

MODUL APLIKASI MOBILE [Pemrograman Android]



Disusun oleh:
Muhammad A. Safi'ie, S.Si, M.Kom

Materi :

Android Layout, UI Control, Listview, Recyclerview, Activity, Broadcast Listener, Service, AsyncTask, AsyncTask Loader, SQLite, Map, Sensor.

Program Diploma III Teknik Informatika FMIPA UNS

Copyright © 2018 by D3 Teknik Informatika | d3ti@mipa.uns.ac.id | <https://d3ti.mipa.uns.ac.id>

Daftar Isi

Daftar Isi	2
Materi 1 : Pengenalan Android Studio	3
Materi 2 : Layout dan UI Control	22
Materi 3 : Struktur Kontrol	31
Materi 4 : Perulangan	37
Materi 5 : Informasi peringatan untuk pengembangan.....	41
Materi 6 : Array dan Listview	43
Materi 7 : ArrayList dan Recyclerview.....	47
Materi 8 : Broadcast Receivers dan Service.....	56
Materi 9 : AsyncTask dan AsyncTaskLoader	67
Materi 10 : Database SQLite	77
Materi 11 : Sensor	89
Materi 12 : Google Map.....	92

Materi 1 : Pengenalan Android Studio

A. Waktu :

Alokasi Waktu : 2 x 100 menit

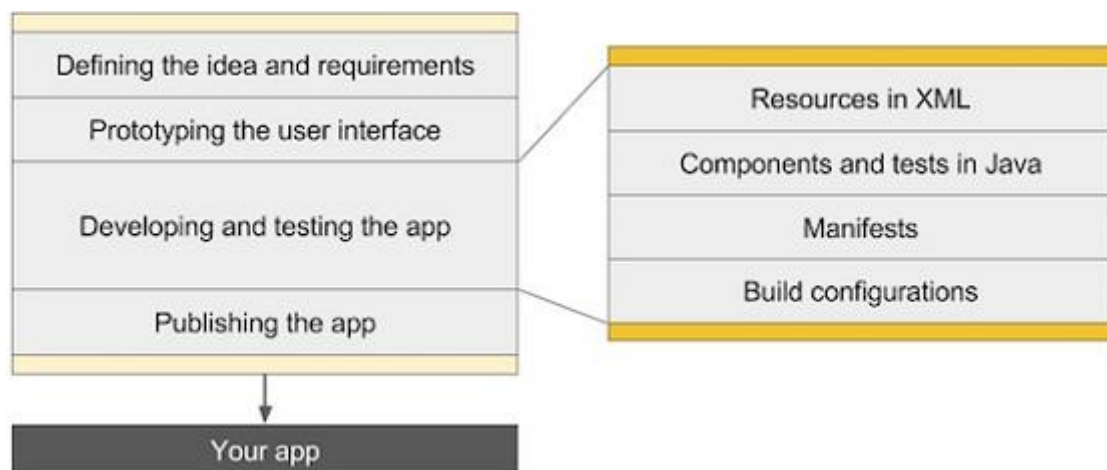
- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa memahami arsitektur protokol stack dalam sistem operasi Android.
2. Mahasiswa memahami file-file resource yang digunakan untuk membangun aplikasi.
3. Mahasiswa dapat menggunakan Android Studio untuk menulis, mengedit, mengcompile dan mendebug.
4. Mahasiswa dapat menggunakan Android Studio untuk mendesain User Interface dan menuliskan kode sumber.

C. Materi:

Proyek aplikasi Android dimulai dengan ide dan definisi persyaratan yang diperlukan untuk mewujudkan ide tersebut. Saat proyek berjalan, maka akan melalui desain, development, dan pengujian.



Gambar 1.1

Diagram 1.1. di atas adalah gambar proses development tingkat tinggi, dengan langkah-langkah berikut:

- Mendefinisikan ide dan persyaratannya: Sebagian besar aplikasi dimulai dengan ide tentang hal yang harus dilakukan, yang didukung pasar dan penelitian pengguna. Selama tahap ini, persyaratan aplikasi didefinisikan.
- Membuat prototipe antarmuka pengguna: Gunakan gambar, rekaan, dan prototipe untuk menampilkan tampilan antarmuka pengguna nantinya, dan cara kerjanya.
- Mengembangkan dan menguji aplikasi: Aplikasi terdiri dari satu atau beberapa aktivitas. Untuk setiap aktivitas, Anda bisa menggunakan Android Studio untuk melakukan hal berikut, tidak dalam urutan tertentu:
 - Buat layout: Tempatkan elemen UI pada layar di layout, dan tetapkan sumber daya string serta item menu, menggunakan Extensible Markup Language (XML).
 - Tulis kode Java: Buat referensi kode sumber untuk komponen dan pengujian, serta gunakan alat pengujian dan debug.
 - Daftarkan aktivitas: Deklarasikan aktivitas dalam file manifes.
 - Definisikan versi: Gunakan konfigurasi pembangunan default atau buat pembangunan khusus untuk versi aplikasi yang berbeda.
- Mempublikasikan aplikasi: Rakit APK (file paket) final dan distribusikan melalui saluran seperti Google Play

D. Praktikum

Android Studio menyediakan alat untuk pengujian, dan mempublikasikan tahap proses development, serta lingkungan development terpadu untuk membuat aplikasi bagi semua perangkat Android. Lingkungan development menyertakan kode template dengan kode contoh untuk fitur aplikasi umum, alat pengujian dan kerangka kerja yang banyak, dan system pembangunan yang fleksibel.

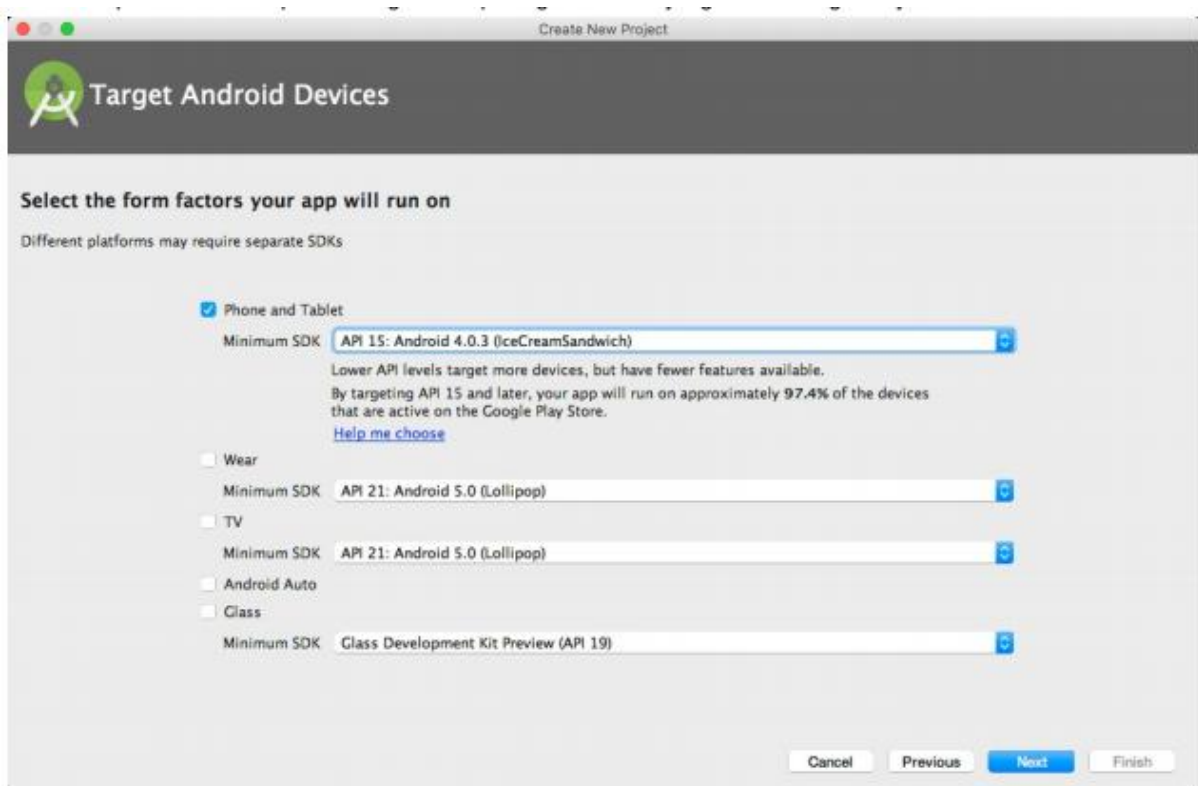
Setelah berhasil memasang Android Studio IDE, klik dua kali ikon aplikasi Android Studio untuk memulainya. Pilih **Start a new Android Studio project** di jendela Welcome, dan beri nama proyek dengan nama yang sama yang ingin Anda gunakan untuk aplikasi.

Saat memilih Domain Perusahaan unik, ingat bahwa aplikasi yang dipublikasikan ke Google Play harus memiliki nama paket unik. Karena domain bersifat unik, yang mengawali nama aplikasi dengan nama Anda, atau nama domain perusahaan, sebaiknya sediakan nama paket unik yang memadai. Jika tidak berencana untuk mempublikasikan aplikasi, Anda bisa menerima domain contoh default. Ketahuilah bahwa mengubah nama paket di lain waktu memerlukan kerja tambahan.

Saat memilih Target Android Devices, Phone and Tablet dipilih secara default, seperti yang ditampilkan dalam gambar di bawah ini. Pilihan yang ditampilkan dalam gambar untuk Minimum SDK — **API 1 5: Android 4.0.3**

(IceCreamSandwich) —membuat aplikasi Anda kompatibel dengan 97% perangkat Android yang aktif di Google Play Store.

Perangkat berbeda menjalankan versi sistem Android yang berbeda, seperti Android 4.0.3 atau Android 4.4. Setiap versi yang berurutan umumnya menambahkan API baru yang tidak tersedia di versi sebelumnya. Untuk menunjukkan rangkaian API yang tersedia, setiap versi menetapkan API level. Misalnya, Android 1.0 adalah API level 1 dan Android 4.0.3 dengan API 15. Minimum SDK mendeklarasikan versi Android minimum untuk aplikasi Anda. Setiap versi Android yang berurutan menyediakan kompatibilitas untuk aplikasi yang dibangun menggunakan API dari versi sebelumnya, sehingga aplikasi Anda akan *selalu* kompatibel dengan versi Android mendatang sambil menggunakan Android API yang di dokumentasikan.



Memilih template

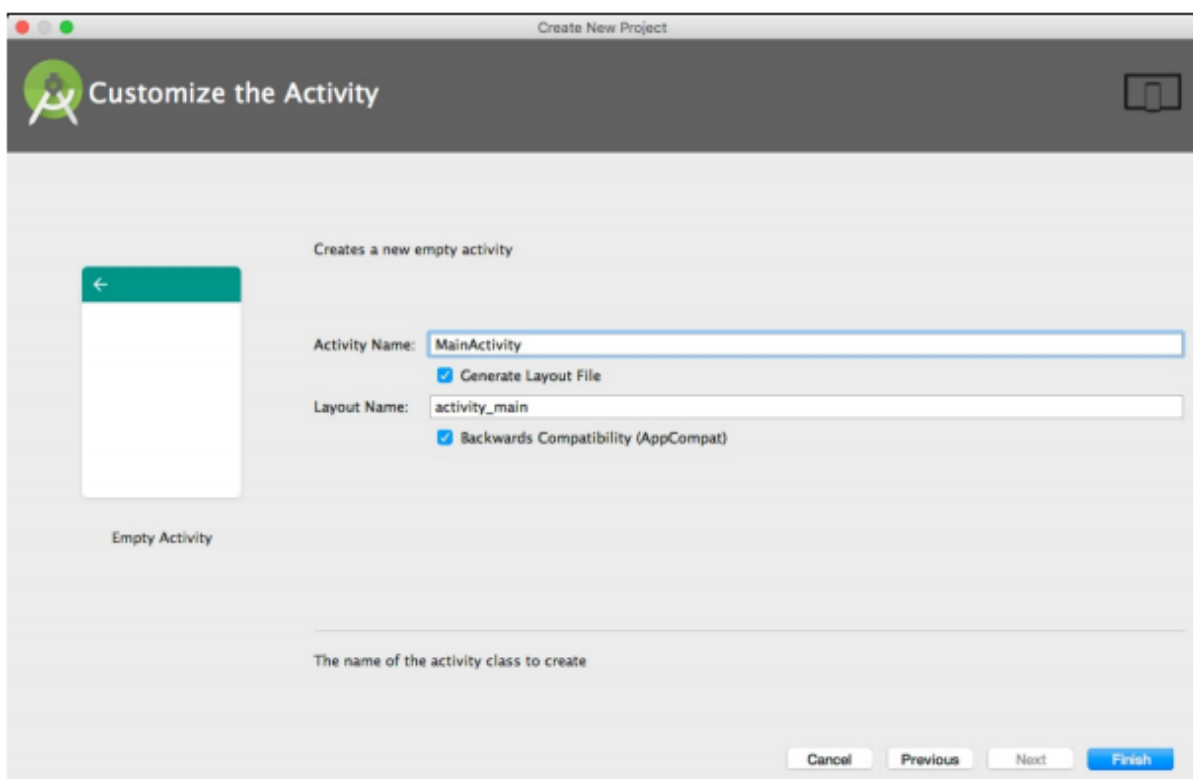
Android Studio mengumpulkan terlebih dulu proyek Anda dengan kode minimum untuk aktivitas dan layout layar berdasarkan template. Tersedia berbagai macam template, mulai dari template kosong virtual (Add No Activity) hingga beragam tipe aktivitas.

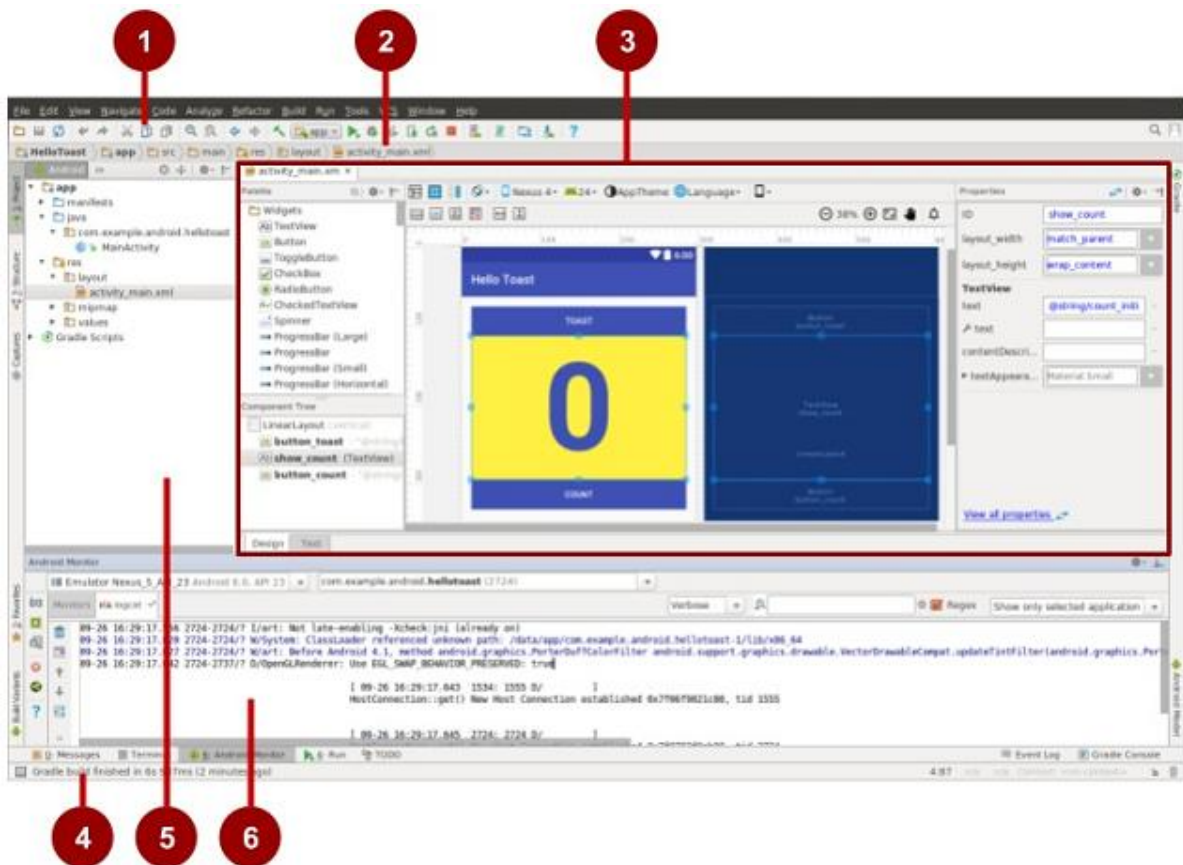
Anda bisa menyesuaikan aktivitas setelah memilih template. Misalnya, template Empty Activity menyediakan satu aktivitas yang disertai satu sumber daya layout untuk layar. Anda bisa memilih untuk menerima nama yang biasa digunakan untuk aktivitas (seperti MainActivity) atau mengubah nama di layar Customize Activity .

Selain itu, jika Anda menggunakan template Empty Activity, pastikan memeriksa yang berikut ini jika belum diperiksa:

- **Generate Layout File:** Biarkan ini dicentang untuk membuat sumber daya layout yang terhubung dengan aktivitas ini, yang biasanya diberi nama `activity_main.xml`. Layout mendefinisikan antarmuka pengguna untuk aktivitas.
- **Backwards Compatibility (AppCompat):** Biarkan ini dicentang untuk menyertakan pustaka AppCompat sehingga aplikasi kompatibel dengan Android versi sebelumnya bahkan jika menggunakan fitur yang hanya ditemukan di versi yang lebih baru.

Android Studio membuat folder untuk proyek yang baru saja dibuat di folder `AndroidStudioProjects` pada komputer Anda.





Jendela utama Android Studio terdiri dari sejumlah area logis, atau *panel*, seperti yang ditampilkan dalam gambar di atas. Dalam gambar di atas:

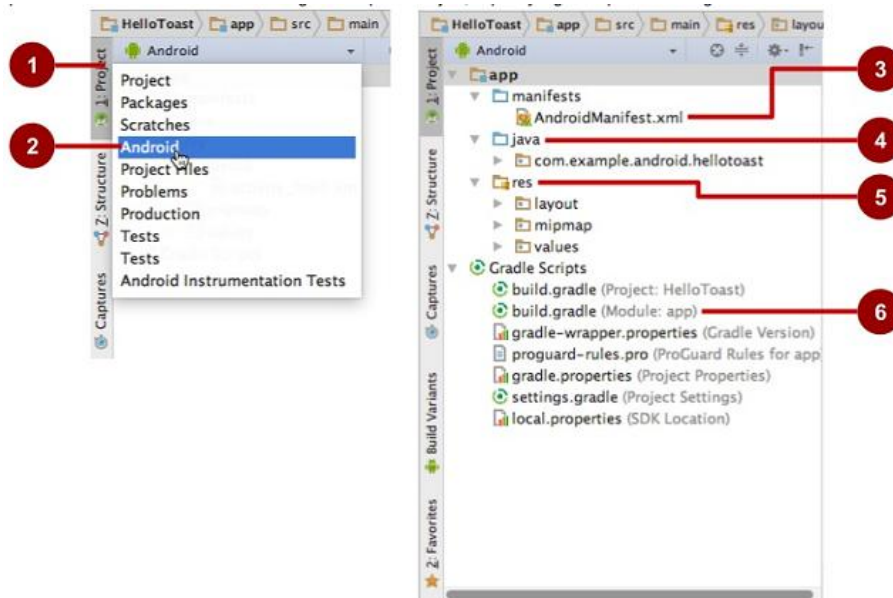
1. **Bilah Alat.** Bilah alat menjalankan beragam tindakan, termasuk menjalankan aplikasi Android dan meluncurkan alat Android.
2. **Bilah Navigasi.** Bilah navigasi memungkinkan navigasi melalui proyek dan membuka file untuk pengeditan. Bilah navigasi menyediakan tampilan struktur proyek yang lebih ringkas.
3. **Panel Editor.** Panel ini menampilkan materi file yang dipilih dalam proyek. Misalnya, setelah memilih layout (seperti yang ditampilkan dalam gambar), panel ini menampilkan editor layout dengan alat untuk mengedit layout. Setelah memilih file kode Java, panel ini menampilkan kode dengan alat untuk mengedit kode.
4. **Bilah Status.** Bilah status menampilkan status proyek dan Android Studio itu sendiri, serta peringatan atau pesan apa pun. Anda bisa mengamati kemajuan pembangunan di bilah status.
5. **Panel Project.** Panel proyek menampilkan file proyek dan hierarki proyek.
6. **Panel Monitor.** Panel monitor menawarkan akses ke daftar TODO untuk mengelola tugas, Android Monitor untuk memantau eksekusi aplikasi (ditampilkan dalam gambar), logcat untuk menampilkan pesan log, dan aplikasi Terminal untuk melakukan aktivitas Terminal.

Menjelajahi proyek

Setiap proyek di Android Studio berisi file AndroidManifest.xml, file kode sumber komponen, dan file sumber daya terkait. Secara default, Android Studio mengatur file proyek Anda berdasarkan jenis file, dan menampilkannya dalam proyek:

Tampilan Android di panel alat (bantu) kiri, seperti yang ditampilkan di bawah ini. Tampilan menyediakan akses cepat ke file kunci proyek Anda.

Untuk beralih kembali ke tampilan ini dari tampilan lainnya, klik tab Project vertikal di kolom paling kiri panel Project, dan pilih Android dari menu munculan di bagian atas panel Project, seperti yang ditampilkan dalam gambar di bawah ini.



Dalam gambar di atas:

1. Tab Project . Klik untuk menampilkan tampilan proyek.
2. Pilihan Android di menu tarik-turun proyek.
3. File AndroidManifest.xml. Digunakan untuk menetapkan informasi tentang aplikasi bagi lingkungan waktu proses Android. Template yang Anda pilih membuat file ini.
4. Folder java. Folder ini menyertakan aktivitas, pengujian, dan komponen lain di kode sumber Java. Setiap aktivitas, service, dan komponen lain didefinisikan sebagai kelas Java, biasanya dalam file miliknya. Nama aktivitas pertama (layar) yang pengguna lihat, yang juga melakukan inisialisasi sumber daya seluruh aplikasi, biasanya adalah MainActivity.
5. Folder res. Folder ini menyimpan sumber daya, seperti layout XML, string UI, dan gambar. Aktivitas biasanya dikaitkan dengan file sumber daya XML yang menetapkan layout tampilannya. File ini biasanya diberi nama setelah aktivitas atau fungsinya.
6. File **build.gradle (Module: App

7.)*. File ini menetapkan konfigurasi pembangunan modul. Template yang Anda pilih membuat file ini, yang mendefinisikan konfigurasi pembangunan, termasuk atribut `minSdkVersion` yang mendeklarasikan versi minimum untuk aplikasi, dan atribut `targetSdkVersion` yang mendeklarasikan versi tertinggi (terbaru) untuk aplikasi yang dioptimalkan. File ini juga menyertakan daftar dependensi*, yaitu pustaka yang diperlukan oleh kode — seperti pustaka `AppCompat` untuk mendukung beragam versi Android.

Menampilkan Manifes Android

Sebelum sistem Android bisa memulai komponen aplikasi, sistem harus mengetahui bahwa komponen ada dengan membaca file `AndroidManifest.xml` aplikasi. Aplikasi harus mendeklarasikan semua komponennya dalam file ini, yang harus berada pada akar direktori proyek aplikasi.

Untuk menampilkan file ini, luaskan folder `manifest` di `Project: Tampilan Android`, dan klik dua kali file (`AndroidManifest.xml`). Isinya muncul di panel pengeditan seperti yang ditampilkan dalam gambar di bawah ini.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.android.helloworld">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Hello World"
        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>
```

Namespace dan tag aplikasi Android

Manifes Android dikodekan di XML dan selalu menggunakan namespace Android:

```
xmlns: android="http://schemas.android.com/apk/res/android"
package="com.example.android.helloworld">
```

Ekspresi `package` menampilkan nama paket unik aplikasi baru. Jangan mengubah ini setelah aplikasi dipublikasikan.

```
<application
...
</application>
```

Tag `<application`, dengan tag `</application>` penutupnya, mendefinisikan setelan manifes untuk keseluruhan aplikasi.

Ikon aplikasi

Metode android: icon menyetel ikon untuk aplikasi:

```
...  
android: allowBackup="true"  
android: icon="@mipmap/ic_launcher"  
...
```

Atribut android: icon menetapkan ikon di folder mipmap (di dalam folder res di Project: tampilan Android) untuk aplikasi.

Ikon muncul di Peluncur untuk meluncurkan aplikasi. Ikon juga digunakan sebagai ikon default untuk komponen aplikasi.

Sumber daya string dan label aplikasi

Seperti yang bisa Anda lihat di gambar sebelumnya, atribut android: label menampilkan string "Hello World" yang disorot. Jika Anda mengeklik string ini, maka string berubah untuk menampilkan sumber daya string @string/app_name :

```
...  
android: label="@string/app_name"  
...
```

Tip: Ctrl - klik atau klik-kanan app_name di panel edit untuk melihat menu konteks. Pilih Go To > Declaration untuk melihat lokasi sumber daya string dideklarasikan: dalam file strings.xml. Bila Anda memilih Go To > Declaration atau membuka file dengan mengeklik dua kali strings.xml di Project: Tampilan Android (di dalam folder values), isinya muncul di panel pengeditan.

Setelah membuka file strings.xml, Anda bisa melihat bahwa nama string app_name disetel ke Hello World . Anda bisa mengubah nama aplikasi dengan mengubah string Hello World ke hal lain. Sumber daya string dijelaskan dalam pelajaran terpisah.

Tema aplikasi

Atribut android: theme menyetel tema aplikasi, yang mendefinisikan penampilan elemen antarmuka pengguna seperti teks:

```
...  
android: theme="@style/AppTheme">  
...
```

Atribut theme disetel ke tema standar AppTheme . Tema dijelaskan dalam pelajaran terpisah.

Mendeklarasikan versi Android

Perangkat yang berbeda mungkin menjalankan versi sistem Android yang berbeda, seperti Android 4.0 atau Android 4.4.

Setiap versi yang berurutan bisa menambahkan API baru yang tidak tersedia di versi sebelumnya. Untuk menunjukkan rangkaian API yang tersedia, setiap versi menetapkan API level. Misalnya, Android 1.0 adalah API level 1 dan Android 4.4 adalah API level 19.

API level memungkinkan developer untuk mendeklarasikan versi minimum dengan aplikasi yang kompatibel, menggunakan tag manifest `<uses-sdk>` dan atribut `minSdkVersion`. Misalnya, Calendar Provider API telah ditambahkan di Android 4.0 (API level 14). Jika aplikasi Anda tidak berfungsi tanpa API tersebut, deklarasikan API level 14 sebagai versi yang didukung minimum aplikasi seperti ini:

```
<manifest . . . >
<uses-sdk android: minSdkVersion="14" android: targetSdkVersion="19" />
. . .
</manifest>
```

Atribut `minSdkVersion` mendeklarasikan versi minimum untuk aplikasi, dan atribut `targetSdkVersion` mendeklarasikan versi tertinggi (terbaru) yang telah dioptimalkan dalam aplikasi. Setiap versi Android yang berurutan menyediakan kompatibilitas untuk aplikasi yang dibangun menggunakan API dari versi sebelumnya, sehingga aplikasi akan selalu kompatibel dengan versi Android mendatang sambil menggunakan Android API yang didokumentasikan.

Atribut `targetSdkVersion` tidak mencegah aplikasi dipasang pada versi Android yang lebih tinggi (yang lebih baru) dibandingkan nilai yang ditetapkan, namun hal ini penting karena menunjukkan pada sistem apakah aplikasi harus mewarisi perubahan perilaku di versi yang baru. Jika Anda tidak memperbarui `targetSdkVersion` ke versi terbaru, sistem akan menganggap bahwa aplikasi memerlukan perilaku kompatibilitas mundur saat dijalankan pada versi terbaru. Misalnya, di antara perubahan perilaku di Android 4.4, alarm yang dibuat dengan AlarmManager API saat ini tidak tepat secara default sehingga sistem bisa mengumpulkan alarm aplikasi dan mempertahankan kekuatan sistem, namun sistem akan menahan perilaku API sebelumnya untuk aplikasi jika API level target Anda lebih rendah daripada "19".

Menampilkan dan mengedit kode Java

Komponen ditulis di Java dan dicantumkan dalam folder modul di folder java dalam Project: Tampilan Android. Setiap nama modul diawali dengan nama domain (seperti `com.example.android`) dan menyertakan nama aplikasi.

Contoh berikut menampilkan komponen aktivitas:

1. Klik folder modul untuk meluaskan dan menampilkan file MainActivity untuk aktivitas yang ditulis di Java (kelas MainActivity).
2. Klik dua kali MainActivity untuk melihat sumber di panel pengeditan, seperti yang ditampilkan dalam gambar di bawah ini.

```

package com.example.android.helloworld;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}

```

Bagian paling atas file MainActivity.java adalah pernyataan package yang mendefinisikan paket aplikasi. Ini diikuti dengan blok import yang diringkas dalam gambar di atas, dengan " . . . ". Klik titik untuk meluaskan blok guna menampilkannya.

Pernyataan import akan mengimpor pustaka yang diperlukan untuk aplikasi, seperti berikut ini, yang mengimpor pustaka AppCompatActivity:

```
import android.support.v7.app.AppCompatActivity;
```

Setiap aktivitas dalam aplikasi diimplementasikan sebagai kelas Java. Deklarasi kelas berikut memperluas kelas AppCompatActivity untuk mengimplementasikan fitur dengan cara yang kompatibel-mundur dengan Android versi sebelumnya:

```

MainActivity kelas publik memperluas AppCompatActivity {
...
}

```

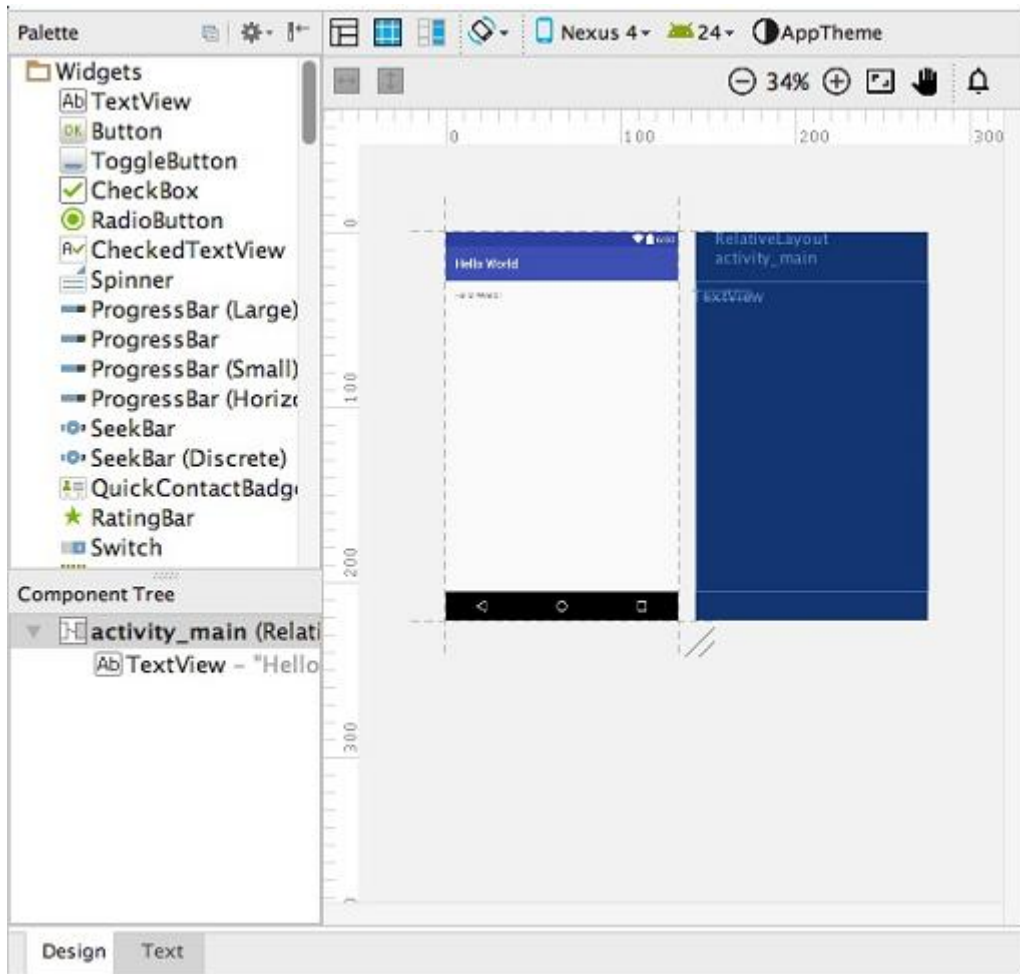
Seperti yang Anda pelajari terdahulu, sebelum sistem Android bisa memulai komponen aplikasi seperti aktivitas, sistem harus mengetahui bahwa aktivitas ada dengan membaca file AndroidManifest.xml aplikasi. Oleh karena itu, setiap aktivitas harus dicantumkan di file AndroidManifest.xml.

Menampilkan dan mengedit layout

Sumber daya layout ditulis dalam XML dan dicantumkan dalam folder layout di folder res dalam Project: Tampilan Android.

Klik res > layout, kemudian klik dua kali activity_main.xml untuk melihat file layout di panel pengeditan.

Android Studio menampilkan tampilan Desain layout, seperti yang ditampilkan dalam gambar di bawah ini. Tampilan ini menyediakan panel Palette elemen antarmuka pengguna, dan petak yang menampilkan layout layar.



Memahami proses pembangunan

Android Application Package (APK) adalah format file paket untuk mendistribusikan dan memasang aplikasi seluler Android. Proses pembangunan melibatkan alat dan proses yang secara otomatis mengonversi setiap proyek menjadi APK.

Android Studio menggunakan Gradle sebagai dasar sistem pembangunan, dengan kemampuan khusus Android lebih banyak yang disediakan oleh Android Plugin for Gradle. Sistem pembangunan ini berjalan sebagai alat (bantu) terintegrasi dari menu Android Studio.

Memahami file build.gradle



Bila Anda membuat proyek, Android Studio secara otomatis menghasilkan file pembangunan penting di folder Gradle Scripts dalam Project: Tampilan Android. File pembangunan Android Studio diberi nama build.gradle seperti yang ditampilkan di bawah ini:

Setiap proyek memiliki hal berikut:

build.gradle (Project: apptitle)

Ini adalah file pembangunan tingkat atas untuk keseluruhan proyek, berada di akar direktori proyek, yang mendefinisikan konfigurasi pembangunan yang berlaku untuk semua modul di proyek Anda. File ini, yang dihasilkan oleh Android Studio, tidak boleh diedit untuk menyertakan dependensi aplikasi.

build.gradle (Module: app)

Android Studio membuat file build. gradle (Module: app) terpisah untuk setiap modul. Anda bisa mengedit setelan pembangunan guna menyediakan opsi pemaketan khusus untuk setiap modul, seperti tipe pembangunan tambahan dan ragam produk, dan untuk menggantikan setelan di file manifes atau build.gradle tingkat atas. File ini paling sering adalah file untuk mengedit ketika mengubah konfigurasi tingkat aplikasi, seperti mendeklarasikan dependensi di bagian dependencies . Yang berikut ini menampilkan isi file build. gradle (Module: app) proyek:

```
apply plugin: ' com. android. application'
android {
  compileSdkVersion 24
  buildToolsVersion "24. 0. 1"
  defaultConfig {
    applicationId "com. example. android. helloworld2"
    minSdkVersion 15
    targetSdkVersion 24
    versionCode 1
    versionName "1. 0"
    testInstrumentationRunner "android. support. test. runner. AndroidJUnitRunner"
  }buildTypes {
    rilis {
      minifyEnabled false
      proguardFiles getDefaultProguardFile( ' proguard- android. txt' ) , ' proguard- rules.
      pro'
    }
  }
}dependencies {
  compile fileTree( dir: ' libs' , include: [ ' *. j ar' ] )
  androidTestCompile( ' com. android. support. test. espresso: espresso- core: 2. 2. 2'
  , {
    exclude group: ' com. android. support' , module: ' support- annotations'
  })
  compile ' com. android. support: appcompat- v7: 24. 2. 1'
```

```
testCompile 'junit:junit:4.12'
```

}
File build.gradle menggunakan sintaks Gradle. Gradle adalah Domain Specific Language (DSL) untuk menjelaskan dan

memanipulasi logika pembangunan dengan menggunakan Groovy, yang merupakan bahasa dinamis untuk Java Virtual

Machine (JVM). Anda tidak perlu mempelajari Groovy untuk melakukan perubahan, karena Android Plugin for Gradle memperkenalkan sebagian besar elemen DSL yang Anda perlukan.

Tip: Untuk mengetahui selengkapnya tentang DSL plugin Android, baca Dokumentasi referensi DSL.

Blok Plugin dan Android.

Dalam file build.gradle (Module: app) di atas, pernyataan pertama menerapkan tugas pembangunan plug-in Gradle

khusus Android:

```
apply plugin: 'com.android.application'
android {
    ...
}
```

Blok android { } menetapkan hal berikut ini untuk versi:

Versi SDK target untuk mengompilasi kode:

```
compileSdkVersion 24
```

Versi alat pembangunan untuk digunakan membangun aplikasi:

```
buildToolsVersion "24.0.1"
```

Blok defaultConfig

Setelan inti dan entri untuk aplikasi ditetapkan di blok defaultConfig { } dalam android { } block:

```
...
defaultConfig {
    applicationId "com.example.hello.helloworld"
    minSdkVersion 15
    targetSdkVersion 23
    versionCode 1
    versionName "1.0"
    testInstrumentationRunner
    "android.support.test.runner.AndroidJUnitRunner"
}
```

...

Setelan `minSdkVersion` dan `targetSdkVersion` menggantikan setelan `AndroidManifest.xml` untuk versi SDK minimum dan versi SDK target. Lihat "Mendeklarasikan versi Android" sebelumnya di bagian ini untuk informasi latar belakang tentang setelan ini.

Pernyataan `testInstrumentationRunner` ini menambahkan dukungan instrumentasi untuk menguji antarmuka pengguna dengan Espresso dan UIAutomator. Hal ini dijelaskan dalam pelajaran terpisah.

Tipe pembangunan

Tipe pembangunan untuk aplikasi ditetapkan di blok `buildTypes { }`, yang mengontrol cara aplikasi dibangun dan dipaketkan.

...

```
buildTypes {
  rilis {
    minifyEnabled false
    proguardFiles getDefaultProguardFile('proguard-android.txt'),
    'proguard-rules.pro'
  }
}
```

..

Tipe pembangunan yang ditetapkan adalah `release` untuk rilis aplikasi. Tipe pembangunan umum lainnya adalah `debug`. Mengonfigurasi tipe pembangunan dijelaskan dalam pelajaran terpisah.

Dependensi

Dependensi untuk aplikasi didefinisikan di blok `dependencies { }`, yang merupakan bagian file `build.gradle` yang paling mungkin berubah saat Anda mulai mengembangkan kode yang bergantung pada pustaka. Blok adalah bagian dari Gradle API standar dan termasuk di luar blok `android { }`.

...

```
dependencies {
  compile fileTree(dir: 'libs', include: ['*.jar'])
  androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2',
  {
    exclude group: 'com.android.support', module: 'support-annotations'
  })
  compile 'com.android.support:appcompat-v7:24.2.0'
  testCompile 'junit:junit:4.12'
}
```

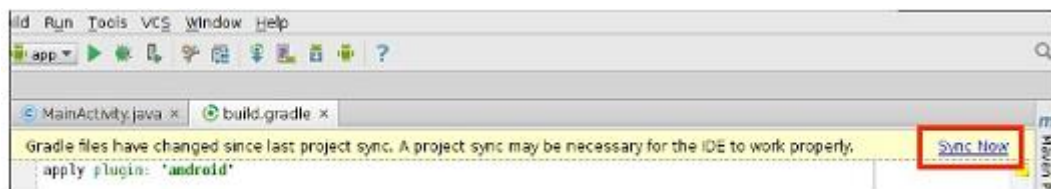
Dalam cuplikan di atas, pernyataan `compile fileTree(dir: 'libs' , include: [' *.jar'])` menambahkan dependensi semua file ".jar" di dalam direktori `libs`. Konfigurasi

compile akan mengompilasi aplikasi utama — segala sesuatu di dalamnya ditambahkan ke classpath kompilasi, dan juga dipaketkan ke dalam APK final.

Menyinkronkan proyek Anda

Bila Anda membuat perubahan pada file konfigurasi pembangunan dalam proyek, Android Studio akan mengharuskan Anda untuk melakukan sinkronisasi file proyek sehingga Android Studio bisa mengimpor perubahan konfigurasi pembangunan dan menjalankan beberapa pemeriksaan untuk memastikan konfigurasi tidak akan menimbulkan kesalahan pembangunan.

Untuk menyinkronkan file proyek, klik Sync Now di bilah notifikasi yang muncul saat membuat perubahan, atau klik Sync Project dari bilah menu. Jika Android Studio memperlihatkan kesalahan apa pun dengan konfigurasi — misalnya, jika kode sumber menggunakan fitur API yang hanya tersedia di API level yang lebih tinggi dari `compileSdkVersion` — jendela Messages muncul untuk menjelaskan masalah.



Menjalankan aplikasi pada emulator atau perangkat

Dengan emulator perangkat virtual, Anda bisa menguji aplikasi pada perangkat yang berbeda seperti tablet atau ponsel cerdas — dengan API level yang berbeda untuk versi Android yang berbeda — untuk memastikan aplikasi terlihat bagus dan berfungsi untuk sebagian besar pengguna. Meskipun ini adalah ide yang bagus, Anda tidak harus bergantung pada memiliki perangkat fisik yang tersedia untuk development aplikasi.



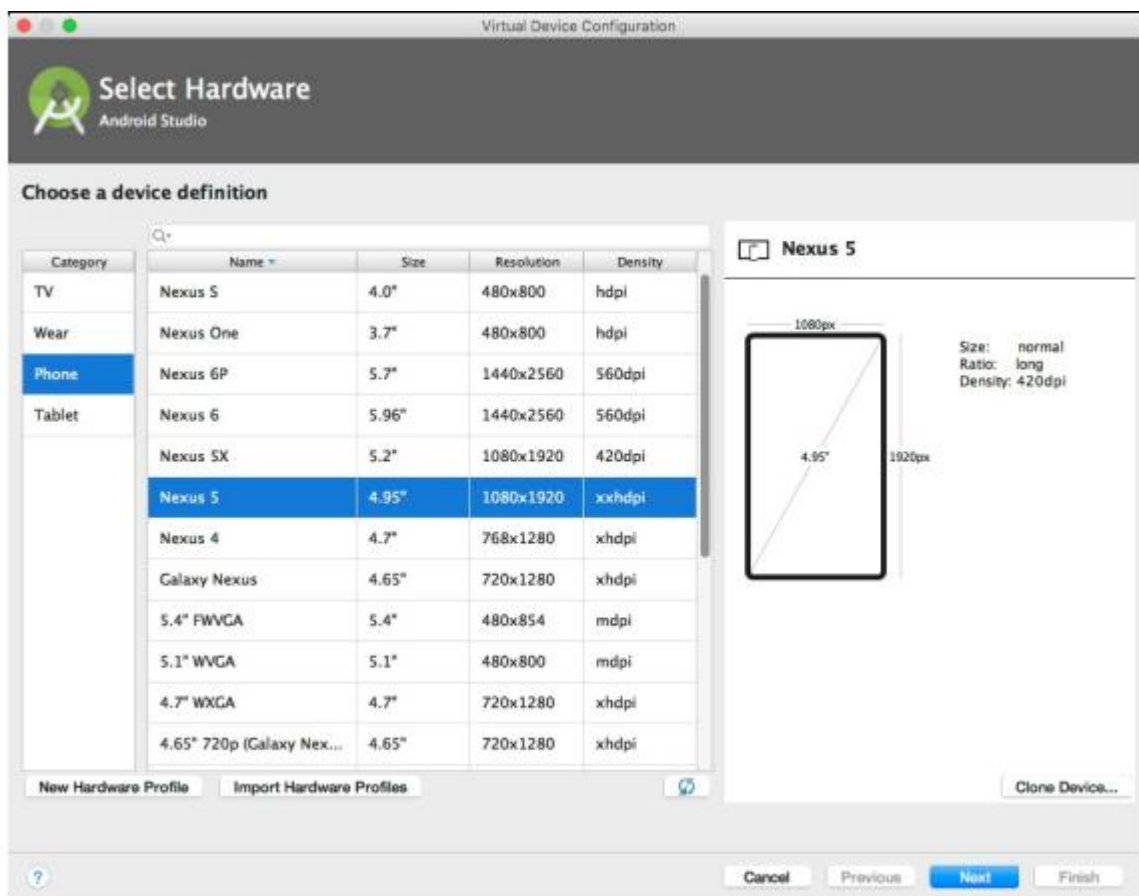
Pengelola Android Virtual Device (AVD) membuat perangkat virtual atau emulator yang menyimulasikan konfigurasi untuk tipe perangkat Android tertentu. Gunakan AVD Manager untuk mendefinisikan karakteristik perangkat keras perangkat dan API levelnya, dan untuk menyimpannya sebagai konfigurasi perangkat virtual. Ketika Anda memulai emulator Android, emulator akan membaca konfigurasi yang ditetapkan dan membuat perangkat emulasi pada komputer yang berperilaku sama persis dengan versi fisik perangkat.

Membuat perangkat virtual

Untuk menjalankan emulator pada komputer, gunakan AVD Manager untuk membuat konfigurasi yang menjelaskan perangkat virtual. Pilih Tools > Android > AVD Manager, atau klik ikon AVD Manager di bilah alat.

Layar "Your Virtual Devices" muncul menampilkan semua perangkat virtual yang dibuat sebelumnya. Klik tombol +Create Virtual Device untuk membuat perangkat virtual baru.

Anda bisa memilih perangkat dari daftar perangkat keras yang telah didefinisikan sebelumnya. Untuk setiap perangkat, tabel menampilkan ukuran tampilan diagonal (Size), resolusi layar dalam piksel (Resolution), dan kepadatan piksel (Density). Misalnya, kepadatan piksel perangkat Nexus 5 adalah xxhdpi, artinya aplikasi menggunakan ikon di folder



xxhdpi dari folder mipmap. Selain itu, aplikasi akan menggunakan layout dan sumber daya dapat digambar dari folder yang didefinisikan untuk kepadatan tersebut. Anda juga memilih versi sistem Android untuk perangkat. Tab Recommended menampilkan sistem yang disarankan untuk perangkat. Lebih banyak versi tersedia di dalam tab x86 Images dan Other Images.

Menjalankan aplikasi pada perangkat virtual

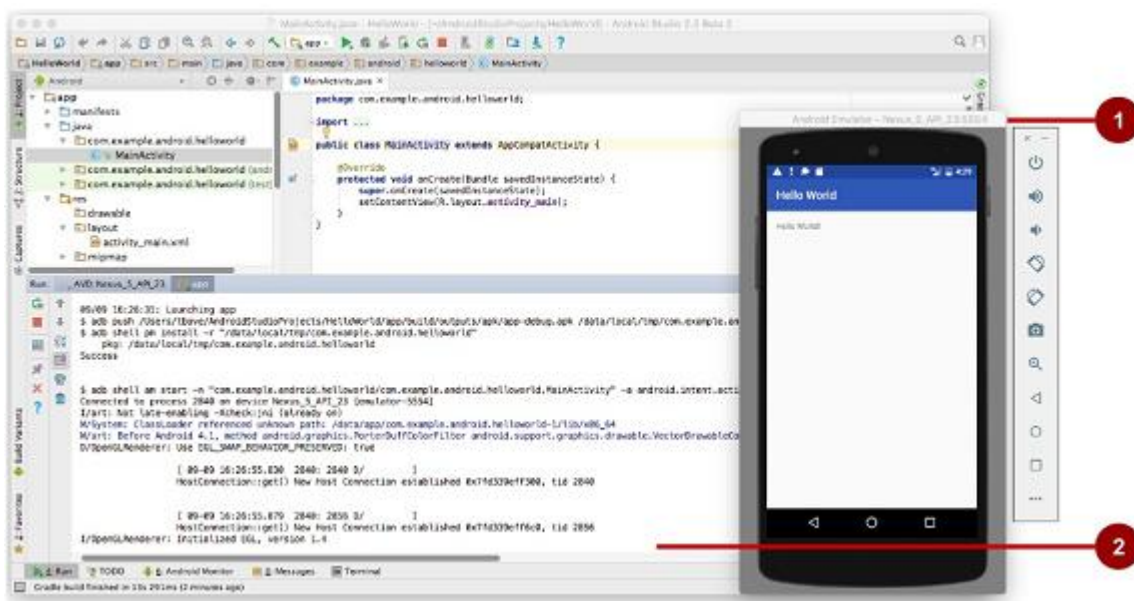
Untuk menjalankan aplikasi pada perangkat virtual yang Anda buat di bagian sebelumnya, ikuti langkah-langkah ini:

1. Di Android Studio, pilih Run > Run app atau klik Run icon di bilah alat.
2. Di jendela Select Deployment Target, pada Emulator yang tersedia, pilih perangkat virtual yang Anda buat, dan klik OK.

Emulator memulai dan mem-booting seperti perangkat fisik. Ini memerlukan waktu beberapa saat, bergantung pada kecepatan komputer Anda. Aplikasi membangun, dan setelah emulator siap, Android Studio mengunggah aplikasi ke emulator dan menjalankannya.

Anda harus melihat aplikasi yang dibuat dari template Empty Activity ("Hello World") seperti yang ditampilkan dalam gambar berikut ini, yang juga menampilkan panel Run Android Studio yang menampilkan tindakan yang dilakukan untuk menjalankan aplikasi di emulator.

Catatan: Saat menguji pada emulator, praktik yang baik adalah memulainya sekali pada awal sesi, dan jangan menutupnya hingga selesai sehingga tidak perlu melalui proses booting kembali.



Dalam gambar di atas:

1. Emulator menjalankan aplikasi.

2. Run Pane. Ini menampilkan tindakan yang dilakukan untuk memasang dan menjalankan aplikasi.

Menjalankan aplikasi pada perangkat fisik

Selalu uji aplikasi Anda pada perangkat fisik, karena pengguna akan menggunakan aplikasi pada perangkat fisik. Meskipun emulator cukup bagus, emulator tidak bisa menampilkan semua kemungkinan keadaan perangkat, seperti yang terjadi jika ada panggilan masuk sewaktu aplikasi berjalan. Untuk menjalankan aplikasi pada perangkat fisik, Anda memerlukan hal berikut ini:

- Perangkat Android seperti ponsel cerdas atau tablet.
- Kabel data untuk menghubungkan perangkat Android ke komputer melalui porta USB.
- Jika Anda menggunakan Linux atau Windows, mungkin perlu melakukan langkah tambahan untuk menjalankan aplikasi pada perangkat keras. Periksa dokumentasi Menggunakan Perangkat Keras. Pada Windows, Anda mungkin perlu memasang driver USB yang sesuai untuk perangkat. Lihat Driver USB OEM.

Untuk memungkinkan Android Studio berkomunikasi dengan perangkat, aktifkan USB Debugging pada perangkat Android.

Pada versi Android 4.2 dan yang lebih baru, layar opsi Developer disembunyikan secara default. Ikuti langkah-langkah ini

untuk mengaktifkan USB Debugging:

1. Pada perangkat fisik, buka Settings dan pilih About phone di bagian bawah layar Settings.
2. Ketuk informasi Build number tujuh kali.

Anda membacanya dengan benar: Ketuk tujuh kali.

3. Kembali ke layar sebelumnya (Settings). Developer Options saat ini muncul di bagian bawah layar. Ketuk Developer options.

4. Pilih USB Debugging.

Sekarang, hubungkan perangkat dan jalankan aplikasi dari Android Studio.

Pemecahan masalah koneksi perangkat

Jika Android Studio tidak mengenali perangkat, coba hal berikut:

1. Putuskan koneksi perangkat dari komputer, kemudian hubungkan kembali.
2. Mulai ulang Android Studio.

3. Jika komputer masih tidak menemukan perangkat atau mendeklarasikannya "tidak sah":

- i. Putuskan koneksi perangkat dari komputer.
- ii. Di perangkat, pilih Settings > Developer Options.
- iii. Ketuk Revoke USB Debugging authorizations.
- iv. Hubungkan kembali perangkat ke komputer.
- v. Bila dikonfirmasi, berikan otorisasi.

4. Anda mungkin perlu memasang driver USB yang sesuai untuk perangkat. Lihat dokumentasi Menggunakan Perangkat Keras.

5. Periksa dokumentasi terakhir, forum pemrograman, atau dapatkan bantuan dari instruktur Anda.

E. Latihan/Tugas

Materi 2 : Layout dan UI Control

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

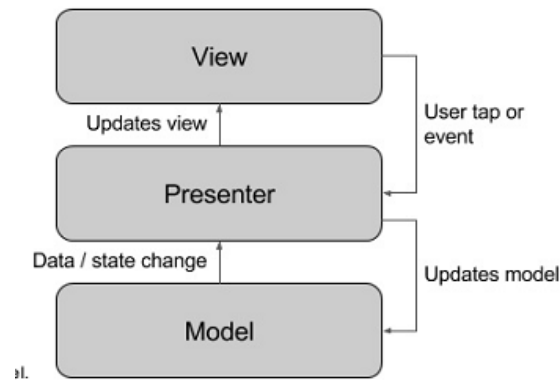
1. Mahasiswa menggunakan macam-macam layout.
2. Mahasiswa memahami karakteristik-karakteristik layout
3. Mahasiswa dapat membuat aplikasi sederhana menggunakan macam-macam layout.

C. Materi:

Pola model-view-presenter

Menautkan aktivitas ke sumber daya layout adalah contoh dari bagian pola arsitektur model-view-presenter (MVP). Pola MVP adalah cara yang sudah umum digunakan untuk mengelompokkan fungsi aplikasi:

- Tampilan. Tampilan adalah elemen antarmuka pengguna yang menampilkan data dan respons terhadap tindakan pengguna. Setiap elemen layar adalah tampilan. Sistem Android menyediakan berbagai jenis tampilan.
- Presenter. Presenter menghubungkan tampilan aplikasi ke model. Presenter menyediakan tampilan dengan data sebagaimana ditetapkan oleh model, dan juga menyediakan masukan pengguna dari tampilan kepada model.
- Model. Model menetapkan struktur data aplikasi dan kode untuk mengakses dan memanipulasi data. Beberapa aplikasi yang Anda buat dalam pelajaran bisa digunakan bersama model untuk mengakses data. Aplikasi Hello Toast tidak menggunakan model data, namun Anda bisa memikirkan logikanya — menampilkan pesan, dan meningkatkan penghitung ketukan — sebagai model.



Tampilan

UI terdiri dari hierarki objek yang disebut tampilan, setiap elemen layar adalah tampilan. Kelas View menyatakan blok pembangunan dasar untuk semua komponen UI, dan kelas dasar untuk kelas yang menyediakan komponen UI interaktif seperti tombol, kotak centang, dan bidang entri teks.

Tampilan memiliki lokasi, yang dinyatakan sebagai pasangan koordinat kiri dan atas, dan dua dimensi, yang dinyatakan sebagai lebar dan tinggi. Unit untuk lokasi dan dimensi adalah piksel yang tidak tergantung perangkat (dp).

Sistem Android menyediakan ratusan tampilan yang telah didefinisikan sebelumnya, termasuk yang menampilkan:

- Teks (TextView)
- Bidang untuk memasukkan dan mengedit teks (EditText)
- Pengguna tombol bisa mengetuk (Button) dan komponen interaktif lainnya
- Teks yang bisa digulir (ScrollView) dan item yang bisa digulir (RecyclerView)
- Gambar (ImageView)

Anda bisa mendefinisikan tampilan untuk muncul di layar dan merespons ketukan pengguna. Tampilan juga bisa didefinisikan untuk menerima masukan teks, atau tidak terlihat hingga diperlukan.

Anda bisa menetapkan tampilan di file sumber daya layout XML. Sumber daya layout ditulis dalam XML dan dicantumkan dalam folder layout di folder res dalam Project: Tampilan Android.

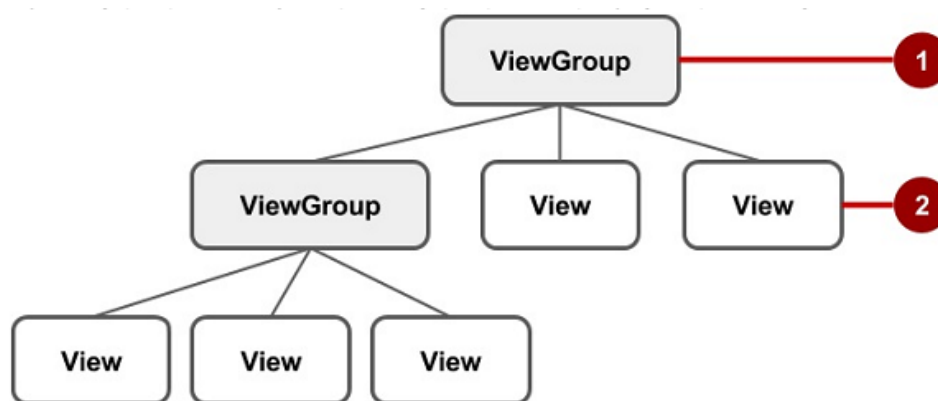
Grup tampilan

Tampilan bisa dikelompokkan bersama di dalam grup tampilan (ViewGroup), yang berfungsi sebagai kontainer tampilan. Hubungannya adalah induk-anak, dalam hal ini induk adalah grup tampilan, dan anak adalah tampilan atau grup tampilan dalam grup. Berikut ini adalah grup tampilan yang umum:

- ScrollView: Grup yang berisi satu tampilan anak lainnya dan memungkinkan pengguliran tampilan anak.
- RecyclerView: Grup yang berisi daftar tampilan atau grup tampilan lainnya dan memungkinkan penggulirannya dengan menambahkan dan membuang tampilan secara dinamis dari layar.

Grup tampilan layout

Tampilan untuk layar dikelola dalam hierarki. Di akar hierarki ini adalah ViewGroup yang berisi layout keseluruhan layar. Layar anak grup tampilan bisa menjadi tampilan atau grup tampilan lain seperti yang ditampilkan dalam gambar berikut.



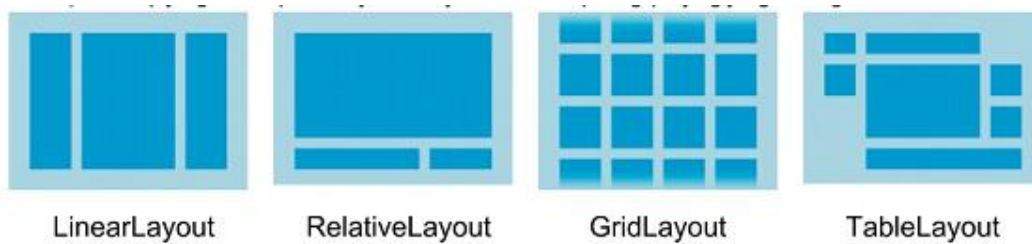
Dalam gambar di atas:

1. Grup tampilan akar.
2. Rangkaian tampilan anak dan grup tampilan pertama yang induknya adalah akar.

Beberapa grup tampilan ditandai sebagai layout karena grup tampilan tersebut mengelola tampilan anak dalam cara khusus dan umumnya digunakan sebagai grup tampilan akar. Beberapa contoh layout adalah:

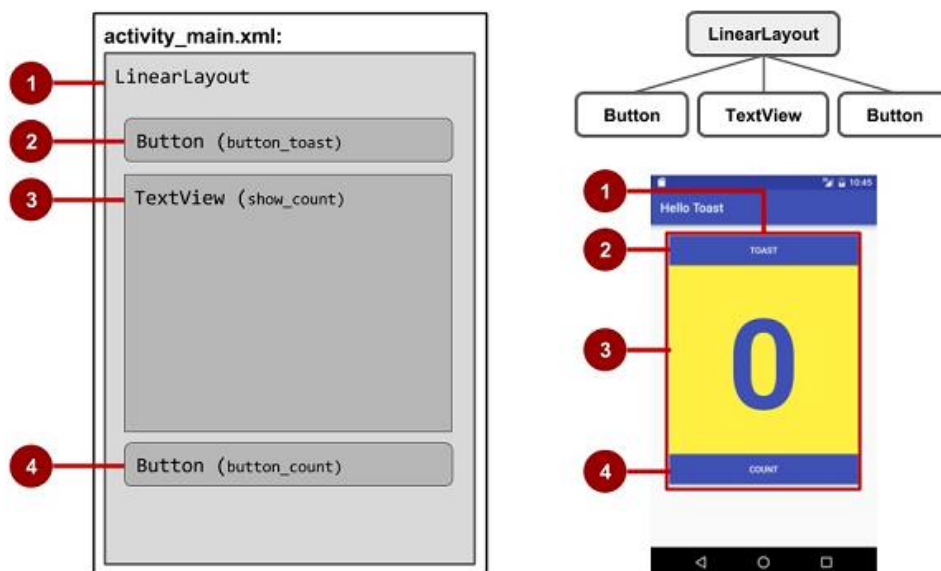
- LinearLayout: Grup tampilan anak yang diposisikan dan disejajarkan secara horizontal atau secara vertikal.
- RelativeLayout: Grup tampilan anak yang setiap tampilannya diposisikan dan disejajarkan relatif terhadap tampilan dalam grup tampilan. Dengan kata lain, posisi tampilan anak bisa dijelaskan dalam hubungan satu sama lain atau dengan grup tampilan induk.
- ConstraintLayout: Grup tampilan anak yang menggunakan titik jangkar, tepi, dan panduan untuk mengontrol cara memosisikan tampilan relatif terhadap elemen lain di layout. ConstraintLayout didesain untuk mempermudah saat menyeret dan melepaskan tampilan di editor layout.
- TableLayout: Grup tampilan anak yang disusun ke dalam baris dan kolom.

- **LinearLayout:** Grup yang memungkinkan Anda menetapkan lokasi pasti (koordinat x/y) tampilan anaknya. Layout mutlak bersifat kurang fleksibel dan lebih sulit dikelola daripada tipe layout lainnya tanpa pemosisian mutlak.
- **FrameLayout:** Grup tampilan anak bertumpuk. FrameLayout didesain untuk memblokir area di layar guna menampilkan satu tampilan. Tampilan anak digambar bertumpuk, dengan anak yang baru saja ditambahkan di atas. Ukuran FrameLayout adalah ukuran tampilan anak terbesarnya.
- **GridLayout:** Grup yang menempatkan layar anaknya dalam kotak persegi panjang yang bisa digulir.



Tip: Ketahui selengkapnya tentang tipe layout yang berbeda di Objek Layout Umum.

Contoh sederhana dari layout dengan tampilan anak adalah aplikasi Hello Toast di salah satu pelajaran awal. Tampilan aplikasi Hello Toast muncul dalam gambar di bawah ini sebagai diagram file layout (`activity_main.xml`), bersama diagram hierarki (kanan atas) dan tangkapan layar layout yang benar-benar selesai (kanan bawah).



Dalam gambar di atas:

1. Layout akar `LinearLayout`, yang berisi semua tampilan anak, disetel ke orientasi vertikal.
2. `Button (button_toast)` tampilan anak. Sebagai tampilan anak pertama, muncul di bagian atas di layout linear.

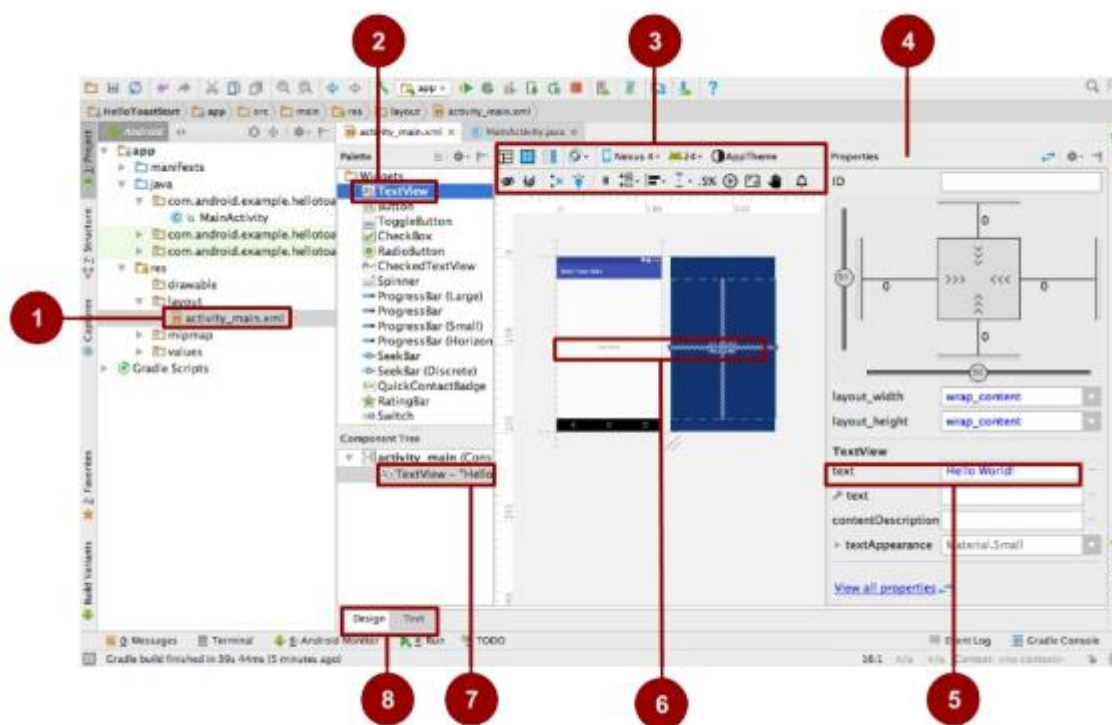
3. TextView (show_count) tampilan anak. Sebagai tampilan anak kedua, muncul di bawah tampilan anak pertama di layout linear.

4. Button (button_count) tampilan anak. Sebagai tampilan anak ketiga, muncul di bawah tampilan anak kedua di layout linear. Hierarki tampilan bisa tumbuh menjadi kompleks untuk aplikasi yang menampilkan banyak tampilan di layar. Penting untuk memahami hierarki tampilan, karena akan memengaruhi apakah tampilan terlihat dan apakah digambar secara efisien.

Menggunakan editor layout

Gunakan editor layout untuk mengedit file layout. Anda bisa menyeret dan melepas objek tampilan ke dalam panel grafik, serta menyusun, mengubah ukuran, dan menetapkan propertinya. Anda akan segera melihat efek perubahan yang dilakukan.

Untuk menggunakan editor layout, buka file layout XML. Editor layout muncul bersama tab Design di bagian bawah yang disorot. (Jika tab Text disorot dan Anda melihat kode XML, klik tab Design.) Untuk template Empty Activity, layout seperti yang ditampilkan dalam gambar di bawah ini.



Dalam gambar di atas:

1. File XML layout. File layout XML, biasanya diberi nama file `activity_main.xml`. Klik dua kali untuk membuka editor layout.

2. Palet elemen UI (tampilan). Panel Palette menyediakan daftar elemen UI dan layout. Tambahkan elemen atau layout ke UI dengan menyeretnya ke panel desain.

3. Bilah alat desain. Bilah alat panel desain menyediakan tombol untuk mengonfigurasi penampilan layout dalam panel desain dan untuk mengedit properti layout. Lihat gambar di bawah ini untuk detail.

Tip: Arahkan kursor ke atas setiap ikon untuk menampilkan keterangan alat yang merangkum fungsinya.

4. Panel Properties. Panel Properties menyediakan kontrol properti untuk tampilan yang dipilih.

5. Kontrol properti. Kontrol properti sesuai dengan atribut XML. Yang ditampilkan dalam gambar adalah properti Text dari TextView yang dipilih, yang disetel ke Hello World! .

6. Panel desain. Seret tampilan dari panel Palette ke panel desain untuk memosisikannya di layout.

7. Component Tree. Panel Component Tree menampilkan hierarki tampilan. Klik tampilan atau grup tampilan dalam panel ini untuk memilihnya. Gambar menampilkan TextView yang dipilih.

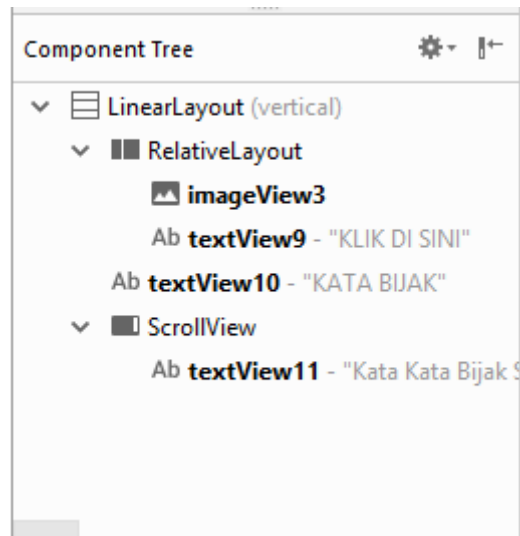
8. Tab Design dan Text. Klik Design untuk melihat editor layout, atau Text untuk melihat kode XML.

D. Praktikum

1. Buat layout android sehingga seperti gambar sebagai berikut:



2. Component tree untuk membuat User Interface ditunjukkan sebagai berikut



3. Desain Activity Main (activity_mail.xml) untuk membuat layout ditunjukkan sebagai berikut:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@android:color/holo_red_dark"
    android:orientation="vertical"
    android:paddingBottom="10dp"
    android:paddingLeft="10dp"
    android:paddingRight="10dp"
    android:paddingTop="10dp"
    android:weightSum="1"
    tools:context="id.ac.uns.simolek2.MainMolek">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="180dp"
        android:layout_marginBottom="10dp"
        android:layout_marginLeft="10dp"
        android:layout_marginRight="10dp"
        android:layout_marginTop="10dp"
        android:background="@android:color/holo_orange_dark"
        android:paddingBottom="10dp"
        android:paddingLeft="10dp"
        android:paddingRight="10dp"
        android:paddingTop="10dp">

        <ImageView
            android:id="@+id/imageView3"
            android:layout_width="300dp"
            android:layout_height="300dp"
            app:srcCompat="@drawable/photo" />

        <TextView
            android:id="@+id/textView9"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="KLIK DI SINI" />

        <TextView
            android:id="@+id/textView10"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="KATA BIJAK" />

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="10dp">

        <TextView
            android:id="@+id/textView11"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Kata Kata Bijak S" />

    </ScrollView>

</RelativeLayout>

</LinearLayout>
```

```

        android:layout_alignBottom="@+id/imageView3"
        android:layout_centerHorizontal="?attr/windowActionBar"
        android:background="@color/colorPrimary"
        android:text="KLIK DI SINI"
        android:textColor="@android:color/holo_green_light"
        android:textColorLink="@android:color/background_light"
        android:textSize="50sp" />

</RelativeLayout>

<TextView
    android:id="@+id/textView10"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="10dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:layout_marginTop="10dp"
    android:layout_weight="0.11"
    android:background="@android:color/holo_blue_light"
    android:fontFamily="casual"
    android:gravity="center"
    android:includeFontPadding="false"
    android:text="KATA BIJAK"
    android:textSize="32sp" />

<ScrollView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignEnd="@+id/textView9"
    android:layout_alignLeft="@+id/textView9"
    android:layout_alignParentTop="true"
    android:layout_alignRight="@+id/textView9"
    android:layout_alignStart="@+id/textView9">

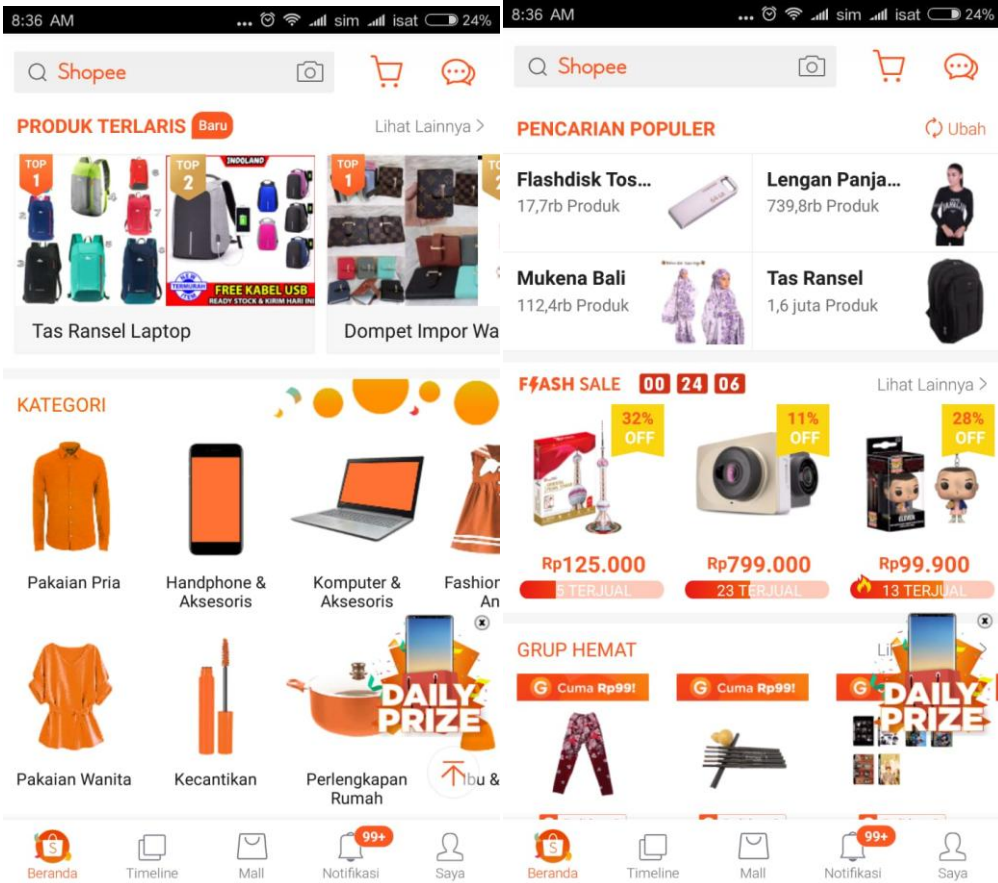
    <TextView
        android:id="@+id/textView11"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_marginBottom="10dp"
        android:layout_marginLeft="10dp"
        android:layout_marginRight="10dp"
        android:layout_marginTop="10dp"
        android:layout_weight="0.39"
        android:background="@android:color/holo_blue_bright"
        android:text="Kata Kata Bijak Singkat Padat Penuh Makna - Kata
        bijak singkat kali ini dibuat dengan jumlah karakter yang sangat terbatas.
        Namun anda jangan khawatir, meskipun singkat dan padat namun tetap memiliki
        makna yang sangat mendalam. Sehingga sangat cocok untuk dibagikan kembali
        atau dishare di berbagai sosial media." />
    </ScrollView>

</LinearLayout>

```

E. Latihan/Tugas

Buat layout untuk User Interface seperti pada gambar contoh (*shopee*), sehingga tampilan sebagai berikut



Materi 3 : Struktur Kontrol

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa menggunakan struktur kontrol yang terdapat dalam Android
2. Mahasiswa membuat aplikasi sederhana menggunakan struktur kontrol dalam Android.

C. Materi:

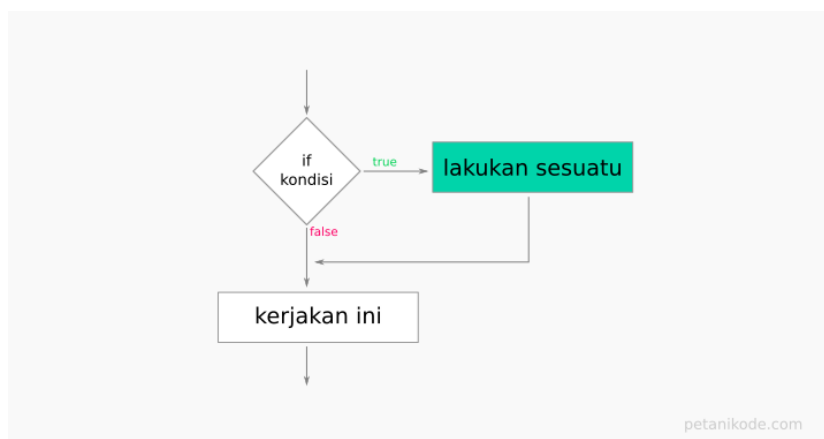
Percabangan hanyalah sebuah istilah yang digunakan untuk menyebut alur program yang bercabang. Percabangan juga dikenal dengan “Control Flow”, “Struktur Kondisi”, “Struktur IF”, “Decision”, dsb.

Sebelumnya, kamu perlu tahu dulu tiga bentuk percabangan pada Java:

- Percabangan IF
- Percabangan IF/ELSE
- Percabangan IF/ELSE/IF atau SWITCH/CASE

1. Percabangan IF

Percabangan ini hanya memiliki satu pilihan. Artinya, pilihan di dalam IF hanya akan dikerjakan kalau kondisinya benar.



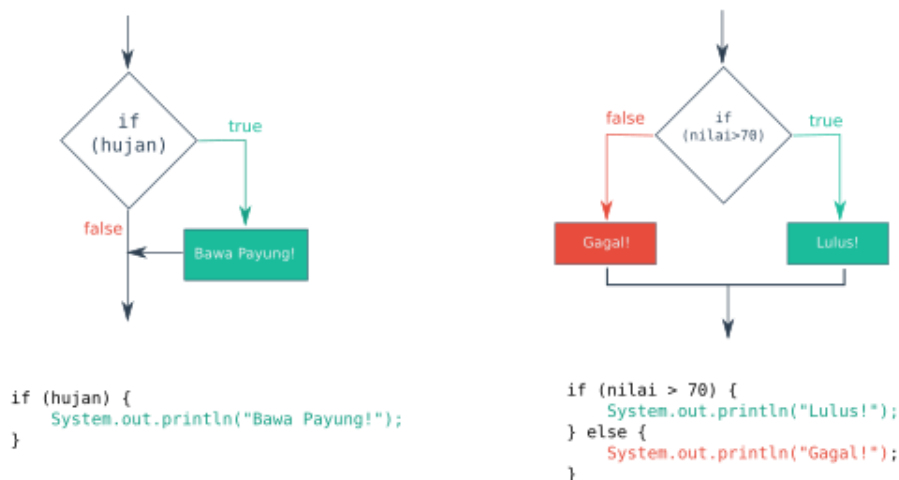
2. Percabangan IF/ELSE

Sedangkan percabangan IF/ELSE memiliki pilihan alternatif kalau kondisinya salah.

IF: “Jika kondisi benar maka kerjakan ini, kalau tidak silahkan lanjut”

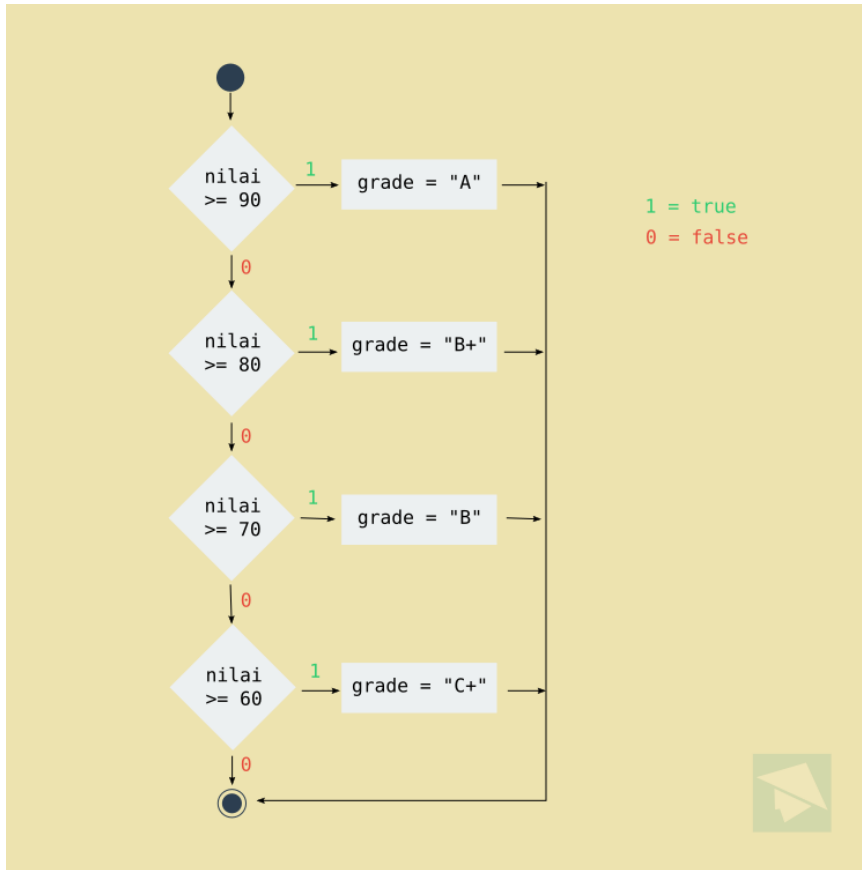
IF/ELSE: “Jika kondisi benar maka kerjakan ini, kalau salah maka kerjakan yang itu, setelah itu lanjut”

Percabangan IF/ELSE dengan Operator Ternary, selain menggunakan struktur seperti di atas, percabangan ini juga dapat menggunakan operator ternary.



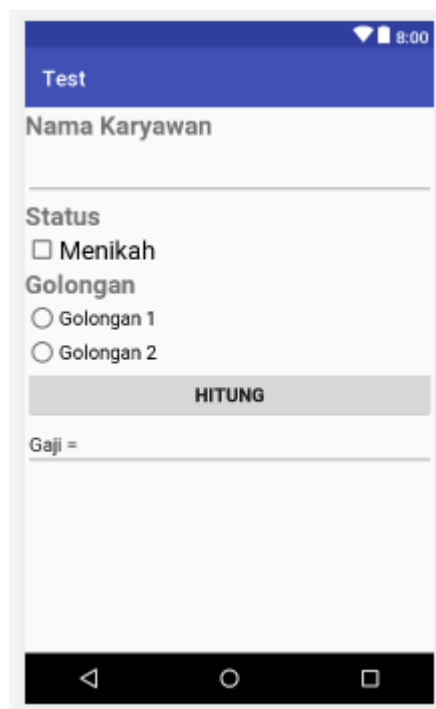
3. Percabangan IF/ELSE/IF dan SWITCH/CASE

Jika percabangan IF/ELSE hanya memiliki dua pilihan saja. Maka percabangan IF/ELSE/IF memiliki lebih dari dua pilihan. Percabangan SWITCH/CASE sebenarnya adalah bentuk lain dari IF/ELSE/IF. Bedanya, percabangan ini menggunakan kata kunci switch dan case. Formatnya juga berbeda, tapi cara kerjanya sama.

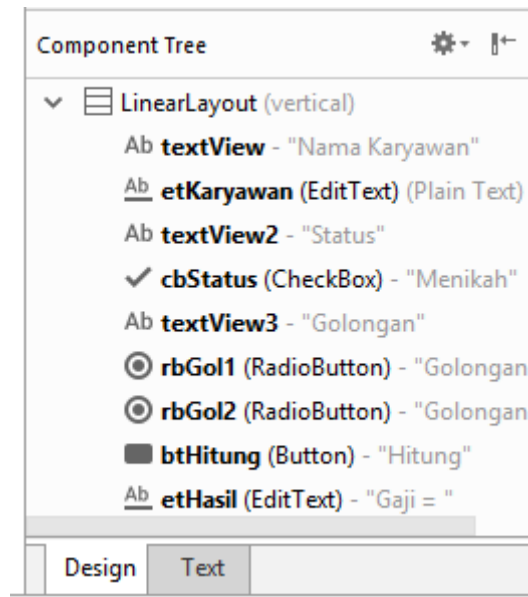


D. Praktikum

1. Buat user interface seperti berikut, atau file activity_main.xml



2. Component tree UI ditunjukkan sebagai berikut:



3. Buat Main Activity.xml sehingga sebagai berikut:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.d3ti.test.MainActivity"
    android:orientation="vertical">

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Nama Karyawan"
        android:textSize="24sp"
        android:textStyle="bold" />

    <EditText
        android:id="@+id/etKaryawan"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:textSize="24sp" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Status"
        android:textSize="24sp"
        android:textStyle="bold" />

    <CheckBox
        android:id="@+id/cbStatus"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Menikah"
        android:checked="true" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Golongan"
        android:textSize="24sp"
        android:textStyle="bold" />

    <RadioGroup
        android:id="@+id/rbGol"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:text="Golongan"
        android:textSize="24sp"
        android:textStyle="bold" />

    <RadioButton
        android:id="@+id/rbGol1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Golongan"
        android:textSize="24sp"
        android:textStyle="bold" />

    <RadioButton
        android:id="@+id/rbGol2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Golongan"
        android:textSize="24sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/btHitung"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hitung"
        android:textSize="24sp"
        android:textStyle="bold" />

    <EditText
        android:id="@+id/etHasil"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:textSize="24sp" />

</LinearLayout>
```

```

        android:textSize="24sp" />

<TextView
    android:id="@+id/textView3"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Golongan"
    android:textSize="24sp"
    android:textStyle="bold" />

<RadioButton
    android:id="@+id/rbGol1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Golongan 1"
    android:textSize="18sp" />

<RadioButton
    android:id="@+id/rbGol2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Golongan 2"
    android:textSize="18sp" />

<Button
    android:id="@+id/btHitung"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Hitung"
    android:textSize="18sp"
    android:textStyle="bold" />

<EditText
    android:id="@+id/etHasil"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="textPersonName"
    android:text="Gaji = " />

</LinearLayout>

```

4. Buat file activity_main.java sebagai berikut

```

package com.d3ti.test;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.EditText;
import android.widget.CheckBox;
import android.widget.RadioButton;

public class MainActivity extends AppCompatActivity {
    Button hitung;
    EditText nama, hasil;
    CheckBox cbstatus;
    RadioButton rb1, rb2;
}

```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    hitung = (Button) findViewById(R.id.btHitung);
    nama = (EditText) findViewById(R.id.etKaryawan);
    hasil = (EditText) findViewById(R.id.etHasil);
    cbstatus = (CheckBox) findViewById(R.id.cbStatus);
    rb1 = (RadioButton) findViewById(R.id.rbGol1);
    rb2 = (RadioButton) findViewById(R.id.rbGol2);

    hitung.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (cbstatus.isChecked() && rb1.isChecked()) {
                float duit = (float) ((0.2*1500000)+1500000);
                hasil.setText("Gaji " + nama.getText() + " " + duit);
            } else if (cbstatus.isChecked() && rb2.isChecked()) {
                float duit = (float) ((0.2*2000000)+2000000);
                hasil.setText("Gaji " + nama.getText() + " " + duit);
            } else if (rb1.isChecked()) {
                hasil.setText("Gaji " + nama.getText() + " Rp.
1.500.000,-");
            } else if (rb2.isChecked()) {
                hasil.setText("Gaji " + nama.getText() + " Rp.
2.000.000,-");
            }
        }
    });
}
}

```

E. Latihan/Tugas

Buat activity yang digunakan untuk membuat invoice barang, dengan data dinamis dari penghitungan pada penjualan barang sedangkan data statisnya berupa label:

PT. SAMPLE SARANA ABADI Ruiko Graha Arteri Mas Jl. Panjang Blok 101 No.1, Jakarta 12223 Phone : (62-21) 58305578 (Hunting) Fax : (62-21) 58453581		INVOICE	Number : 00000145 Inv. Date : 11 February 2013 Payment Term : Cash/Tunai Due Date : 11 February 2013 Salesman : DIMAS TYO Currency : SGD			
Customer : SAMPLE INDONESIA, PT. Phone : 021-3985656 Fax : 35425521						
No.	Product Description	Quantity UOM	Unit Price	Gross Amt.	Discount Detail %	Net Amount
1	GOLD RING, (749X.133)	6 Pcs	19.04	114.24	0%	114.24
2	RING	6 Pcs	325.86	1,955.16	0%	1,955.16
3	VALVE	1 Pcs	672.08	672.08	0%	672.08
4	BALL	12 Pcs	37.72	452.64	0%	452.64
Inword : Tiga Ribu Lima Ratus Tiga Belas Singapore Dollar Koma Lima Ratus Tiga Puluh Dua				Sincerely, Yuda Haryanto	Gross Total 3,194.12 Discount Total 0.00 Down Payment 3,513.53 Tax 319.41 Freight Charge 0.00 Net Total 3,513.53	
Remark : TRANSFER VIA BCA-IBR A/C : 164-800-3321 A/N : PT. SAMPLE SARANA ABADI						

PEMBAYARAN DENGAN CHEQUE/BO DIANGGAP LUNAS, APABILA SUDAH DAPAT DIJANGKAN

Sumber gambar : katakatabajak.co.id

Materi 4 : Perulangan

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

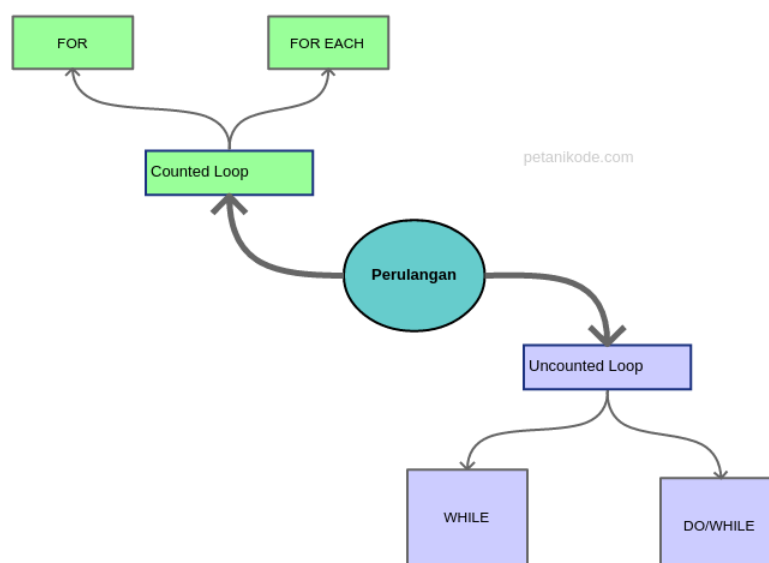
1. Mahasiswa menggunakan perulangan yang terdapat dalam Android
2. Mahasiswa dapat membuat aplikasi sederhana menggunakan perulangan pada Android.

C. Materi:

Perulangan dalam pemrograman dibagi menjadi dua jenis:

- Counted loop: Perulangan yang jumlah pengulangannya terhitung atau tentu.
- Uncounted loop: Perulangan yang jumlah pengulangannya tidak terhitung atau tidak tentu.

Counted loop terdiri dari perulangan For dan For each. Sedangkan Uncounted loop terdiri dari perulangan While dan Do/While.



Counted Loop

Seperti yang sudah dijelaskan, perulangan ini memiliki jumlah pengulangan yang tentu dan terhitung. Perulangan ini terdiri dari perulangan For dan For each.

Perulangan For

Format penulisan perulangan For di java adalah sebagai berikut:

```
for( int hitungan = 0; hitungan <= 10; hitungan++ ){  
    // blok kode yang akan diulang  
}
```

Uncounted Loop

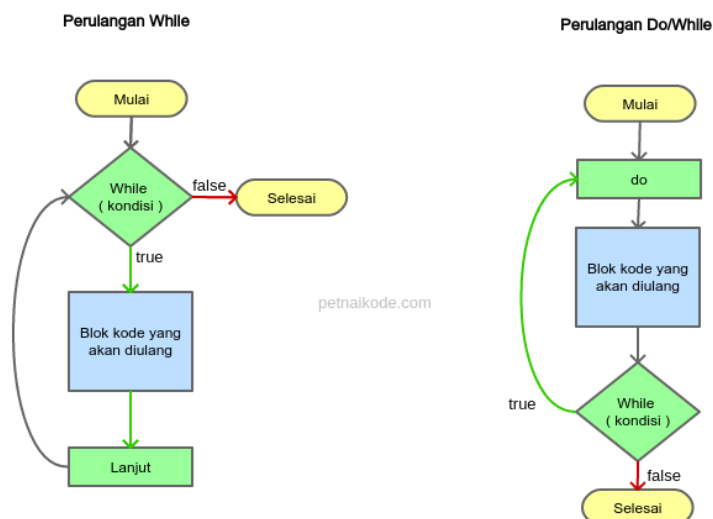
Seperti yang sudah dijelaskan di awal tadi, perulangan ini tidak jelas jumlah pengulangannya. Tapi, tidak menutup kemungkinan juga, jumlah pengulangannya dapat ditentukan. Perulangan uncounted loop terdiri dari perulangan While dan Do/While. Flow Chart Perulangan While dan Do/While

Perulangan While

While bisa kita artikan selama. Cara kerja perulangan ini seperti percabangan, ia akan melakukan perulangan selama kondisinya bernilai true.

Struktur penulisan perulangan while:

```
while ( kondisi ) {  
    // blok kode yang akan diulang  
}
```



Perulangan Do/While

Cara kerja perulangan Do/While sebenarnya sama seperti perulangan While. Bedanya, Do/While melakukan satu kali perulangan dulu. Kemudian mengecek kondisinya.

Struktur penulisannya seperti ini:

```
do {  
    // blok kode yang akan diulang  
} while (kondisi);
```

Jadi kerjakan dulu (Do), baru di cek kondisinya while(kondisi). Kalau kondisi bernilai true, maka lanjutkan perulangan.

D. Praktikum

1. Buat file activity_main.xml sebagai berikut

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:tools="http://schemas.android.com/tools"  
    android:id="@+id/activity_main"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:gravity="center"  
    tools:context="com.anonimeact.looperhandler.MainActivity">  
  
    <TextView  
        android:id="@+id/tv_count"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:textSize="24sp"  
        android:gravity="center"  
        android:text="Perulangan ke : " />  
</LinearLayout>
```

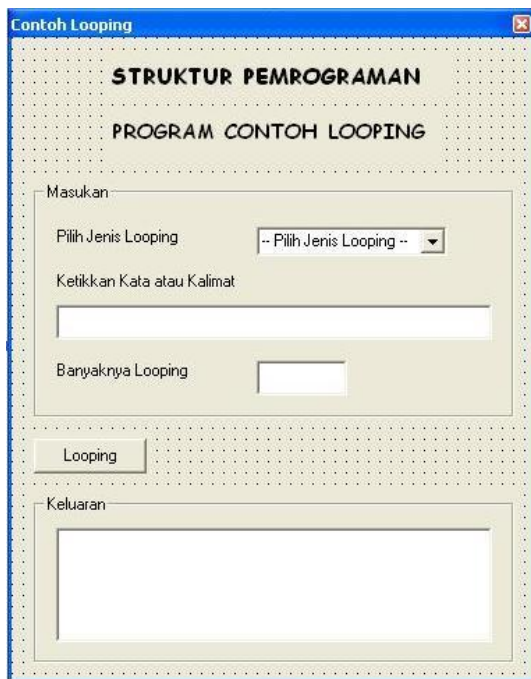
2. Buat file MainActivity.java sebagai berikut

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:tools="http://schemas.android.com/tools"  
    android:id="@+id/activity_main"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:gravity="center"  
    tools:context="com.anonimeact.looperhandler.MainActivity">  
  
    <TextView  
        android:id="@+id/tv_count"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"
```

```
        android:textSize="24sp"  
        android:gravity="center"  
        android:text="Perulangan ke : " />  
</LinearLayout>
```

E. Latihan/Tugas

Buatlah activity perulangan seperti pada gambar berikut



Materi 5 : Informasi peringatan untuk pengembangan

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa dapat menggunakan perangkat-perangkat untuk memberitahukan kesalahan pemrograman.
2. Mahasiswa dapat membuat aplikasi sederhana dengan mengeluarkan informasi-informasi kesalahan.

C. Materi:

Errors dan Exceptions adalah ketentuan yang mungkin terjadi hampir di setiap kode. Menyadari bagaimana menangani mereka memainkan bagian penting dari seberapa efisien aplikasi nantinya. Metode paling umum yang digunakan di Android Studio adalah *try-catch*. Serta dalam pemrograman java, yang terdiri dari membuat node "*try*" untuk memasukkan rutin yang mungkin masuk ke dalam kesalahan. Kemudian, dengan "*catch*" memberi tahu aplikasi apa yang harus dilakukan ketika menangkap pengecualian tertentu.

Dengan struktur ini, kode akan mencapai "*catch*" segera setelah ada pengecualian yang tertangkap. Jika itu terjadi dalam berikutnya akan diabaikan dan kode akan mengeksekusi "*catch*". Kemudian dapat mencetak pesan pengecualian, atau mencetak jejak stack, atau bahkan menentukan rutin alternatif jika diperlukan.

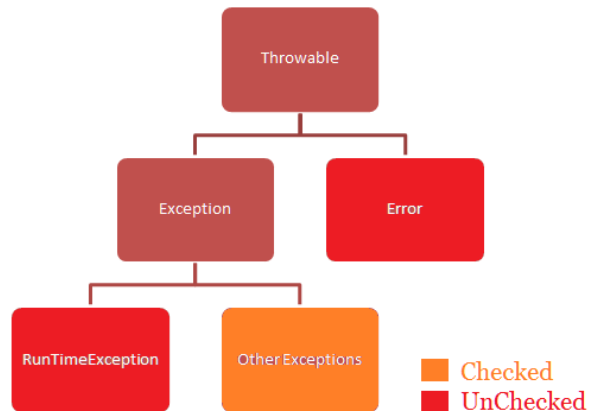
Penting untuk memperhatikan bahwa beberapa kesalahan dapat diharapkan, misalnya ketika membaca input pengguna dan menghitung data menjadi variabel. Beberapa kesalahan mungkin muncul yang dapat diprediksi, seperti:

- Array out of index exception
- Input is not in a valid format

Namun, beberapa kesalahan mungkin muncul dari logika atau struktur kode itu sendiri. Itu bisa lebih sulit diprediksi, tetapi harus selalu diperhitungkan saat melakukan kode Anda. Tidak ada yang ingin pengguna akhir mendapatkan kesalahan seperti:

- Stack overflow

- NullerPointerException



D. Praktikum

```

public void ExampleMethod() throws IOException
{
    if (conditionForError == true)
    {
        throw new IOException("Custom exception msg");
    }

    try
    {
        // routine 1
        // routine 2
        // routine 3
    }
    catch (ExceptionRoutine1 e)
    {
        // That's good for diagnosing the problem
        System.out.println("Thrown exception: " + e.getMessage());
    }
    catch (ExceptionRoutine2 e)
    {
        // custom exception
        throw new Exception("Error in Routine 2", e);
    }
    catch (Exception e)
    {
        // General error can be anything*
        // captured by the java class Exception
        // print in the console detailed technical info
        e.printStackTrace();
    }
}
  
```

E. Latihan/Tugas

Materi 6 : Array dan ListView

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa menggunakan Array dan menampilkannya menggunakan listView.
2. Mahasiswa dapat membuat aplikasi sederhana dengan menggunakan Array dan menampilkannya dalam listView.

C. Materi:

ListView adalah kelompok tampilan yang menampilkan daftar item yang dapat digulir. Item daftar secara otomatis dimasukkan ke daftar dengan menggunakan Adapter yang menarik materi dari sumber seperti larik atau kueri database dan mengonversi setiap hasil item menjadi sebuah tampilan yang dimasukkan ke dalam daftar.



Menggunakan Loader

Penggunaan CursorLoader adalah cara standar untuk kueri Cursor sebagai tugas asinkron agar dapat menghindari pemblokiran thread utama aplikasi Anda dengan kueri. Bila CursorLoader menerima hasil Cursor, LoaderCallbacks akan menerima callback ke `onLoadFinished()`, yang merupakan tempat Anda memperbarui Adapter dengan Cursor baru dan tampilan daftar kemudian menampilkan hasilnya.

Walaupun CursorLoader API pertama kali diperkenalkan dalam Android 3.0 (API level 11), API ini juga tersedia di Pustaka Dukungan sehingga aplikasi Anda dapat menggunakannya saat mendukung perangkat yang menjalankan Android 1.6 atau yang lebih tinggi.

D. Praktikum

1. Buat UI sehingga file layout main_activity.xml sebagai berikut:

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/rl"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10dp"
    tools:context=".MainActivity"
    android:background="#e8d25b"
    >
    <TextView
        android:id="@+id/tv"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textStyle="italic"
        android:textColor="#5f65ff"
        android:background="#d5daff"
        android:padding="5dp"
        android:text="Choose your most favorite."
        />
    <ListView
        android:id="@+id/lv"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/tv"
        />
</RelativeLayout>
```

2. Buat file ActivityMain.java

```
package com.d3ti.safiie99.mytestapplication;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    @Override
```

```

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    ListView lv = (ListView) findViewById(R.id.lv);
    final TextView tv = (TextView) findViewById(R.id.tv);

    String[] fruits = new String[] {
        "Japanese Persimmon",
        "Kakadu lime",
        "Illawarra Plum",
        "Malay Apple ",
        "Mamoncillo",
        "Persian lime"
    };

    // Create a List from String Array elements
    List<String> fruits_list = new
ArrayList<String>(Arrays.asList(fruits));

    // Create a ArrayAdapter from List
    ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>
(this, android.R.layout.simple_list_item_1, fruits_list);

    // Populate ListView with items from ArrayAdapter
    lv.setAdapter(arrayAdapter);
    lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int
position, long id) {
            String selectedItem = (String)
parent.getItemAtPosition(position);

            // Display the selected item text on TextView
            tv.setText("Your favorite : " + selectedItem);
        }
    });
}
}


```


E. Latihan/Tugas


Buat activity sebagai berikut:


6:31 PM 0.05K/s 100%


List view


 **Apple**
The apple tree is a deciduous tree in the rose family best known for its sweet, pomaceous fruit, the apple.

 **Banana**
The banana is an edible fruit – botanically a berry – produced by several kinds of large herbaceous flowering plants in the genus *Musa*.

 **Lemon**
The lemon, *Citrus limon* Osbeck, is a species of small evergreen tree in the flowering plant family Rutaceae, native to Asia.

 **Cherry**
A cherry is the fruit of many plants of the genus *Prunus*, and is a fleshy drupe.

 **Strawberry**
The garden strawberry is a widely grown hybrid species of the genus *Fragaria*, collectively known as the strawberries.

 **Avocado**
The avocado is a tree, long thought to have originated in South Central Mexico, classified as a member of the flowering plant family Lauraceae.

Materi 7 : ArrayList dan Recyclerview

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa menggunakan ArrayList dan menampilkannya ke dalam Recyclerview
2. Mahasiswa dapat membuat aplikasi sederhana menggunakan ArrayList dan menampilkannya ke dalam Recyclerview.

C. Materi:

Kelas RecyclerView adalah versi ListView yang lebih canggih dan fleksibel. Widget ini adalah kontainer untuk menampilkan rangkaian data besar yang bisa digulir secara sangat efisien dengan mempertahankan tampilan dalam jumlah terbatas. Gunakan widget RecyclerView bila Anda perlu menampilkan banyak data yang bisa digulir, atau kumpulan data dengan elemen yang berubah pada waktu proses berdasarkan aksi pengguna atau kejadian jaringan.

Komponen RecyclerView

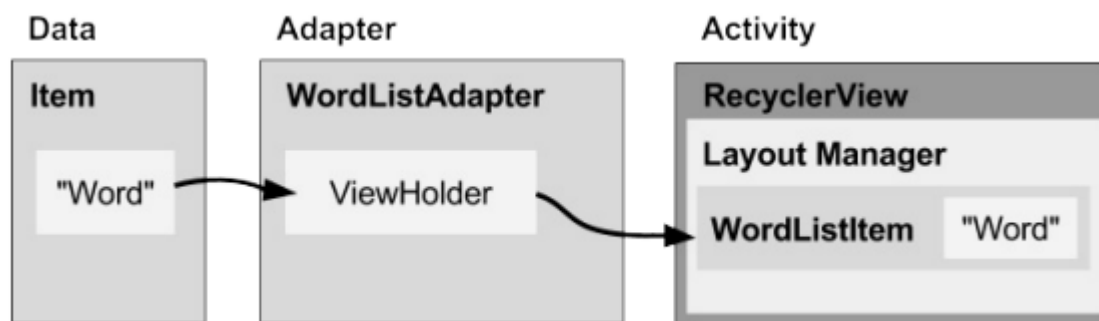
Untuk menampilkan data dalam RecyclerView, memerlukan bagian berikut:

- Data. Tidak penting dari mana asal data. Anda bisa membuat data secara lokal, seperti yang Anda lakukan dalam latihan, mendapatkannya dari database perangkat seperti yang akan Anda lakukan dalam praktik nanti, atau menariknya dari awan.
- RecyclerView. Daftar gulir yang berisi item daftar.
- Instance RecyclerView sebagaimana didefinisikan dalam file layout aktivitas Anda akan bertindak sebagai kontainer tampilan.
- Layout untuk satu item data. Semua item daftar tampak sama, sehingga Anda bisa menggunakan layout yang sama
 - untuk semuanya. Layout item harus dibuat secara terpisah dari layout aktivitas, sehingga satu per satu tampilan item bisa dibuat dan diisi data.
- Pengelola layout. Pengelola layout menangani penyusunan (layout) komponen antarmuka pengguna dalam suatu tampilan. Semua grup tampilan memiliki pengelola layout. Untuk LinearLayout, sistem Android menangani layout untuk

Anda. RecyclerView memerlukan pengelola layout eksplisit untuk mengelola susunan item daftar yang terdapat di dalamnya. Layout ini bisa vertikal, horizontal, atau berupa petak.

- Pengelola layout adalah instance dari RecyclerView.LayoutManager untuk menyusun layout item dalam RecyclerView
- Adapter. Adapter menghubungkan data Anda dengan RecyclerView. Adapter menyiapkan data dan cara menampilkan data dalam view holder. Bila data berubah, adapter akan memperbarui materi tampilan item daftar terkait dalam RecyclerView. Adapter juga merupakan ekstensi dari RecyclerView.Adapter. Adapter menggunakan ViewHolder untuk menampung tampilan yang menyusun setiap item dalam RecyclerView, dan mengikat data untuk ditampilkan dalam tampilan yang menampilkannya.
- View holder. View holder memperluas kelas ViewHolder. View holder berisi tampilan informasi untuk menampilkan satu item dari layout item. View holder digunakan oleh adapter untuk menyediakan data, yang merupakan ekstensi dari RecyclerView.ViewHolder

Diagram di bawah ini menampilkan hubungan antara komponen-komponen ini



Data

Semua data yang bisa ditampilkan akan ditampilkan dalam RecyclerView.

- Teks
- Gambar
- Ikon

Data bisa berasal dari sumber mana pun.

- Dibuat oleh aplikasi. Misalnya, kata acak untuk permainan.
- Dari database lokal. Misalnya, daftar kontak.
- Dari storage awan atau internet. Misalnya judul berita.

RecyclerView

RecyclerView adalah:

- Suatu grup Tampilan untuk kontainer yang bisa digulir

- Ideal untuk daftar item serupa yang panjang
- Hanya menggunakan tampilan dalam jumlah terbatas yang digunakan kembali saat tampilan tersebut tidak tampak di layar. Hal ini menghemat memori dan mempercepat pembaruan item daftar saat pengguna menggulir data, karena tidak perlu membuat tampilan baru untuk setiap item yang muncul.
- Secara umum, RecyclerView menyimpan sebanyak mungkin tampilan item yang muat di layar, plus sedikit tambahan pada setiap akhir daftar untuk memastikan pengguliran berjalan cepat dan lancar.

Layout Item

Layout adalah sebuah item daftar yang disimpan dalam file terpisah sehingga adapter bisa membuat tampilan item dan mengedit materinya secara independen dari layout aktivitas.

Pengelola Layout

Pengelola layout memosisikan tampilan item di dalam grup tampilan, seperti RecyclerView dan menentukan kapan harus menggunakan kembali tampilan item yang tidak lagi terlihat oleh pengguna. Untuk menggunakan kembali (atau mendaur ulang) tampilan, pengelola layout bisa meminta adapter untuk mengganti materi tampilan dengan elemen lain dari kumpulan data. Mendaur ulang tampilan dengan cara ini akan meningkatkan kinerja karena menghindari pembuatan tampilan yang tidak diperlukan atau melakukan pencarian `findViewById()` yang mahal.

- RecyclerView menyediakan semua pengelola layout bawaan ini:
- `LinearLayoutManager` menampilkan item dalam daftar gulir vertikal atau horizontal. `GridLayoutManager` menampilkan item dalam petak.
- `StaggeredGridLayoutManager` menampilkan item dalam petak zigzag. Untuk membuat pengelola layout khusus, perluas kelas `RecyclerView.LayoutManager`.

Animasi

Animasi untuk menambahkan dan menghapus item diaktifkan secara default dalam RecyclerView. Untuk menyesuaikan

animasi ini, perluas kelas `RecyclerView.ItemAnimator` dan gunakan metode `RecyclerView.setItemAnimator()`.

Adapter

Adapter membantu dua antarmuka yang tidak kompatibel untuk bekerja bersama. Dalam RecyclerView, adapter menghubungkan data dengan tampilan. Adapter bertindak sebagai perantara antara data dan tampilan. Adapter menerima atau mengambil data, melakukan semua pekerjaan yang diperlukan agar bisa ditampilkan dalam suatu tampilan, dan menempatkan data dalam tampilan. Misalnya, adapter bisa menerima data dari database sebagai objek `Cursor`, mengekstrak kata dan definisinya, mengonversinya menjadi string, dan menempatkan string dalam suatu

tampilan item yang memiliki tampilan teks, satu untuk kata dan satu untuk definisi. Anda akan mengetahui selengkapnya tentang cursor dalam bab berikutnya.

RecyclerView.Adapter mengimplementasikan sebuah view holder, dan harus mengganti callback berikut:

onCreateViewHolder() memekarkan tampilan item dan mengembalikan view holder baru yang memuatnya. Metode ini dipanggil bila RecyclerView memerlukan view holder baru untuk menyatakan suatu item.

onBindViewHolder() menyetel materi item pada posisi yang ditentukan dalam RecyclerView. Proses ini dipanggil oleh RecyclerView, misalnya, bila sebuah item baru bergulir ke tampilan.

View holder

RecyclerView.ViewHolder menjelaskan tampilan data dan metadata tentang tempatnya dalam RecyclerView. Setiap view holder menampung satu rangkaian data. Adapter menambahkan data ke view holder untuk ditampilkan oleh pengelola layout.

Definisikan layout view holder Anda dalam file sumber daya XML. Layout ini bisa berisi (hampir) semua tipe tampilan, termasuk elemen yang bisa diklik.

Mengimplementasikan RecyclerView

Mengimplementasikan RecyclerView memerlukan langkah-langkah berikut:

1. Tambahkan dependensi RecyclerView ke file app/build.gradle aplikasi.
2. Tambahkan RecyclerView ke layout aktivitas
3. Buat file XML layout untuk satu item
4. Perluas RecyclerView.Adapter dan implementasikan metode onCreateViewHolder serta onBindViewHolder.
5. Perluas RecyclerView.ViewHolder untuk membuat view holder bagi layout item Anda. Anda bisa menambahkan perilaku klik dengan mengganti metode onClick.
6. Dalam aktivitas Anda, dalam metode onCreate method, buat RecyclerView dan inialisasi dengan adapter serta pengelola layout.

D. Praktikum

1. Buat file layout sebagai berikut

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <android.support.v7.widget.RecyclerView
        android:id="@+id/rv_movielist"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</RelativeLayout>

```

2. Buat file item_row_film.xml

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="16dp"
    android:orientation="horizontal">
    <de.hdodenhof.circleimageview.CircleImageView
        android:id="@+id/img_gambar"
        android:layout_width="55dp"
        android:layout_height="55dp"
        android:layout_marginRight="16dp"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@+id/img_gambar"
        android:layout_centerVertical="true"
        android:orientation="vertical">
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/tv_item_judul"
            android:text="Name"
            android:layout_marginBottom="8dp"/>
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/tv_item_tglrilis"
            android:text="Deskripsi"/>
    </LinearLayout>
</RelativeLayout>

```

3. Buat file FilmModel.java

```

package com.d3ti.safiie99.a03_catalogue_movie;

```

```

public class FilmModel {
    private String judul, tgl_rilis, gambar;

    /* public FilmModel(String judul, String tgl_rilis, String gambar) {
        this.judul = judul;
        this.tgl_rilis = tgl_rilis;
        this.gambar = gambar;
    } */

    public String getJudul() {
        return judul;
    }

    public void setJudul(String judul) {
        this.judul = judul;
    }

    public String getTgl_rilis() {
        return tgl_rilis;
    }

    public void setTgl_rilis(String tgl_rilis) {
        this.tgl_rilis = tgl_rilis;
    }

    public String getGambar() {
        return gambar;
    }

    public void setGambar(String gambar) {
        this.gambar = gambar;
    }
}

```

4. Buat file FilmData.Java

```

package com.d3ti.safiie99.a03_catalogue_movie;

import java.util.ArrayList;

public class FilmData {
    public static String[][] data = new String[][] {
        {"Avenger", "Avenger Age of
Ultron", "https://upload.wikimedia.org/wikipedia/en/f/f9/TheAvengers2012Post
er.jpg"},
        {"Captain America", "The
Fury", "https://upload.wikimedia.org/wikipedia/en/f/f9/TheAvengers2012Poster
.jpg"},
        {"Thor", "Palu
Godam", "https://upload.wikimedia.org/wikipedia/en/f/f9/TheAvengers2012Poste
r.jpg"},
        {"Aquaman", "Aqua
Galon", "https://upload.wikimedia.org/wikipedia/en/f/f9/TheAvengers2012Poste
r.jpg"},
        {"Wonder Woman", "Lady's
Kgniht", "https://upload.wikimedia.org/wikipedia/en/f/f9/TheAvengers2012Post
er.jpg"}
    };
}

```

```

public static ArrayList<FilmModel> getListFilm() {
    FilmModel fd = null;
    ArrayList<FilmModel> listFilm = new ArrayList<>();
    for (int i=0; i<data.length; i++) {
        fd = new FilmModel();
        fd.setJudul(data[i][0]);
        fd.setTgl_rilis(data[i][1]);
        fd.setGambar(data[i][2]);

        listFilm.add(fd);
    }

    return listFilm;
}
} //Class Film Data

```

5. Buat file ListFilmAdapter.java

```

package com.d3ti.safiie99.a03_catalogue_movie;

import android.content.Context;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import com.bumptech.glide.Glide;

import java.util.ArrayList;

public class ListFilmAdapter extends
RecyclerView.Adapter<ListFilmAdapter.ListViewHolder> {
    private Context context;

    public ArrayList<FilmModel> getListFilmModel() {
        return listFilmModel;
    }

    public void setListFilmModel(ArrayList<FilmModel> listFilmModel) {
        this.listFilmModel=listFilmModel;
    }

    private ArrayList<FilmModel> listFilmModel;

    public ListFilmAdapter(Context context) {
        this.context = context;
    }

    @Override
    public ListFilmAdapter.ListViewHolder onCreateViewHolder(ViewGroup
parent, int viewType) {
        View itemRow =
LayoutInflater.from(parent.getContext()).inflate(R.layout.item_row_film,
parent, false);

```

```

        return new ListViewHolder(itemRow);
    }

    @Override
    public void onBindViewHolder(ListFilmAdapter.ListViewHolder holder, int
position) {
holder.tvJudul.setText(getListFilmModel().get(position).getJudul());
holder.tvTglRilis.setText(getListFilmModel().get(position).getTgl_rilis());

Glide.with(context).load(getListFilmModel().get(position).getGambar())
        .override(55,55)
        .crossFade()
        .into(holder.imgGambar);
    }

    @Override
    public int getItemCount() {
        return getListFilmModel().size();
    }

    class ListViewHolder extends RecyclerView.ViewHolder {
        TextView tvJudul;
        TextView tvTglRilis;
        ImageView imgGambar;

        public ListViewHolder(View itemView) {
            super(itemView);
            tvJudul = (TextView) itemView.findViewById(R.id.tv_item_judul);
            tvTglRilis = (TextView)
itemView.findViewById(R.id.tv_item_tgl_rilis);
            imgGambar = (ImageView) itemView.findViewById(R.id.img_gambar);
        }
    }
}

```

6. Buat file MainActivity.java

```

package com.d3ti.safiie99.a03_catalogue_movie;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {
    private RecyclerView rvFilm;
    private ArrayList<FilmModel> listFilm;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}

```

```
rvFilm = (RecyclerView) findViewById(R.id.rv_ListFilm);
rvFilm.setHasFixedSize(true);

listFilm = new ArrayList<>();
listFilm.addAll(FilmData.getListFilm());

showRecyclerView();
}

private void showRecyclerView() {
    rvFilm.setLayoutManager(new LinearLayoutManager(this));
    ListFilmAdapter listFilmAdapter = new ListFilmAdapter(this);
    listFilmAdapter.setListFilmModel(listFilm);
    rvFilm.setAdapter(listFilmAdapter);

}

}
```

E. Latihan/Tugas

Selesaikan praktikum diatas

Materi 8 : Broadcast Receivers dan Service

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa menggunakan broadcast receivers dan service untuk menangkap sinyal Sistem Android.
2. Mahasiswa dapat membuat aplikasi sederhana menggunakan broadcast receivers dan service untuk menangkap sinyal Sistem Android.

C. Materi:

Intent eksplisit digunakan untuk memulai aktivitas tertentu yang benar-benar memenuhi syarat, serta untuk meneruskan informasi di antara aktivitas di aplikasi Anda. Intent implisit digunakan untuk memulai aktivitas berdasarkan komponen terdaftar yang diketahui sistem, misalnya fungsionalitas umum.

Dalam pelajaran ini Anda akan mempelajari tentang Intent broadcast, yang tidak memulai aktivitas namun dikirimkan ke penerima broadcast.

Intent broadcast

Intent yang telah Anda lihat sejauh ini selalu menghasilkan aktivitas yang akan diluncurkan, baik aktivitas tertentu dari aplikasi Anda maupun aktivitas dari aplikasi lain yang bisa memenuhi aksi yang diminta. Namun kadang-kadang Intent tidak memiliki penerima tertentu, dan kadang-kadang Anda tidak ingin aktivitas diluncurkan sebagai respons terhadap Intent. Misalnya, bila aplikasi Anda menerima Intent sistem yang menunjukkan bahwa keadaan jaringan di perangkat sudah berubah, Anda mungkin tidak ingin meluncurkan aktivitas, melainkan mungkin ingin menonaktifkan beberapa fungsionalitas aplikasi.

Karena alasan ini, ada tipe Intent ketiga yang bisa dikirimkan ke aplikasi yang tertarik: Intent broadcast. Meskipun Intent broadcast didefinisikan dengan cara yang sama seperti Intent implisit, setiap tipe Intent memiliki karakteristik pembeda yang penting:

- Intent broadcast dikirim menggunakan `sendBroadcast()` atau metode terkait, sementara tipe Intent lainnya menggunakan `startActivity()` untuk memulai aktivitas. Bila menyiarkan Intent, Anda tidak akan pernah menemukan atau

memulai aktivitas. Demikian juga, tidak ada cara bagi penerima broadcast untuk melihat atau menangkap Intent yang digunakan bersama `startActivity()` .

- Intent broadcast adalah operasi latar belakang yang biasanya tidak diketahui pengguna. Memulai aktivitas bersama Intent, di sisi lain, adalah operasi latar depan yang memodifikasi apa yang sedang berinteraksi dengan pengguna.

Ada dua tipe Intent broadcast, yang dikirimkan oleh sistem (Intent broadcast sistem), dan yang dikirimkan oleh aplikasi Anda (Intent broadcast khusus).

Intent broadcast sistem

Sistem mengirimkan Intent broadcast sistem bila ada kejadian sistem yang mungkin menarik aplikasi. Misalnya:

- Bila perangkat sedang booting, sistem akan mengirim sebuah Intent broadcast sistem `ACTION_BOOT_COMPLETED` . Intent ini berisi nilai konstanta `"android.intent.action.BOOT_COMPLETED"` .
- Bila perangkat terhubung ke daya eksternal, sistem akan mengirim `ACTION_POWER_CONNECTED` , yang berisi nilai konstanta `"android.intent.action.ACTION_POWER_CONNECTED"` . Bila perangkat terlepas dari daya eksternal, sistem akan mengirim `ACTION_POWER_DISCONNECTED` .
- Bila memori perangkat minim, sistem akan mengirim `ACTION_DEVICE_STORAGE_LOW` . Intent ini berisi nilai konstanta `"android.intent.action.DEVICE_STORAGE_LOW"` .

`ACTION_DEVICE_STORAGE_LOW` merupakan broadcast melekat, Intentnya adalah nilai broadcast bertahan di cache. Jika Anda perlu mengetahui apakah penerima broadcast memproses nilai yang ada di cache (melekat) atau nilai yang disiarkan pada saat ini, gunakan `isInitialStickyBroadcast()` .

Untuk informasi selengkapnya tentang broadcast sistem umum, kunjungi referensi Intent . Untuk menerima Intent broadcast sistem, Anda perlu membuat penerima broadcast.

Intent broadcast khusus

Intent broadcast khusus adalah Intent broadcast yang dikirim aplikasi Anda. Gunakan Intent broadcast khusus bila Anda ingin agar aplikasi mengambil suatu aksi tanpa meluncurkan aktivitas, misalnya bila Anda ingin membiarkan aplikasi lainnya mengetahui bahwa data telah diunduh ke perangkat dan tersedia untuk digunakan.

Untuk membuat Intent broadcast khusus, buat aksi Intent khusus. Untuk mengirim broadcast khusus ke aplikasi lainnya, teruskan Intent ke `sendBroadcast()` , `sendOrderedBroadcast()` , atau `sendStickyBroadcast()` . (Untuk detail tentang metode ini, lihat dokumentasi referensi Context .)

Misalnya, metode berikut membuat sebuah Intent dan menyiarkannya ke semua [penerima broadcast] yang tertarik(#broadcast_receivers):

```
public void sendBroadcastIntent( ) {  
Intent intent = new Intent( ) ;  
intent.setAction( "com. example. myproj ect. ACTION_SHOW_TOAST" ) ;  
sendBroadcast( intent ) ;  
}
```

Catatan: Bila Anda menetapkan aksi untuk Intent, gunakan nama paket unik (com. example. myproj ect dalam contoh) untuk memastikan Intent tersebut tidak konflik dengan Intent yang disiarkan dari aplikasi lain atau dari sistem.

Broadcast Listener

Intent broadcast tidak ditargetkan pada penerima tertentu. Sebagai gantinya, aplikasi yang tertarik mendaftarkan komponen untuk "mendengarkan" jenis Intent ini. Komponen pendengar ini disebut penerima broadcast. Gunakan penerima broadcast untuk merespons pesan yang disiarkan dari aplikasi lain atau dari sistem. Untuk membuat penerima broadcast:

1. Definisikan sub kelas dari kelas BroadcastReceiver dan implementasikan metode onReceive() .
2. Daftarkan penerima broadcast secara dinamis di Java, atau secara statis di file manifest aplikasi Anda.

Sebagai bagian dari langkah ini, gunakan filter Intent untuk menetapkan jenis Intent broadcast yang ingin Anda terima.

Service

Service adalah komponen aplikasi yang menjalankan operasi yang berjalan lama, biasanya di latar belakang. Service tidak menyediakan antarmuka pengguna (UI). (Aktivitas, di sisi lain, menyediakan UI.)

Service bisa dimulai, diikat, atau keduanya:

- Service yang dimulai adalah service yang komponen aplikasinya memulai dengan memanggil startService() . Gunakan service yang dimulai untuk tugas yang berjalan di latar belakang guna menjalankan operasi yang berjalan lama. Selain itu, gunakan service yang dimulai untuk tugas yang menjalankan pekerjaan untuk proses jarak jauh.
- Service terikat adalah service yang komponen aplikasinya diikat ke dirinya sendiri dengan memanggil bindService() .

Gunakan service terikat untuk tugas yang berinteraksi dengan komponen aplikasi lain guna menjalankan komunikasi interproses (IPC). Misalnya, service terikat mungkin menangani transaksi jaringan, menjalankan I/O file, memutar musik, atau berinteraksi dengan penyedia materi.

Jika service Anda akan melakukan pekerjaan yang banyak membutuhkan CPU atau operasi pemblokiran (seperti pemutaran MP3 atau jaringan), buat thread baru dalam service untuk melakukan pekerjaan itu. Dengan menggunakan thread terpisah, Anda akan mengurangi risiko kesalahan Aplikasi Tidak Merespons (Application Not Responding/ANR) dan thread utama aplikasi bisa terus disediakan untuk interaksi pengguna dengan aktivitas Anda.

Untuk mengimplementasikan suatu jenis service dalam aplikasi Anda:

1. Deklarasikan service di manifes.
2. Buat kode implementasi, seperti yang dijelaskan dalam Service yang dimulai dan Service terikat, di bawah ini.
3. Kelola daur hidup service.

Mendeklarasikan service di manifes

Sebagaimana dengan aktivitas dan komponen lainnya, Anda harus mendeklarasikan semua service dalam file manifes aplikasi. Untuk mendeklarasikan service, tambahkan elemen `<service>` sebagai anak dari elemen `<application>` .

Misalnya:

```
<manifest . . . >
. . .
<application . . . >
<service android: name="ExampleService"
android: exported="false" />
. . .
</application>
</manifest>
```

Untuk memblokir akses ke service dari aplikasi lainnya, deklarasikan service sebagai privat. Caranya, setel atribut `android: exported` ke `false` . Ini akan menghentikan aplikasi lain dari memulai service Anda, bahkan bila menggunakan maksud eksplisit.

startService

Cara memulai service:

1. Komponen aplikasi seperti aktivitas memanggil `startService()` dan meneruskannya di Intent . Dalam hal ini

Intent menetapkan service dan menyertakan data yang akan digunakan oleh service.

2. Sistem akan memanggil metode `onCreate()` service dan callback lainnya yang sesuai di thread utama. Tergantung service untuk mengimplementasikan callback

tersebut dengan perilaku yang sesuai, seperti membuat thread sekunder yang akan digunakan.

3. Sistem akan memanggil metode `onStartCommand()` service, dengan meneruskan Intent yang disediakan oleh klien di langkah 1 . (Klien dalam konteks ini adalah komponen aplikasi yang memanggil service.)

Setelah dimulai, service bisa berjalan di latar belakang tanpa dibatasi waktu, bahkan jika komponen yang memulainya telah dimusnahkan. Biasanya, service yang dimulai menjalankan operasi tunggal dan tidak mengembalikan hasil ke pemanggil. Misalnya, service dapat mengunduh atau mengunggah file melalui jaringan. Bila operasi selesai, service harus berhenti sendiri dengan memanggil `stopSelf()` , atau komponen lain bisa menghentikannya dengan memanggil `stopService()` .

Misalnya, anggaplah aktivitas perlu menyimpan data ke database online. Aktivitas akan memulai service pendamping dengan meneruskan Intent ke `startService()` . Service menerima maksud di `onStartCommand()` , menghubungkan ke Internet, dan menjalankan transaksi database. Bila transaksi selesai, service akan menggunakan `stopSelf()` untuk menghentikan dirinya sendiri dan dimusnahkan. (Ini adalah contoh service yang ingin Anda jalankan di thread pekerja, sebagai ganti thread utama.)

IntentService

Sebagian besar service yang dimulai tidak perlu menangani beberapa permintaan secara bersamaan, dan jika service melakukannya, maka akan mengakibatkan skenario multi-threading yang berbahaya. Karena itu, sebaiknya Anda mengimplementasikan service menggunakan kelas `IntentService` .

- `IntentService` adalah subkelas yang berguna dari `Service` :
- `IntentService` secara otomatis menyediakan thread pekerja untuk menangani Intent .
- `IntentService` menangani beberapa kode boilerplate yang diperlukan service umum (seperti memulai dan menghentikan service).

`IntentService` bisa membuat antrian pekerjaan yang meneruskan satu maksud untuk setiap kalinya ke implementasi `onHandleIntent()` , sehingga Anda tidak perlu mengkhawatirkan multi-threading.

Untuk mengimplementasikan `IntentService` :

1. Sediakan konstruktor kecil untuk service.
2. Buat implementasi `onHandleIntent()` untuk melakukan pekerjaan yang disediakan klien.

Inilah contoh implementasi `IntentService` :

BindService

Service "terikat" bila komponen aplikasi mengikatnya dengan memanggil `bindService()` . Service terikat menawarkan antarmuka klien-server yang memungkinkan komponen berinteraksi dengan service, mengirim permintaan, dan mendapatkan hasil, kadang-kadang menggunakan komunikasi interproses (IPC) untuk mengirim dan menerima informasi di seluruh proses. Service terikat hanya berjalan selama komponen aplikasi terikat padanya. Beberapa komponen bisa diikat ke service sekaligus, namun bila semuanya telah dilepas, service akan dimusnahkan.

Service terikat umumnya tidak mengizinkan memulai komponen dengan memanggil `startService()` .

Mengimplementasikan `bindService`

Untuk mengimplementasikan service terikat, definisikan antarmuka yang menetapkan cara klien bisa berkomunikasi dengan service. Antarmuka ini, yang dikembalikan service Anda dari metode callback `onBind()` , harus berupa implementasi `IBinder` . Untuk mengambil antarmuka `IBinder` , komponen aplikasi klien memanggil `bindService()` . Setelah klien menerima `IBinder` , klien berinteraksi dengan service melalui antarmuka itu. Ada sejumlah cara untuk mengimplementasikan service terikat, dan implementasi tersebut lebih rumit daripada service yang dimulai. Untuk detail selengkapnya tentang service terikat, lihat Service Terikat.

Mengikat ke service

Untuk mengikat ke service yang dideklarasikan di manifes dan diimplementasikan oleh komponen aplikasi, gunakan `bindService()` dengan Intent eksplisit.

Perhatian: Jangan gunakan maksud implisit untuk mengikat ke service. Melakukannya adalah bahaya keamanan, karena Anda tidak bisa memastikan service yang akan merespons maksud tersebut, dan pengguna tidak bisa melihat service mana yang dimulai. Mulai dengan Android 5.0 (API level 21), sistem membuat pengecualian jika Anda memanggil `bindService()` dengan Intent implisit.

Daur hidup service

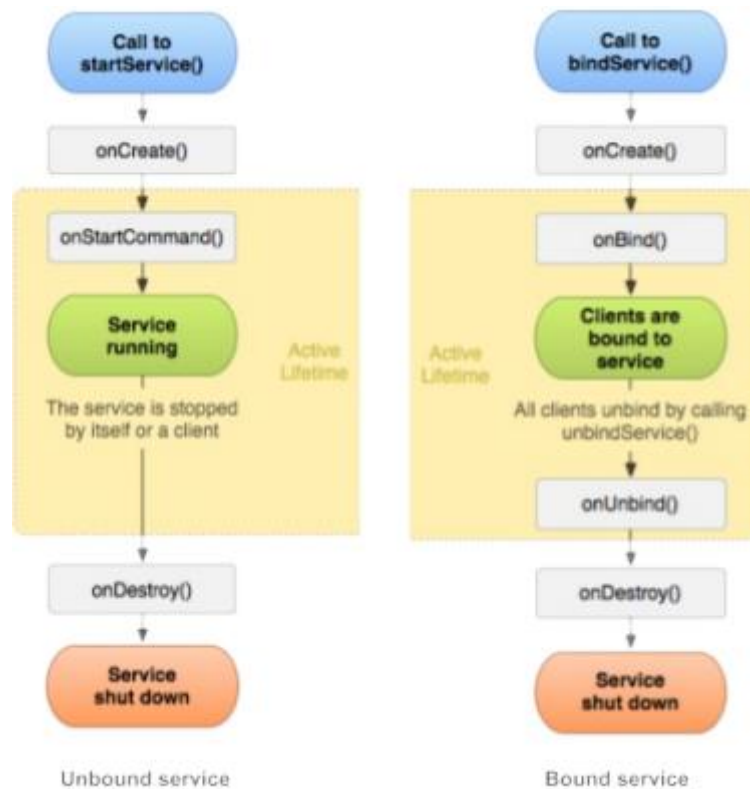
Daur hidup service lebih sederhana daripada aktivitas. Akan tetapi, ini jauh lebih penting karena Anda memerhatikan dari dekat cara service dibuat dan dimusnahkan. Karena tidak memiliki UI, service bisa terus berjalan di latar belakang tanpa diketahui pengguna, bahkan jika pengguna beralih ke aplikasi lain. Ini menghabiskan sumber daya dan menguras baterai.

Seperti aktivitas, service memiliki metode callback daur hidup yang bisa Anda implementasikan untuk memantau perubahan keadaan service dan melakukan pekerjaan pada waktu yang sesuai. Service kerangka berikut memperagakan setiap metode daur hidup:

Daur hidup service yang dimulai vs. service terikat

Service terikat hanya tersedia untuk menyajikan komponen aplikasi yang terikat padanya, sehingga bila tidak ada lagi komponen yang diikat ke service tersebut, sistem akan memusnahkannya. Service terikat tidak perlu dihentikan secara eksplisit seperti halnya service yang dimulai (menggunakan `stopService()` atau `stopSelf()`).

Diagram di bawah ini menampilkan perbandingan antara daur hidup service yang dimulai dan terikat.



Service Foreground

Walaupun sebagian besar service berjalan di latar belakang, sebagian lagi ada yang berjalan di latar depan. Service latar depan adalah service yang diketahui pengguna, jadi ini bukan service yang bakal dimatikan sistem bila memori tinggal sedikit.

Misalnya, pemutar musik yang memutar musik dari service harus disetel untuk berjalan di latar depan, karena pengguna mengetahui operasinya. Notifikasi di bilah status dapat menunjukkan lagu saat ini dan memungkinkan pengguna meluncurkan aktivitas untuk berinteraksi dengan pemutar musik.

Untuk meminta agar service berjalan di latar depan, panggil `startForeground()` sebagai ganti `startService()`. Metode ini menggunakan dua parameter: integer yang secara unik mengidentifikasi notifikasi dan `Notification` untuk bilah status.

Notifikasi ini sedang berlangsung, artinya tidak bisa ditutup. Notifikasi tetap berada di bilah status hingga service dihentikan atau dibuang dari latar depan.

Service terjadwal

Untuk API level 21 dan yang lebih tinggi, Anda bisa meluncurkan service menggunakan JobScheduler API. Untuk menggunakan JobScheduler, Anda perlu mendaftarkan tugas dan menetapkan persyaratannya untuk jaringan dan pengaturan waktu. Sistem menjadwalkan tugas untuk dieksekusi di waktu yang tepat.

Antarmuka JobScheduler menyediakan banyak metode untuk mendefinisikan ketentuan eksekusi service. Untuk detailnya, lihat JobScheduler reference.

D. Praktikum

1. Tambahkan di file AndroidManifest.xml sehingga menjadi sebagai berikut:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.d3ti.safiie99.mybroadcast2">

    <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>

        <service
            android:name=".DownloadService"
            android:exported="false"></service>
    </application>

</manifest>
```

2. File activity_main.xml

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="Hello World!"
        android:orientation="vertical"
```

```

        android:paddingBottom="16dp"
        android:paddingTop="16dp"
        android:paddingLeft="16dp"
        android:paddingRight="16dp"
    />
    <Button
        android:id="@+id/bt_Download"
        android:text="Donload a File"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>

```

3. Buat file DownloadService.java

```

package com.d3ti.safiie99.mybroadcast2;

import android.app.IntentService;
import android.content.Intent;
import android.util.Log;

/**
 * An {@link IntentService} subclass for handling asynchronous task
 requests in
 * a service on a separate handler thread.
 * <p>
 * TODO: Customize class - update intent actions and extra parameters.
 */
public class DownloadService extends IntentService {
    // TODO: Rename actions, choose action names that describe tasks that
    this
    // IntentService can perform, e.g. ACTION_FETCH_NEW_ITEMS
    //public static final String ACTION_FOO =
    "com.d3ti.safiie99.mybroadcast2.action.FOO";
    //public static final String ACTION_BAZ =
    "com.d3ti.safiie99.mybroadcast2.action.BAZ";
    public static final String TAG = "DownloadService";

    // TODO: Rename parameters
    //public static final String EXTRA_PARAM1 =
    "com.d3ti.safiie99.mybroadcast2.extra.PARAM1";
    //public static final String EXTRA_PARAM2 =
    "com.d3ti.safiie99.mybroadcast2.extra.PARAM2";

    public DownloadService() {
        super("DownloadService");
    }

    @Override
    protected void onHandleIntent(Intent intent) {
        /*if (intent != null) {
            final String action = intent.getAction();
            if (ACTION_FOO.equals(action)) {
                final String param1 = intent.getStringExtra(EXTRA_PARAM1);
                final String param2 = intent.getStringExtra(EXTRA_PARAM2);
                handleActionFoo(param1, param2);
            } else if (ACTION_BAZ.equals(action)) {
                final String param1 = intent.getStringExtra(EXTRA_PARAM1);
                final String param2 = intent.getStringExtra(EXTRA_PARAM2);
                handleActionBaz(param1, param2);
            }
        }
        */
    }
}

```



```

    } */

    Log.d(TAG, "Download service dijalankam");
    if (intent != null) {
        try {
            Thread.sleep(1000);
        } catch (InterruptedException ie) {
            ie.printStackTrace();
        }
        Intent notifyFinishIntent = new
Intent(MainActivity.ACTION_DOWNLOAD_STATUS);
        sendBroadcast(notifyFinishIntent);
    }
}

/**
 * Handle action Foo in the provided background thread with the
provided
 * parameters.
 */
private void handleActionFoo(String param1, String param2) {
    // TODO: Handle action Foo
    throw new UnsupportedOperationException("Not yet implemented");
}

/**
 * Handle action Baz in the provided background thread with the
provided
 * parameters.
 */
private void handleActionBaz(String param1, String param2) {
    // TODO: Handle action Baz
    throw new UnsupportedOperationException("Not yet implemented");
}
}

```

4. Buat file MainActivity.java

```

package com.d3ti.safii99.mybroadcast2;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity implements
View.OnClickListener {
    private Button bt_download;
    public static final String ACTION_DOWNLOAD_STATUS="donload_status";
    private BroadcastReceiver downloadReceiver;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

```

```

super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

setTitle("Incoming Message");
bt_download = (Button) findViewById(R.id.bt_Download);
bt_download.setOnClickListener(this);

/*bt_download.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent downloadServiceIntent = new Intent(this,
DownloadService.class);
        startService(downloadServiceIntent);
    }
});*/
downloadReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(context, "Download selesai",
Toast.LENGTH_SHORT).show();
    }
};
IntentFilter downloadIntenFilter = new
IntentFilter(ACTION_DOWNLOAD_STATUS);
registerReceiver(downloadReceiver, downloadIntenFilter);

} //onCreate

@Override
public void onClick(View view) {
    if (view.getId() == R.id.bt_Download) {
        Intent downloadServiceIntent = new Intent(this,
DownloadService.class);
        startService(downloadServiceIntent);
    }
}

@Override
protected void onDestroy() {
    super.onDestroy();
    if(downloadReceiver != null) {
        unregisterReceiver(downloadReceiver);
    }
}

} //Mainactivity

```

E. Latihan/Tugas

Selesaikan tugas coding diatas

Materi 9 : AsyncTask dan AsyncTaskLoader

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa menggunakan AsyncTask dan AsyncTaskLoader untuk membuat aplikasi background.
2. Mahasiswa dapat membuat aplikasi sederhana menggunakan AsyncTask dan AsyncTaskLoader untuk membuat aplikasi background.

C. Materi:

Ada dua cara untuk melakukan pemrosesan latar belakang di Android: menggunakan kelas AsyncTask , atau menggunakan kerangka kerja Loader , yang menyertakan kelas AsyncTaskLoader yang menggunakan AsyncTask . Di sebagian besar situasi, Anda akan memilih kerangka kerja Loader , namun penting untuk mengetahui cara kerja AsyncTask , sehingga Anda bisa membuat pilihan yang bagus.

Dalam bab ini Anda akan mempelajari alasan pentingnya memproses beberapa tugas di latar belakang, di luar thread UI. Anda akan mempelajari cara menggunakan AsyncTask , bila tidak menggunakan AsyncTask , dan dasar-dasar penggunaan loader.

Thread UI

Bila aplikasi Android dimulai, aplikasi membuat thread utama, yang sering disebut thread UI. Thread UI mengirimkan kejadian ke widget antarmuka pengguna (UI) yang sesuai, dan ini merupakan tempat aplikasi Anda berinteraksi dengan komponen dari toolkit UI Android (komponen dari paket android.widget dan android.view).

Model thread Android memiliki dua aturan:

- 1 . Jangan memblokir thread UI.

Thread UI perlu memberikan perhatiannya untuk menggambar UI dan menjaga aplikasi tetap responsif terhadap masukan pengguna. Jika semuanya terjadi di thread UI, operasi panjang seperti akses jaringan atau kueri database bisa

memblokir seluruh UI. Dari perspektif pengguna, aplikasi tersebut akan mogok. Lebih buruk lagi, jika thread UI diblokir selama lebih dari beberapa detik (saat ini sekitar 5 detik) pengguna akan ditampilkan dialog "application not responding" (ANR). Pengguna bisa memutuskan untuk keluar dari aplikasi dan mencopot pemasangannya.

Untuk memastikan aplikasi Anda tidak memblokir thread UI:

- Selesaikan semua pekerjaan dalam waktu kurang dari 16 ms untuk setiap layar UI.
- Jangan menjalankan tugas asinkron dan tugas lain yang berjalan lama pada thread UI. Sebagai gantinya, implementasikan tugas pada thread latar belakang menggunakan AsyncTask (untuk tugas singkat atau yang bisa disela) atau AsyncTaskLoader (untuk tugas berprioritas tinggi, atau tugas yang perlu melaporkan kembali ke pengguna atau UI).

Jangan menggunakan thread latar belakang untuk memanipulasi UI Anda, karena toolkit UI Android bukan thread-safe.

AsyncTask

Gunakan kelas AsyncTask untuk mengimplementasikan tugas asinkron yang berjalan lama di thread pekerja. (Thread pekerja adalah thread yang bukan thread utama atau thread UI.) AsyncTask memungkinkan Anda menjalankan operasi latar belakang dan mempublikasikan hasil di thread UI tanpa memanipulasi thread atau penangan.

Bila AsyncTask dieksekusi, maka akan melalui empat langkah:

1. `onPreExecute()` dipanggil di thread UI sebelum tugas dieksekusi. Langkah ini biasanya digunakan untuk

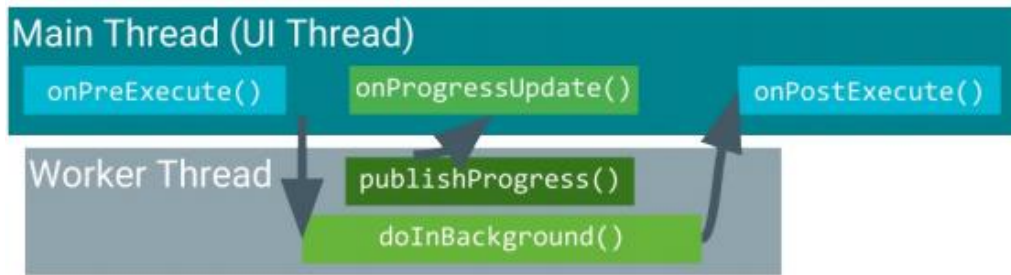
mempersiapkan tugas, misalnya dengan menampilkan bilah kemajuan di UI.

2. `doInBackground(Params...)` dipanggil pada thread latar belakang segera setelah `onPreExecute()` selesai. Langkah ini menjalankan komputasi latar belakang, mengembalikan hasil, dan meneruskan hasilnya ke `onPostExecute()`. Metode `doInBackground()` juga bisa memanggil `publishProgress(Progress...)` untuk mempublikasikan satu atau beberapa unit kemajuan.

3. `onProgressUpdate(Progress...)` berjalan di thread UI setelah `publishProgress(Progress...)` dipanggil. Gunakan `onProgressUpdate()` untuk melaporkan suatu bentuk kemajuan ke thread UI sewaktu komputasi latar belakang dieksekusi. Misalnya, Anda bisa menggunakannya untuk meneruskan data guna menganimasikan bilah kemajuan atau menampilkan log di bidang teks.

4. `onPostExecute(Result)` berjalan di thread UI setelah komputasi latar belakang selesai.

Untuk detail selengkapnya mengenai metode ini, lihat referensi AsyncTask . Di bawah ini adalah diagram urutan pemanggilan.



Penggunaan AsyncTask

Untuk menggunakan kelas AsyncTask , definisikan subkelas AsyncTask yang menggantikan metode doInBackground(Params. .) (dan biasanya juga metode onPostExecute(Result)). Bagian ini menjelaskan parameter dan penggunaan AsyncTask , kemudian menampilkan contoh lengkap.

Parameter AsyncTask

Di subkelas AsyncTask , sediakan tipe data untuk tiga jenis parameter.

- "Params" menetapkan tipe parameter yang diteruskan ke doInBackground() sebagai larik.
- "Progress" menetapkan tipe parameter yang diteruskan ke publishProgress() di thread latar belakang. Parameter ini selanjutnya diteruskan ke metode onProgressUpdate() di thread utama.
- "Result" menetapkan tipe parameter yang dikembalikan doInBackground() . Parameter ini secara otomatis diteruskan ke onPostExecute() di thread utama.

Tetapkan tipe data untuk setiap tipe parameter ini, atau gunakan Void jika tipe parameter tidak akan digunakan. Misalnya:

```
public class MyAsyncTask extends AsyncTask <String, Void, Bitmap>{
```

Dalam deklarasi kelas ini:

- Tipe parameter "Params" adalah String , yang berarti bahwa MyAsyncTask memerlukan satu atau beberapa string sebagai parameter di doInBackground() , misalnya untuk digunakan di kueri.
- Tipe parameter "Progress" adalah Void , yang berarti bahwa MyAsyncTask tidak akan menggunakan metode publishProgress() atau onProgressUpdate() .
- Tipe parameter "Result" adalah Bitmap . MyAsyncTask mengembalikan Bitmap di doInBackground() , yang diteruskan ke dalam onPostExecute() .

Contoh AsyncTask

```

private class DownloadFilesTask extends AsyncTask<URL, Integer, Long> {
protected Long doInBackground( URL. . . urls) {
int count = urls. length;
long totalSize = 0;
for ( int i = 0; i < count; i++) {
totalSize += Downloader. downloadFile( urls[i] ) ;
publishProgress( ( int) ( ( i / ( float) count) * 100) ) ;
// Escape early if cancel( ) is called
if ( isCancelled( ) ) break;
}return totalSize;
}p
rotected void onProgressUpdate( Integer. . . progress) {
setProgressPercent( progress[0] ) ;
}p
rotected void onPostExecute( Long result) {
showDialog( "Downloaded " + result + " bytes" ) ;
}
}

```

Contoh di atas melewati tiga dari empat langkah-langkah AsyncTask dasar:

- doInBackground() mengunduh materi, tugas yang berjalan lama. Langkah ini menghitung persentase file yang diunduh dari indeks loop for dan meneruskannya ke publishProgress(). Pemeriksaan untuk isCancelled() di dalam loop for memastikan bahwa tugas telah dibatalkan, sistem tidak menunggu hingga loop selesai.
- onProgressUpdate() memperbarui kemajuan persentase. Ini dipanggil setiap kali metode publishProgress() dipanggil di dalam doInBackground() , yang memperbarui kemajuan persentase.
- doInBackground() menghitung jumlah total byte yang diunduh dan mengembalikannya. onPostExecute() menerima hasil yang dikembalikan dan meneruskannya ke dalam onPostExecute() , yang ditampilkan di dialog.

Tipe parameter yang digunakan dalam contoh ini adalah:

- URL untuk tipe parameter "Params". Tipe URL berarti Anda bisa meneruskan sejumlah URL ke dalam panggilan, dan URL secara otomatis diteruskan ke dalam metode doInBackground() sebagai larik.
- Integer untuk tipe parameter "Progress".
- Long untuk tipe parameter "Result".

Mengeksekusi AsyncTask

Setelah Anda mendefinisikan subkelas AsyncTask , buat instance-nya di thread UI. Kemudian panggil execute() di instance, dengan meneruskan sejumlah parameter. (Parameter tersebut sesuai dengan tipe parameter "Params" yang dibahas di atas).

Misalnya, untuk mengeksekusi tugas DownloadFilesTask yang didefinisikan di atas, gunakan baris kode berikut:

```
new DownloadFilesTask( ) . execute( url1, url2, url3) ;
```

Membatalkan AsyncTask

Anda bisa membatalkan tugas kapan saja, dari thread apa pun, dengan memanggil metode `cancel()` .

- Metode `cancel()` akan mengembalikan `false` jika tugas tidak bisa dibatalkan, biasanya karena sudah diselesaikan secara normal. Jika tidak, `cancel()` akan mengembalikan `true` .
- Untuk mengetahui apakah tugas sudah dibatalkan, periksa nilai yang dikembalikan `isCancelled()` secara berkala dari `doInBackground(Object[])` , misalnya dari dalam loop seperti yang ditampilkan dalam contoh di atas. Metode `isCancelled()` akan mengembalikan `true` jika tugas dibatalkan sebelum diselesaikan secara normal.
- Setelah tugas AsyncTask dibatalkan, `onPostExecute()` tidak akan digunakan setelah `doInBackground()` dikembalikan. Sebagai gantinya, `onCancelled(Object)` akan dipanggil. Implementasi default `onCancelled(Object)` cukup memanggil `onCancelled()` dan mengabaikan hasil.
- Secara default, tugas yang sedang diproses boleh diselesaikan. Untuk memperbolehkan `cancel()` menyela thread yang sedang mengeksekusi tugas, teruskan `true` untuk nilai `mayInterruptIfRunning` .

Keterbatasan AsyncTask

AsyncTask tidak praktis untuk beberapa kasus penggunaan:

- Perubahan pada konfigurasi perangkat menyebabkan masalah. Bila konfigurasi perangkat berubah sewaktu AsyncTask berjalan, misalnya jika pengguna mengubah orientasi layar, aktivitas yang membuat AsyncTask akan dimusnahkan dan dibuat ulang. Metode AsyncTask tidak dapat mengakses aktivitas yang baru saja dibuat dan hasil AsyncTask tidak akan dipublikasikan.
- Objek AsyncTask lama tetap ada, dan aplikasi Anda bisa kehabisan memori atau mogok. Jika aktivitas yang membuat AsyncTask dimusnahkan, AsyncTask tidak akan dimusnahkan bersamanya. Misalnya, jika pengguna keluar dari aplikasi setelah AsyncTask dimulai, AsyncTask akan terus menggunakan sumber daya kecuali jika Anda memanggil `cancel()` .

Bila menggunakan AsyncTask :

- Tugas singkat atau yang bisa disela.
- Tugas yang tidak perlu untuk melaporkan kembali ke UI atau pengguna.
- Tugas dengan prioritas rendah yang bisa ditinggalkan sebelum selesai.

Untuk semua situasi lainnya, gunakan `AsyncTaskLoader` , adalah bagian dari kerangka kerja Loader yang akan dijelaskan berikutnya.

Loader

Tugas latar belakang biasanya digunakan untuk memuat data seperti laporan prakiraan cuaca atau ulasan film. Pemuatan data bisa jadi banyak menggunakan memori, dan Anda ingin data tersedia sekalipun jika konfigurasi perangkat berubah. Untuk situasi ini, gunakan loader, yang berupa rangkaian kelas yang memfasilitasi pemuatan data ke dalam aktivitas.

Loader menggunakan kelas `LoaderManager` untuk mengelola satu atau beberapa loader. `LoaderManager` menyertakan serangkaian callback bila loader telah dibuat, bila pemuatan datanya selesai, dan bila disetel ulang. Memulai loader

Gunakan kelas `LoaderManager` untuk mengelola satu atau beberapa instance Loader dalam aktivitas atau fragmen.

Gunakan `initLoader()` untuk melakukan inisialisasi dan mengaktifkannya. Biasanya, Anda melakukan ini dalam metode `onCreate()` aktivitas. Misalnya:

```
// Prepare the loader. Either reconnect with an existing one,  
// or start a new one.  
getLoaderManager().initLoader(0, null, this);
```

Jika Anda menggunakan Pustaka Dukungan, buat panggilan ini menggunakan `getSupportLoaderManager()` sebagai ganti

```
getLoaderManager().initLoader(0, null, this);  
getSupportLoaderManager().initLoader(0, null, this);
```

Metode `initLoader()` memerlukan tiga parameter:

- ID unik yang mengidentifikasi loader. ID ini bisa berupa apa saja yang Anda inginkan.
- Argumen opsional yang disediakan ke loader saat pembuatan, dalam bentuk `Bundle`. Jika loader sudah ada, parameter ini akan diabaikan.
- Implementasi `LoaderCallbacks`, yang dipanggil oleh `LoaderManager` untuk melaporkan kejadian loader. Dalam contoh ini, kelas lokal mengimplementasikan antarmuka `LoaderManager`. `LoaderCallbacks`, sehingga meneruskan referensi ke dirinya sendiri, `this`.

Panggilan `initLoader()` memiliki dua kemungkinan hasil:

- Jika loader yang ditetapkan melalui ID sudah ada, maka loader yang dibuat terakhir menggunakan ID itu akan digunakan kembali.
- Jika loader yang ditetapkan melalui ID tidak ada, `initLoader()` akan memicu metode `onCreateLoader()`. Di sinilah Anda mengimplementasikan kode untuk membuat instance dan mengembalikan loader baru.

Catatan: Bila `initLoader()` membuat loader atau menggunakan kembali loader yang ada, implementasi

LoaderCallbacks yang diberikan akan dikaitkan dengan loader dan dipanggil bila keadaan loader berubah. Jika loader yang diminta sudah ada dan sudah menghasilkan data, maka sistem segera memanggil `onLoadFinished()` (selama `initLoader()`), jadi bersiaplah jika hal ini terjadi.

Masukkan panggilan ke `initLoader()` di `onCreate()` sehingga aktivitas bisa dihubungkan kembali ke loader yang sama bila konfigurasi berubah. Dengan cara itu, loader tidak kehilangan data yang sudah dimuatnya.

Memulai ulang loader

Bila `initLoader()` menggunakan kembali loader yang ada, maka data yang telah dimuat loader tidak akan diganti, namun kadang-kadang Anda perlu menggantinya. Misalnya, bila Anda menggunakan kueri pengguna untuk melakukan penelusuran dan pengguna memasukkan kueri baru, Anda perlu memuat ulang data dengan menggunakan istilah penelusuran baru. Dalam situasi ini, gunakan metode `restartLoader()` dan teruskan ID loader yang ingin dimulai ulang. Hal ini akan memaksa muatan data lain dengan data masukan baru.

Tentang metode `restartLoader()` :

`restartLoader()` menggunakan argumen yang sama dengan `initLoader()`. `restartLoader()` akan memicu metode `onCreateLoader()`, seperti yang dilakukan `initLoader()` saat membuat loader baru. Jika sudah ada loader dengan ID yang diberikan, `restartLoader()` akan memulai ulang loader yang diidentifikasi dan mengganti datanya. Jika tidak ada loader dengan ID yang diberikan, `restartLoader()` akan memulai loader baru.

Callback LoaderManager

Objek `LoaderManager` secara otomatis memanggil `onStartLoading()` saat membuat loader. Setelah itu, `LoaderManager` akan mengelola keadaan loader berdasarkan pada keadaan aktivitas dan data, misalnya dengan memanggil `onLoadFinished()` bila data telah dimuat. Untuk berinteraksi dengan loader, gunakan salah satu callback `LoaderManager` di aktivitas yang memerlukan data: Panggil `onCreateLoader()` agar bisa membuat instance dan mengembalikan loader baru untuk ID yang diberikan.

Panggil `onLoadFinished()` bila loader yang dibuat sebelumnya selesai memuat. Di sinilah Anda biasanya ingin memindahkan data ke dalam tampilan aktivitas.

Panggil `onLoaderReset()` bila loader yang dibuat sebelumnya sedang disetel ulang, sehingga datanya tidak tersedia.

Di sinilah aplikasi harus membuang semua referensi apa pun yang dimilikinya ke data loader. Subkelas `Loader` bertanggung jawab atas pemuatan data sebenarnya. Subkelas `Loader` yang Anda gunakan bergantung pada tipe data yang dimuat, namun salah satu yang paling mudah adalah `AsyncTaskLoader`, yang akan

dijelaskan berikutnya. `AsyncTaskLoader` menggunakan `AsyncTask` untuk menjalankan tugas pada thread pekerja.

AsyncTaskLoader

`AsyncTaskLoader` adalah loader yang setara dengan `AsyncTask`. `AsyncTaskLoader` menyediakan metode,

`loadInBackground()`, yang dijalankan di thread terpisah. Hasil `loadInBackground()` secara otomatis dikirimkan ke thread UI, melalui `onLoadFinished()` `LoaderManager` callback.

Penggunaan `AsyncTaskLoader`

Untuk mendefinisikan subkelas `AsyncTaskLoader`, buat kelas yang memperluas `AsyncTaskLoader<D>`, dalam hal ini `D` adalah tipe data yang sedang Anda muat. Misalnya, `AsyncTaskLoader` ini akan memuat daftar string: `public static class StringListLoader extends AsyncTaskLoader<List<String>> {}`

Berikutnya, implementasikan konstruktor yang cocok dengan implementasi super kelas: Konstruktor menggunakan konteks aplikasi sebagai argumen dan meneruskannya ke panggilan untuk `super()`. Jika loader Anda memerlukan informasi tambahan untuk melakukan pemuatan, konstruktor bisa mengambil argumen tambahan. Dalam contoh yang ditampilkan di bawah ini, konstruktor menggunakan sebuah istilah kueri.

```
public StringListLoader( Context context, String queryString) {
    super( context) ;
    mQueryString = queryString;
}
```

Untuk melakukan pemuatan, gunakan metode penggantian `loadInBackground()`, akibat metode `doInBackground()` dari `AsyncTask`. Misalnya:

```
@Override
public List<String> loadInBackground() {
    List<String> data = new ArrayList<String>;
    //TODO: Load the data from the network or from a database
    return data;
}
```

Mengimplementasikan callback

Gunakan konstruktor di callback `onCreateLoader()` `LoaderManager`, yang merupakan tempat membuat loader baru.

Misalnya, callback `onCreateLoader()` ini menggunakan konstruktor `StringListLoader` yang didefinisikan di atas:

```
@Override
public Loader<List<String>> onCreateLoader( int id, Bundle args) {
```

```
return new StringListLoader( this, args. getString( "queryString" ) ) ;
}
```

Hasil `loadInBackground()` secara otomatis diteruskan ke dalam callback `onLoadFinished()` , di sinilah Anda bisa menampilkan hasil di UI. Misalnya:

```
public void onLoadFinished( Loader<List<String>> loader, List<String> data) {
mAdapter. setData( data) ;
}
```

Callback `onLoaderReset()` hanya dipanggil bila loader akan dimusnahkan, sehingga seringkali Anda bisa mengosongkan `onLoaderReset()` , karena Anda tidak akan mencoba mengakses data setelah loader ini dimusnahkan.

Bila Anda menggunakan `AsyncTaskLoader` , data Anda akan bertahan bila ada perubahan konfigurasi perangkat. Jika aktivitas Anda dimusnahkan secara permanen, loader ini akan dimusnahkan bersamanya, tanpa tugas yang menanti dan mengonsumsi sumber daya sistem.

Loader juga memiliki manfaat lain, misalnya loader bisa memantau perubahan sumber data dan memuat ulang data jika terjadi perubahan. Anda akan mengetahui selengkapnya tentang loader tertentu di pelajaran berikutnya.

D. Praktikum

1. Buat file `main_activity.xml` sebagai berikut:

```
<?xml version="1.0" encoding="utf-8" ?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginBottom="16dp"
    android:layout_marginTop="16dp"
    android:layout_marginLeft="16dp"
    android:layout_marginRight="16dp"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/tv_Status"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Status Aplikasi" />

</RelativeLayout>
```

2. Source Code `AsyncTask`

```
package com.d3ti.safiie99.myasynctask;
```

```

import android.os.AsyncTask;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    public static final String DEMO_ASYNC = "DemoAsync";
    private TextView tv_status;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        tv_status = (TextView) findViewById(R.id.tv_Status);
        DemoAsync da = new DemoAsync();
        da.execute("Hallo ini demo AsyncTask");
    }

    private class DemoAsync extends AsyncTask<String, Void, String> {
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            tv_status.setText("Status : onPreExecute");
        }

        @Override
        protected String doInBackground(String... params) {
            Log.d(DEMO_ASYNC, "status : doInBackground");
            try {
                Thread.sleep(5000);
            } catch (Exception e) {
                Log.d(DEMO_ASYNC, e.getMessage());
            }
            return params[0];
        }
    }
}

```

E. Latihan/Tugas

Selesaikan tugas coding diatas

Materi 10 : Database SQLite

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa dapat menggunakan class SQLite dan SQLiteHelper
2. Mahasiswa menggunakan database SQLite untuk insert, update, edit dan delete.

C. Materi:

SQLite

SQLite pustaka perangkat lunak yang mengimplementasikan mesin database SQL yang:

- mandiri (tidak memerlukan komponen lain)
- tanpa server (tidak memerlukan server-backend)
- konfigurasi-nol (tidak perlu dikonfigurasi untuk aplikasi Anda)
- transaksional (perubahan dalam satu transaksi di SQLite akan diterapkan pada semua bagian atau tidak diterapkan sama sekali)

SQLite adalah mesin database yang paling banyak diterapkan di seluruh dunia. Kode sumber untuk SQLite berada dalam domain publik.

Transaksi

Transaksi adalah urutan operasi yang dijalankan sebagai satu unit kerja logis. Unit kerja logis harus menunjukkan empat properti, yaitu properti atomisitas, konsistensi, isolasi, dan ketahanan (ACID), untuk memenuhi syarat sebagai transaksi.

Semua perubahan dalam satu transaksi di SQLite diterapkan pada semua bagian atau tidak diterapkan sama sekali), bahkan jika tindakan menulis perubahan ke disk terputus karena

- program mogok,
- sistem operasi mogok, atau
- listrik mati.

Contoh transaksi:

- Mengirimkan uang dari rekening tabungan ke rekening giro.
- Memasukkan suatu istilah dan definisi ke dalam kamus.
- Mengikat changelist ke cabang utama.

ACID

- Atomisitas. Semua modifikasi data dijalankan, atau tidak ada yang dijalankan.
- Konsistensi. Bila selesai, suatu transaksi harus meninggalkan semua data dalam keadaan konsisten.
- Isolasi. Modifikasi yang dibuat oleh transaksi yang terjadi bersamaan harus diisolasi dari modifikasi yang dibuat oleh transaksi bersamaan lainnya. Suatu transaksi mengenali data baik dalam keadaan sebelum transaksi bersamaan lainnya memodifikasinya, maupun mengenali data setelah transaksi kedua selesai, namun tidak mengenali keadaan di antara keduanya.
- Ketahanan. Setelah transaksi selesai, efeknya diterapkan permanen dalam sistem. Modifikasi tetap ada bahkan seandainya listrik mati.

Bahasa kueri

Anda menggunakan bahasa kueri SQL khusus untuk berinteraksi dengan database. Kueri bisa sangat kompleks, namun operasi dasarnya adalah

- menyisipkan baris
- menghapus baris
- memperbarui nilai dalam baris
- mengambil baris yang memenuhi kriteria tertentu

Di Android, objek database menyediakan metode praktis untuk menyisipkan, menghapus, dan memperbarui database.

Anda hanya perlu memahami SQL untuk mengambil data.

Struktur kueri

Kueri SQL sangat terstruktur dan berisi bagian dasar berikut:

```
SELECT kata, keterangan FROM WORD_LIST_TABLE WHERE kata="alpha"
```

Versi generik kueri contoh:

```
SELECT kolom FROM tabel WHERE kolom="value"
```

Bagian:

- SELECT kolom—memilih kolom untuk dikembalikan. Gunakan * untuk mengembalikan semua kolom.
- FROM tabel—menetapkan tabel yang dipakai untuk mendapatkan hasil.
- WHERE—kata kunci untuk ketentuan yang harus terpenuhi.
- kolom="value"—ketentuan yang harus dipenuhi.
operator umum: =, LIKE, <, >
- AND, OR—menghubungkan beberapa ketentuan dengan operator logika.
- ORDER BY—mengabaikan urutan default, atau menetapkan ASC untuk menaik, DESC untuk menurun.
- LIMIT adalah kata kunci yang sangat berguna jika Anda ingin mendapatkan jumlah hasil yang terbatas.

Cursor

SQLiteDatabase selalu menyajikan hasil berupa Cursor dalam format tabel yang menyerupai database SQL. Anda bisa menganggap data sebagai larik baris. Cursor adalah pointer ke dalam satu baris data terstruktur. Kelas Cursor menyediakan metode untuk menggerakkan kursor melalui struktur data, dan metode untuk mendapatkan data dari bidangbidang di setiap baris.

Kelas Cursor memiliki sejumlah subkelas yang mengimplementasikan kursor untuk tipe data tertentu.

- SQLiteCursor mengekspos hasil kueri dari sebuah SQLiteDatabase. SQLiteCursor tidak disinkronkan secara internal, sehingga kode yang menggunakan SQLiteCursor dari beberapa thread harus melakukan sinkronisasi sendiri saat menggunakan SQLiteCursor.
- MatrixCursor adalah implementasi kursor lengkap dan tidak tetap, yang didukung oleh larik objek yang secara otomatis meluaskan kapasitas internal bila diperlukan.

Beberapa operasi yang umum pada kursor adalah:

- getCount() mengembalikan jumlah baris dalam kursor.
- getColumnNames() mengembalikan larik string yang berisi nama semua kolom dalam rangkaian hasil dalam urutan pencantumannya dalam hasil.
- getPosition() mengembalikan posisi kursor saat ini dalam rangkaian baris.
- Getter tersedia untuk tipe data tertentu, seperti getString(int column) dan getInt(int column).
- Operasi seperti moveToFirst() dan moveToNext() akan menggerakkan kursor.
- close() membebaskan semua sumber daya dan membuat kursor menjadi tidak valid sama sekali. Ingat untuk menutup panggilan guna membebaskan sumber daya!

D. Praktikum

1. Desain User Interface

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.d3ti.sqlite.MainActivity"
    android:orientation="vertical">

    <TextView
        android:id="@+id/textView3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Input Nama dan Hobi"
        android:textSize="24sp"
        android:textStyle="bold" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="60dp"
        android:orientation="horizontal">

        <EditText
            android:id="@+id/inNama"
            android:layout_width="186dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:ems="10"
            android:inputType="textPersonName"
            android:text="Nama" />

        <EditText
            android:id="@+id/inHobi"
            android:layout_width="184dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:ems="10"
            android:inputType="textPersonName"
            android:text="Hobi" />

        <Button
            android:id="@+id/btTambah"
            android:layout_width="186dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Tambah" />
    </LinearLayout>

    <TextView
        android:id="@+id/textView4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Sebelum mengubah data pilih dahulu baris yang akan
dirubah"
        android:textSize="18sp"
        android:textStyle="bold" />
```



```

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Update Baris Ke"
        android:textSize="18sp"
        android:textStyle="bold" />

    <EditText
        android:id="@+id/edById"
        android:layout_width="103dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:text="Baris" />

    <Button
        android:id="@+id/btGetId"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="GET" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="59dp"
    android:orientation="horizontal">

    <EditText
        android:id="@+id/edNama"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:ems="10"
        android:inputType="textPersonName"
        android:text="Nama" />

    <EditText
        android:id="@+id/edHobi"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:ems="10"
        android:inputType="textPersonName"
        android:text="Hobi" />

    <Button
        android:id="@+id/btUpdate"
        android:layout_width="119dp"
        android:layout_height="wrap_content"
        android:text="UPDATE" />

</LinearLayout>

```

```

<TextView
    android:id="@+id/textView6"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Tulis Nama Baris yang akan di delete"
    android:textSize="18sp" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/textView7"
        android:layout_width="155dp"
        android:layout_height="wrap_content"
        android:text="Delete Baris ke"
        android:textSize="18sp" />

    <EditText
        android:id="@+id/delById"
        android:layout_width="124dp"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:text="Baris" />

    <Button
        android:id="@+id/btDelete"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="DELETE" />

</LinearLayout>

<TableLayout
    android:id="@+id/tabel_data"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:visibility="visible">

    <TableRow
        android:id="@+id/tableRow1"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:visibility="visible">

        <TextView
            android:id="@+id/tvNo"
            android:layout_width="50dp"
            android:layout_height="wrap_content"
            android:text="No"
            android:textSize="18sp"
            android:textStyle="bold" />

        <TextView
            android:id="@+id/tvNama"
            android:layout_width="100dp"
            android:layout_height="wrap_content"
            android:text="NAMA"
            android:textSize="18sp"

```

```

        android:textStyle="bold" />

        <TextView
            android:id="@+id/tvHobi"
            android:layout_width="100dp"
            android:layout_height="wrap_content"
            android:text="HOBI"
            android:textSize="18sp"
            android:textStyle="bold" />
    </TableRow>

</TableLayout>

</LinearLayout>

```

2. Source Code file DatabaseManager.java

```

/* File DatabaseManager.java */
package com.d3ti.sqlite;

/**
 * Created by ahmed.safiie on 07/11/2017.
 */

import java.util.ArrayList;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;
import android.widget.Toast;

public class DatabaseManager {
    private static final String ROW_ID = "_id";
    private static final String ROW_NAMA = "nama";
    private static final String ROW_HOBI = "hobi";

    private static final String NAMA_DB = "DatabaseTest";
    private static final String NAMA_TABEL = "hobi";
    private static final int DB_VERSION = 1;

    private static final String CREATE_TABLE = "CREATE TABLE "+NAMA_TABEL+"
(" +ROW_ID+ " integer PRIMARY KEY autoincrement," + ROW_NAMA+ " text,"
+ROW_HOBI+ " text)";
    private final Context context;
    private DatabaseOpenHelper dbHelper;
    private SQLiteDatabase db;

    public DatabaseManager(Context ctx) {
        this.context = ctx;
        dbHelper = new DatabaseOpenHelper(context);
        db = dbHelper.getWritableDatabase();
    }

    private static class DatabaseOpenHelper extends SQLiteOpenHelper {
        public DatabaseOpenHelper(Context context) {

```

```

        super(context, NAMA_DB, null, DB_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(CREATE_TABLE);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {
        db.execSQL("DROP TABLE IF EXISTS "+NAMA_DB);
        onCreate(db);
    }
}

public void close() {
    dbHelper.close();
}

public void addRow(String nama, String hobi) {
    ContentValues values = new ContentValues();
    values.put(ROW_NAMA, nama);
    values.put(ROW_HOBI, hobi);

    try {
        db.insert(NAMA_TABEL, null, values);
    } catch (Exception e) {
        Log.e("DB Error", e.toString());
        e.printStackTrace();
    }
}

public ArrayList<ArrayList<Object>> ambilSemuaBaris() {
    ArrayList<ArrayList<Object>> dataArray = new
ArrayList<ArrayList<Object>>();
    Cursor cur;
    try {
        cur = db.query(NAMA_TABEL, new String[] {ROW_ID, ROW_NAMA,
ROW_HOBI}, null, null, null, null, null);
        cur.moveToFirst();
        if (!cur.isAfterLast()) {
            do {
                ArrayList<Object> dataList = new ArrayList<>();
                dataList.add(cur.getLong(0));
                dataList.add(cur.getString(1));
                dataList.add(cur.getString(2));
                dataArray.add(dataList);
            } while (cur.moveToNext());
        }
    } catch (Exception e) {
        e.printStackTrace();
        Log.e("DB Error", e.toString());
        Toast.makeText(context, "Gagal mengambil semua baris: "
+e.toString(), Toast.LENGTH_SHORT).show();
    }
    return dataArray;
}

public ArrayList<Object> ambilBaris(long rowId) {
    ArrayList<Object> arrbaris = new ArrayList<Object>();

```

```

    Cursor cur;
    try {
        //
        cur = db.query(NAMA_TABEL, new String[] {ROW_ID, ROW_NAMA,
ROW_HOBI}, ROW_ID + "=" + rowId, null, null, null, null, null);
        cur.moveToFirst();

        if(!cur.isAfterLast()) {
            do {
                arrbaris.add(cur.getLong(0));
                arrbaris.add(cur.getString(1));
                arrbaris.add(cur.getString(2));
            } while (cur.moveToNext());
            String r = String.valueOf(arrbaris);
            Toast.makeText(context, "Data: " + r,
Toast.LENGTH_SHORT).show();
        }
        cur.close();
    } catch (Exception e) {
        //
        e.printStackTrace();
        Log.e("Error", e.toString());
        Toast.makeText(context, "Apalagi ini" + e.toString(),
Toast.LENGTH_SHORT).show();
    }
    return arrbaris;
}

public void updateBaris(long rowId, String nama, String hobi) {
    ContentValues cv = new ContentValues();
    cv.put(ROW_NAMA, nama);
    cv.put(ROW_HOBI, hobi);

    try {
        //
        db.update(NAMA_TABEL, cv, ROW_ID + "=" + rowId, null);
    } catch (Exception e) {
        //
        e.printStackTrace();
        Log.e("DB Error", e.toString());
    }
}

public void deleteBaris(long idBaris) {
    try {
        //
        db.delete(NAMA_TABEL, ROW_ID + "=" + idBaris, null);
    } catch (Exception e) {
        //
        e.printStackTrace();
        Log.e("Error", e.toString());
    }
}

} /* End of Code */

```

3. File Main Activity

```

/* File MainActivity */
package com.d3ti.sqlite;

```

```

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

import java.util.ArrayList;
import android.app.Activity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    DatabaseManager dm;
    EditText nama, hobi, getId, updateNama, updateHobi, idDel;
    Button btAdd, btGetId, btUpdate, btDelete;
    TableLayout tabel4data;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        dm = new DatabaseManager(this);
        setupView();
        fungsiButton();
        updateTable();
    }

    public void setupView() {
        tabel4data = (TableLayout) findViewById(R.id.tabel_data);
        nama = (EditText) findViewById(R.id.inNama);
        hobi = (EditText) findViewById(R.id.inHobi);
        updateNama = (EditText) findViewById(R.id.edNama);
        updateHobi = (EditText) findViewById(R.id.edHobi);
        getId = (EditText) findViewById(R.id.edById);
        idDel = (EditText) findViewById(R.id.delById);

        btAdd = (Button) findViewById(R.id.btTambah);
        btGetId = (Button) findViewById(R.id.btGetId);
        btUpdate = (Button) findViewById(R.id.btUpdate);
        btDelete = (Button) findViewById(R.id.btDelete);
    }

    public void fungsiButton() {
        btAdd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                simpKamuta();
                kosongkanField();
            }
        });
        btGetId.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ambilBaris();
            }
        });
    }
}

```

```

    });
    btUpdate.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            updateBaris();
            kosongkanField();
        }
    });
    btDelete.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            deleteData();
            kosongkanField();
        }
    });
}

protected void kosongkanField() {
    nama.setText("");
    hobi.setText("");
    updateNama.setText("");
    updateHobi.setText("");
    getId.setText("");
    idDel.setText("");
}

private void deleteData() {
    dm.deleteBaris(Long.parseLong(idDel.getText().toString()));
    updateTable();
}

protected void updateBaris() {
    dm.updateBaris(Long.parseLong(idDel.getText().toString()),
updateNama.getText().toString(), updateHobi.getText().toString());
    updateTable();
}

private void ambilBaris() {
    try {
        //
        ArrayList<Object> baris;
        baris =
dm.ambilBaris(Long.parseLong(getId.getText().toString()));
        updateNama.setText((String) baris.get(1));
        updateHobi.setText((String) baris.get(2));
    } catch (NumberFormatException e) {
        //
        e.printStackTrace();
        Log.e("Error DB", e.toString());
        Toast.makeText(getBaseContext(), e.toString(),
Toast.LENGTH_LONG).show();
    }
}

protected void simpKamuta() {
    try {
        dm.addRow(nama.getText().toString(),
hobi.getText().toString());
        updateTable();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

```

        Toast.makeText(getApplicationContext(), "Gagal simpan" + e.toString(),
Toast.LENGTH_LONG).show();
    }
}

protected void updateTable() {
    while (tabel4data.getChildCount() > 1) {
        tabel4data.removeViewAt(1);
    }
    ArrayList<ArrayList<Object>> data = dm.ambilSemuaBaris();
    for (int posisi = 0; posisi < data.size(); posisi++) {
        TableRow tabelBaris = new TableRow(this);
        ArrayList<Object> baris = data.get(posisi);

        TextView tvId = new TextView(this);
        tvId.setText(baris.get(0).toString());
        tabelBaris.addView(tvId);

        TextView tvNama = new TextView(this);
        tvNama.setText(baris.get(1).toString());
        tabelBaris.addView(tvNama);

        TextView tvHobi = new TextView(this);
        tvHobi.setText(baris.get(2).toString());
        tabelBaris.addView(tvHobi);

        tabel4data.addView(tabelBaris);
    }
}
} /* End of Code */

```

E. Latihan/Tugas

Buat file yang digunakan untuk menyimpan todo-list

Materi 11 : Sensor

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa dapat menggunakan sensor camera.
2. Mahasiswa dapat menggunakan sensor orientasi.
3. Mahasiswa dapat membuat aplikasi sederhana menggunakan sensor yang terdapat pada hardware android.

C. Materi:

Sebagian besar perangkat pada Android memiliki sensor internal yang mengukur gerakan, orientasi, dan berbagai kondisi lingkungan. Sensor ini mampu menyediakan data dengan presisi dan akurasi tinggi, dan berguna jika Anda ingin memantau pergerakan atau pemosisian perangkat tiga dimensi, atau Anda ingin memantau perubahan di lingkungan sekitar dekat perangkat. Misalnya, game mungkin melacak pembacaan dari sensor gravitasi untuk menyimpulkan gerakan dan gerakan pengguna yang kompleks, seperti kemiringan, goyangan, rotasi, atau ayunan. Demikian juga, aplikasi cuaca mungkin menggunakan sensor suhu dan sensor kelembapan perangkat untuk menghitung dan melaporkan dewpoint, atau aplikasi perjalanan mungkin menggunakan sensor medan dan accelerometer geomagnetik untuk melaporkan bantalan kompas.

Platform Android mendukung tiga kategori sensor yang luas:

- Sensor gerak, sensor ini mengukur gaya akselerasi dan gaya rotasi sepanjang tiga sumbu. Kategori ini termasuk akselerometer, sensor gravitasi, giroskop, dan sensor vektor rotasi.
- Sensor lingkungan, sensor-sensor ini mengukur berbagai parameter lingkungan, seperti suhu udara dan tekanan, pencahayaan, dan kelembaban. Kategori ini termasuk barometer, fotometer, dan termometer.
- Sensor posisi, sensor-sensor ini mengukur posisi fisik perangkat. Kategori ini termasuk sensor orientasi dan magnetometer.

Pengantar Sensor

Framework sensor Android memungkinkan Anda mengakses banyak jenis sensor. Beberapa dari sensor ini berbasis perangkat keras dan beberapa perangkat lunak berbasis. Sensor berbasis perangkat keras adalah komponen fisik yang dibuat menjadi perangkat handset atau tablet. Mereka memperoleh data mereka dengan langsung mengukur sifat lingkungan tertentu, seperti akselerasi, kekuatan medan geomagnetik, atau perubahan sudut. Sensor berbasis perangkat lunak bukanlah perangkat fisik, meskipun mereka meniru sensor berbasis perangkat keras. Sensor berbasis perangkat lunak mengambil data mereka dari satu atau lebih dari sensor berbasis perangkat keras dan kadang-kadang disebut sensor virtual atau sensor sintetis. Sensor percepatan linear dan sensor gravitasi adalah contoh dari sensor berbasis perangkat lunak. Tabel 1 merangkum sensor yang didukung oleh platform Android.

Sensor	Type	Description	Common Uses
TYPE_ACCELEROMETER	Hardware	Mengukur gaya akselerasi dalam m/s^2 pada ketiga sumbu putar, x, y dan z	deteksi gerakan
TYPE_AMBIENT_TEMPERATURE	Hardware	Mengukur suhu ruang dalam $^{\circ}C$	Monitoring air temperatures.
TYPE_GRAVITY	Software or Hardware	Mengukur gaya pada gravitasi dalam m/s^2 yang diterapkan pada ketiga sumbu yaitu x,y,dan z	Deteksi gerakan (shake, tilt, etc.).
TYPE_GYROSCOPE	Hardware	Mengukur rerata rotasi dalam rad/s dalam ketiga sumbu, yaitu: x, y dan z.	Deteksi rotasi (spin, turn, etc.).
TYPE_LIGHT	Hardware	Mengukur level cahaya (iluminasi) dalam lx	Kecerahan screen
TYPE_LINEAR_ACCELERATION	Software or Hardware	Mengukur gaya percepatan dalam m/s^2 pada ketiga sumbu, yaity: x, y dan z.	Monitoring akselerasi.
TYPE_ORIENTATION	Software	Mengukur derajat rotasi dimana divais terdapat dalam 3 sumbu, yaitu x,y dan z.	Pengukuran posisi divais.
TYPE_PROXIMITY	Hardware	Mengukur	Posisi telp pada

		perkiraan dari suatu object, biasanya digunakan untuk mengukur headset digunakan.	saat telepon
<u>TYPE RELATIVE HUMIDITY</u>	Hardware	Mengukur kelembaban relatif dalam (%).	Monitoring kelembaban relatif

Framework Sensor

Untuk dapat mengakses sensor ini dan memperoleh data dengan menggunakan kerangka sensor Android. Framework sensor adalah bagian dari paket android.hardware dan termasuk kelas dan antarmuka berikut:

- **SensorManager**, untuk dapat menggunakan kelas ini untuk membuat turunan dari layanan sensor. Kelas ini menyediakan berbagai metode untuk mengakses dan mendaftarkan sensor, mendaftarkan dan tidak mendaftarkan pendengar peristiwa sensor, dan memperoleh informasi orientasi. Kelas ini juga menyediakan beberapa konstanta sensor yang digunakan untuk melaporkan keakuratan sensor, mengatur tingkat perolehan data, dan mengkalibrasi sensor.
- **Sensor**, untuk dapat menggunakan kelas ini untuk membuat instance dari sensor tertentu. Kelas ini menyediakan berbagai metode yang memungkinkan Anda menentukan kemampuan sensor.
- **SensorEvent**, sistem menggunakan kelas ini untuk membuat objek peristiwa sensor, yang menyediakan informasi tentang peristiwa sensor. Objek peristiwa sensor mencakup informasi berikut: data sensor mentah, jenis sensor yang menghasilkan peristiwa, keakuratan data, dan stempel waktu untuk peristiwa tersebut.
- **SensorEventListener**, untuk dapat menggunakan antarmuka ini untuk membuat dua metode panggilan balik yang menerima pemberitahuan (peristiwa sensor) ketika nilai-nilai sensor berubah atau ketika akurasi sensor berubah.

D. Praktikum

E. Latihan/Tugas

Materi 12 : Google Map

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa dapat menggunakan sensor GPS.
2. Mahasiswa dapat membuat aplikasi sederhana menggunakan google map untuk ditempelkan.

C. Materi:

Trend teknologi sekarang dan kedepan salah satunya adalah teknologi mobile dengan fitur LBS nya. Dengan LBS, kita bisa mencari suatu lokasi dengan sangat mudah, mencari posisi teman, mempublish lokasi keberadaan kita sendiri, mencari rute jalan atau lainnya.

Suatu gadget katakanlah Android mendapatkan lokasi bisa dari GPS , Network Location Provider atau bisa juga keduanya. Cell-id (Simcard) dan wifi adalah contoh Network Location

Provider.

Menentukan lokasi menggunakan Android itu gampanggampang susah, karena semakin akurat suatu lokasi semakin boros juga baterainya. Berikut ini adalah hal-hal yang menjadi

kendala untuk menentukan lokasi

1. Multitude sumber lokasi

GPS, simcard dan wifi dapat dikombinasikan untuk mendapatkan lokasi yang akurat, namun efeknya jatuh ke baterai.

2. Perpindahan pengguna

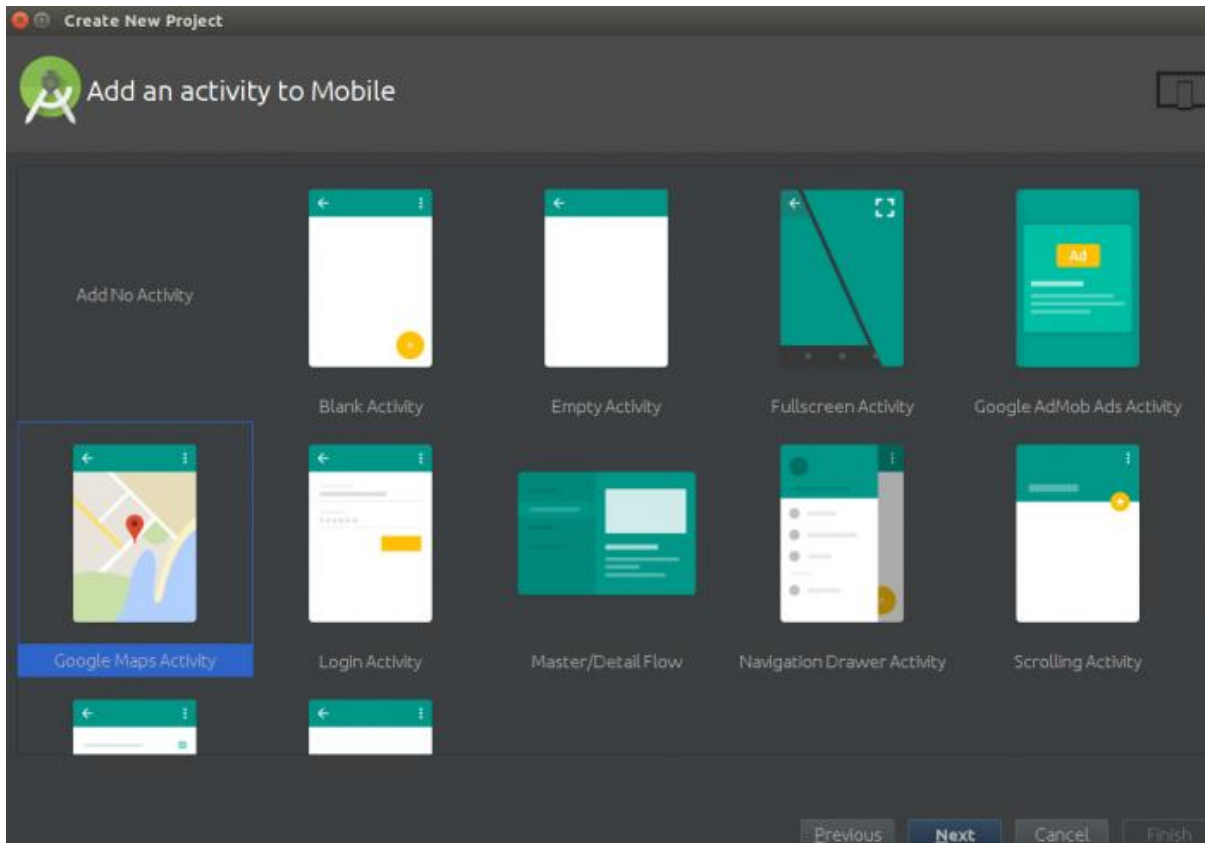
Karena seorang pengguna melakukan perpindahan, maka aplikasi juga akan mererefresh lokasi secara berulang.

3. Akurasi

Akurasi terhadap posisi suatu lokasi tidak konsisten. Posisi 10 menit yang lalu bias jadi lebih akurat dibandingkan posisi yang terbaru

D. Praktikum

1. Buat Activity baru sebagai berikut:



2. Tambahkan user permission pada file Android Manifest sehingga kode lengkapnya seperti berikut:

```
/* Android Manifest */
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.d3ti.a02_peta">
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"
    />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission
    android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"
    />
    <!--
    use      The ACCESS_COARSE/FINE_LOCATION permissions are not required to
    fine     Google Maps Android API v2, but you must specify either coarse or
    location permissions for the 'MyLocation' functionality.
    -->
    <uses-permission
    android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
    />
</manifest>
```

```

<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">

    <!--
        The API key for Google Maps-based APIs is defined as a string
        resource.
        (See the file "res/values/google_maps_api.xml").
        Note that the API key is linked to the encryption key used to
        sign the APK.
        You need a different API key for each encryption key,
        including the release key that is used to
        sign the APK for publishing.
        You can define the keys for the debug and release targets in
        src/debug/ and src/release/.
    -->
    <meta-data
        android:name="com.google.android.geo.API_KEY"
        android:value="@string/google_maps_key" />

    <activity
        android:name=".MapsActivity"
        android:label="@string/title_activity_maps">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

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

</manifest>

```

3. Masukkan google map api key ke dalam google_maps_api, sehingga memiliki kode lengkap sebagai berikut

```

<resources>
    <!--
        TODO: Before you run your application, you need a Google Maps API key.

        To get one, follow this link, follow the directions and press "Create"
        at the end:

        https://console.developers.google.com/flows/enableapi?apiid=maps\_android\_backend&keyType=CLIENT\_SIDE\_ANDROID&r=B2:4A:9D:BD:DA:4E:FA:A0:4A:3F:8E:10:D0:02:B1:73:16:D3:D1:D0%3Bcom.d3ti.a02\_peta

        You can also add your credentials to an existing key, using these
        values:

        Package name:
        B2:4A:9D:BD:DA:4E:FA:A0:4A:3F:8E:10:D0:02:B1:73:16:D3:D1:D0
    -->

```

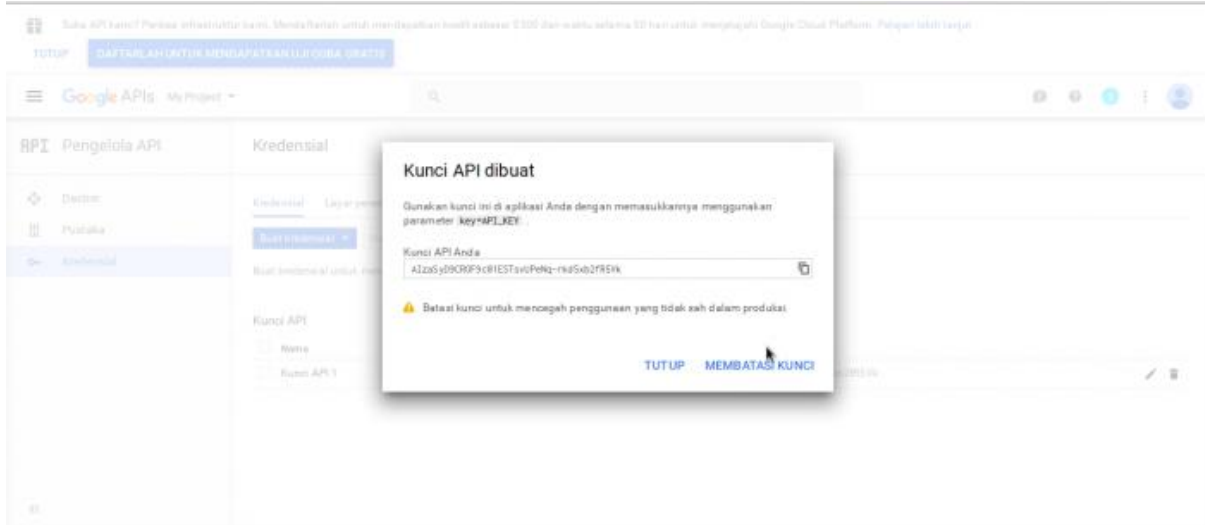
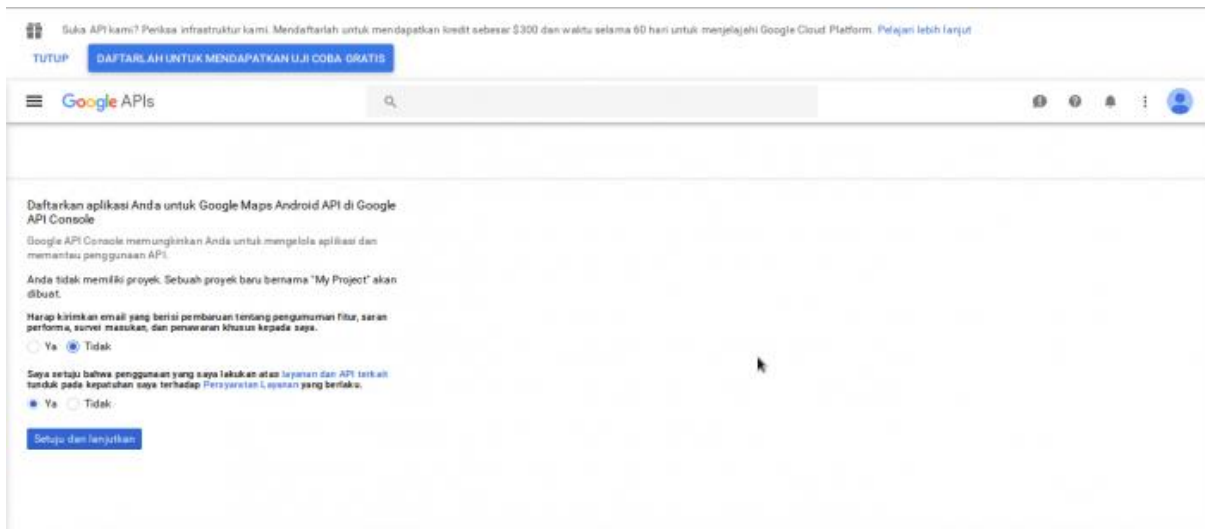
SHA-1 certificate fingerprint:
B2:4A:9D:BD:DA:4E:FA:A0:4A:3F:8E:10:D0:02:B1:73:16:D3:D1:D0

Alternatively, follow the directions here:
<https://developers.google.com/maps/documentation/android/start#get-key>

Once you have your key (it starts with "AIza"), replace the
"google_maps_key"
string in this file.

```
-->  
<string name="google_maps_key" templateMergeStrategy="preserve"  
translatable="false">AIzaSyBkE0xqOYU-fOY8QR507HC7AL3yXnnWND0</string>  
</resources>
```

4. Copy paste console developer api ke dalam url pada web browser sehingga muncul aktif google map api developer



setelah muncul api key masukkan ke dalam google_maps_api.xml

Source Code

```

/* File Main Activity */

package com.d3ti.a02_peta;

import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Build;
import android.support.v4.app.ActivityCompat;
import android.support.v4.app.FragmentActivity;
import android.os.Bundle;
import android.support.v4.content.ContextCompat;
import android.widget.Toast;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.location.LocationListener;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;

public class MapsActivity extends FragmentActivity implements
    OnMapReadyCallback,
        GoogleApiClient.ConnectionCallbacks,
        GoogleApiClient.OnConnectionFailedListener,
        LocationListener {

    private GoogleMap mMap;
    GoogleApiClient mGoogleApiClient;
    Location mLastLocation;
    Marker mCurrLocationMarker;
    LocationRequest mLocationRequest;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        // Obtain the SupportMapFragment and get notified when the map is
        // ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment)
        getSupportFragmentManager()
            .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }

    /**
     * Manipulates the map once available.
     * This callback is triggered when the map is ready to be used.
     * This is where we can add markers or lines, add listeners or move the
     * camera. In this case,
     * we just add a marker near Sydney, Australia.
     */
}

```



```

    * If Google Play services is not installed on the device, the user
    will be prompted to install
    * it inside the SupportMapFragment. This method will only be triggered
    once the user has
    * installed Google Play services and returned to the app.
    */
    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        mMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {
                buildGoogleApiClient();
                mMap.setMyLocationEnabled(true);
            }
        } else {
            buildGoogleApiClient();
            mMap.setMyLocationEnabled(true);
        }

        // Add a marker in Sydney and move the camera
        /* LatLng sydney = new LatLng(-34, 151);
        mMap.addMarker(new MarkerOptions().position(sydney).title("Marker
in Sydney"));
        mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney)); */
    }

    public synchronized void buildGoogleApiClient() {
        mGoogleApiClient = new GoogleApiClient.Builder(this)
            .addConnectionCallbacks(this)
            .addOnConnectionFailedListener(this)
            .addApi(LocationServices.API)
            .build();
        mGoogleApiClient.connect();
    }

    @Override
    public void onLocationChanged(Location location) {
        mLastLocation = location;
        if (mCurrLocationMarker != null) {
            mCurrLocationMarker.remove();
        }

        //Place current location marker
        LatLng latLng = new LatLng(location.getLatitude(),
location.getLongitude());
        MarkerOptions markerOptions = new MarkerOptions();
        markerOptions.position(latLng);
        markerOptions.title("Current Position");

markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFa
ctory.HUE_MAGENTA));
        mCurrLocationMarker = mMap.addMarker(markerOptions);

        //move map camera
        mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
        mMap.animateCamera(CameraUpdateFactory.zoomTo(11));
    }

```

```

        //stop location updates
        if (mGoogleApiClient != null) {
LocationServices.FusedLocationApi.removeLocationUpdates(mGoogleApiClient,
this);
        }
    }

    @Override
    public void onConnected(Bundle bundle) {
        mLocationRequest = new LocationRequest();
        mLocationRequest.setInterval(1000);
        mLocationRequest.setFastestInterval(1000);

mLocationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURA
CY);
        if (ContextCompat.checkSelfPermission(this,
            Manifest.permission.ACCESS_FINE_LOCATION)
            == PackageManager.PERMISSION_GRANTED) {

LocationServices.FusedLocationApi.requestLocationUpdates(mGoogleApiClient,
mLocationRequest, this);
        }

    }

    @Override
    public void onConnectionSuspended(int i) {

    }

    @Override
    public void onConnectionFailed(ConnectionResult connectionResult) {

    }

    public static final int MY_PERMISSIONS_REQUEST_LOCATION = 99;
    public boolean checkLocationPermission() {
        if (ContextCompat.checkSelfPermission(this,
            Manifest.permission.ACCESS_FINE_LOCATION)
            != PackageManager.PERMISSION_GRANTED) {

            // Asking user if explanation is needed
            if (ActivityCompat.shouldShowRequestPermissionRationale(this,
                Manifest.permission.ACCESS_FINE_LOCATION)) {

                // Show an explanation to the user *asynchronously* --
                // don't block
                // this thread waiting for the user's response! After the
                // user
                // sees the explanation, try again to request the
                // permission.

                //Prompt the user once explanation has been shown
                ActivityCompat.requestPermissions(this,
                    new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},
                    MY_PERMISSIONS_REQUEST_LOCATION);
            }
        }
    }

```

```

    } else {
        // No explanation needed, we can request the permission.
        ActivityCompat.requestPermissions(this,
            new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},
            MY_PERMISSIONS_REQUEST_LOCATION);
    }
    return false;
} else {
    return true;
}
}

@Override
public void onRequestPermissionsResult(int requestCode, String
permissions[], int[] grantResults) {
    switch (requestCode) {
        case MY_PERMISSIONS_REQUEST_LOCATION: {
            // If request is cancelled, the result arrays are empty.
            if (grantResults.length > 0
                && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {

                // permission was granted. Do the
                // contacts-related task you need to do.
                if (ContextCompat.checkSelfPermission(this,
                    Manifest.permission.ACCESS_FINE_LOCATION)
                    == PackageManager.PERMISSION_GRANTED) {

                    if (mGoogleApiClient == null) {
                        buildGoogleApiClient();
                    }
                    mMap.setMyLocationEnabled(true);
                }

            } else {

                // Permission denied, Disable the functionality that
                // depends on this permission.
                Toast.makeText(this, "permission denied",
Toast.LENGTH_LONG).show();
            }
            return;
        }

        // other 'case' lines to check for other permissions this app
        // might request.
        // You can add here other case statements according to your
        // requirement.
    }
}

} /* End of Code */

```

E. Latihan/Tugas

Selesaikan tugas coding diatas.

Materi 13: Android-PHP

A. Waktu :

Alokasi Waktu : 2 x 100 menit

- Praktikum Terbimbing: 1 x 100 menit
- Praktikum Mandiri: 1 x 100 menit

B. Tujuan Pembelajaran:

1. Mahasiswa dapat membuat aplikasi sederhana menggunakan client-server

C. Materi:

(Isikan materi secukupnya sehingga mudah diketahui oleh peserta)

Ketentuan :

- Sajikan materi secukupnya, dan semudahnya, jangan terlalu detail/dalam mengingat peserta masih baru dalam PTIK
- Gambar ilustrasi disajikan seperlunya.
- Jumlah halaman tiap materi maksimal 5 halaman (sudah termasuk latihan)

D. Praktikum

E. Latihan/Tugas