



# Android Dasar

Eko Kurniawan Khannedy

# Eko Kurniawan Khannedy

- Technical architect at one of the biggest ecommerce company in Indonesia
- 10+ years experiences
- [www.programmerzamannow.com](http://www.programmerzamannow.com)
- [youtube.com/c/ProgrammerZamanNow](https://youtube.com/c/ProgrammerZamanNow)





# Eko Kurniawan Khannedy

- Telegram : [@khannedy](https://t.me/khannedy)
- Facebook : [fb.com/ProgrammerZamanNow](https://fb.com/ProgrammerZamanNow)
- Instagram : [instagram.com/programmerzamannow](https://instagram.com/programmerzamannow)
- Youtube : [youtube.com/c/ProgrammerZamanNow](https://youtube.com/c/ProgrammerZamanNow)
- Telegram Channel : [t.me/ProgrammerZamanNow](https://t.me/ProgrammerZamanNow)
- Email : [echo.khannedy@gmail.com](mailto:echo.khannedy@gmail.com)



# Sebelum Belajar

- Mengerti Pemrograman Kotlin
- Mengerti Gradle
- Sudah mengikuti Kelas Roadmap Kotlin oleh Programmer Zaman Now
- Jika sudah mengikuti Kelas Roadmap Java oleh Programmer Zaman Now akan lebih baik lagi



# Agenda

- Pengenalan Android Development
- Membuat Project
- Menjalankan Project
- Activity
- Layout
- Find View
- Dan lain-lain

---

# Pengenalan Android



# Sejarah Android

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



# Versi Android

- Layaknya sistem operasi, Android sendiri sudah berkembang sejak pertama kali dikenalkan
- Ketika materi ini dibuat, Android sudah versi 11
- Namun bukan berarti semua perangkat sudah menggunakan Android versi 11, bisa jadi beberapa perangkat tidak mendukung sistem operasi Android versi terbaru



# Distribusi Versi Android

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.1 Jelly Bean	16	
4.2 Jelly Bean	17	99.8%
4.3 Jelly Bean	18	99.5%
4.4 KitKat	19	99.4%
5.0 Lollipop	21	98.0%
5.1 Lollipop	22	97.3%
6.0 Marshmallow	23	94.1%
7.0 Nougat	24	89.0%
7.1 Nougat	25	85.6%
8.0 Oreo	26	82.7%
8.1 Oreo	27	78.7%
9.0 Pie	28	69.0%
10. Q	29	50.8%
11. R	30	24.3%



# Android Development

- Saat pertama kali dikenalkan, untuk membuat aplikasi di Android, kita bisa membuatnya menggunakan bahasa pemrograman Java
- Namun bukan berarti semua fitur pemrograman Java bisa digunakan
- Android hanya menggunakan bahasa pemrograman Java, dan sedikit fitur yang terdapat di Java
- Oleh karena itu, kita perlu belajar lagi standard library atau API yang terdapat di Android, karena walaupun menggunakan bahasa yang sama, namun teknologinya berbeda
- <https://developer.android.com/?hl=id>



# Adopsi Pemrograman Kotlin

- Pada tahun 2017, Google mengumumkan bahwa Kotlin menjadi bahasa pemrograman utama yang sekarang digunakan untuk pengembangan aplikasi Android, menggantikan Java
- Kotlin adalah bahasa pemrograman yang dibuat oleh perusahaan JetBrains yang pertama kali dirilis sekitar tahun 2011
- Saat ini, Kotlin semakin populer karena Kotlin bisa bekerja dengan baik dengan ekosistem Java, sehingga tidak hanya programmer Android, programmer Backend Java pun sekarang banyak yang menggunakan Kotlin
- <https://developer.android.com/kotlin?hl=id>



# Android Studio

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

---

# Android SDK



# Android SDK

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

---

# Android Compatibility



# Android Compatibility

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



# Gambar API Level

Save location

Language

Minimum SDK

- API 25: Android 7.1.1 (Nougat)
- API 26: Android 8.0 (Oreo)
- API 27: Android 8.1 (Oreo)
- API 28: Android 9.0 (Pie)
- API 29: Android 10.0 (Q)**
- API 30: Android 11.0 (R)
- API 31: Android 12.0 (S)
- API 32



# Menentukan API Level

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



# Android Release Note

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

---

# Membuat Project



# Membuat Project

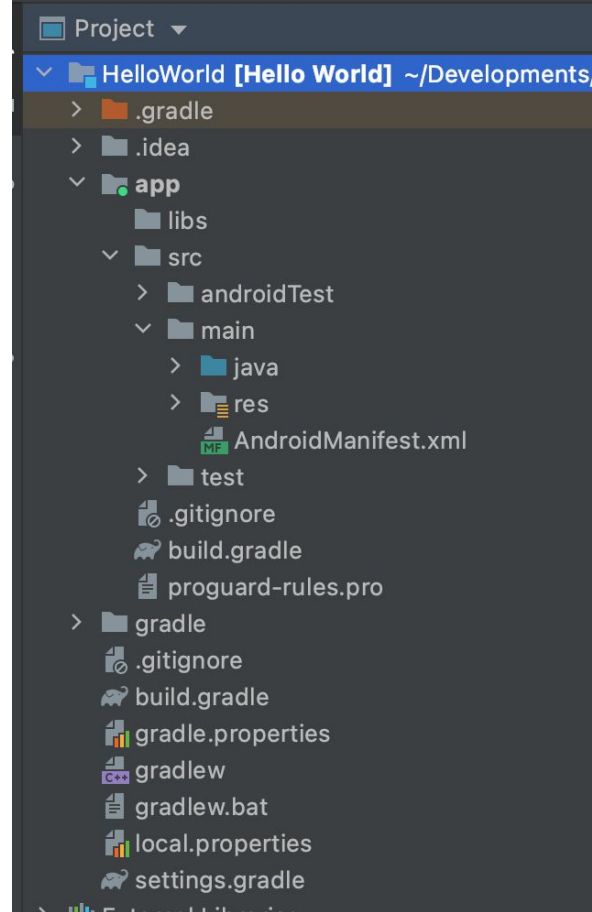
- Buat project baru menggunakan Empty Activity

---

# Struktur Project



# Struktur Project





Gradle



# Gradle

- Android menggunakan Gradle untuk project management nya, oleh karena itu di awal saya jelaskan bahwa teman-teman harus sudah mengerti tentang Kotlin dan Gradle
- Secara default, Android Studio akan membuat multi module Gradle Project, dimana hanya terdapat 1 module, yaitu app

---

# Android Virtual Device

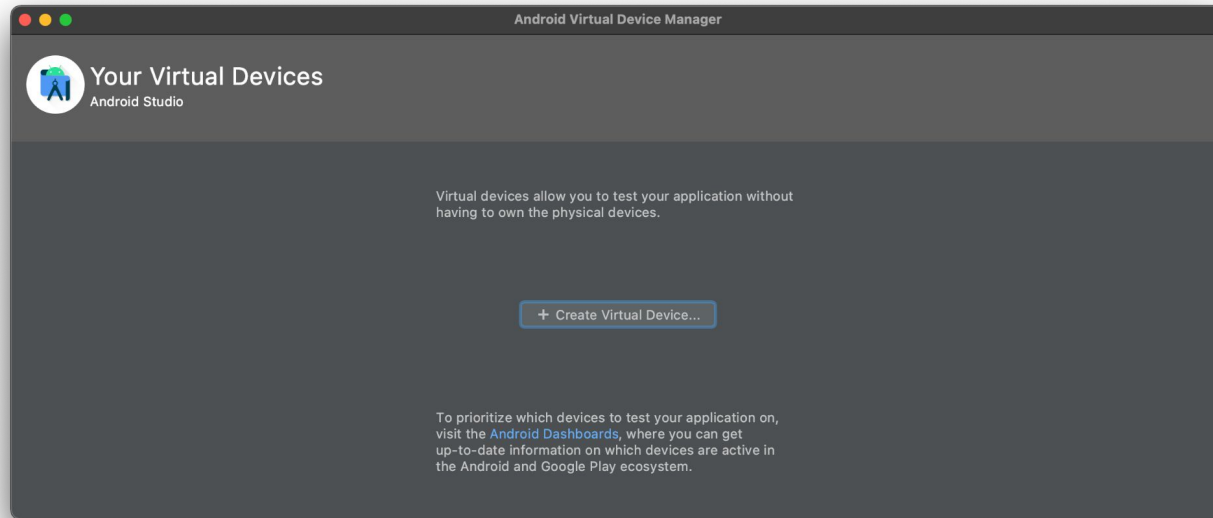


# Android Virtual Device

- Saat kita membuat aplikasi Android, kita tidak bisa menjalankan aplikasinya di sistem operasi Windows, Mac atau Linux yang kita gunakan
- Kita membutuhkan sistem operasi Android untuk menjalankan aplikasi Android kita
- Ada dua cara menjalankan aplikasi Android kita, pertama menggunakan device Android kita, kedua membuat Virtual Device
- Dan untungnya Android SDK mendukung kita untuk membuat Virtual Device, sehingga akan memudahkan kita menjalankan sistem operasi Android secara virtual



# Membuat Virtual Device



---

# Android Development Mode



# Android Development Mode

- Selain menggunakan Android Virtual Device, kita juga bisa menggunakan smartphone Android untuk mencoba aplikasi Android kita
- Namun sebelum kita bisa menggunakan smartphone Android kita, terlebih dahulu kita perlu menjalankan Development Mode di smartphone Android kita
- Biasanya tiap merek smartphone memiliki cara sendiri-sendiri untuk mengaktifkan Development Mode, oleh karena itu disarankan untuk mencari di internet cara mengaktifkan Development Mode pada merek smartphone yang kita gunakan



# Windows

- Khusus jika kita menggunakan Windows, kita juga perlu menginstall Driver agar smartphone Android kita bisa terdeteksi
- <https://developer.android.com/studio/run/oem-usb?hl=id>

---

# Build Configuration

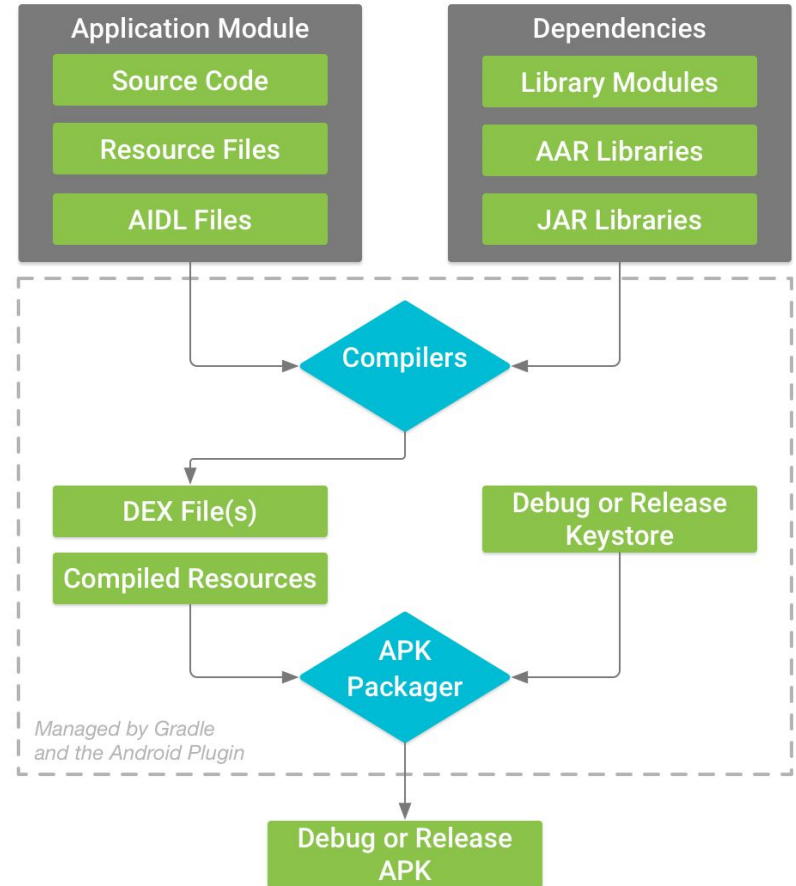


# Android Build System

- Android build system akan melakukan kompilasi aplikasi dari kode program dan resource dan mem-package semuanya menjadi sebuah aplikasi Android
- Android Studio menggunakan Gradle, untuk melakukan otomatisasi semua proses tersebut, sehingga kita tidak perlu melakukannya secara manual lagi menggunakan Android Build System
- Gradle dan Android Build System sendiri sebenarnya bisa berjalan secara independen, oleh karena itu kita wajib menggunakan Android Studio, namun dengan menggunakan Android Studio, akan mempermudah kita ketika membuat aplikasi Android

# Proses Build

- Compiler akan melakukan kompulasi semua kode kita menjadi DEX (Dalvix Executable) file
- API Packager akan menandai file apakah ini versi debug atau release, sebelum akhirnya dijadikan aplikasi file APK (Android Application Package)





# Konfigurasi Modul Aplikasi

- Setiap module di project, terdapat gradle file build.gradle yang berisikan konfigurasi dari module aplikasi
- Kita perlu menentukan konfigurasi pada modul aplikasi sesuai dengan build configuration yang kita gunakan, misal kita harus pastikan sdk dan target sdk nya sesuai dengan yang kita gunakan misalnya
- Selain kita juga perlu menentukan applicationId, yang mana itu adalah id dari aplikasi kita, dan harus unik secara global, artinya tidak boleh ada yang sama



## Kode : Build Configuration

```
android {  
    compileSdk 32  
  
    defaultConfig {  
        applicationId "com.programmerzamannow.helloworld"  
        minSdk 29  
        targetSdk 32  
        versionCode 1  
        versionName "1.0"  
  
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  
    }  
}
```

---

# Menjalankan Aplikasi



# Android Application Package

- Sebelum menjalankan aplikasi kita, kita perlu mem-package aplikasi Android kita dalam format APK (Android Application Package)
- Namun hal ini tidak perlu kita lakukan manual, karena secara otomatis Android Studio akan menggunakan Gradle untuk membuat APK secara otomatis, lalu mengirimnya ke Device Android yang kita pilih (Virtual Device atau Device Asli)

---

# Manifest File



# Manifest File

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

# Kode : Contoh Manifest File

```
AndroidManifest.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="com.programmerzamannow.helloworld">
4
5     <application
6         android:allowBackup="true"
7         android:icon="@mipmap/ic_launcher"
8         android:label="@string/app_name"
9         android:roundIcon="@mipmap/ic_launcher_round"
10        android:supportRtl="true"
11        android:theme="@style/Theme.HelloWorld">
12        <activity
13            android:name=".MainActivity"
14            android:exported="true">
15            <intent-filter>
16                <action android:name="android.intent.action.MAIN" />
17
18                <category android:name="android.intent.category.LAUNCHER" />
19            </intent-filter>
```

---

# Activity



# Activity

- Saat kita membuat aplikasi seperti di Java atau di Kotlin, biasanya ketika akan membuat main function sebagai function yang akan diluncurkan ketika aplikasi berjalan
- Di Android tidak seperti itu, Android memiliki fitur yang bernama Activity, dimana nanti object Activity tersebut akan secara otomatis dijalankan oleh Android
- Pada kelas ini kita tidak akan membahas detail tentang Activity, kita akan membahasnya nanti di kelas khusus tentang Activity, karena materi Activity sangat banyak



# Class Activity

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



# Mendaftarkan Activity

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



## Kode : Contoh Manifest File

```
<activity
    android:name=".MainActivity"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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



# Activity Callback

- Activity itu memiliki banyak sekali function, nanti kita akan bahas di kelas tersendiri
- Salah satu function yang dipanggil ketika Activity dibuat adalah onCreate()



## Kode : Activity onCreate()

```
class MainActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
    }  
}
```



# Layout



# Layout

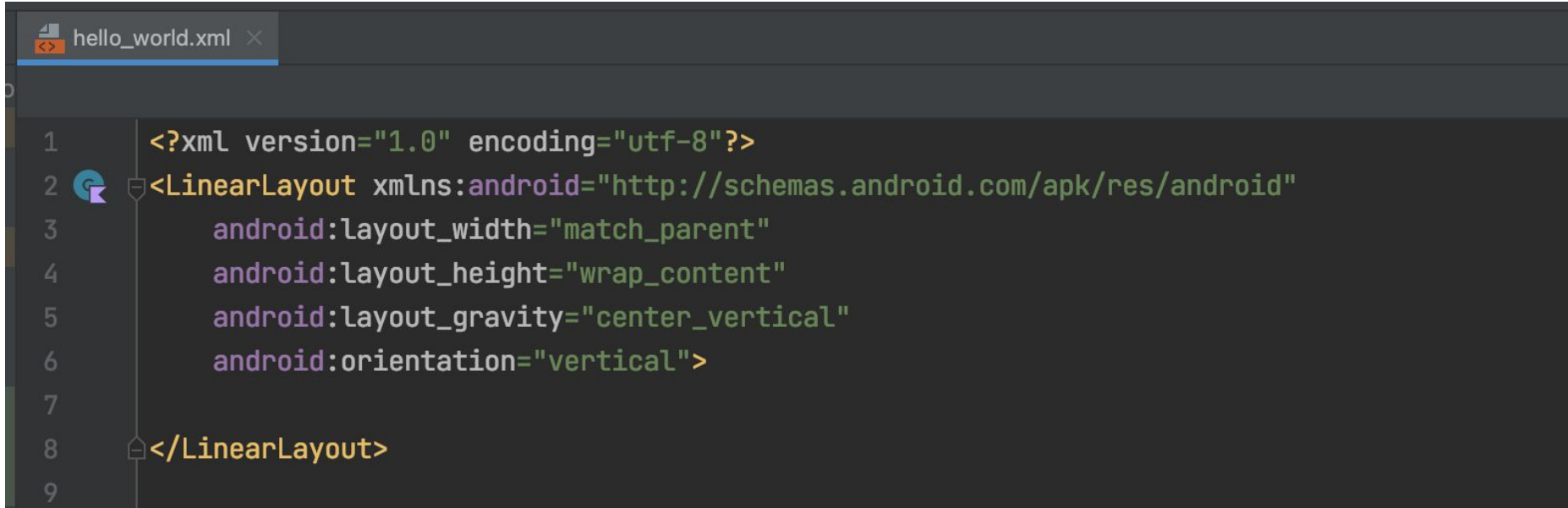
- Activity bukanlah tampilan UI, tapi biasanya Activity itu akan menampilkan tampilan UI
- Android memisahkan antara kode program dan tampilan UI, namanya adalah Layout
- Layout merupakan kode yang berisikan tampilan UI
- Layout di Android menggunakan XML, sehingga yang terbiasa menggunakan HTML, akan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



## R Class

- Saat kita membuat Layout baru, secara otomatis Android akan melakukan auto generate sebuah variable berisikan id layout yang sesuai dengan nama Layout nya
- Dan untuk menentukan Layout mana yang akan ditampilkan di Activity, kita bisa menggunakan function setContentView(layout\_id)

# Kode : Layout



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="match_parent"
4      android:layout_height="wrap_content"
5      android:layout_gravity="center_vertical"
6      android:orientation="vertical">
7
8  </LinearLayout>
9
```



## Kode : Activity

```
class MainActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.hello_world)  
    }  
}
```

---

**View**



# View

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



# Menambah View

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



## Kode : Menambah View

```
<TextView
```

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

```
<EditText
```

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

```
<Button
```

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

```
<TextView
```

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

---

**View ID**



## Find View by Id

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



# View ID

- Untuk menambahkan View ID, kita bisa menambahkan atribut id pada komponen nya
- Namun caranya kita perlu menggunakan nilai `@+id/namaIdNya`
- Android Build System akan secara otomatis membuatkan id tersebut sebagai id component di class R, mirip seperti pada Layout
- Best Practice penamaan view ID adalah nama diikuti dengan jenis komponen, misal : `nameEditText`, `sayHelloButton`, `firstNameEditText`, `lastNameEditText`, `registerButton`, dan lain-lain

# Kode : View ID

```
<EditText
    android:id="@+id/nameEditText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal" />

<Button
    android:id="@+id/sayHelloButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:text="Say Hello" />

<TextView
    android:id="@+id/sayHelloTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"/>
```

Component Tree



SAY HELLO

## Kode : Find View by Id

```
class MainActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.hello_world)  
  
        val nameEditText: EditText = findViewById(R.id.nameEditText)  
        val sayHelloButton: Button = findViewById(R.id.sayHelloButton)  
        val sayHelloTextView: TextView = findViewById(R.id.sayHelloTextView)  
  
        sayHelloTextView.text = "Hi"  
    }  
}
```

---

# Action Listener



# Action Listener

- Beberapa jenis component memiliki Action Listener
- Yaitu object yang bisa kita tambahkan ke komponen ketika sebuah aksi dilakukan ke komponen tersebut
- Detail dari Action Listener akan kita bahas di kelas khusus membahas Android View
- Contoh sederhana, pada Button, terdapat Click Listener yang bisa digunakan untuk menambahkan object Listener ketika tombol di Click menggunakan function `setOnClickListener(listener)`



## Kode : Action Listener

```
val nameEditText: EditText = findViewById(R.id.nameEditText)
val sayHelloButton: Button = findViewById(R.id.sayHelloButton)
val sayHelloTextView: TextView = findViewById(R.id.sayHelloTextView)

sayHelloTextView.text = "Hi"

sayHelloButton.setOnClickListener { it: View!
    val name = nameEditText.text.toString()
    sayHelloTextView.text = "Hi $name"
}
```



Lateinit



# Masalah Dengan Find View By Id

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



# Lateinit

- Salah satu rekomendasinya adalah kita simpan object komponen nya sebagai variable di class nya
- Namun karena pada kotlin jika kita ingin membuat variable yang bisa nullable harus menggunakan tanda ?, namun agar lebih mudah, kita bisa menggunakan lateinit, agar variable yang kita buat tidak perlu menjadi nullable



## Kode : Lateinit

```
private lateinit var nameEditText: EditText
private lateinit var sayHelloButton: Button
private lateinit var sayHelloTextView: TextView

private fun initComponents() {
    nameEditText = findViewById(R.id.nameEditText)
    sayHelloButton = findViewById(R.id.sayHelloButton)
    sayHelloTextView = findViewById(R.id.sayHelloTextView)
}

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.hello_world)

    initComponents()
}
```