BAB I
PENDAHULUAN

1.1 Tujuan Pembelajaran :

1. Mahasiswa mengenal *Android.*
2. Mahasiswa mengenal *Android Studio* sebagai *IDE (Integrated Development Environment)* resmi dari *Google* untuk *Android.*

1.2 *Software* yang dibutuhkan :

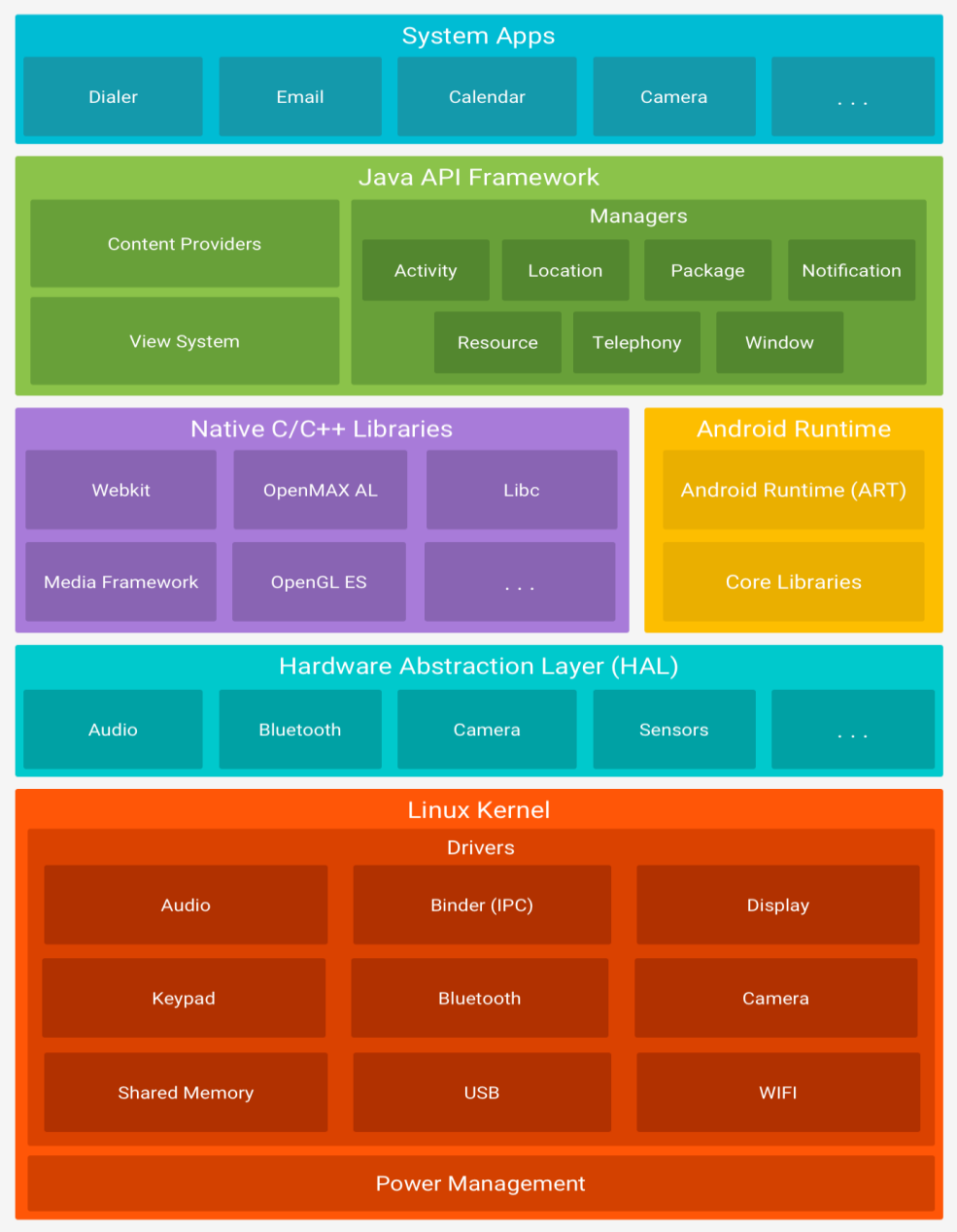
1. *Java JDK*
2. *Android Studio 3.5*
3. *SDK API 29*

1.3 Mengenal Android

Android adalah sistem operasi berbasis *Linux* yang dirancang untuk perangkat bergerak layar sentuh seperti telepon pintar dan komputer *tablet*. *Android* awalnya dikembangkan oleh *Android, Inc.*, dengan dukungan finansial dari *Google*, yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya *Open Handset Alliance*, konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler. Ponsel *Android* pertama mulai dijual pada bulan Oktober 2008.

1.4 Arsitektur *Platform*

Android adalah tumpukan perangkat lunak berbasis *Linux* sumber terbuka yang dibuat untuk berbagai perangkat dan faktor bentuk. *Diagram* pada Gambar 1.1 menunjukkan komponen besar dari platform Android.



Gambar 1.1 Arsitektur perangkat lunak Android.

1.4.1 *Linux Kernel*

Fondasi *platform Android* adalah *kernel Linux*. Sebagai contoh, *Android Runtime (ART)* bergantung pada *kernel Linux* untuk fungsionalitas dasar seperti *threading* dan pengelolaan memori tingkat rendah. Menggunakan *kernel Linux* memungkinkan *Android* untuk memanfaatkan fitur keamanan inti dan memungkinkan produsen perangkat untuk mengembangkan *driver* perangkat keras untuk *kernel* yang cukup dikenal.

1.4.2 *Hardware Abstraction Layer (HAL)*

Hardware Abstraction Layer (HAL) memberikan antarmuka standar yang mengungkap kemampuan perangkat keras perangkat ke kerangka kerja API Java yang lebih tinggi. *HAL* terdiri atas beberapa modul pustaka, masing-masing menerapkan antarmuka untuk komponen perangkat keras tertentu, seperti modul kamera atau bluetooth. Ketika *API* kerangka kerja melakukan panggilan untuk mengakses perangkat keras, sistem *Android* memuat modul pustaka untuk komponen perangkat keras tersebut.

1.4.3 *Android Runtime*

Untuk perangkat yang menjalankan *Android versi 5.0 (API level 21)* atau lebih tinggi, setiap aplikasi menjalankan proses masing-masing dengan tahap *Android Runtime (ART). ART* ditulis guna menjalankan beberapa mesin virtual pada perangkat bermemori rendah dengan mengeksekusi file *DEX*, format *bytecode* yang dirancang khusus untuk *Android* yang dioptimalkan untuk *footprint* memori minimal. Buat rantai aplikasi, misalnya *Jack*, mengumpulkan sumber *Java* ke *bytecode DEX*, yang dapat berjalan pada *platform Android*.

Beberapa fitur utama *ART* mencakup:

a. Kompilasi mendahului waktu (*AOT*) dan tepat waktu (*JIT*)
b. Pengumpulan sampah (*GC*) yang dioptimalkan
c. Di *Android 9 (API level 28)* dan yang lebih tinggi, konversi file format *Dalvik Executable (DEX)* paket aplikasi menjadi kode mesin yang lebih ringkas.
d. Dukungan *debugging* yang lebih baik, mencakup profiler penyampelan terpisah, pengecualian diagnostik mendetail dan laporan kerusakan dan kemampuan untuk mengatur titik pantau guna memantau bidang tertentu.

Sebelum ke *Android* versi 5.0 *(API level 21), Dalvik* adalah waktu proses *Android*. Jika aplikasi Anda berjalan baik pada *ART*, semestinya berfungsi baik juga pada *Dalvik*, tetapi mungkin tidak sebaliknya. Android juga menyertakan serangkaian pustaka waktu proses inti yang menyediakan sebagian besar fungsi bahasa pemrograman *Java*, termasuk beberapa fitur bahasa *Java 8*, yang digunakan kerangka kerja *API Java*.

1.4.4 Pustaka C/C++ Bawaan

Banyak komponen dan layanan sistem *Android* inti seperti *ART* dan *HAL* dibuat dari kode bawaan yang memerlukan pustaka bawaan yang tertulis dalam *C* dan *C++*. *Platform Android* memungkinkan kerangka kerja *API Java* meningkatkan fungsi beberapa pustaka bawaan pada aplikasi. Misalnya, Anda dapat mengakses *OpenGL ES* melalui kerangka kerja *API OpenGL Java Android* guna menambahkan dukungan untuk menggambar dan memanipulasi grafik *2D* dan *3D* pada aplikasi Anda. Jika Anda mengembangkan aplikasi yang memerlukan kode *C* atau *C++*, Anda dapat menggunakan *Android NDK* untuk mengakses beberapa pustaka *platform* bawaan langsung dari kode asal.

1.4.5 Kerangka Kerja *API Java*

Keseluruhan rangkaian fitur pada *Android OS* tersedia untuk Anda melalui *API* yang ditulis dalam bahasa *Java*. *API* ini membentuk elemen dasar yang harus Anda buat aplikasi *Android* dengan menyederhanakan penggunaan ulang inti, komponen dan layanan sistem modular, yang mencakup berikut ini:

a. Tampilan Sistem yang kaya dan luas dapat Anda gunakan untuk membuat *UI* aplikasi, termasuk daftar, kisi, kotak teks, tombol, dan bahkan browser web yang dapat disematkan
b. Pengelola Sumber Daya, memberikan akses ke sumber daya bukan kode seperti *string* yang dilokalkan, grafik, dan *file layout*.
c. Pengelola Notifikasi yang mengaktifkan semua aplikasi guna menampilkan lansiran khusus pada bilah status.

d. Pengelola Aktivitas yang mengelola siklus hidup aplikasi dan memberikan back-stack navigasi yang umum.
e. Penyedia Materi yang memungkinkan aplikasi mengakses data dari aplikasi lainnya, seperti aplikasi Kontak, atau untuk berbagi data milik sendiri.
f. Developer memiliki akses penuh ke *API* kerangka kerja yang digunakan oleh aplikasi sistem *Android*.

1.4.6 Aplikasi Sistem

Android dilengkapi dengan serangkaian aplikasi inti untuk *email*, perpesanan SMS, kalender, menjelajahi *internet*, kontak, dll. Aplikasi yang disertai dengan *platform* tidak memiliki status khusus pada aplikasi yang pengguna ingin instal. Jadi, aplikasi pihak ketiga dapat menjadi *browser web* utama, pengolah pesan *SMS* atau bahkan *keyboard* utama (beberapa pengecualian berlaku, seperti aplikasi *Settings* sistem). Aplikasi sistem berfungsi sebagai aplikasi untuk pengguna dan memberikan kemampuan kunci yang dapat diakses oleh *developer* dari aplikasi mereka sendiri. Misalnya, jika aplikasi Anda ingin mengirimkan pesan *SMS*, Anda tidak perlu membangun fungsi tersebut sendiri. Anda bisa juga menjalankan aplikasi *SMS* mana saja yang telah diinstal guna mengirimkan pesan kepada penerima yang Anda cantumkan.

1.5 Mengenal *Android Studio*

Android Studio adalah Lingkungan Pengembangan Terpadu (*Integrated Development Environment/IDE*) resmi untuk pengembangan aplikasi *Android,* yang didasarkan pada *IntelliJ IDEA*. Selain sebagai editor kode dan fitur *developer IntelliJ* yang andal, *Android Studio* menyediakan banyak fitur yang meningkatkan produktivitas dalam membuat aplikasi *Android*, seperti:

a. Sistem *build* berbasis *Gradle* yang fleksibel.
b. *Emulator* yang cepat dan kaya fitur.
c. Lingkungan terpadu tempat sehingga bisa mengembangkan aplikasi untuk semua perangkat *Android.*

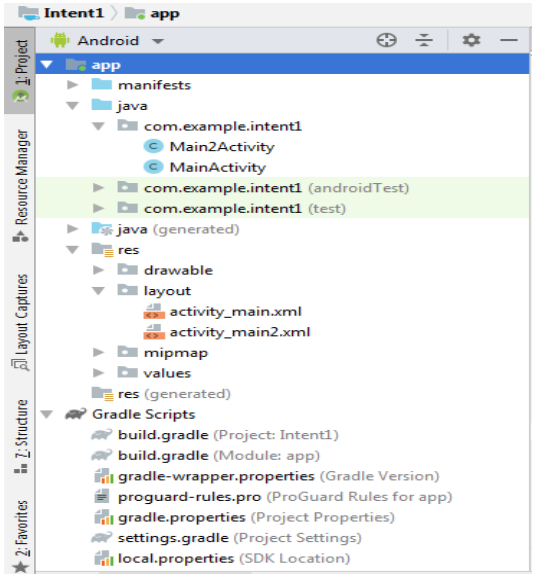
d. Terapkan Perubahan untuk melakukan *push* pada perubahan kode dan *resource* ke aplikasi yang sedang berjalan tanpa memulai ulang aplikasi
e. *Template* kode dan integrasi *GitHub* untuk membantu Anda membuat fitur aplikasi umum dan mengimpor kode sampel
f. *Framework* dan fitur pengujian yang lengkap
g. Fitur *lint* untuk merekam performa, kegunaan, kompatibilitas versi, dan masalah lainnya
h. Dukungan *C++* dan *NDK*
i. Dukungan bawaan untuk *Google Cloud Platform*, yang memudahkan integrasi *Google Cloud Messaging* dan *App Engine.*

1.5.1 Struktur *project*

Setiap *project* di *Android Studio* berisi satu atau beberapa modul dengan file kode sumber dan file *resource*. Jenis modul meliputi:

a. Modul aplikasi *Android*
b. Modul *library*
c. Modul *Google App Engine*

Secara *default*, *Android Studio* menampilkan *file project* dalam tampilan *project Android*, seperti yang ditunjukkan pada Gambar 1.2, yang disusun menurut modul untuk memberikan akses cepat ke file sumber utama project yang telah dibuat.

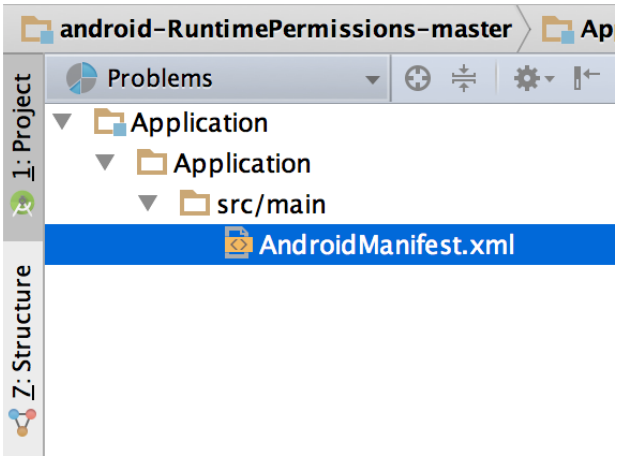


Gambar 1.2 Tampilan *file project* pada *Android Studio 3.5*

Semua *file build* terlihat di tingkat teratas di bagian *Gradle Script* dan setiap modul aplikasi berisi folder berikut:

a. *manifests:* Berisi *file AndroidManifest.xml*.
b. *java*: Berisi file kode sumber *Java*, termasuk kode pengujian *JUnit*.
c. *res*: Berisi semua *resource* non-kode, seperti tata letak *XML*, string *UI*, dan gambar *bitmap*.

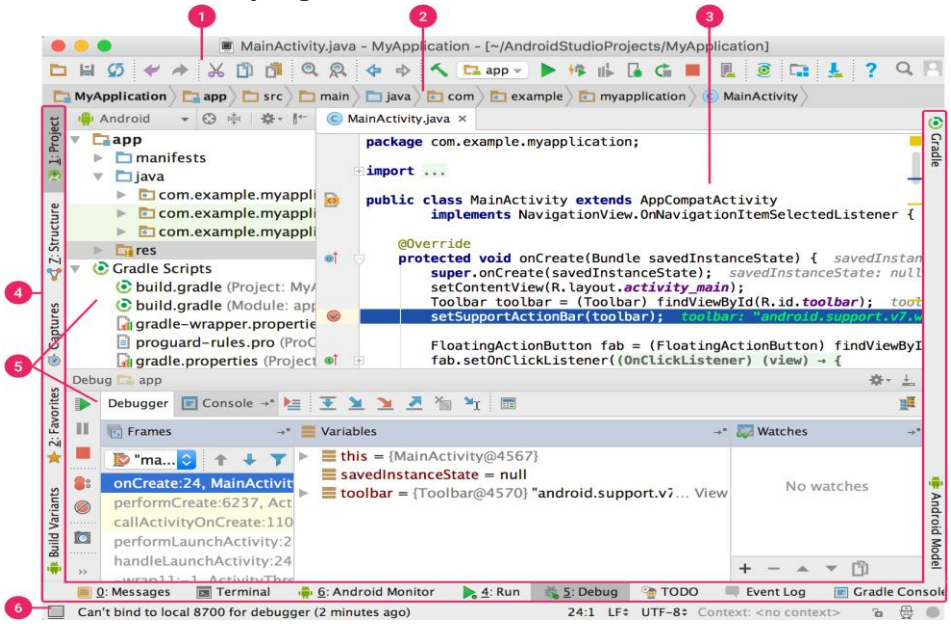
Untuk melihat struktur *file project* sebenarnya, pilih *Project* dari menu drop-down ***Project*** (pada Gambar 1.1 , ditampilkan sebagai **Android**). Praktikan juga dapat menyesuaikan tampilan *file project* untuk berfokus pada aspek spesifik dari pengembangan aplikasi yang telah dibuat. Misalnya, memilih tampilan *Problems* pada *project* yang telah dibuat akan menampilkan *link* ke file sumber yang berisi *error coding* dan *sintaks* yang dikenali, seperti *tag* penutup elemen *XML* yang tidak ada dalam *file* tata letak, seperti ditunjukkan Gambar 1.3.



Gambar 1.3 *File project* dalam tampilan *Problems*

1.5.2 Antar Muka Pengguna (IDE)

Jendela utama *Android Studio* terdiri dari beberapa area logis yang diidentifikasi dalam Gambar 1.4.



Gambar 1.4 Jendela utama *Android Studio*

1. *Toolbar*, memungkinkan praktikan melakukan berbagai tindakan, termasuk menjalankan aplikasi dan meluncurkan fitur *Android*.
2. *Menu navigasi*, membantu praktikan menjelajah *project* dan membuka *file* untuk diedit. Menu ini memberikan tampilan struktur yang lebih ringkas yang terlihat di jendela *Project*.
3. *Jendela editor*, adalah tempat membuat dan memodifikasi kode. Tergantung jenis file yang ada, editor ini dapat berubah. Misalnya, saat menampilkan file tata letak, editor akan menampilkan *Layout Editor*.
4. *Panel jendela fitur*, berada di sisi luar jendela *IDE* dan berisi tombol-tombol yang memungkinkan Anda memperluas atau menciutkan setiap jendela fitur.
5. *Jendela fitur*, memberi akses ke tugas tertentu seperti pengelolaan *project*, penelusuran, kontrol versi, dan banyak lagi. Jendela ini dapat diperluas dan diciutkan jendela ini.
6. *Status bar*, menampilkan status *project* dan *IDE* itu sendiri, serta semua peringatan atau pesan.

1.5.3 Struktur *project Android*

Untuk melihat struktur *file* sesungguhnya dari suatu *project* termasuk semua file yang disembunyikan dari tampilan *Android*, pilih *Project* dari menu *drop-down* di bagian atas jendela *Project*. Jika memilih tampilan *Project*, maka akan dapat melihat banyak *file* dan direktori lainnya. Yang terpenting di antaranya adalah sebagai berikut:

`module-name/`

`build/`

Berisi *output build*.

`libs/`

Berisi *library* pribadi.

`src/`

Berisi semua *file* kode dan *resource* untuk modul dalam sub-direktori berikut:

`androidTest/`

Berisi kode untuk pengujian instrumentasi yang dijalankan pada perangkat *Android*.

`main/`

Berisi *file set* sumber *"main"*: kode dan *resource Android* yang digunakan bersama oleh semua varian *build* (*file* untuk varian *build* lain berada dalam direktori, seperti `src/debug/` untuk jenis *build debug*).

`AndroidManifest.xml`

Menjelaskan sifat aplikasi dan masing-masing komponennya.

`java/`

Berisi sumber kode *Java*.

`jni/`

Berisi kode *native* yang menggunakan *Java Native Interface (JNI)*.

`gen/`

Berisi *file Java* yang dihasilkan oleh *Android Studio*, seperti *file* `R.java` dan antarmuka yang dibuat dari *file AIDL*.

`res/`

Berisi *resource* aplikasi, seperti *file* yang dapat digambar, *file* tata letak, dan *string UI*.

`assets/`

Berisi *file* yang harus dikompilasi menjadi sebuah *file* `.apk` apa adanya. Direktori ini dapat dibuka dengan cara yang sama seperti sistem *file* umumnya dengan menggunakan *URI* dan membaca *file* sebagai aliran *byte* menggunakan `AssetManager` .

`test/`

Berisi kode untuk pengujian lokal yang dijalankan pada *JVM host*.

`build.gradle` **(modul)**

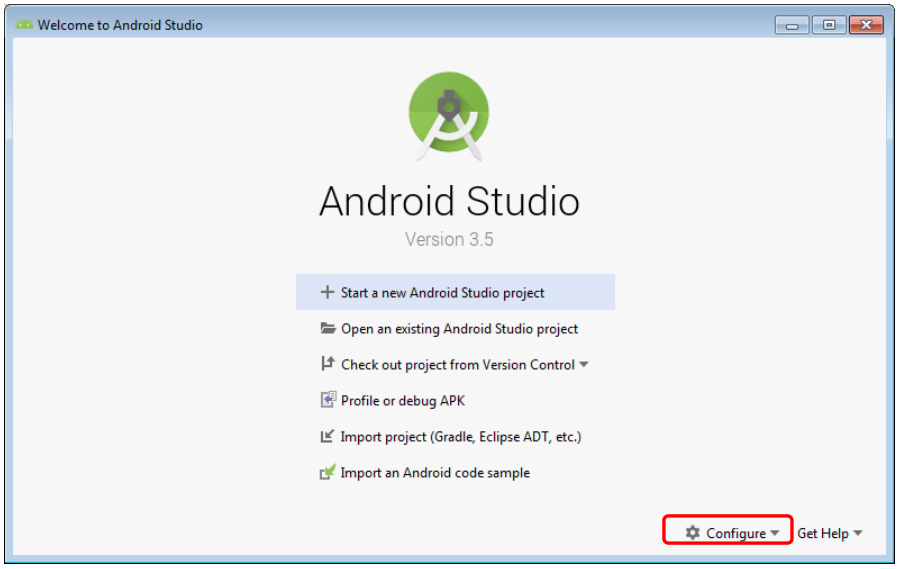
Ini mendefinisikan konfigurasi *build* spesifik modul.

`build.gradle` **(project)**

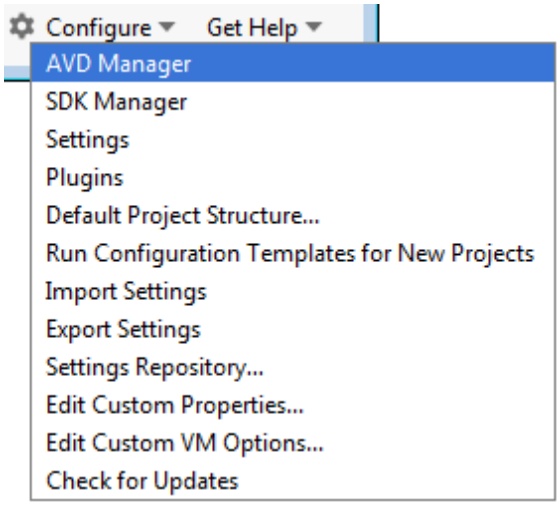
File ini mendefinisikan konfigurasi *build* yang berlaku untuk semua modul. File ini integral bagi *project*, jadi harus disimpan dalam kontrol revisi bersama semua kode sumber lainnya.

1.5.4 *Android Virtual Device (AVD)*

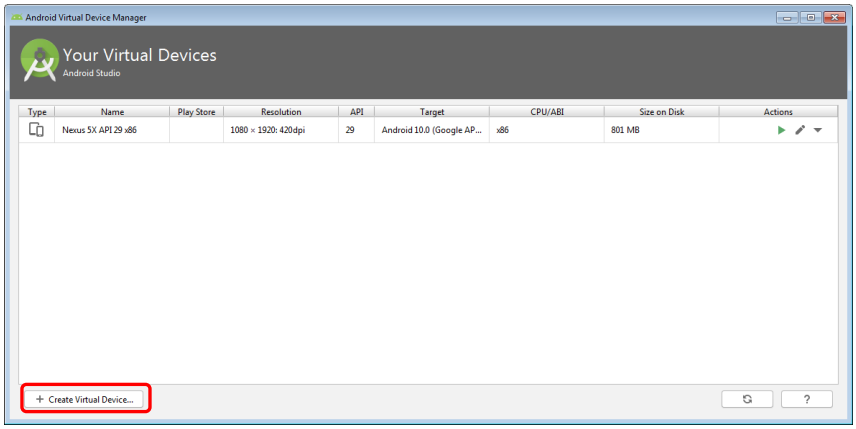
Android Virtual Device (AVD) adalah konfigurasi yang mendefinisikan karakteristik ponsel *Android, tablet, Wear OS, Android TV*, atau perangkat *Automotive OS* yang ingin disimulasikan dalam *Android Emulator*. Dengan *AVD* ini, praktikan bisa mengembangkan dan mencoba aplikasi *Android* tanpa harus menggunakan perangkat *Android* yang sebenarnya. Praktikan bisa menentukan karakteristik *AVD*, misalkan menentukan versi *Android*, jenis dan ukuran layar, besarnya memori, dan lain sebagainya. *AVD* dapat dibuat sebanyak yang diinginkan. Untuk dapat menggunakan *AVD* ini, *Android Studio* telah menyediakan fitur *AVD Manager*. *AVD Manager* adalah antarmuka yang dapat diluncurkan dari *Android Studio* yang dapat membantu membuat dan mengelola *AVD*. Cara penggunaannya pada Jendela utama *Android Studio* seperti adalah sebagai berikut.



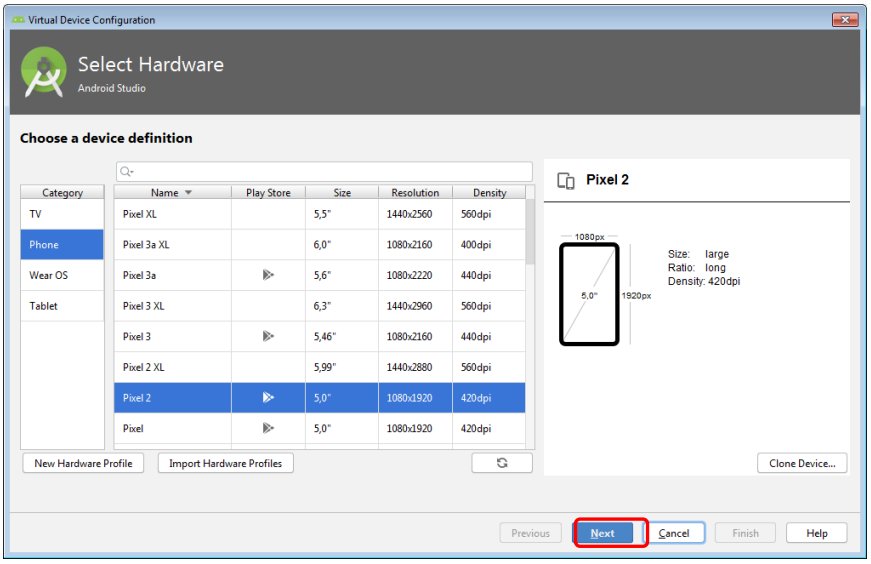
Gambar 1.5 Jendela Utama *Android Studio 3.5*



Gambar 1.6 Menu *Configure* pada *Android Studio 3.5*



Gambar 1.7 Jendela utama *Android Virtual Device Manager*



Gambar 1.8 Jendela Konfigurasi *Virtual Device*

BAB II
INTENT

2.1 Tujuan Pembelajaran :

1. Mahasiswa mengetahui konsep *Intent.*
2. Mahasiswa dapat membuat *Intent.*

2.2 *Software* yang dibutuhkan :

1. *Java JDK*
2. *Android Studio 3.5*
3. *SDK API 29*

2.3 Intent

Intent adalah mekanisme untuk melakukan sebuah aksi dan komunikasi antar komponen aplikasi pada *platform Android*. Atau dapat juga dijelaskan sebagai sebuah *object* yang memungkinkan kita untuk dapat berkomunikasi antara *Activity* satu dengan *Activity* lainnya. Ada tiga penggunaan umum *Intent* dalam aplikasi *Android* yaitu :

1. Memindahkan satu *Activity* ke *Activity* lain dengan atau tidak membawa data.
2. Menjalankan *background Service*, misal melakukan sinkronisasi ke *server* dan menjalankan proses berulang (*Periodic / Scheduler Task*).
3. Mengirimkan objek *Broadcast* ke aplikasi yang membutuhkan. Misalkan jika aplikasi membutuhkan proses menjalankan sebuah *Background Service* setiap aplikasi melakukan *Booting*. Aplikasi harus bisa menerima objek *Broadcast* yang dikirimkan oleh *Android* untuk *event Booting* tersebut.

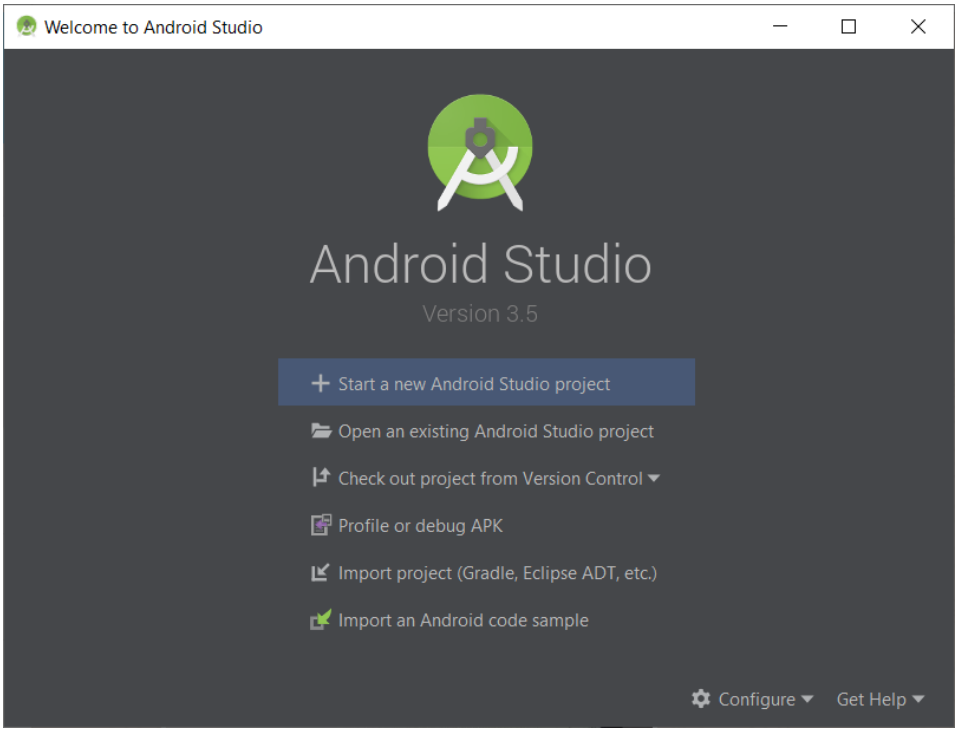
Intent memiliki dua bentuk, yaitu :

1. *Explicit Intent*, adalah tipe *intent* yang digunakan untuk menjalankan komponen aplikasi dengan tahu detail dari nama kelas yang dituju misal : com.praktik.activity.DetailActivity. Umumnya intent ini digunakan untuk mengaktifkan komponen pada aplikasi yang sama.

2. *Implicit Intent*, adalah tipe *intent* yang tidak memerlukan detail nama kelas yang ingin diaktifkan. Ini memungkinkan komponen dari aplikasi lain bisa merespon *request intent* yang dijalankan. Penggunaan tipe *intent* ini umumnya diperuntukkn guna menjalankan fitur / fungsi dari komponen aplikasi lain. Contohnya, ketika kita membutuhkan aplikasi kita untuk mengambil foto , daripada kita harus membuat sendiri fungsi kamera lebih baik kita menyerahkan proses tersebut pada aplikasi kamera bawaan dari *device* atau aplikasi kamera lain yang telah terinstal sebelumnya pada *device*. Atau jika kita membutuhkan fungsi berbagi konten, kita dapat memanfaatkan *intent* untuk menampilkan mana saja aplikasi yang bisa menawarkan fungsi berbagi *(share)* konten.

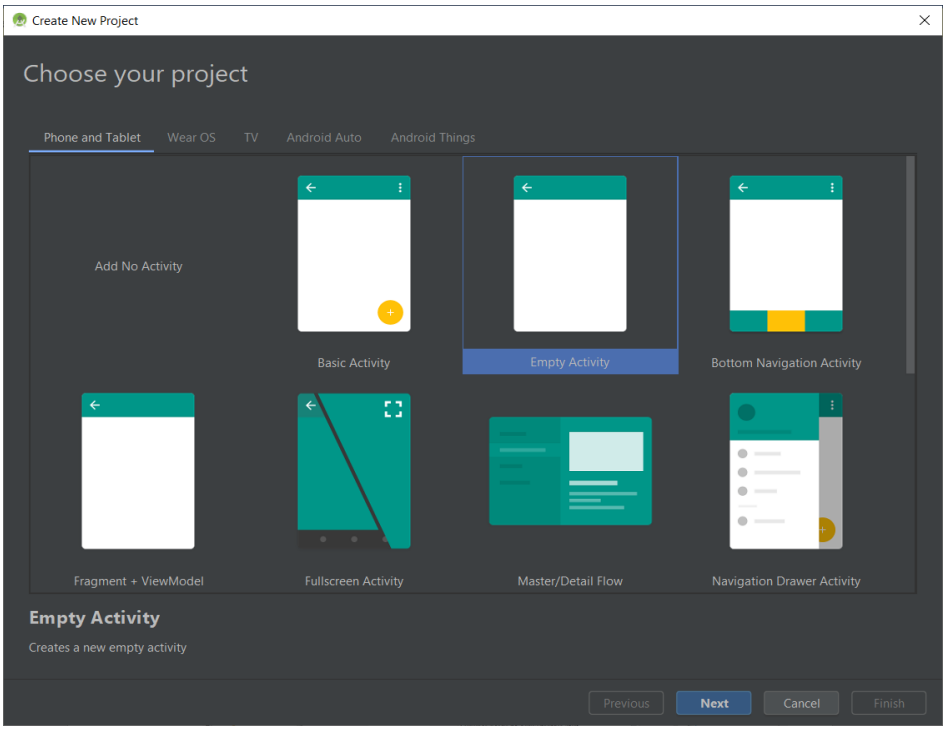
2.4 Langkah – Langkah Praktikum

1. Membuat *project* baru pada *Android Studio 3.5*, dengan memilih *Start a new Android Studio Project* seperti Gambar 2.1.



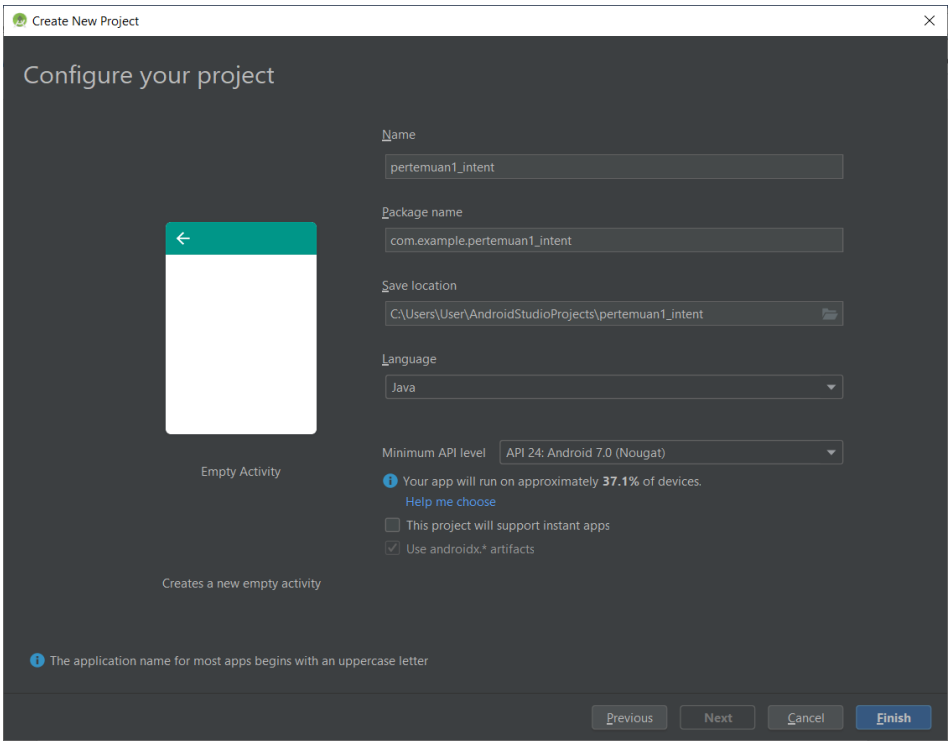
Gambar 2.1 *Start a new Android Studio Project*

2. Praktikan memilih *Empty Activity* pada *Tab Phone and Tablet*, seperti Gambar 2.2.



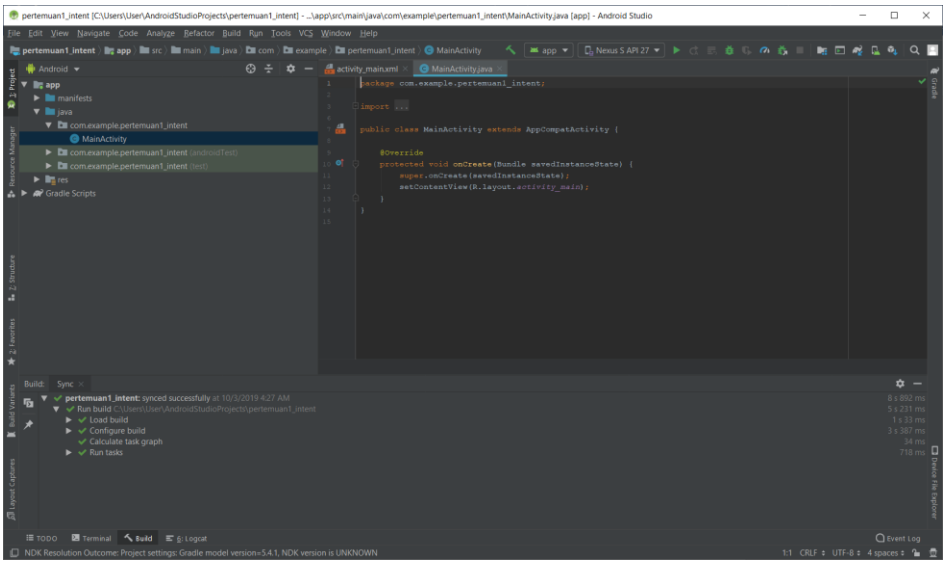
Gambar 2.2 *Empty Activity* pada *Tab Phone and Tablet*

3. Praktikan mengisi identitas *project*, seperti Gambar 2.2



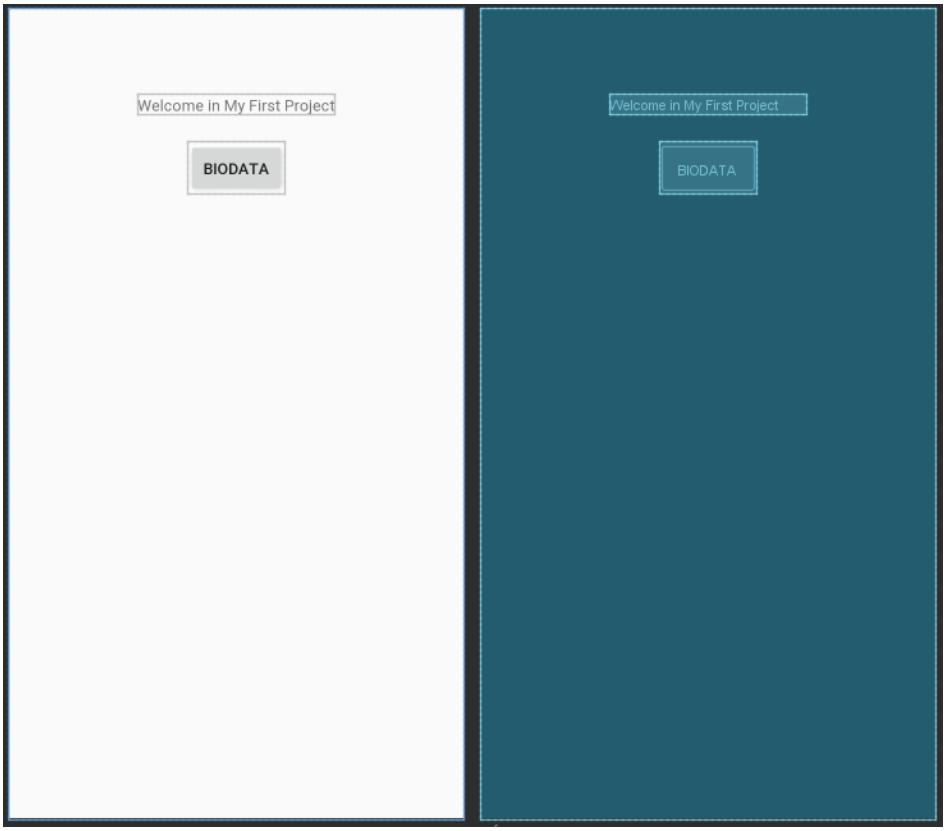
Gambar 2.3 *Empty Activity* pada *Tab Phone and Tablet*

4. Praktikan akan diantarkan pada *Window IDE Android Studio 3.5*, seperti Gambar 2.4.



Gambar 2.4 *Window IDE Android Studio 3.5*

5. Praktikan membuat tampilan activity_main.xml seperti gambar 2.5.

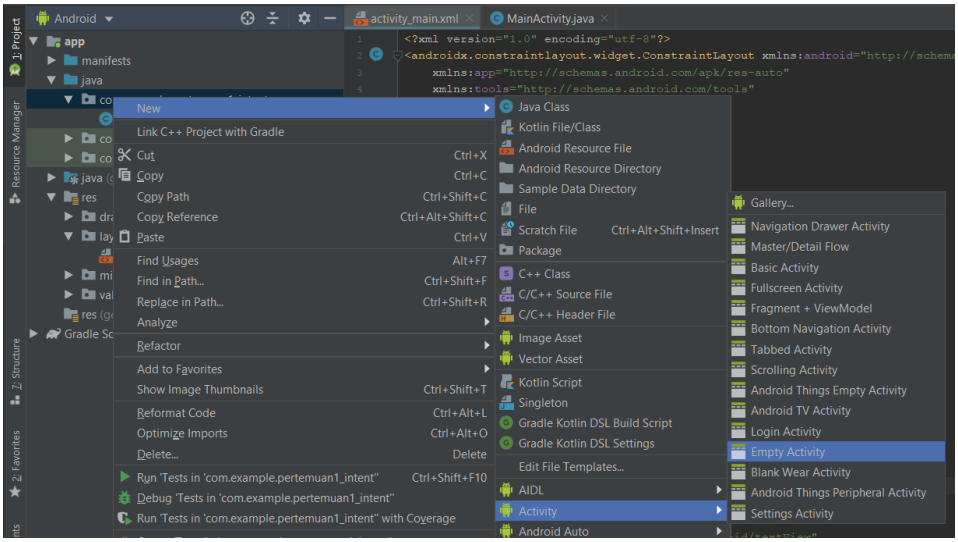


Gambar 2.5 *Design dan Blueprint activity_main.xml*

Source code activity_main.xml :

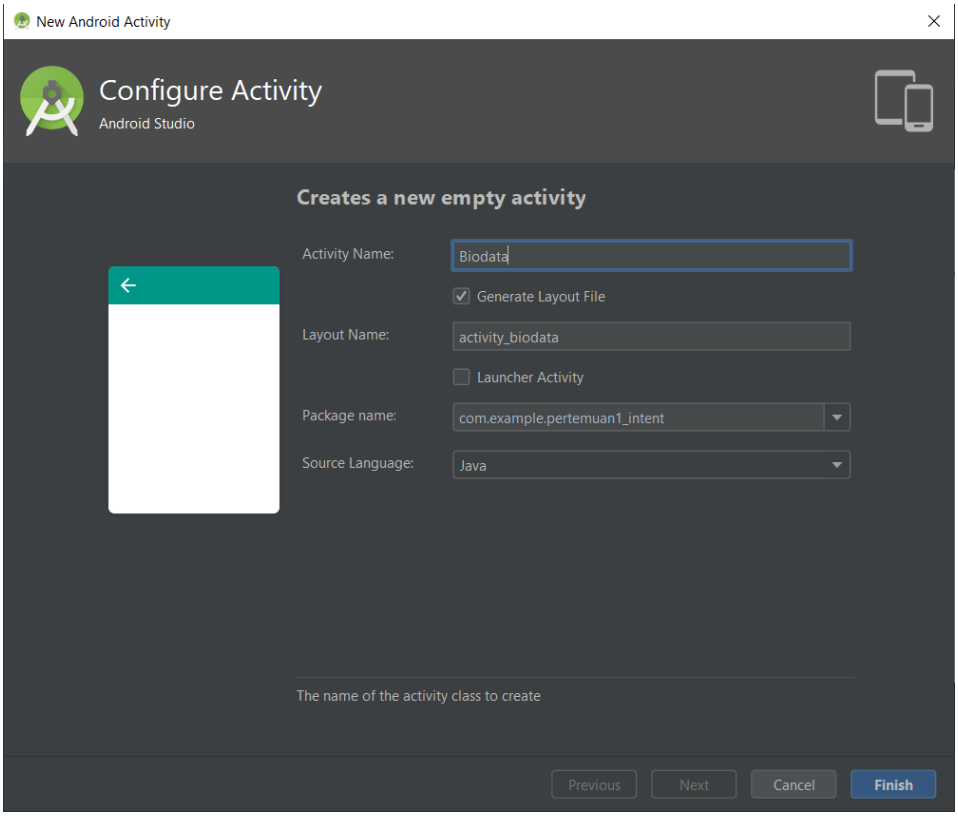
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"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

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/welcome_in_my_first_project"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.108" />

    <Button
        android:id="@+id/btn_biodata"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/biodata"
        android:onClick="biodata"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        app:layout_constraintVertical_bias="0.04" />



</androidx.constraintlayout.widget.ConstraintLayout>
```

6. Praktikan membuat *Activity* baru dengan cara pilih *Tab Project* (pada *window* sebelah kiri) >> letakkan kursor pada nama *package* yang telah dibuat >> klik kanan >> pilih *New* >> pilih *Activity* >> pilih *Empty Activity*, seperti Gambar 2.6.



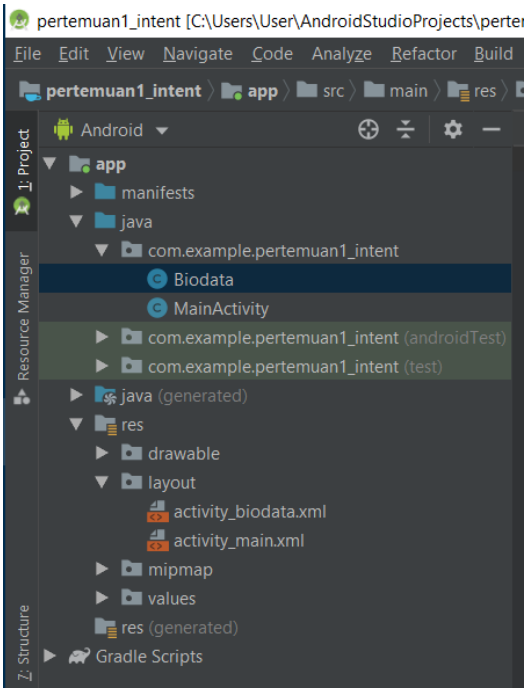
Gambar 2.5 Membuat *empty activity* baru

7. Praktikan mengisi identitas *activity*, seperti Gambar 2.6



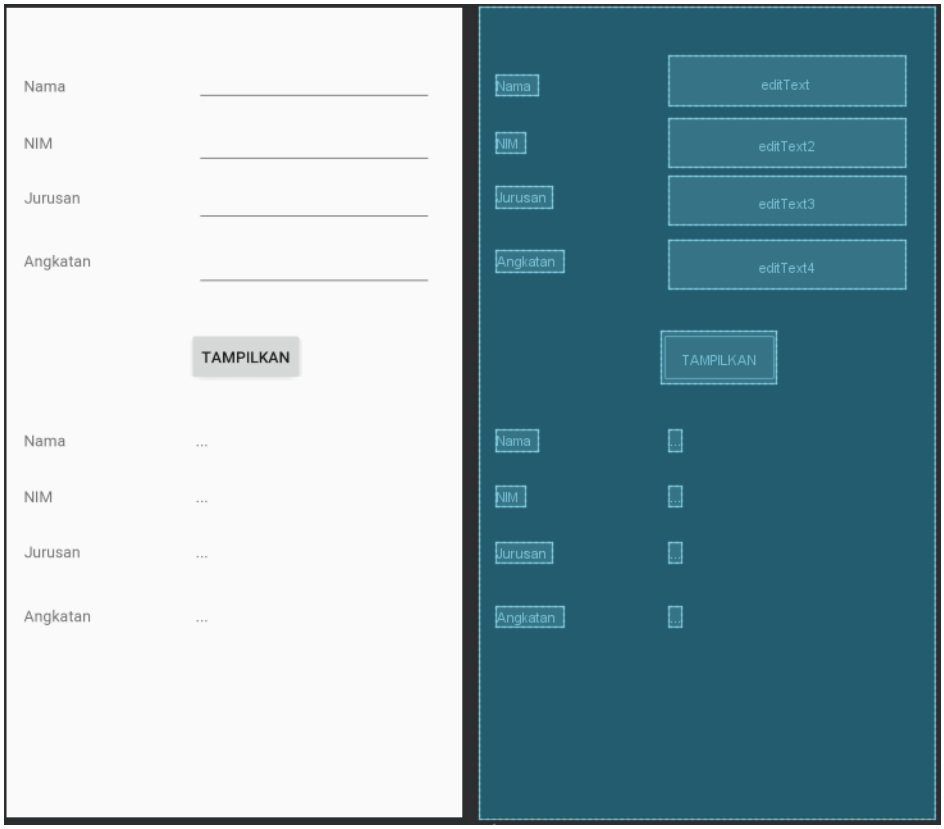
Gambar 2.6 *Configure Activity*

8. Tampilan Tab Project berisi *activity* Biodata yang praktikan buat.



Gambar 2.7 *Tab Project* setelah dibuat *activity* baru

9. Praktikan membuat tampilan activity_biodata.xml seperti gambar 2.8.



Gambar 2.8 *Design* dan *Blueprint* activity_biodata.xml

Source code activity_biodata.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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=".Biodata">

    <TextView
        android:id="@+id/output_angkatan"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="40dp"
        android:text="...."
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toEndOf="@+id/textView17"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.757" />

    <TextView
        android:id="@+id/output_jurusan"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="48dp"
        android:text="...."
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toEndOf="@+id/textView18"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.676" />

    <TextView
        android:id="@+id/output_nim"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="72dp"
        android:text="...."
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toEndOf="@+id/textView16"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.606" />

    <TextView
        android:id="@+id/textView15"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Nama"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.042"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.535" />

    <TextView
        android:id="@+id/textView17"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Angkatan"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.045"
        app:layout_constraintStart_toStartOf="parent"
```

```
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.757" />

    <TextView
        android:id="@+id/textView18"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Jurusan"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.044"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.676" />

    <TextView
        android:id="@+id/textView16"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="NIM"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.041"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.606" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Nama"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.042"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.085" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="NIM"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.041"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.157" />

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Angkatan"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.045"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.307" />

    <TextView
        android:id="@+id/textView4"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Jurusan"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.044"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.227" />

    <EditText
        android:id="@+id/input_nama"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.863"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.062" />

    <EditText
        android:id="@+id/input_jurusan"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.863"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.221" />

    <EditText
        android:id="@+id/input_nim"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.863"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.145" />

    <EditText
        android:id="@+id/input_angkatan"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.863"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.306" />

    <Button
        android:id="@+id/btn_tampil"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:text="tampilkan"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.534"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.426" />

    <TextView
        android:id="@+id/output_nama"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="60dp"
        android:text="...."
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toEndOf="@+id/textView15"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.535" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

10. *Source code* MainActivity.java

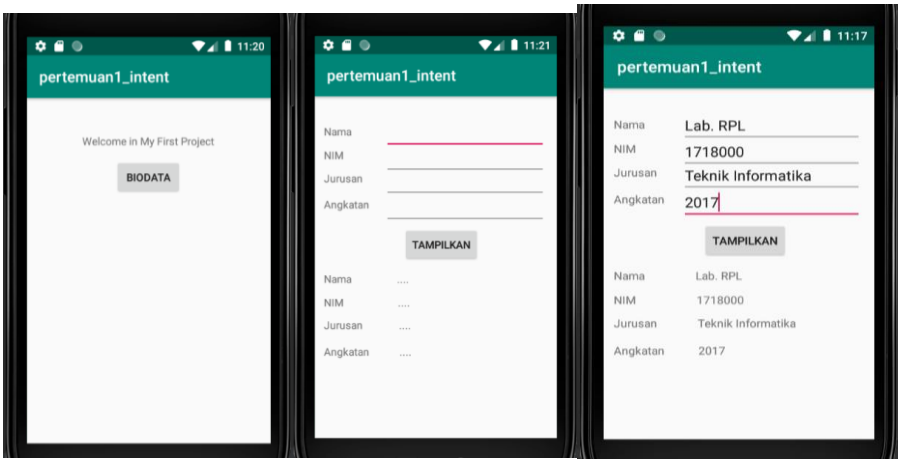
```
package com.example.pertemuan1_intent;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;

public class MainActivity extends AppCompatActivity {

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

    public void biodata(View view){
        Intent a = new Intent(MainActivity.this,
Biodata.class);
        startActivity(a);
    }
}
```

11. *Source code* Biodata.java

```
package com.example.pertemuan1_intent;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class Biodata extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_biodata);

        final EditText nama =
(EditText)findViewById(R.id.input_nama);
        final EditText nim =
(EditText)findViewById(R.id.input_nim);
        final EditText jurusan =
(EditText)findViewById(R.id.input_jurusan);
        final EditText angkatan =
(EditText)findViewById(R.id.input_angkatan);

        final TextView nama_t =
(TextView)findViewById(R.id.output_nama);
        final TextView nim_t =
(TextView)findViewById(R.id.output_nim);
        final TextView jurusan_t =
(TextView)findViewById(R.id.output_jurusan);
        final TextView angkatan_t =
(TextView)findViewById(R.id.output_angkatan);

        final Button tampil =
(Button)findViewById(R.id.btn_tampil);
        tampil.setOnClickListener(new View.OnClickListener(){
            @Override
            public void onClick(View v) {
                String nama1 = nama.getText().toString();
                String nim1 = nim.getText().toString();
                String jurusan1 = jurusan.getText().toString();
                String angkatan1 =
angkatan.getText().toString();

                nama_t.setText(nama1);
                nim_t.setText(nim1);
                jurusan_t.setText(jurusan1);
                angkatan_t.setText(angkatan1);
            }
        });
    }
}
```

12. Tampilan Hasil



BAB III
FRAGMENT

3.1 Tujuan Pembelajaran

1. Mahasiswa mengetahui konsep *fragment.*
2. Mahasiswa dapat membuat *fragment* ke dalam aplikasi.
3. Mahasiswa dapat mengirimkan data antar *fragment*.
4. Mahasiswa dapat menambahkan *fragment* sebagai *dialog*.

3.2 *Software* yang dibutuhkan

1. *Java JDK*
2. *Android Studio 3.5*
3. *SDK API 29*

3.2 Fragment

1. *Fragment* merupakan komponen yang memiliki fungsi untuk menampilkan antar muka ke pengguna melalui *Activity* dengan memiliki *layout xml* sendiri.
2. *Fragment* memiliki daur hidup sendiri dan bergantung penuh pada daur hidup *Activity* dimana ia ditanamkan.
3. Penggunaan *fragment* lebih kepada pemecahan komponen tampilan aplikasi untuk menjadi fleksibel dan dapat digunakan kembali *(reusable).*
4. *Fragment* adalah sebuah *reuseable class* yang mengimplement beberapa fitur sebuah *Activity*.
5. Satu *activity* bisa ditempeli lebih dari satu *fragment*.
6. Tidak seperti *Activity*, *fragment* tidak perlu didaftarkan ke dalam *file AndroidManifest.xml*.
7. Satu kelas java dinyatakan sebuah *fragment* ketika kelas tersebut meng-*extends* (*inherit*) kelas *fragment*.
8. Melalui *Android support library*, *fragment* dapat kompatibel sampai *Android API* level 10 *Gingerbread.*

9. Analogi yang mendekati dengan *fragment* pada *platform* lain adalah penggunaan komponen *frame* pada aplikasi berbasis *web.*

Ada beberapa *state* yang perlu diketahui sebelum menggunakan *fragment* :

a. *Resumed*

Fragment dapat dilihat ketika *Activity* sedang berjalan.

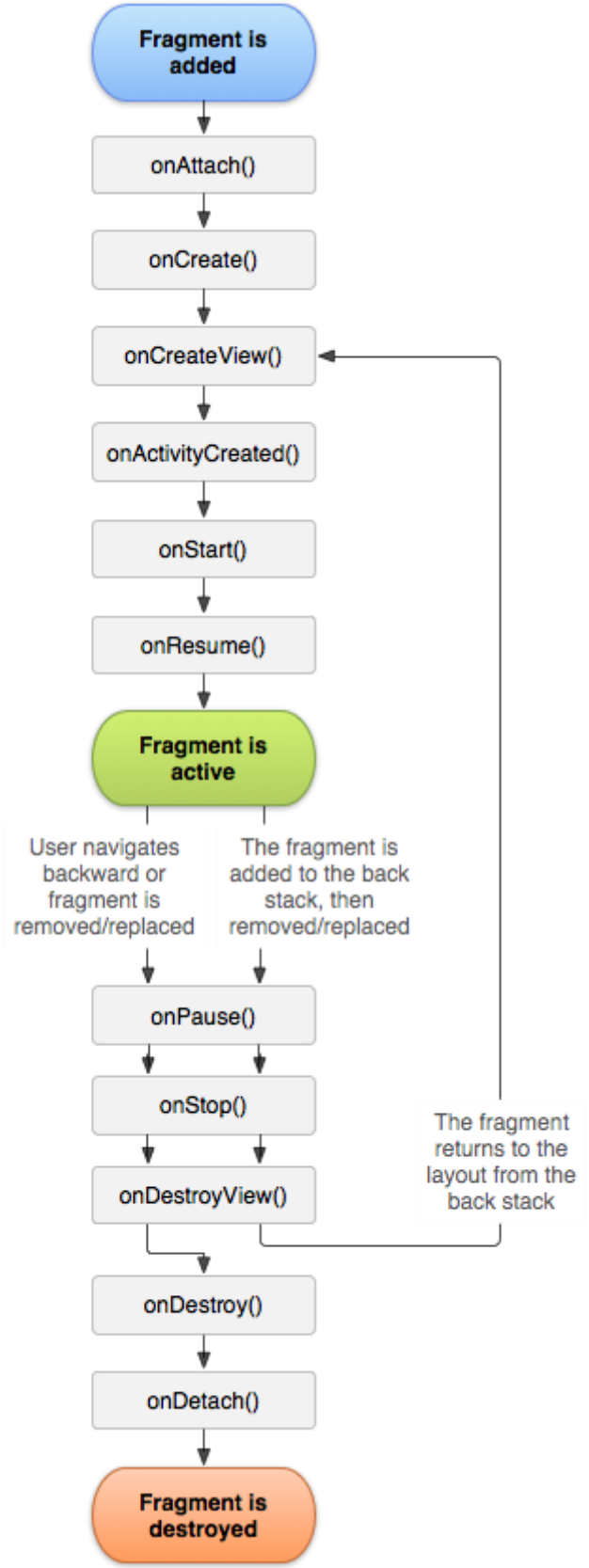
b. *Paused*

Ketika ada *Activity* lain yang menutupi sebagian dari *Activity* dimana *fragment* ditambahkan. Yang dimaksud menutupi sebagian adalah ketika *Activity*nya tidak tertutup sepenuhnya oleh *Activity* lain, jadi masih ada bagian dari *Activity* yang masih bisa kelihatan di layar.

c. *Stopped*

Ketika *fragment* tidak kelihatan di layar. Bisa jadi karena *Activity* dimana *fragment* itu ditambahkan, mengalami *state stopped* atau bahkan *fragment* itu sendiri sudah di *remove* dari *Activity* dan dilemparkan ke *Back Stack*. Pada kondisi ini *fragment* masih hidup dengan semua informasinya, akan tetapi sudah tidak kelihatan di layar dan akan di *destroy* atau *kill* ketika *Activity*nya di *destroy*.

Gambar 3.1 menunjukkan *method* yang akan dipanggil di dalam *fragment* ketika terjadi perubahan pada sebuah *Activity*. Gambar 3.1 menunjukkan bahwa perubahan *state* dari sebuah *Activity* akan mempengaruhi *lifecycle* dari sebuah *fragment*. Hal ini karena *fragment* merupakan komponen *view* yang bisa ditambahkan (*embed*) ke dalam *Activity*



Gambar 3.1 Fragment Lifecycle

3.3 Langkah – Langkah Praktikum

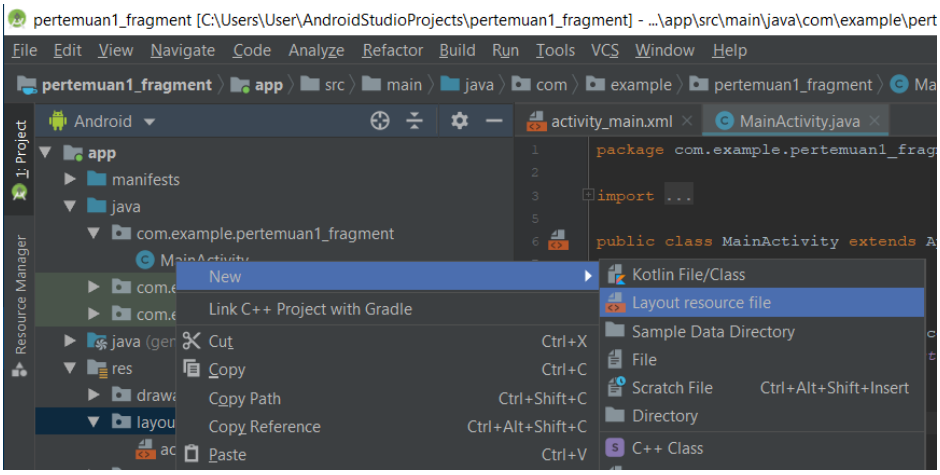
1. Praktikan membuat project dengan kriteria sebagai berikut :

   Project Name : *pertemuan1_fragment*

   Minimum API Level : *24*

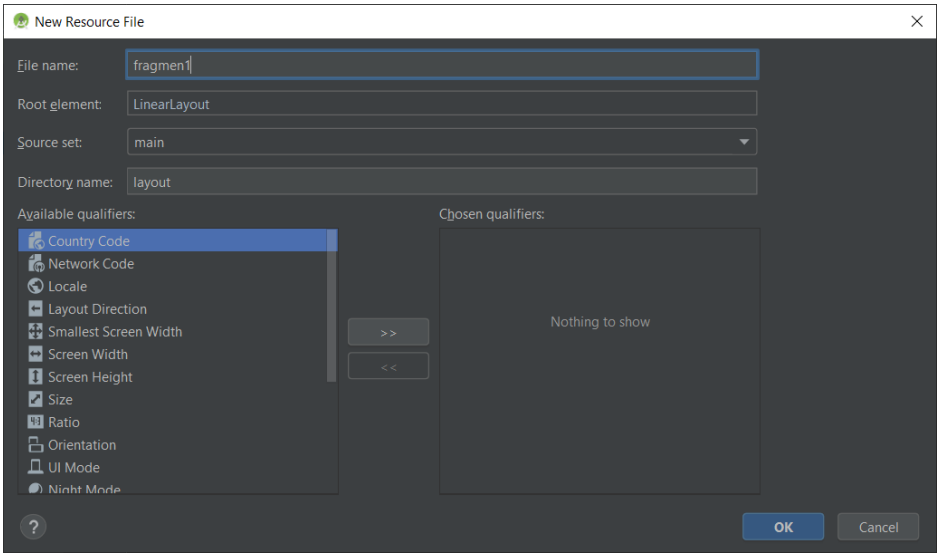
   Default Activity : *Empty Activity*

2. Praktikan membuat file *layout* baru dengan cara pilih *Tab Project* (pada *window* sebelah kiri) >> letakkan kursor pada *res/layout* >> klik kanan >> pilih *New* >> pilih *Layout resource file*, seperti Gambar 3.2



Gambar 3.2 Membuat *file layout xml* baru

3. Praktikan mengisi identitas untuk file xml yang dibuat, dengan ketentuan seperti gambar 3.3.

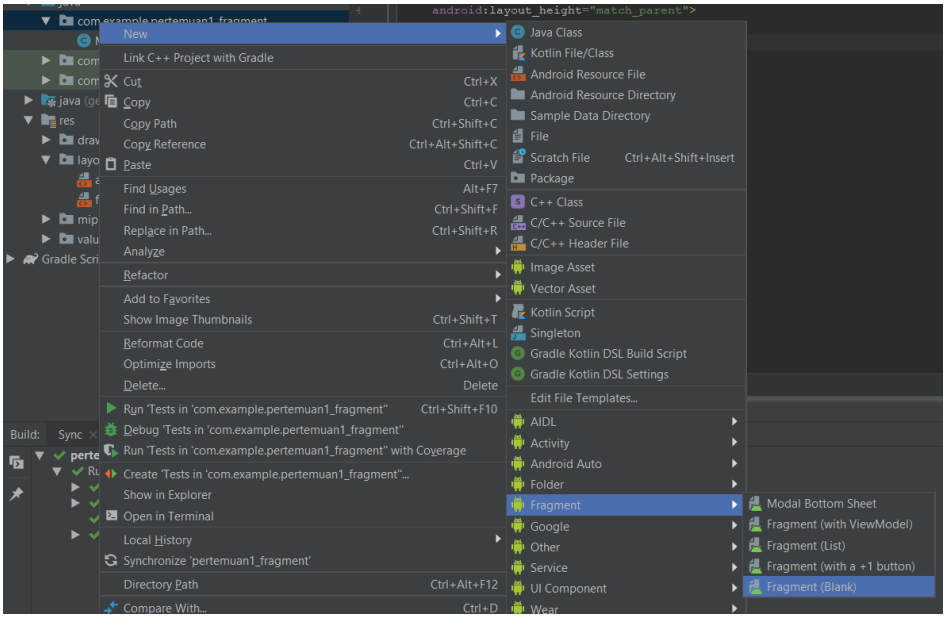


Gambar 3.3 Identitas untuk file *layout* yang dibuat

4. Source code fragment1.xml

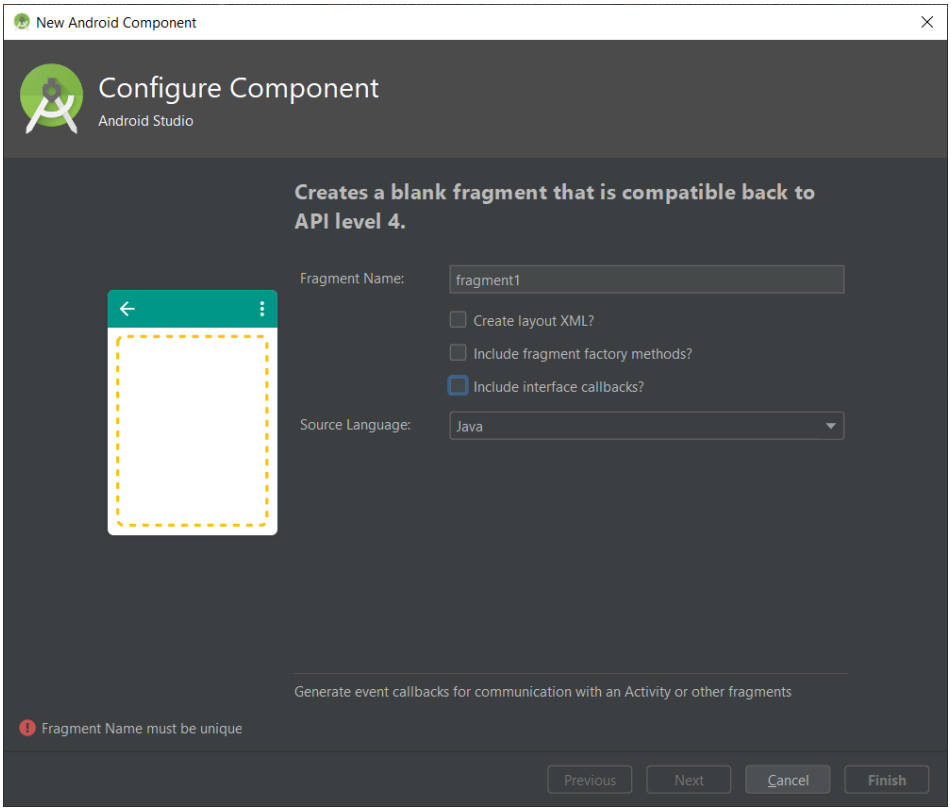
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/colorPrimary">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Ini adalah fragment 1"
        android:textSize="25sp"
        android:textColor="#000000" />

</LinearLayout>
```

5. Praktikan membuat *Fragment* baru dengan cara pilih *Tab Project* (pada *window* sebelah kiri) >> letakkan kursor pada nama *package* yang telah dibuat >> klik kanan >> pilih *New* >> pilih *Fragment* >> pilih *Fragment(Blank)*, seperti Gambar 3.4.



Gambar 3.4 Membuat *fragment* baru

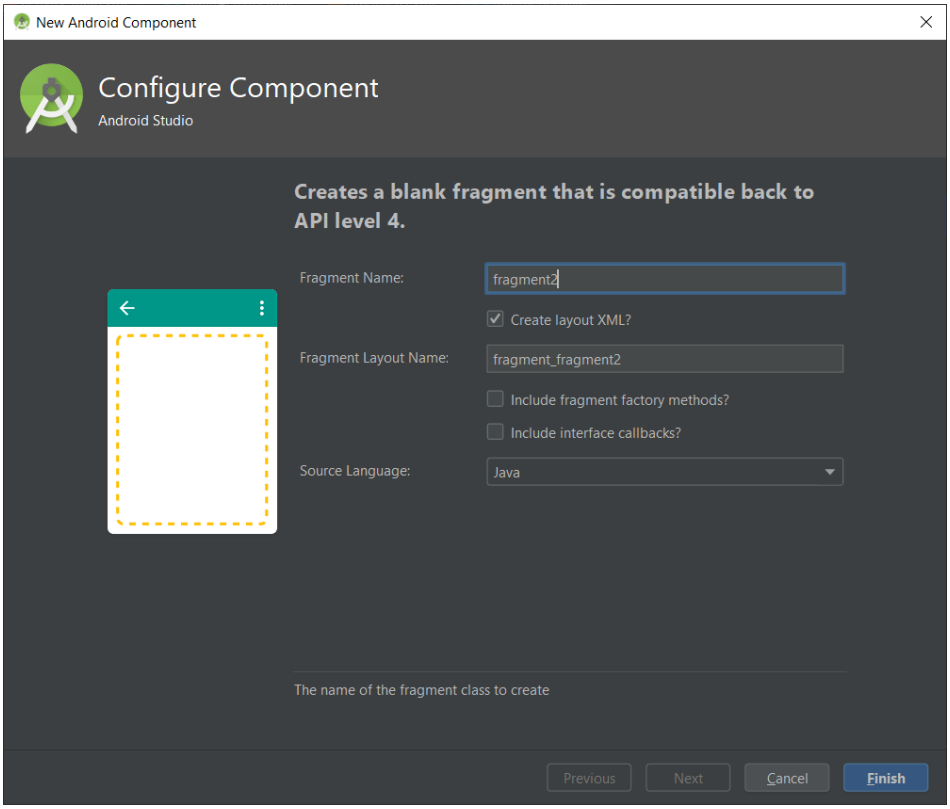
6. Membuat fragment baru dengan kreteria sebagai berikut.



7. Source code fragment1.java

```
package com.example.pertemuan1_fragment;


import android.os.Bundle;

import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

/**
 * A simple {@link Fragment} subclass.
 */
public class fragment1 extends Fragment {



    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                             Bundle savedInstanceState) {
        return
inflater.inflate(R.layout.fragmen1,container,false);
    }

}
```

8. Membuat fragment baru dengan kriteria sebagai berikut.



9. Source code fragment_fragment2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#000000"
    tools:context=".fragment2">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Ini adalah fragment 2"
        android:textSize="25sp"
        android:textColor="#ffffff" />

</FrameLayout>
```

10. Source code fragment2.java

```
package com.example.pertemuan1_fragment;


import android.os.Bundle;

import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
```

```
/**
 * A simple {@link Fragment} subclass.
 */
public class fragment2 extends Fragment {


    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_fragment2,
container, false);
    }

}
```

11. Source code activity_main.xml

```
<?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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    tools:context=".MainActivity" >

    <fragment
        android:id="@+id/fragment1"

android:name="com.example.pertemuan1_fragment.fragment1"
        android:layout_width="0px"
        android:layout_height="match_parent"
        android:layout_weight="1" />

    <fragment
        android:id="@+id/fragment2"

android:name="com.example.pertemuan1_fragment.fragment2"
        android:layout_width="0px"
        android:layout_height="match_parent"
        android:layout_weight="1" />

</LinearLayout>
```

12. Tampilan hasil