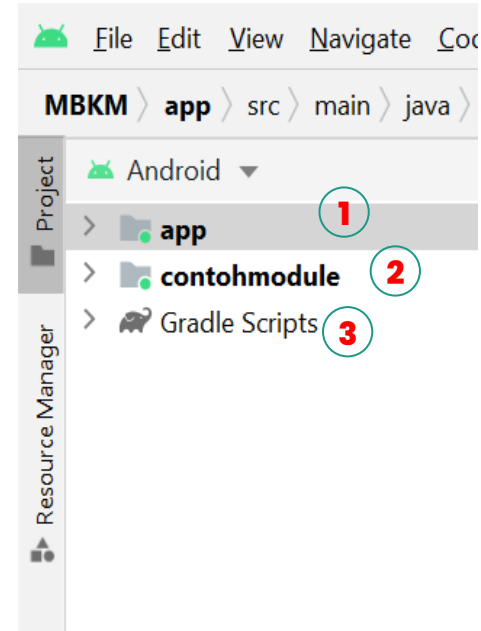
---

# Struktur Project



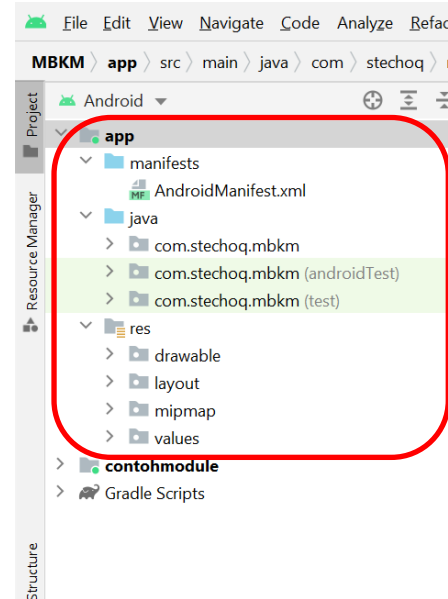
# Struktur Project

- **Modul Aplikasi** : modul utama aplikasi android.
- **Modul Library** : modul tambahan yang sifatnya opsional/tambahan jika diperlukan.
- **Gradle** : program build-tool yang berfungsi untuk melakukan build (compile dan packaging) secara otomatis.



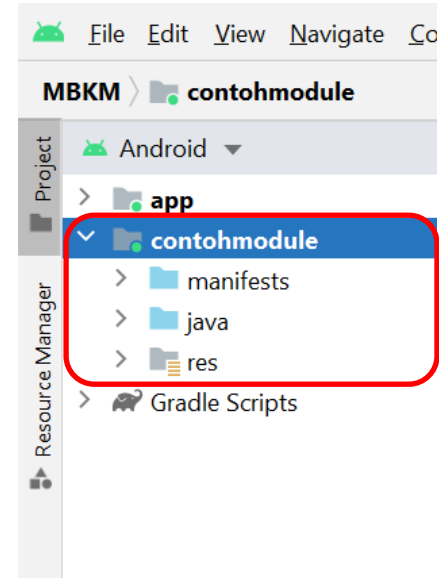
# Modul Aplikasi

- **manifest** : berisi tentang package (paket) pada proyek kita, seperti : deklarasi Activity, Services, User Permission, Content Provider, dll
- **java** : berisi package utama, androidTest dan test. Untuk package utama berisi data class proyek, contoh : MainActivity.java/MainActivity.kt
- **res** : folder resource yang berisi data gambar, layout XML, icon aplikasi, value dari warna, tema, dimension, string, dll



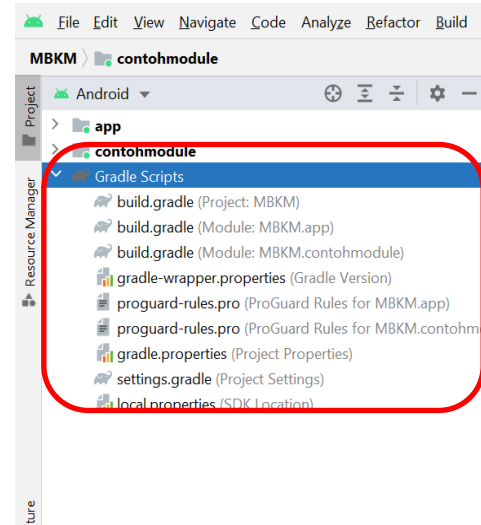
# Modul Library

- Merupakan modul tambahan/opsional yang bisa kita tambahkan sendiri. Dapat berupa modul atau berupa format kompresi dari modul (.jar).
- Untuk menambahkan modul tambahan bisa dilakukan di Gradle Scripts.



# Gradle

- Android menggunakan Gradle untuk project management nya.
- Secara default, Android Studio akan membuat multi module Gradle Project, dimana hanya terdapat 1 module, yaitu app



---

# Manifest File



# Manifest File

- Setiap project Android, wajib memiliki Manifest File yaitu AndroidManifest.xml
- Manifest File berisikan informasi dari aplikasi yang kita buat, seperti informasi Activity, Permission, Intent, Provider, Receiver, Service, dan lain-lain
- <https://developer.android.com/guide/topics/manifest/manifest-intro?hl=id>

# Kode : Contoh Manifest File

```
AndroidManifest.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools"
4      package="com.staygrateful.app.mbk2022">
5
6      <application
7          android:allowBackup="true"
8          android:dataExtractionRules="@xml/data_extraction_rules"
9          android:fullBackupContent="@xml/backup_rules"
10         android:icon="@mipmap/ic_launcher"
11         android:label="MBKM 2022"
12         android:roundIcon="@mipmap/ic_launcher_round"
13         android:supportsRtl="true"
14         android:theme="@style/Theme.MBK2022"
15         tools:targetApi="31">
16         <activity
17             android:name=".MainActivity"
18             android:exported="true">
```



Resource





# Resource

- Resource adalah file tambahan atau konten statis yang biasanya digunakan di kode program kita
- Seperti contohnya text, gambar, layout, animasi, dan lain-lain
- Android sendiri mendukung banyak sekali resource, pertanyaannya, bagaimana untuk mengambil data resource nya?
- Untuk mengambil data resource nya, kita bisa menggunakan class Resources
- <https://developer.android.com/reference/android/content/res/resources>



## Jenis-Jenis Resource (1)

Resource Directory	Keterangan
animator	XML file definisi property animations
anim	XML file definisi tween animations
color	XML file definisi warna
drawable	Bitmap files (png, jpg, gif) atau XML file drawable resource
mipmap	Drawable file untuk icon launcher



## Jenis-Jenis Resource (2)

Resource Directory	Keterangan
layout	XML file definisi layout user interface
menu	XML file definisi app menu
raw	File yang utuh lainnya
values	XML file yang berisi value seperti string, integer dan lain-lain
xml	XML file yang bisa dibaca menggunakan Resources.getXML()
font	Font files

---

# String Resource



# Values Resource

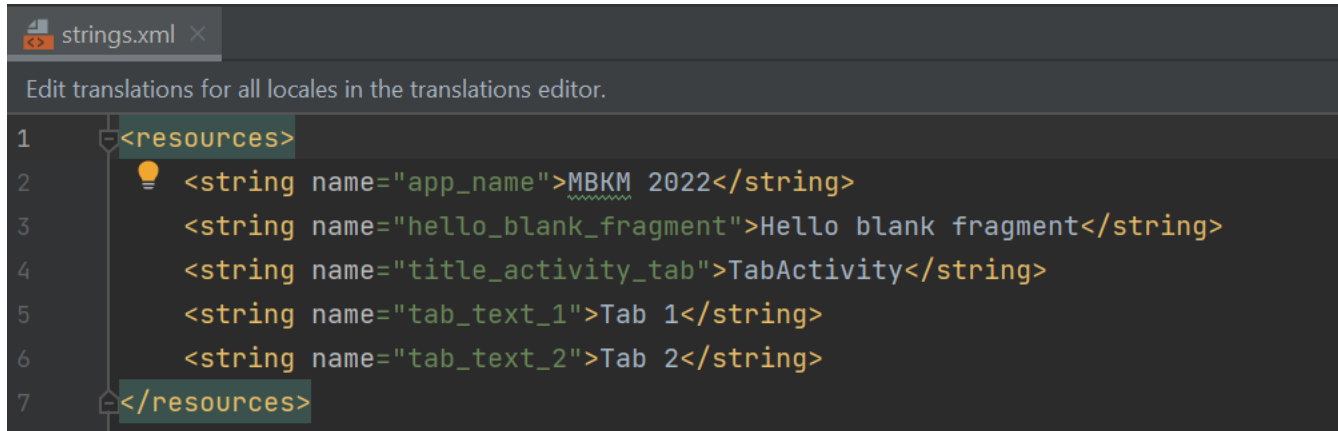
- Values resource merupakan jenis resource yang biasanya digunakan untuk menyimpan data-data statis yang digunakan di kode program kita, misal string, integer, boolean, color dan lain-lain



# String Resource

- String merupakan resource yang berisi teks
- Rekomendasinya ketika kita membuat tulisan untuk kita tampilkan di halaman UI aplikasi Android kita, disarankan jangan meng-hardcode pada kode program, lebih baik menggunakan String resource
- Hal ini karena jika kita ingin mengubah text nya, tidak perlu mengubah kode program, selain itu data text nya bisa digunakan ulang di halaman UI berbeda

## Kode : String Resource



```
strings.xml x
Edit translations for all locales in the translations editor.
1 <resources>
2   <string name="app_name">MBKM 2022</string>
3   <string name="hello_blank_fragment">Hello blank fragment</string>
4   <string name="title_activity_tab">TabActivity</string>
5   <string name="tab_text_1">Tab 1</string>
6   <string name="tab_text_2">Tab 2</string>
7 </resources>
```



## Kode : String Resource di Layout

```
14      <TextView
15          android:id="@+id/title"
16          android:layout_width="wrap_content"
17          android:layout_height="wrap_content"
18          android:gravity="center"
19          android:minHeight="?actionBarSize"
20          android:padding="@dimen/appbar_padding"
21          android:text="@string/app_name"
22          android:textAppearance="@style/TextAppearance.Widget..
```





# String Formatting

- Kadang saat menggunakan String resource, kita butuh membuat String yang datanya dinamis dan memiliki parameter, contoh pada kasus kita adalah tulisan Hello \$name
- Artinya \$name tersebut bisa berubah-ubah
- String resource mendukung hal tersebut, kita bisa menggunakan formatting string pada string resource, caranya cukup gunakan format %index\$s untuk parameter string, atau %index\$d untuk angka desimal

# Kode : String Formatting Resource

```
strings.xml
1 <resources>
2     <string name="app_name">MBKM 2022</string>
3     <string name="hello_blank_fragment">Hello blank fragment</string>
4     <string name="title_activity_tab">TabActivity</string>
5     <string name="tab_text_1">Tab 1</string>
6     <string name="tab_text_2">Tab 2</string>
7
8     <string name="title">MBKM 2022</string>
9     <string name="hello">Hallo %2$s</string>
10
11     <string-array name="jenis_kelamin">
12         <item>Laki-Laki</item>
13         <item>Perempuan</item>
14     </string-array>
15 </resources>
```

## Kode : Formatting String

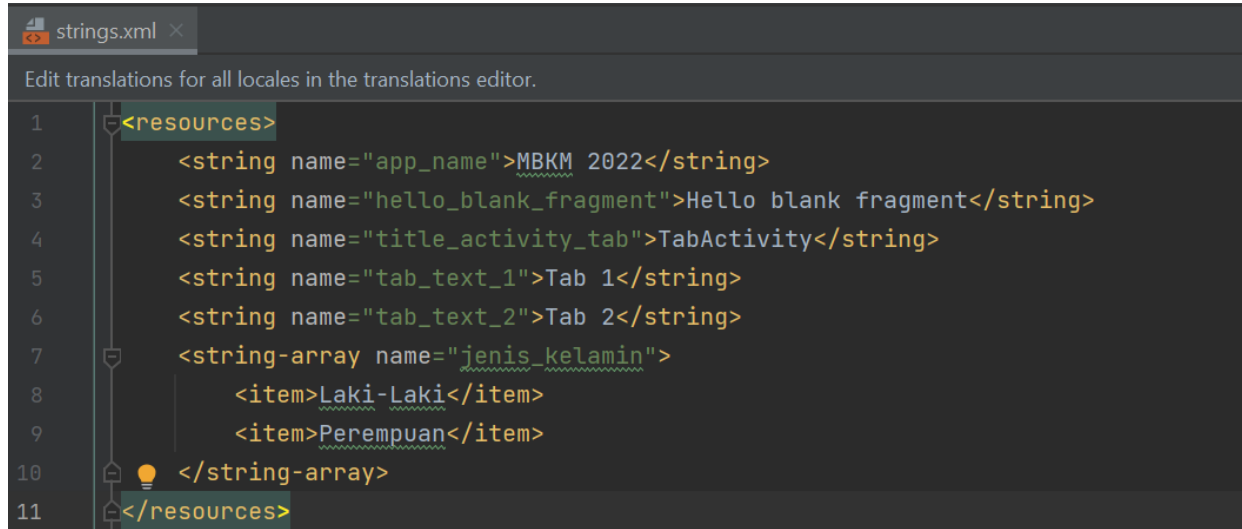
```
13      @Override
14      protected void onCreate(@Nullable Bundle savedInstanceState) {
15          super.onCreate(savedInstanceState);
16          setContentView(R.layout.activity_main);
17
18          final String title = getResources().getString(R.string.title);
19
20          final String hello = getResources().getString(R.string.hello, ...formatArgs: "Bapak", "Ibuk");
21
22          final String[] jenisKelamin = getResources().getStringArray(R.array.jenis_kelamin);
23      }
24  }
```



# String Array Resource

- Values resource juga bisa kita tambahkan tipe resource berupa String Array
- Kita bisa menggunakan tag `<string-array>` untuk menambahkan tipe String Array Resource
- Dan di dalamnya untuk menambahkan tiap datanya, kita bisa gunakan tag `<item>`
- String Array Resource secara otomatis akan terdapat di property array di class R
- Untuk mengambil String Array Resource, kita bisa gunakan function `getStringArray(resourceId)`

# Kode : String Array Resource



```
strings.xml x
Edit translations for all locales in the translations editor.
1 <resources>
2     <string name="app_name">MBKM 2022</string>
3     <string name="hello_blank_fragment">Hello blank fragment</string>
4     <string name="title_activity_tab">TabActivity</string>
5     <string name="tab_text_1">Tab 1</string>
6     <string name="tab_text_2">Tab 2</string>
7     <string-array name="jenis_kelamin">
8         <item>Laki-Laki</item>
9         <item>Perempuan</item>
10    </string-array>
11 </resources>
```

## Kode : Mengambil String Array Resource

```
10 public class MainActivity extends AppCompatActivity {  
11  
12     @Override  
13     protected void onCreate(@Nullable Bundle savedInstanceState) {  
14         super.onCreate(savedInstanceState);  
15         setContentView(R.layout.activity_main);  
16  
17         final String[] jenisKelamin = getResources().getStringArray(R.array.jenis_kelamin);  
18     }  
19 }
```

---

# Layout Resource

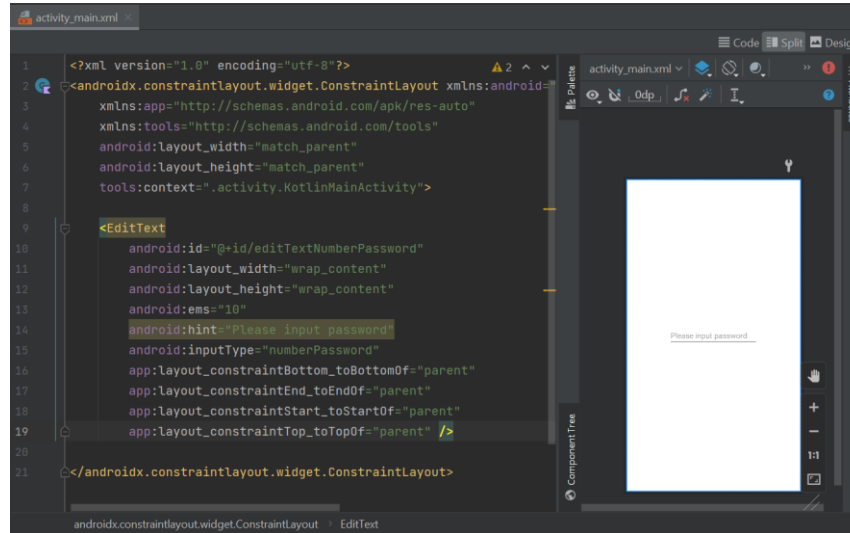


# Layout Resource

- Layout resource adalah definisi dari tampilan untuk UI
- Di dalam layout, kita bisa mendefinisikan isi View atau ViewGroup
- View adalah single komponen, sedangkan ViewGroup adalah container atau wadah untuk satu atau lebih komponen View
- Contoh ViewGroup seperti LinearLayout, RelativeLayout, FrameLayout, dan lain-lain
- Layout juga bisa menambahkan layout lain, dengan menggunakan tag `<include>`
- Setiap layout harus memiliki satu root element, jika misal kita tidak ingin memiliki root element, misal untuk digunakan sebagai include di layout lain, kita bisa gunakan root tag `<merge>`



# Kode : Layout



---

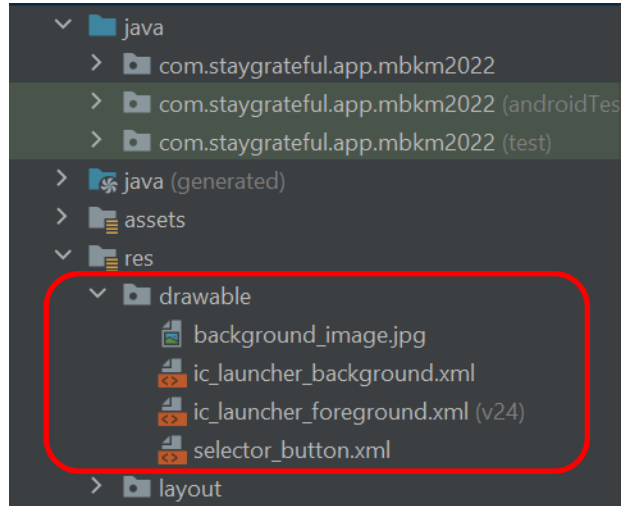
# Drawable Resource



# Drawable Resource

- Drawable Resource adalah jenis resource dengan konsep graphics yang bisa digambar di layar
- Ada banyak jenis Drawable, seperti Bitmap File (file gambar), Nine Patch File, Layer List, State List dan lain-lain
- <https://developer.android.com/guide/topics/resources/drawable-resource?hl=id>
- Drawable direpresentasikan dalam class Drawable, dan untuk mendapatkannya, kita bisa gunakan function `getDrawable(resourceId)` pada class Resources
- <https://developer.android.com/reference/android/graphics/drawable/Drawable>

# Menambah Drawable



---

# Asset Manager

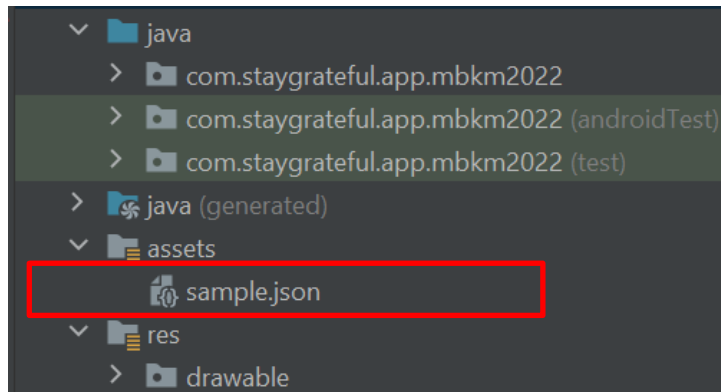


# Asset Manager

- Kadang kita ingin menambahkan resource di Aplikasi Android kita, tapi bukan resource yang di manage oleh Android, misal kita ingin menambahkan file JSON atau TXT misalnya
- Untuk kasus seperti ini, kita bisa menggunakan AssetManager
- AssetManager adalah class yang bisa kita gunakan untuk mengakses resource secara manual
- <https://developer.android.com/reference/android/content/res/AssetManager>
- AssetManager akan mengambil resource pada directory assets, sehingga kita perlu membuatnya terlebih dahulu
- Dan untuk mendapatkan AssetManager, kita bisa gunakan function `getAssets()` di Context / Activity



# Asset Manager





## Kode : Asset Manager

```
15  @ public static InputStream openStreamAssetsResources(Context context, @NonNull String fileName) {  
16      try {  
17          return context.getAssets().open(fileName);  
18      } catch (Exception e) {  
19          e.printStackTrace();  
20      }  
21      return null;  
22  }
```



---

# Raw Resource

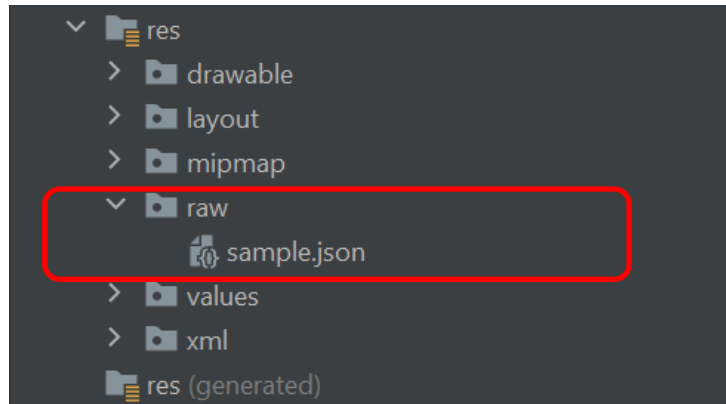


# Raw Resource

- Salah satu masalah ketika menggunakan Asset Manager adalah, kita harus mengetikkan lokasi resource nya menggunakan string. Hal ini akan rentan kesalahan, terutama jika lokasi file tidak ada, maka otomatis aplikasi kita akan error. Dan error ini tidak bisa dideteksi ketika kompilasi
- Android juga sebenarnya mendukung Raw Resource, memang tidak se flexible AssetManager, namun pada kasus yang tidak terlalu banyak assetnya, kita bisa gunakan Raw Resource
- Raw Resource akan secara otomatis men-generate id di class R, sehingga kesalahan tidak akan mungkin terjadi ketika aplikasinya sudah berjalan



# Raw Resource





## Kode : Raw Resource

```
15  @ public static InputStream openStreamRawResources(Context context, @RawRes int id) {  
16      try {  
17          return context.getResources().openRawResource(id);  
18      } catch (Exception e) {  
19          e.printStackTrace();  
20      }  
21      return null;  
22  }
```



# Raw vs Assets

Scenario	Assets Folder	Res/Raw Folder
<b>Flexible File Name</b>	YES	NO
<b>Store in subdirectory</b>	YES	NO
<b>Compile-time checking</b>	NO	YES
<b>List filenames at runtime</b>	YES	NO
<b>Filename accessible from XML</b>	NO	YES

---

# Screen Compatibility



# Screen Compatibility

- Android berjalan di perangkat dengan berbagai ukuran, dari smartphone, tablet sampai tv
- Untuk mengkategorisasikan perangkat berdasarkan ukuran layar, Android mendefinisikan dalam dua karakteristik, screen size (ukuran layar) dan screen density (kepadatan pixel layar, atau DPI)
- Untuk mempermudah dan menggeneralisasi semua varian ini, Android membagi menjadi beberapa grup
- Screen size : small, normal, large dan xlarge
- Screen density : mdpi (medium), hdpi (high), xhdpi (extra high), xxhdpi (extra-extra high)
- Secara default, aplikasi kita akan kompatibel dengan semua screen size dan density, karena Android akan melakukan penyesuaian layout UI dan gambar untuk tiap perangkat layar
- Namun untuk pengalaman yang lebih baik, ada baiknya kita menggunakan resource yang sesuai dengan tiap jenis ukuran layar



# Pixel Densities

- Saat kita menggunakan perangkat yang kepadatan pixelnya berbeda-beda, maka agak menyulitkan ketika kita ingin membuat komponent yang ukurannya fix, misal menggunakan pixel
- Contoh jika kita menggunakan komponen dengan ukuran 100px , maka akan terlihat besar di layar yang density nya kecil, tapi akan terlihat kecil jika diukuran layar yang density nya besar
- Oleh karena itu, di Android, kita jarang menggunakan ukuran px, melainkan ukuran dp (density-independent pixels)
- Ukuran dp bisa secara otomatis membesar sesuai dengan screen density.
- Jika kita ingin menghitung berapa pixel dari dp, kita bisa gunakan rumus :  
$$px = dp * (dpi / 160)$$



Density qualifier	Description
<b>ldpi</b>	Resources for low-density ( <i>ldpi</i> ) screens (~120dpi).
<b>mdpi</b>	Resources for medium-density ( <i>mdpi</i> ) screens (~160dpi). (This is the baseline density.)
<b>hdpi</b>	Resources for high-density ( <i>hdpi</i> ) screens (~240dpi).
<b>xhdpi</b>	Resources for extra-high-density ( <i>xhdpi</i> ) screens (~320dpi).
<b>xxhdpi</b>	Resources for extra-extra-high-density ( <i>xxhdpi</i> ) screens (~480dpi).
<b>xxxhdpi</b>	Resources for extra-extra-extra-high-density ( <i>xxxhdpi</i> ) uses (~640dpi).



## Kode : Density Independent Pixels

```
9      <ImageView
10          android:id="@+id/imageView"
11          android:layout_width="wrap_content"
12          android:layout_height="250dp"
13          android:src="@tools:sample/backgrounds/scenic"
14          app:layout_constraintBottom_toBottomOf="parent"
15          app:layout_constraintEnd_toEndOf="parent"
16          app:layout_constraintStart_toStartOf="parent"
17          app:layout_constraintTop_toTopOf="parent" />
```

---

# Activity

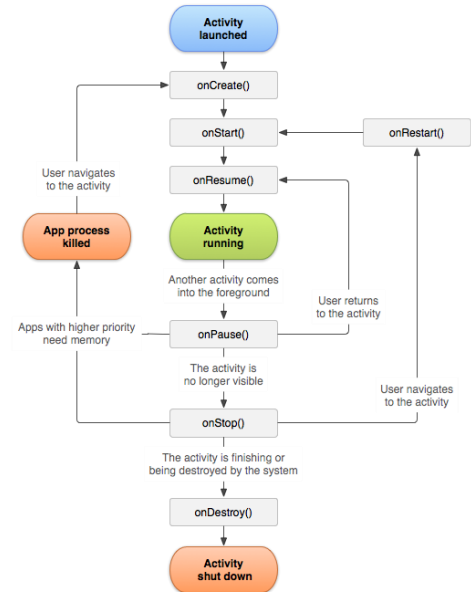


# Activity

- Saat kita membuat aplikasi seperti di Java atau di Kotlin, main function sebagai function yang akan diluncurkan ketika aplikasi berjalan.
- Di Android tidak seperti itu, Android memiliki fitur yang bernama Activity, dimana nanti object Activity tersebut akan secara otomatis dijalankan oleh Android
- Activity yang dibuat harus dideklarasikan di file AndroidManifest.xml
- Sebuah activity memiliki siklus hidup.

# Activity Lifecycle

- Activity tidak hidup abadi selamanya, dia bisa diciptakan dan dihancurkan. Activity memiliki siklus hidup (lifecycle) yang merupakan kondisi yang akan dialami saat diciptakan sampai dihancurkan.
- Ada beberapa kondisi yang akan dialami oleh Activity seperti yang tergambar pada flow chart berikut ini.





# Class Activity

- Untuk membuat Activity, kita perlu membuat class turunan dari Activity
- Saat kita membuat project Android, secara otomatis akan ada sebuah MainActivity yang merupakan class turunan dari AppCompatActivity
- AppCompatActivity merupakan turunan dari class Activity yang memungkinkan kita menggunakan fitur baru Android di versi Android lama, oleh karena itu direkomendasikan menggunakan class AppCompatActivity
- <https://developer.android.com/reference/android/app/Activity>



# Mendaftarkan Activity

- Untuk memberitahu kepada Android, bahwa kita membuat Activity, kita harus mendaftarkannya di Manifest File
- Selain itu, terkadang kita perlu menambahkan intent untuk menambahkan informasi seperti misalnya, menandai sebuah Activity bahwa ini adalah Main Activity, dan menandai bahwa Activity ini harus dijalankan ketika aplikasi Android diluncurkan/dibuka (LAUNCH)

## Kode : Contoh Manifest Activity

```
25 <activity
26     android:name=".activity.MainActivity"
27     android:exported="true" >
28     <intent-filter>
29         <action android:name="android.intent.action.MAIN" />
30
31         <category android:name="android.intent.category.LAUNCHER" />
32     </intent-filter>
33 </activity>
```



## Kode : Activity onCreate()

```
MainActivity.java x
1  package com.staygrateful.app.mbkm2022;
2
3  import ...
7
8  public class MainActivity extends AppCompatActivity {
9
10     @Override
11     protected void onCreate(@Nullable Bundle savedInstanceState) {
12         super.onCreate(savedInstanceState);
13         setContentView(R.layout.activity_main);
14     }
15 }
```

—

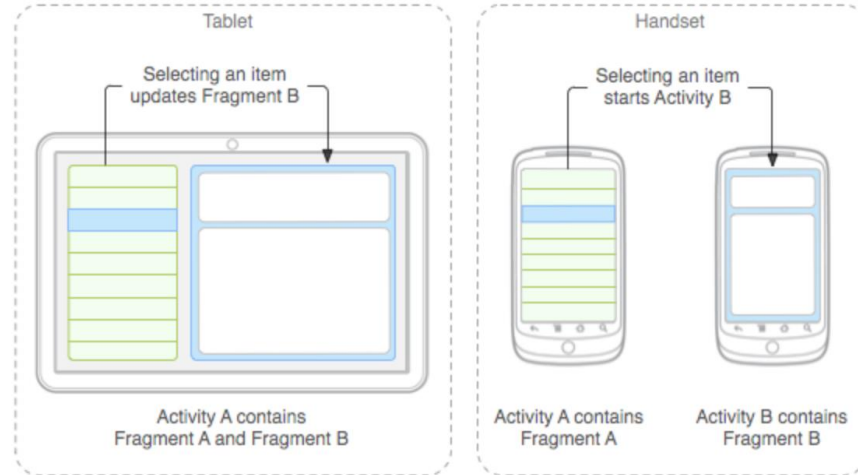
Fragment



# Fragment

- Komponen yang dapat digunakan kembali dari antarmuka pengguna android activity yang digunakan untuk membuat UI yang dinamis dan fleksibel.
- Fragmen memiliki siklus hidup itu sendiri tetapi selalu tertanam dengan aktivitas sehingga siklus hidup fragmen secara langsung dipengaruhi oleh siklus hidup aktivitas.
- Kita bisa menyisipkan fragmen ke dalam layout aktivitas dengan mendeklarasikan fragmen dalam file layout aktivitas, sebagai elemen <fragment>, atau dari kode aplikasi dengan menambahkannya ke ViewGroup yang ada.

# Gambaran Fragment





# Kegunaan Fragment

- Sebelum Fragment diperkenalkan, kita hanya dapat menampilkan satu activity di layar pada satu waktu tertentu sehingga tidak dapat membagi layar.
- Dengan bantuan Fragment, kita dapat membagi layar di berbagai bagian dan mengontrol bagian-bagian yang berbeda secara terpisah.
- Dengan menggunakan Fragment, kita dapat membentuk beberapa Fragment dalam satu activity.
- Fragment memiliki event, layout, dan status mereka sendiri. Hal ini memberikan fleksibilitas dan juga mengatasi limitasi dalam activity di layar pada suatu waktu.



# Kode : Fragment

```
Samplefragment.java
1  package com.staygrateful.app.mbk2022.fragment;
2
3  import ...
4
12
13  public class SampleFragment extends Fragment {
14
15      public SampleFragment() {
16          // Required empty public constructor
17      }
18
19      @Override
20      public void onCreate(Bundle savedInstanceState) {
21          super.onCreate(savedInstanceState);
22      }
23
24
25      @Override
26      public View onCreateView(LayoutInflater inflater, ViewGroup container,
27                              Bundle savedInstanceState) {
28          // Inflate the layout for this fragment
29          return inflater.inflate(R.layout.fragment_sample, container, attachToRoot: false);
30      }
31  }
```

---

**View**



# View

- Semua komponen UI di Android adalah turunan View
- Ada banyak sekali komponen yang terdapat di Android
- Bahkan Layout sendiri adalah turunan dari class View
- <https://developer.android.com/reference/android/view/View>





# Menambah View

- Untuk menambah View, kita bisa menyebutkan View yang ingin kita gunakan di Layout yang sudah kita buat
- Setiap komponen View memiliki banyak atribut yang bisa kita ubah, misal Button, Label dan Text memiliki atribut text yang bisa kita ubah untuk mengubah tulisan text nya



## Kode XML : Menambah View

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:text="Name" />

<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal" />
```

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:text="Say Hello" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"/>
```



# Find View by Id

- Saat kita menambahkan komponen ke Layout, kadang kita ingin mendapatkan object dari komponen tersebut
- Caranya adalah kita bisa menggunakan function `findViewById()`
- Namun sebelum kita bisa menggunakan function tersebut, kita perlu menambahkan id terhadap komponen yang ingin kita ambil objectnya



## Kekurangan Find View By Id

- Saat kita menggunakan function `findViewById()`, maka aplikasi kita akan secara otomatis mencari View berdasarkan id kita
- Saat komponen dan layout kita masih sedikit, maka menggunakan `findViewById()` secara terus menerus tidak akan bermasalah, namun saat nanti komponen dan layout kita semakin banyak, maka akan membuat kode program kita semakin lambat
- Oleh karena itu tidak disarankan selalu memanggil function ini setiap kali kita butuh komponent

---

# View Binding



# View Binding

- Fitur yang memungkinkan kita untuk binding sebuah properti ke elemen view.
- Library ini secara otomatis akan memberi akses langsung ke semua view yang ada di dalam XML.
- Setelah diaktifkan dalam sebuah modul, view binding akan menghasilkan class binding untuk setiap file tata letak XML yang ada dalam modul tersebut.
- Instance class binding berisi referensi langsung ke semua tampilan yang memiliki ID di tata letak yang terkait.
- <https://developer.android.com/topic/libraries/view-binding>



# Perbandingan View Binding

Method	Elegance	Compile Type Safety	Build Speed
findViewById	✗	✗	✓
Data Binding	✓	✓	✗
Butterknife	✓	✗	✗
Kotlin Synthethic	✓	✗	✓
View Binding	✓	✓	✓



# Mengaktifkan View Binding

- Fitur View Binding secara default tidak aktif. Untuk mengaktifkannya, kita perlu menambahkan kode berikut pada build.gradle di level module yang akan menggunakan View Binding.

```
1 android {  
2     ...  
3     buildFeatures {  
4         viewBinding true  
5     }  
6 }
```





# Layout



# Layout

- Activity bukanlah tampilan UI, tapi biasanya Activity itu akan menampilkan tampilan UI
- Android memisahkan antara kode program dan tampilan UI, namanya adalah Layout
- Layout merupakan kode yang berisikan tampilan UI
- Layout di Android menggunakan XML, sehingga yang terbiasa menggunakan HTML, akan mudah beradaptasi

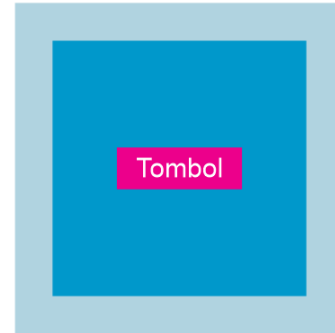


# Jenis – Jenis Layout (umum)

- `FrameLayout`
- `LinearLayout`
- `RelativeLayout`
- `ConstraintLayout`
- `TableLayout`
- `GridLayout`

# FrameLayout

- Layout yang digunakan untuk membuat objek yang saling bertindihan, contohnya seperti gambar berikut.
- Layout standar dan paling ringan
- Biasanya digunakan sebagai container fragment
- Hanya memiliki atribut **width**, **height**, **gravity**, **margin**

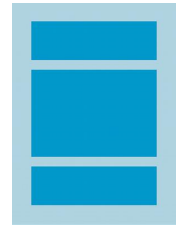


# LinearLayout

- Layout yang digunakan untuk membuat objek yang **horizontal** atau **vertikal**, contohnya seperti gambar berikut.
- Untuk menggunakan layout horizontal atau vertical menggunakan atribut orientation.
- Memiliki semua atribut pada FrameLayout serta atribut orientation



Horizontal



Vertikal

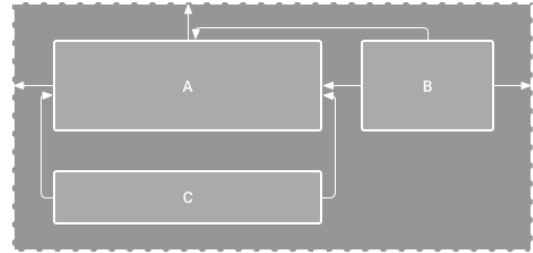
# RelativeLayout

- Tata letak objek atau komponen secara bebas tanpa aturan.
- Tata letak objek tergantung pada view yang lainnya.
- Memiliki semua atribut pada `FrameLayout` serta atribut `above`, `below`, `toStartOf`, `toEndOf`, dsb.



# ConstraintLayout

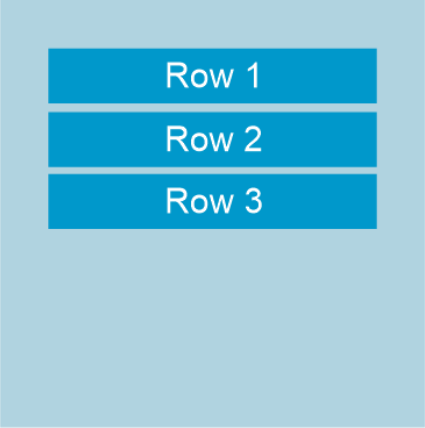
- Hampir sama seperti RelativeLayout hanya saja ConstraintLayout lebih fleksibel dibandingkan RelativeLayout.
- Tata letak yang lebih baru, fleksibel, kompleks dan responsive.





# TableLayout

- Untuk merancang layout menggunakan baris dan kolom.
- Tidak akan ada garis kolom, baris, atau cell yang ditampilkan meskipun namanya adalah table layout.
- Row/ baris digunakan untuk menyimpan satu jenis record (TableRow).
- Kolom adalah sub bagian terbagi dari setiap baris dan satu baris dapat menampung beberapa jenis kolom.



Row 1
Row 2
Row 3



# GridLayout

- Hampir mirip seperti TableLayout bedanya GridLayout tidak memerlukan objek baris (TableRow) dalam implementasinya.
- Harus menentukan jumlah baris dan kolom nya.
- Hanya mendukung Android 4.0 keatas



—

Intent



# Intent

- Intent merupakan mekanisme untuk melakukan sebuah action dan komunikasi antar komponen aplikasi.
- Contoh, Ketika tekan tombol untuk membuka peta, kamera, atau halaman atau activity lainnya. Perpindahan inilah yang dinamakan Intent.
- Intent dapat digunakan pada Activity, Service, Broadcast.



# Jenis-Jenis Intent

- Intent Implicit : Berfungsi melakukan perpindahan activity (halaman) menuju ke aplikasi internal smartphone kamu. Contohnya ketika hendak membuka sebuah kamera.
- Intent Explicit : Berfungsi melakukan perpindahan activity (halaman) ke activity (halaman) lainnya dalam satu aplikasi yang sama. Explicit intent bekerja dengan menggunakan nama kelas yang dituju, misal `com.stechoq.activity.DetailActivity`. Umumnya intent ini digunakan untuk mengaktifkan komponen pada satu aplikasi.

# Intent Implicit

```
53 @ public static void shareImplicitIntent(Context context) {  
54     //Perintah Intent Implicit untuk share ke aplikasi lain, misal sosmed  
55     Intent intent = new Intent(Intent.ACTION_SEND);  
56  
57     //Data yang ingin dishare  
58     intent.putExtra(Intent.EXTRA_TEXT, value: "Hallo saya share ke sosial media");  
59     intent.setType("text/plain");  
60  
61     //Menjalankan perintah  
62     context.startActivity(Intent.createChooser(intent, title: "Share to :"));  
63 }
```



# Intent Explicit

```
65 public static void shareExplicitIntent(Activity activity, Class<Activity> activityToOpen) {  
66     //Activity yang dituju  
67     Intent intent = new Intent(activity, activityToOpen);  
68  
69     //Data  
70     intent.putExtra( name: "MY_DATA", value: "Hallo saya share ke sosial media");  
71  
72     //Menjalankan perintah  
73     activity.startActivity(intent);  
74 }
```

---

# Recycler View



# RecyclerView

- Layout yang digunakan untuk membuat objek yang berdasarkan list atau array.
- Sama seperti ListView atau GridView, implementasinya harus menggunakan adapter.
- RecyclerView mendaur ulang elemen individual tersebut. Saat item di-scroll keluar layar, RecyclerView tidak merusak tampilannya.
- Memiliki performa yang jauh lebih baik dibandingkan ListView atau GridView.
- Mendukung layout linear, grid dan staggered.





# Layout pada RecyclerView

- Item di RecyclerView diatur oleh class LayoutManager. Library RecyclerView menyediakan tiga pengelola layout, yang menangani situasi tata letak paling umum :



Linear



Grid



Staggered



# Implementasi RecyclerView

- Hal pertama, tentukan bagaimana daftar atau list yang ingin ditampilkan. Ingin ditampilkan dalam bentuk list linear, grid atau staggered.
- Buat desain tampilan dan perilaku setiap elemen dalam daftar. Berdasarkan desain ini, perluas class ViewHolder. Versi ViewHolder menyediakan semua fungsi untuk item daftar atau list.
- Tentukan Adapter yang mengatribusikan data dengan tampilan ViewHolder.

---

# Rest Api

# Rest Api

- REST API adalah sebuah program API (Application Program Interface) yang menggunakan protokol HTTP (GET, POST, PUT, DELETE) untuk digunakan mengolah data.
- Desain yang umum digunakan untuk REST API, berbentuk nouns (kata benda). Silahkan perhatikan gambar berikut untuk lebih memahami desain REST API.

Resource	POST create	GET read	PUT update	DELETE delete
/dogs	create a new dog	list dogs	bulk update dogs	delete all dogs
/dogs/1234	error	show Bo	if exists update Bo if not error	delete Bo



# Retrofit

- REST API adalah sebuah program API (Application Program Interface) yang menggunakan protokol HTTP (GET, POST, PUT, DELETE) untuk digunakan mengolah data.
- Desain yang umum digunakan untuk REST API, berbentuk nouns (kata benda). Silahkan perhatikan gambar berikut untuk lebih memahami desain REST API.



# Implement Retrofit

- Import dependensi Retrofit dan Gson pada gradle.
- Pastikan sudah memiliki permission atau izin akses internet (Lihat Manifest file).
- Membuat object sesuai response api.
- Membuat interface api.
- Membuat object retrofit client.



# Import Dependensi

- Import dependensi Retrofit dan Gson pada gradle module seperti gambar berikut :

```
60      //Retrofit
61      implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
62      implementation 'com.squareup.retrofit2:retrofit:2.9.0'
63
64      //Okhttp3
65      implementation 'com.squareup.okhttp3:okhttp:4.9.3'
66      implementation 'com.squareup.okhttp3:logging-interceptor:4.9.1'
```

# Izin Akses Internet

- Pastikan sudah menambahkan permission atau izin untuk akses internet pada file manifest. Dapat dilihat seperti gambar berikut :

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:tools="http://schemas.android.com/tools"
4   package="com.staygrateful.app.mbk2022">
5
6   <uses-permission android:name="android.permission.INTERNET"/>
7
8   <application
9     android:allowBackup="true"
10    android:dataExtractionRules="@xml/data_extraction_rules"
11    android:fullBackupContent="@xml/backup_rules"
```





# Membuat Object

- Membuat sebuah object sesuai response dari api. Berikut contoh sample object nya:

```
5 public class Student {
6
7     @SerializedName("name")
8     private String name;
9     @SerializedName("email")
10    private String email;
11
12    public String getName() {
13        return name;
14    }
15
16    public void setName(String name) {
17        this.name = name;
18    }
19
20    public String getEmail() {
21        return email;
22    }
23
24    public void setEmail(String email) {
25        this.email = email;
26    }
27 }
```



# Membuat Interface Api

- Membuat sebuah interface api yang terdiri dari data-data api. Berikut contoh sample interface nya:

```
10 public interface Service {  
11     @GET("photos")  
12     Call<List<ResponseData>> getAllPhotos();  
13  
14     @GET("photos")  
15     Call<JsonElement> getAllData();  
16 }
```



# Membuat Retrofit Client

- Membuat sebuah retrofit client untuk mengakses ke api server. Berikut contoh sample pembuatan nya:

```
25 @      public static Retrofit getRetrofit(OkHttpClient okHttpClient){
26         final Gson gson = new GsonBuilder()
27             .setLenient()
28             .create();
29         return new retrofit2.Retrofit.Builder()
30             .client(okHttpClient != null ? okHttpClient : new OkHttpClient())
31             .baseUrl(BASE_URL)
32             .addConverterFactory(GsonConverterFactory.create(gson))
33             .build();
34     }
```

## Kode : Contoh Implementasi

```
42 public static void getAllData() {  
43     final OkHttpClient ohc = ApiClient.getOkHttpClient(ApiClient.getInterceptor());  
44     Service service = ApiClient.getRetrofit(ohc).create(Service.class);  
45     Call<JsonElement> call = service.getAllData();  
46     call.enqueue(new Callback<JsonElement>() {  
47         @Override  
48         public void onResponse(@NonNull Call<JsonElement> call,  
49                               @NonNull Response<JsonElement> response) {  
50             System.out.println("Data response body : " + response.body());  
51         }  
52  
53         @Override  
54         public void onFailure(@NonNull Call<JsonElement> call,  
55                               @NonNull Throwable t) {  
56             System.out.println("Data response error : " + t.getLocalizedMessage());  
57         }  
58     });  
59 }
```

---

# Debugging



# Debugging

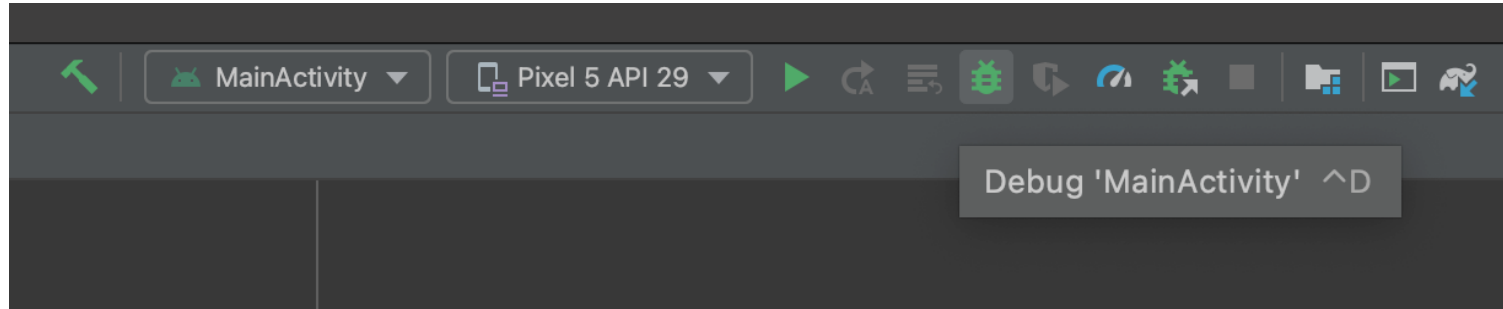
- Saat membuat aplikasi, kesalahan sering terjadi
- Dan kadang, agak sulit untuk mencari kesalahan tersebut, sehingga kita butuh menelusuri alur kode program kita satu per satu
- Untungnya, Android memiliki fitur debugging, dimana kita bisa menghentikan program yang sedang berjalan, dan melanjutkan jalannya program secara bertahap sesuai yang kita inginkan
- Fitur ini sangat tergantung dengan Android Studio



## Kode : Debugging

```
7  
8     private fun printHello(name: String) {  
9         Log.i( tag: "DEBUG", name)  
10    }  
11  
12 }
```

# Menjalankan dalam Mode Debug





—

# Testing



# Testing

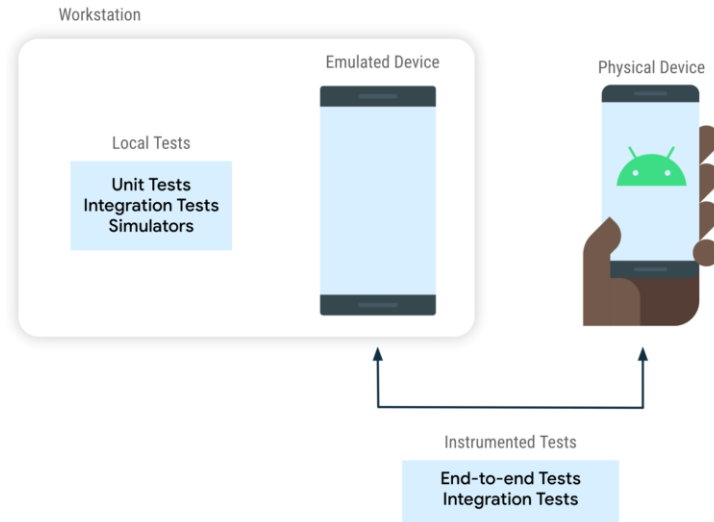
- Testing adalah salah satu tahapan dalam pembuatan aplikasi, termasuk di Android
- Android sudah mendukung testing dengan baik, baik itu unit test sampai end to end test



# Jenis Testing di Android

- Instrumented Test, yaitu test yang di jalankan di perangkat Android, baik itu physical atau emulator. Aplikasi akan di install di perangkat Android, lalu test akan melakukan pengetesan pada aplikasi yang berjalan dengan mengirim perintah-perintah. Instrumented Test bisa dibilang juga Integration Test atau End to End Test.
- Local Test, yaitu test yang dieksekusi di development machine kita, misal di laptop atau di server. Biasanya local test itu kecil dan cepat, dan biasanya di dalam local test tidak membutuhkan device Android untuk berjalan. Kita bisa bilang juga dengan nama Unit Test

# Jenis Testing di Android





# Kenapa Automation Testing Penting?

- Dengan menjalankan test terhadap aplikasi kita secara konsisten, kita bisa memverifikasi kebenaran aplikasi kita, fungsionalitas aplikasi, sebelum kita merilis aplikasi ke public
- Sebenarnya kita bisa saja melakukan test secara manual, dengan cara menjalankan aplikasi di Device, lalu mencoba semua fitur.
- Namun, manual test sulit untuk dikerjakan secara konsisten, rentang kesalahan dan jika butuh cepat, tidak bisa dilakukan secara otomatis juga.
- Oleh karena itu automation test sangat penting dalam pengembangan aplikasi, terutama aplikasi Android

---

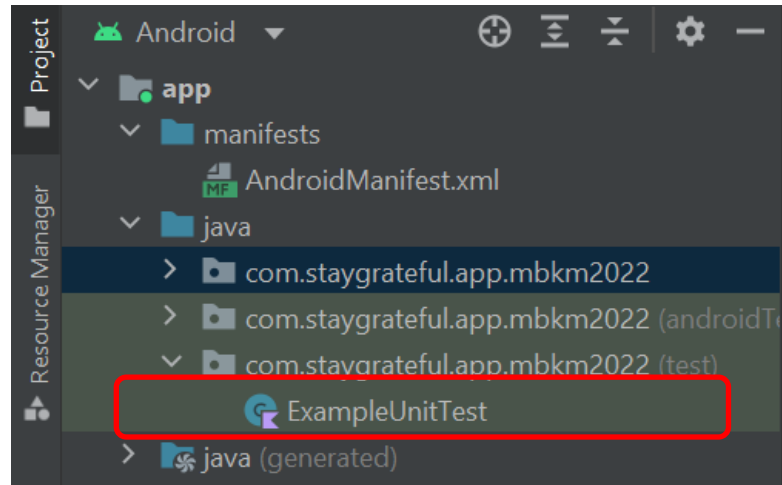
# Local Test



# Local Test

- Android menggunakan JUnit untuk melakukan Unit Test atau Local Test
- Pada saat ini, versi JUnit yang digunakan oleh Android masih menggunakan JUnit 4, padahal versi terbaru adalah JUnit 5
- Namun jangan khawatir jika kita sudah terbiasa menggunakan JUnit 5, harusnya akan lebih mudah menggunakan JUnit 4

# Folder Unit Test





---

# Instrumentation Test



# Instrumentation Test

- Unit Test hanya bisa digunakan untuk melakukan pengetesan kode program kita
- Saat kita membuat aplikasi Android, kadang banyak sekali ketergantungan dengan sistem operasi Android nya itu sendiri, oleh karena itu kadang agak sulit untuk membuat unit test ketika kita butuh melakukan pengetesan interaksi dengan UI aplikasi kita
- Android mendukung Instrumentation Test, atau sederhananya adalah UI Test Automation
- Dengan menggunakan Instrumentation Test, kita bisa membuat automation test mirip robot, yang mensimulasikan pengguna aplikasi kita



# Espresso

- JUnit hanya digunakan untuk library Unit Test, untuk melakukan UI Test, Android menggunakan library bernama Espresso
- Dengan Espresso, kita bisa membuat UI Test dengan mudah
- <https://developer.android.com/training/testing/espresso>



# Activity Scenario

- Saat kita melakukan instrumentation test, biasanya kita akan melakukan UI Test, dan untuk menampilkan UI, kita biasanya butuh Activity
- Dalam Instrumentation Test, kita bisa menggunakan ActivityScenario untuk menjalankan sebuah Activity
- <https://developer.android.com/reference/androidx/test/core/app/ActivityScenario>



## Kode : Activity Scenario

```
23  @RunWith(AndroidJUnit4.class)
24  ▶ public class ActivityInstrumentTest {
25
26      private ActivityScenario<MainActivity> mainActivityScenario;
27
28      @Before
29      public void setup() {
30          mainActivityScenario = ActivityScenario.launch(MainActivity.class);
31      }
32
33      @Test
34      ▶ public void test() {
35          //Test here...!
36      }
37
38      @After
39      public void finish() {
40          mainActivityScenario.close();
41      }
42  }
```



# Activity Scenario Rule

- Menjalankan Activity secara manual menggunakan Activity Scenario tidak direkomendasikan ketika kita membuat Instrumentation Test
- Kita bisa memanfaatkan ActivityScenarioRule yang bisa di integrasikan dengan JUnit Rule, yang secara otomatis bisa menjalankan Activity ketika unit test mulai dan menghentikan Activity ketika unit test selesai
- <https://developer.android.com/reference/androidx/test/ext/junit/rules/ActivityScenarioRule>



## Kode : Activity Scenario Rule

```
23  @RunWith(AndroidJUnit4.class)
24  ▶ public class ActivityInstrumentTest {
25
26      @Rule
27      public ActivityScenarioRule<MainActivity> mainActivityScenarioRule =
28          new ActivityScenarioRule<>(MainActivity.class);
29
30      @Test
31  ▶ public void test() {
32      💡 // Test here...!
33  }
34  }
```



# Espresso Package

- Espresso berisikan banyak class yang bisa digunakan untuk mempermudah kita melakukan Instrumentation Test
- Package Espresso ada di `androidx.test.espresso`
- <https://developer.android.com/reference/androidx/test/espresso/package-summary?hl=en>



## Kode : Instrumentation Test

```
23  @RunWith(AndroidJUnit4.class)
24  >> public class ActivityInstrumentTest {
25
26      @Rule
27      public ActivityScenarioRule<MainActivity> mainActivityScenarioRule = new ActivityScenarioRule<>(MainActivity.class);
28
29      @Test
30  ▶  public void test() {
31          Espresso.onView(ViewMatchers.withId(R.id.inputPassword))
32              .perform(ViewActions.typeText(stringToBeTyped: "Password Saya"));
33
34          Espresso.onView(ViewMatchers.withId(R.id.inputPassword))
35              .check(ViewAssertions.matches(ViewMatchers.withText("Password Aku")));
36      }
37  }
```

---

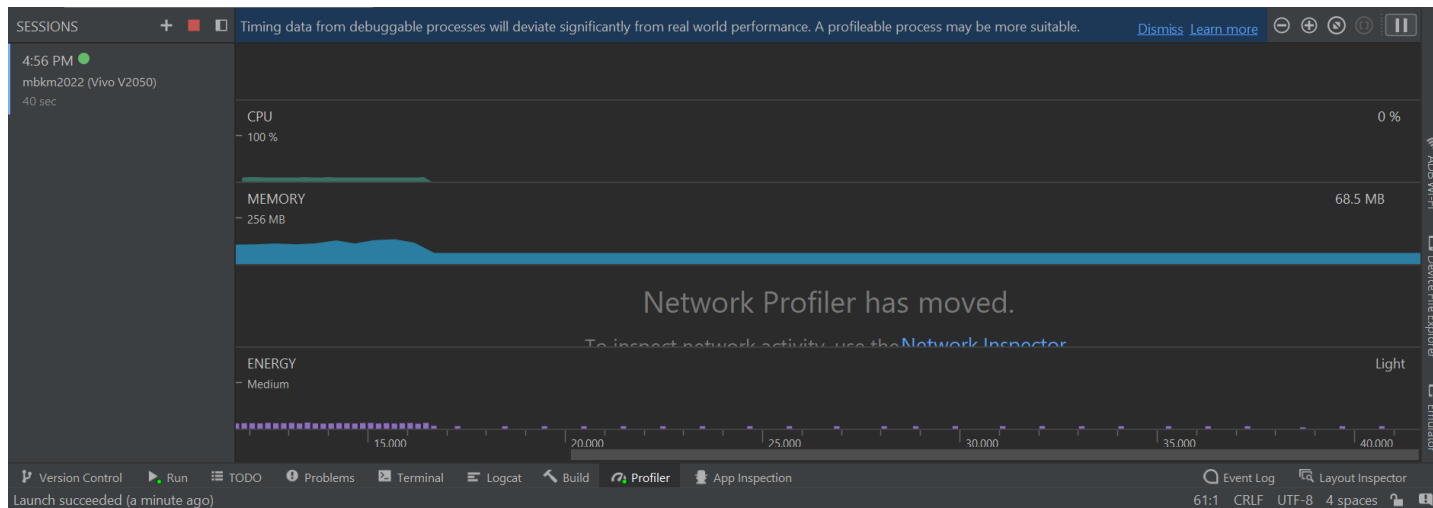
# Profiling



# Profiling

- Debugging, Unit Test dan Instrumentation Test adalah salah satu cara untuk meminimalisir kesalahan ketika membuat aplikasi
- Namun kadang-kadang ada masalah yang mungkin tidak terdeteksi oleh debugging, unit test atau bahkan instrumentation test
- Contoh yang sering terjadi adalah masalah memory leak
- Pada kasus ini, fitur Profiling sangatlah berguna
- Android mendukung fitur Profiling, dimana kita bisa memonitor aplikasi kita yang berjalan
- Kita bisa memonitor penggunaan memory, cpu, dan lain-lain

# Profiler





# Materi Selesai

# Quiz

---



## Quiz (1) – Knowledge Check

- Berikan penjelasan apa yang kalian ketahui tentang :
  - a. Activity
  - b. Fragment
  - c. View
  - d. Intent
  - e. Service
- Apa perbedaan fragment dan Activity? Dan ketika apa fragment itu dibutuhkan?
- Apa yang kalian ketahui tentang Main Thread pada Android?



## Quiz (2) – Membuat Kode

- Buatlah 2 activity sederhana yang menampilkan informasi sebagai berikut :
  - a. Login (username, password, login button, register button)
  - b. Register (fullname, email, username, password, confirm password, register button)
- Buatlah sebuah aplikasi sederhana dengan mengimplementasikan berikut ini :
  - a. Implementasi api GET dari situs <https://jsonplaceholder.typicode.com/photos>
  - b. Menggunakan RecyclerView untuk menampilkan list dari api.
  - c. Menampilkan image dari api list. Bisa menggunakan Glide, Picasso atau yang lainnya.





## Quiz - Prosedur

- Buatlah kelompok yang terdiri 5 orang
- Durasi pengerjaan kurang lebih 3 jam, bisa fleksibel tergantung waktu.
- Kirim hasil pengerjaan kuis ke alamat email [riky.fathoni@gmail.com](mailto:riky.fathoni@gmail.com)