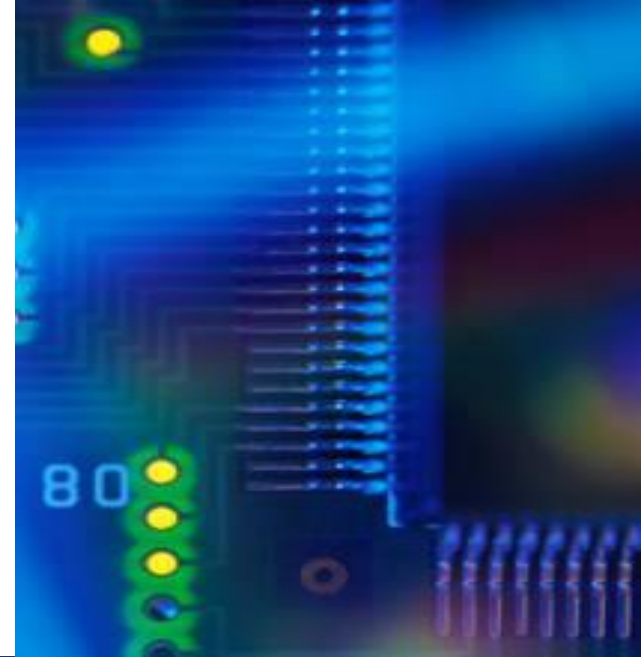




KEMENTERIAN KOMUNIKASI DAN INFORMATIKA  
REPUBLIK INDONESIA

*Menuju Masyarakat Informasi Indonesia*



## JUNIOR MOBILE PROGRAMMER

**Menunjukkan platform operating system dan bahasa pemrograman di dalam perangkat lunak**

# Deskripsi Singkat

## Deskripsi Singkat mengenai Topik

Topik ini menjelaskan bahasa pemrograman berbasis mobile (Dasar Java untuk mobile programming)

## Tujuan Pelatihan

1. Peserta mampu menentukan mobile pemrograman berbasis mobile jenis bahasa pemrogramannya.
2. Peserta mampu membandingkan perbedaannya bahasa pemrograman berbasis mobile.
3. Peserta mampu Mengkonfigurasi Perangkat lunak terkait penggunaan bahasa pemrograman berbasis mobile sesuai dengan spesifikasinya.
4. Peserta mampu Menghasilkan Alur program untuk pembuatan aplikasi berbasis mobile.
5. Menentukan tipe-data variabel dan konstanta dalam salah satu bahasa pemrograman berbasis mobile
6. Menentukan konsep struktur kondisi dan perulangan dalam salah satu bahasa pemrograman berbasis mobile.
7. Menjelaskan Konsep layout dan objek dalam salah satu bahasa pemrograman berbasis mobile.
8. Membangun aplikasi mobile sederhana dengan bahasa pemrograman mobile.

## Materi Yang akan disampaikan:

1. Kategori dan Jenis Bahasa Pemrograman
2. Pengaturan Ruang Kerja
3. Dasar Alur Pembuatan Software
4. Konsep Variabel dan Konstanta
5. Struktur Kondisi dan Perulangan
6. Konsep Layout
7. Membuat Sebuah Aplikasi Sederhana

## Kategori Aplikasi dan Jenis Bahasa Pemrograman

## Kategori Aplikasi

## Pelatihan

### **Native mobile application**

dikembangkan dalam bahasa pemrograman yang berasal dari perangkat dan sistem operasi, dan perlu membuat sebuah aplikasi untuk platform tertentu.

### **Hybrid mobile application**

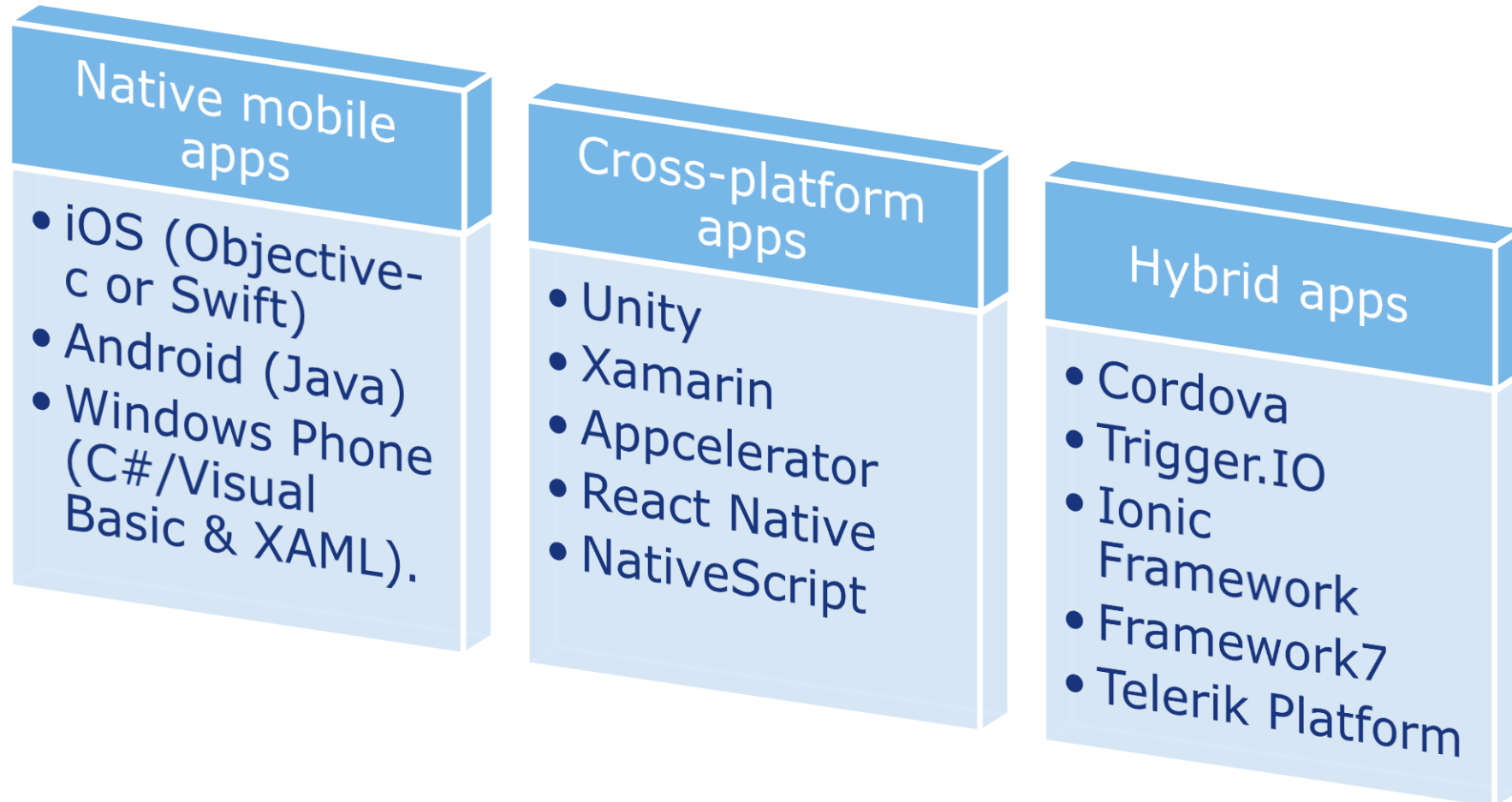
adalah aplikasi lintas platform namun membuat antarmuka pengguna menggunakan browser web, memanfaatkan HTML, CSS, dan Javascript.

### **Cross-platform mobile application**

dikembangkan menggunakan Bahasa seperti Javascript, dan bukan native terhadap sistem operasi smartphone.

# Kategori dan Jenis Bahasa Pemrograman

## Pelatihan



## Pengaturan environment

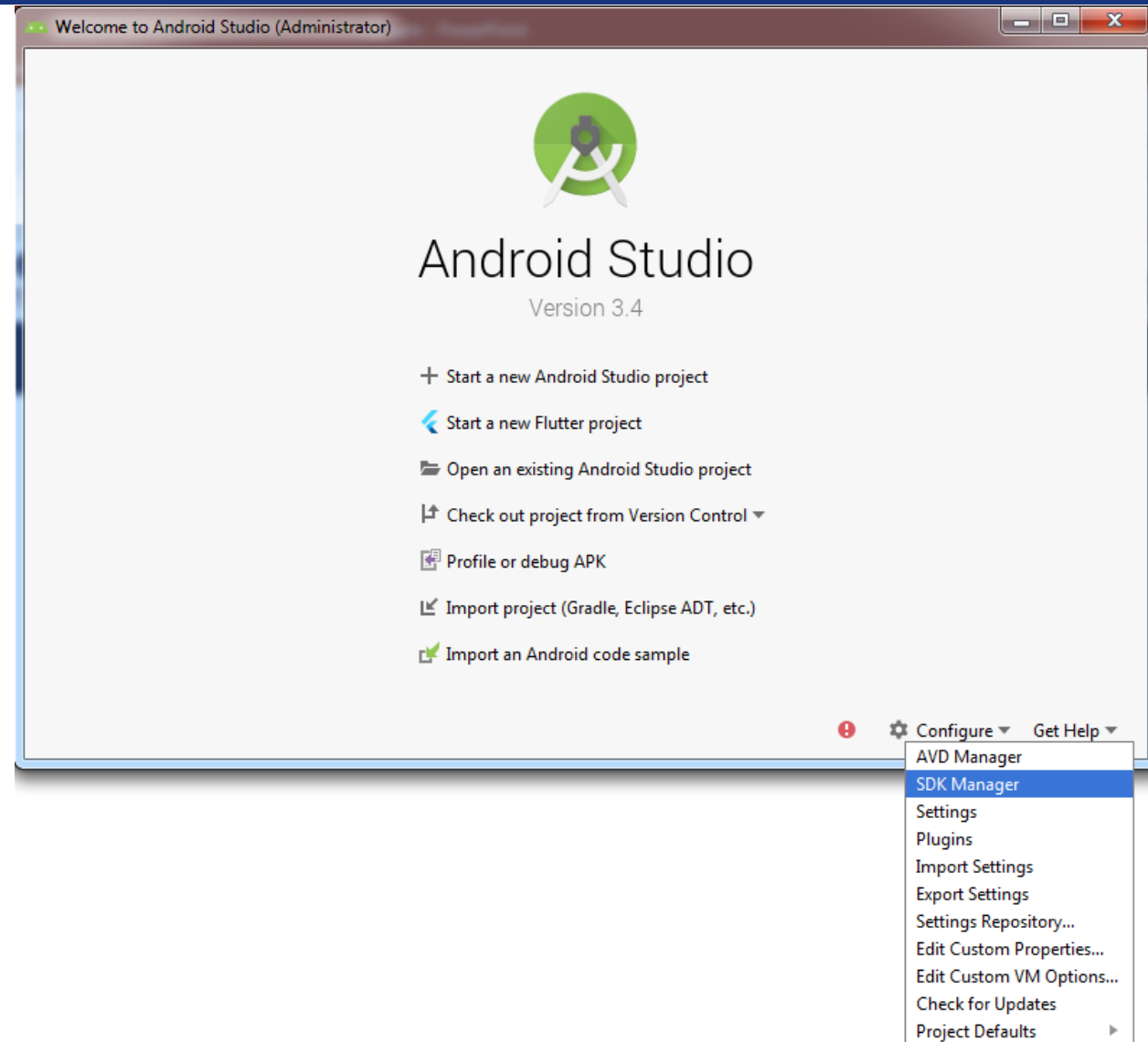
## Pelatihan

1. Java Development Kit (JDK)
2. Android Studio
3. Software Development Kit (SDK)

# Instalasi Software Development Kit

# Pelatihan

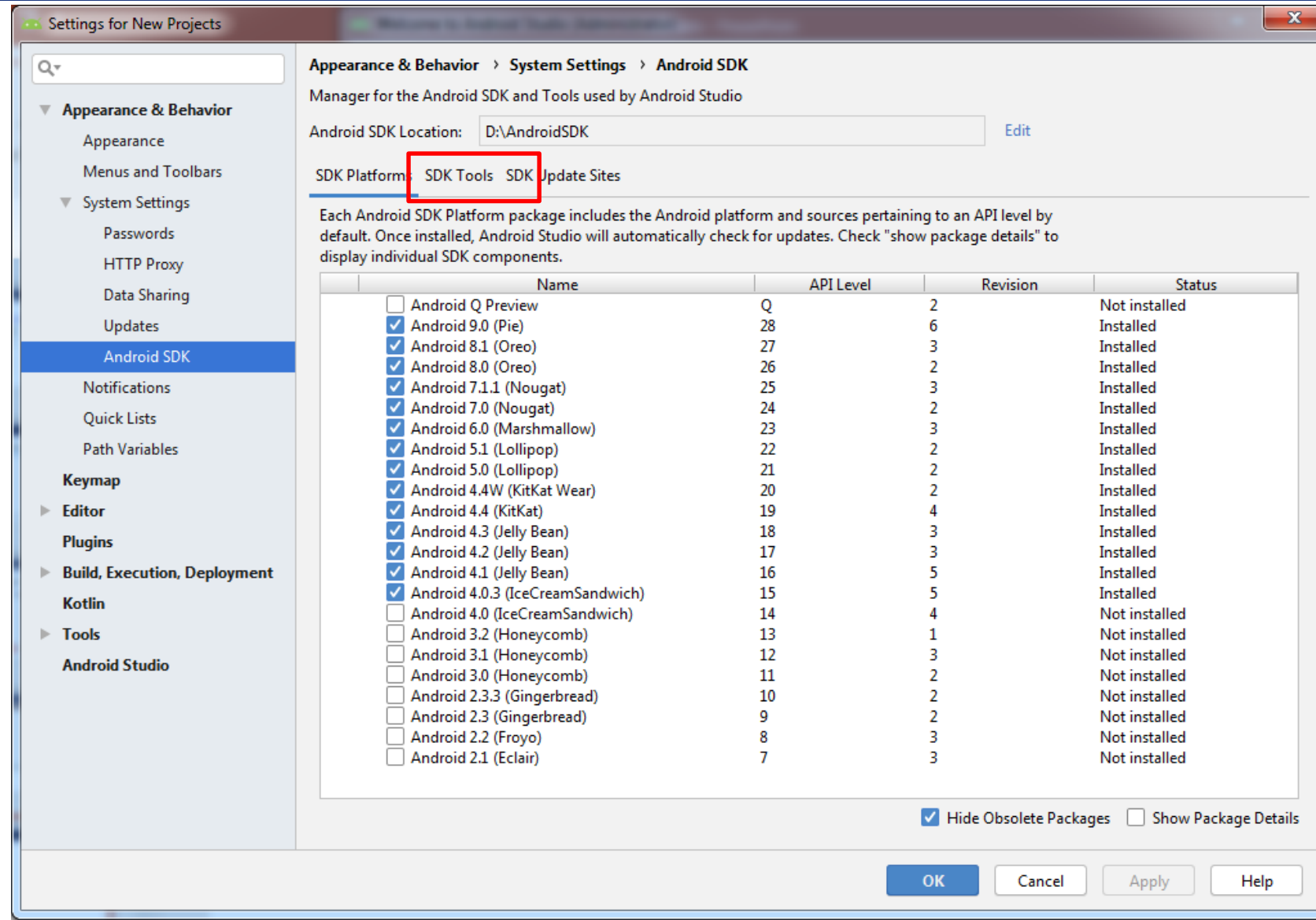
1. Buka android Studio
2. Pilih SDK Manager



# Instalasi Software Development Kit

# Pelatihan

## 3. Pilih SDK Tools



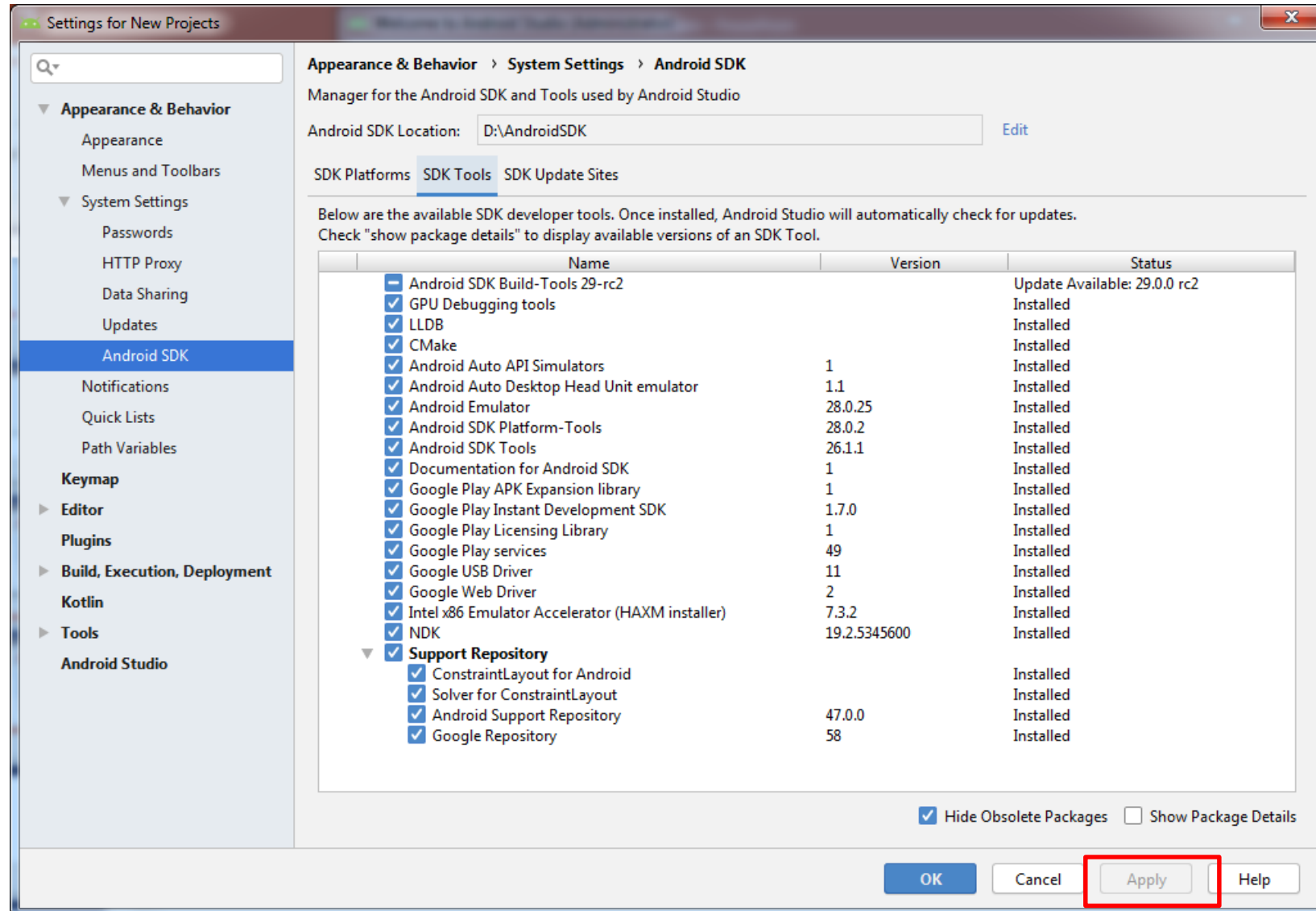


# Instalasi Software Development Kit

# Pelatihan

4. Centang pada bagian yang dibutuhkan. Kemudian tekan tombol apply untuk melakukan instalasi.

SDK manager akan mendownload dan menginstall kebutuhan tersebut.



# Alur pengembangan aplikasi

## Dasar Alur Kerja

## Pelatihan

Alur kerja untuk mengembangkan aplikasi untuk Android secara konseptual sama dengan platform aplikasi lainnya. Namun, untuk secara efisien membangun aplikasi yang dirancang dengan baik untuk Android, Anda memerlukan beberapa alat khusus..

### 1. Siapkan ruang kerja anda

Instalasi Android Studio, dan membuat project baru.



## Dasar Alur Kerja

## Pelatihan

### 2. Menulis code pada aplikasi anda dan menambahkan sumber daya

- ☐ Mulailah mengerjakan aplikasi.
- ☐ Android Studio memiliki berbagai alat yang dapat membantu Anda bekerja lebih cepat, menulis kode kualitas, merancang UI, dan membuat sumber daya untuk berbagai jenis perangkat.

Write

Write code

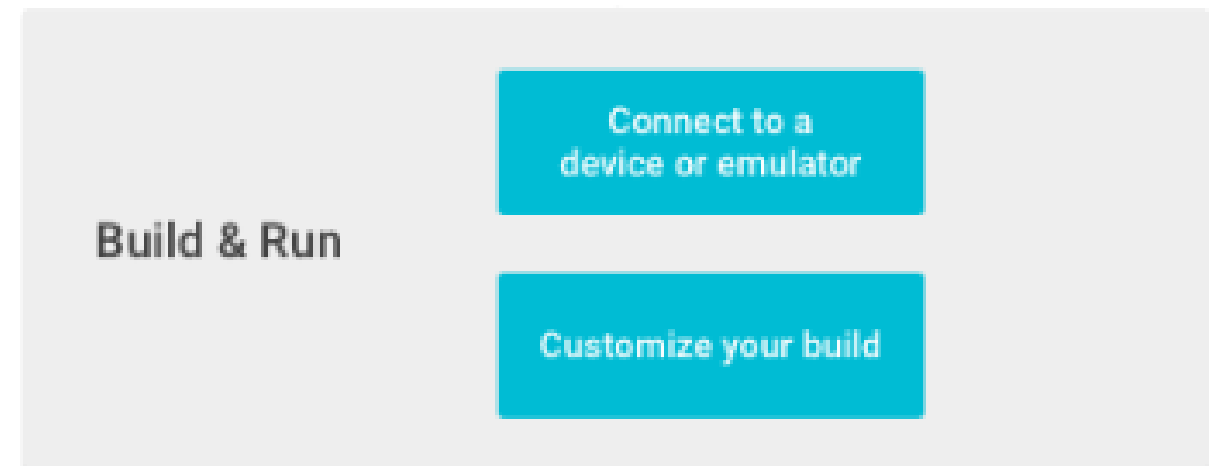
Add assets

## Dasar Alur Kerja

## Pelatihan

### 3. Membangun dan menjalankan aplikasi anda

- ☐ Jalankan aplikasi pada smarphone atau emulator.
- ☐ Customize build untuk membentuk ukuran aplikasi yang lebih kecil.

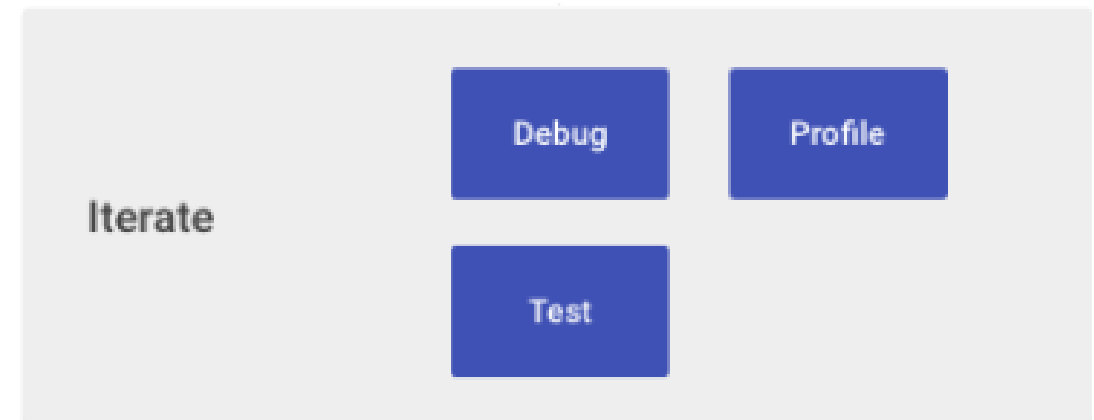


## Dasar Alur Kerja

## Pelatihan

### 4. Step perulangan

- ☐ Temukan bug dan tingkatkan kinerja aplikasi.
- ☐ Profile berguna untuk menganalisis kinerja seperti penggunaan memori, lalu lintas jaringan, dampak CPU, dan lainnya.
- ☐ Testing aplikasi anda



## Dasar Alur Kerja

## Pelatihan

### 5. Pasarkan aplikasi anda

- ☐ Atur versi aplikasi anda.
- ☐ Buat key dan tanda tangani aplikasi.
- ☐ Ketika update aplikasi, pastikan versi diatur lebih tinggi dan gunakan key yang sama.

Lebih detail pada link berikut ini

<https://developer.android.com/studio/publish/index.html>

Publish

Version

Sign

# Dasar



# Dasar

- ❑ Aplikasi android dapat dibuat dengan menggunakan Bahasa java, kotlin dan C++.
- ❑ Android SDK mengkompilasi kode, data dan file sumber daya apa pun ke dalam APK (Android application package), yang merupakan file arsip dengan akhiran .apk.
- ❑ Satu file apk memiliki semua konten dari aplikasi android dan dapat digunakan untuk menginstal aplikasi.

# Dasar (Lanjutan)

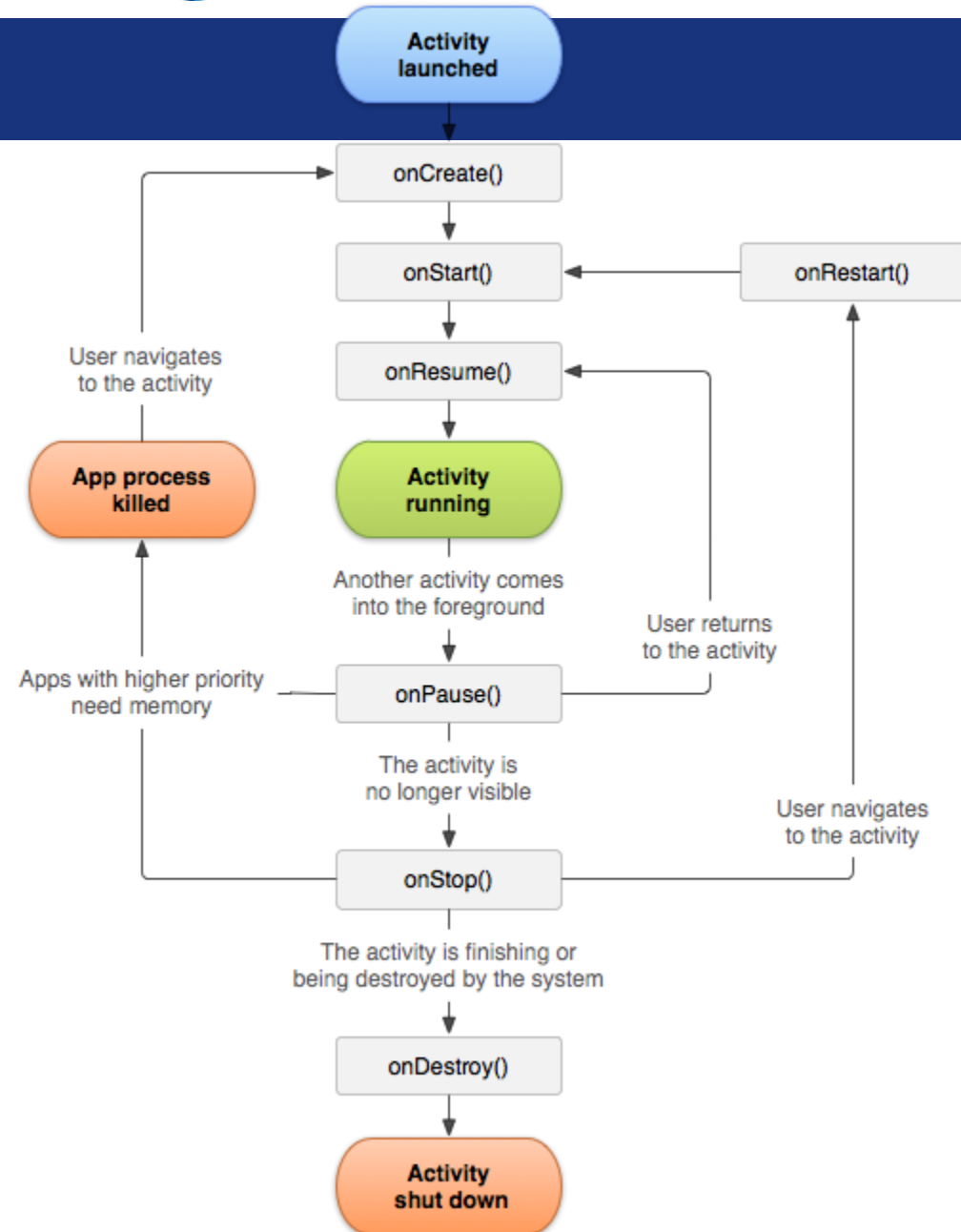
## Komponen

Terdapat 4 buah tipe komponen

- ☐ Activities
- ☐ Services
- ☐ Broadcast receivers
- ☐ Content providers

## Activities

- ☐ Sebuah activity akan menyajikan user interface (UI) kepada pengguna, sehingga pengguna dapat melakukan interaksi.
- ☐ Satu activity biasanya dipakai untuk menampilkan aplikasi atau bertindak sebagai user interface (UI) saat aplikasi diperlihatkan kepada user.
- ☐ Untuk pindah dari satu activity ke activity lain, dapat dilakukan dengan satu even, misalnya klik tombol, memilih opsi atau menggunakan triggers tertentu.



## Service

- ☐ Service merupakan sebuah komponen yang digunakan untuk membuat aplikasi tetap menjalankan fungsi pada background.
- ☐ Service tidak memiliki user interface.
- ☐ Contoh dari penggunaan service adalah ketika memainkan music.

## Contoh penerapan

- ☐ Transaksi jaringan
- ☐ Memutar musik
- ☐ Melakukan file I/O
- ☐ Berinteraksi dengan penyedia materi

## Karakteristik layanan

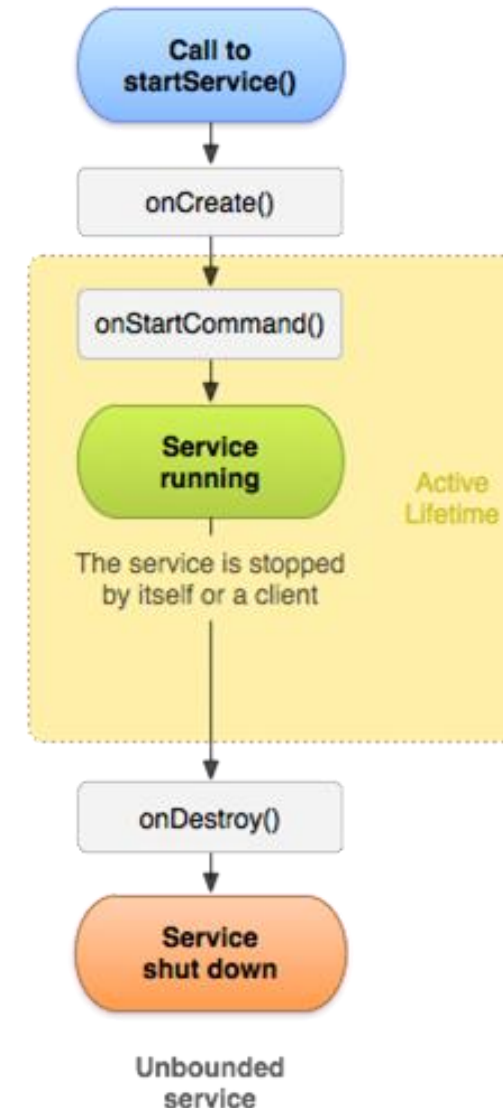
- ☐ Dimulai dengan Intent
- ☐ Bisa tetap berjalan ketika pengguna beralih aplikasi
- ☐ Daur hidup—yang harus Anda kelola
- ☐ Aplikasi lain bisa menggunakan layanan—mengelola izin
- ☐ Berjalan di thread utama dari proses hosting-nya

## **Service: tidak terikat (unbound service)**

- ☐ Dimulai dengan Intent
- ☐ Bisa tetap berjalan ketika pengguna beralih aplikasi
- ☐ Berjalan di thread utama dari proses hosting-nya
- ☐ Daur hidup yang harus Anda kelola, jika tidak dihentikan, akan terus berjalan.

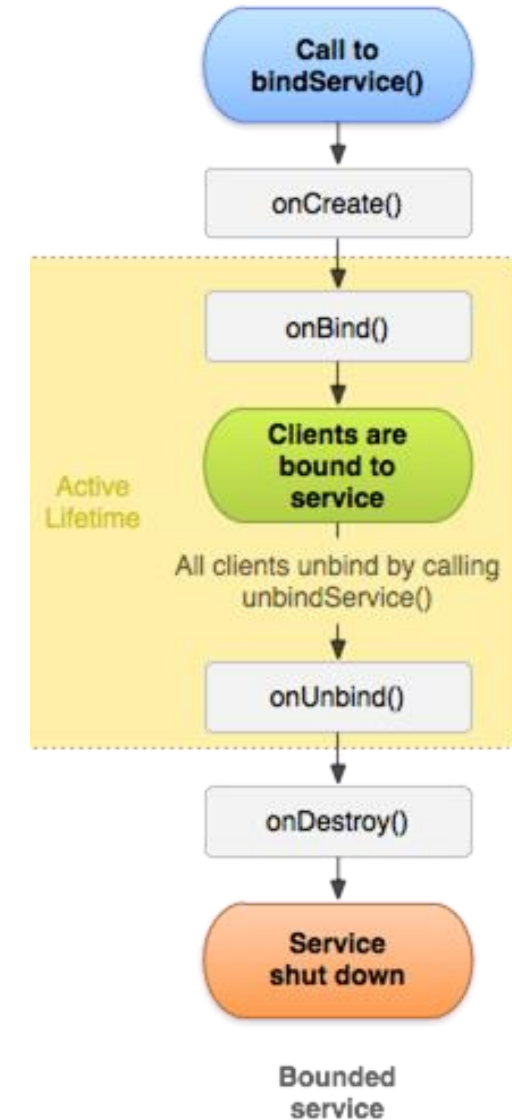


- Dimulai dengan startService()
- Berjalan tanpa batas waktu sampai berhenti sendiri
- Biasanya tidak memperbarui UI



## Service: terikat (bound service)

- ❑ Menawarkan antarmuka klien-server yang memungkinkan komponen untuk berinteraksi dengan service
- ❑ Klien mengirimkan permintaan dan mendapatkan hasil
- ❑ Dimulai dengan `bindService()`
- ❑ Berakhir ketika semua klien melepaskan kaitan

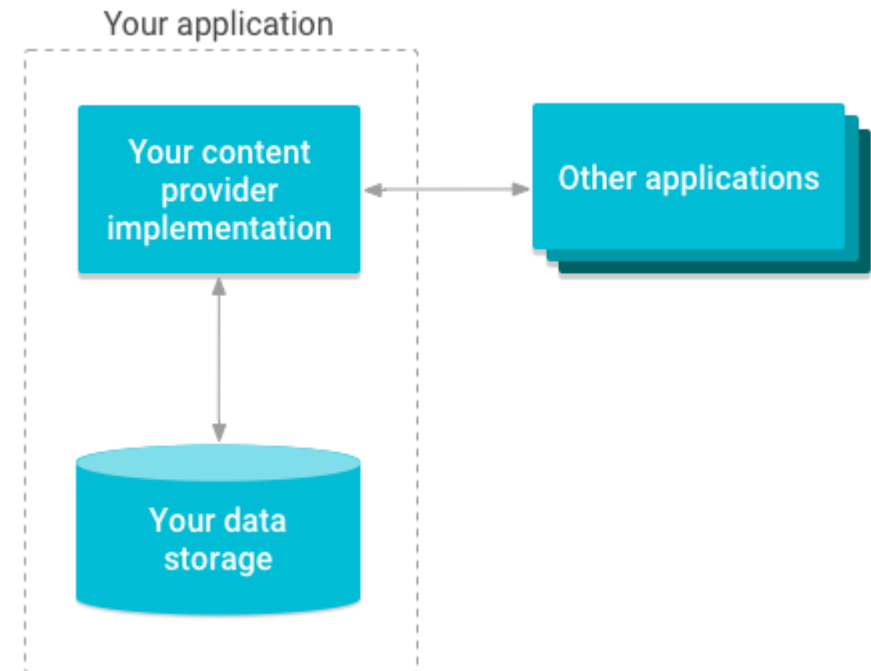


## Broadcast receivers

- ☐ Broadcast receiver berfungsi menerima dan berekasi untuk menyampaikan notifikasi.
- ☐ Contohnya adalah menampilkan notifikasi zona waktu berubah, baterai lemah, atau
- ☐ pengubahan referensi bahasa yang digunakan.

## Content providers

- ❑ Komponen pengambil data yang diminta aplikasi dari repositori
- ❑ Aplikasi ini tidak perlu mengetahui di mana atau bagaimana data disimpan, diformat, atau diakses



## **JAVA (Konsep variabel, Kondisi dan Perulangan)**

# Dasar Java Android

# Pelatihan

Apa itu Java?

Java adalah bahasa pemrograman yang terkenal, dibuat pada tahun 1995. Dimiliki oleh Oracle.

Digunakan untuk :

1. Aplikasi Mobile (spesial Android apps)
2. Aplikasi Desktop
3. Aplikasi Web
4. Web Server dan Application server
5. Game
6. Koneksi Basisdata
7. Dan lain lain

# Dasar Java Android

# Pelatihan

Mengapa Menggunakan Java?

1. Java berfungsi pada berbagai platform (Windows, Mac, Linux, Raspberry Pi, dll.)
2. Salah satu bahasa pemrograman paling populer di dunia
3. Mudah dipelajari dan mudah digunakan
4. Open-source dan gratis
5. Aman, cepat dan kuat
6. Memiliki dukungan komunitas yang sangat besar (puluhan juta pengembang)

Command Palette... Ctrl+Shift+P  
Snippets...

Build System >

Build Ctrl+B

Build With... Ctrl+Shift+B

Cancel Build Ctrl+Break

Build Results >

✓ Save All on Build

Record Macro Ctrl+Q

Playback Macro Ctrl+Shift+Q

Save Macro...

Macros >

Developer >

Install Package Control...

✓ Automatic

ActionScript

Ant

C Single File

C++ Single File

Cargo

D

D dub

Erlang

Haskell

JavaC

Lua

Make

Python

R

Ruby

Rust

ShellScript

Syntax Tests

New Build System...



Command Palette... Ctrl+Shift+P  
Snippets...

Build System >

Build Ctrl+B

Build With... Ctrl+Shift+B

Cancel Build Ctrl+Break

Build Results >

✓ Save All on Build

Record Macro Ctrl+Q

Playback Macro Ctrl+Shift+Q

Save Macro...

Macros >

Developer >

Install Package Control...

Automatic

ActionScript

Ant

C Single File

C++ Single File

Cargo

D

D dub

Erlang

Haskell

✓ JavaC

Lua

Make

Python

R

Ruby

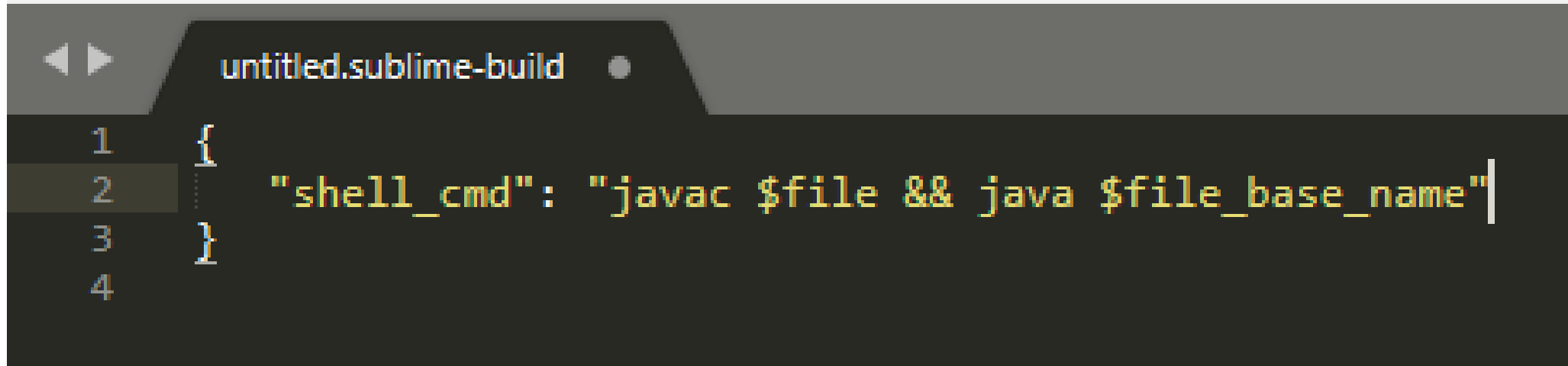
Rust

ShellScript

Syntax Tests

New Build System...

File Edit Selection Find View Goto Tools Project Preferences Help



```
untitled.sublime-build
1 {
2   "shell_cmd": "javac $file && java $file_base_name"
3 }
4
```



Save As



← → ↕ ↑ > LPA > AppData > Roaming > Sublime Text 3 > Packages > User

Search User



Organize ▾

New folder



This PC

3D Objects

Desktop

Documents

Downloads

Music

Pictures

Videos

Local Disk (C:)

DATA (E:)

Name

Date modified

Type

Size

Preferences.sublime-settings

25/04/2019 3:00

SUBLIME-SETTIN...

1 KB

File name: RunJava.sublime-build

Save as type: JSON (\*.json;\*.sublime-settings;\*.sublime-menu;\*.sublime-keymap;\*.sublime-mousemap;\*.sublime-theme;\*.sublime-build;\*.sublime-project;\*.sul

^ Hide Folders

Save

Cancel

New File Ctrl+N

Open File... Ctrl+O

Open Folder...

Open Recent >

Reopen with Encoding >

New View into File

Save Ctrl+S

Save with Encoding >

Save As... Ctrl+Shift+S

Save All

New Window Ctrl+Shift+N

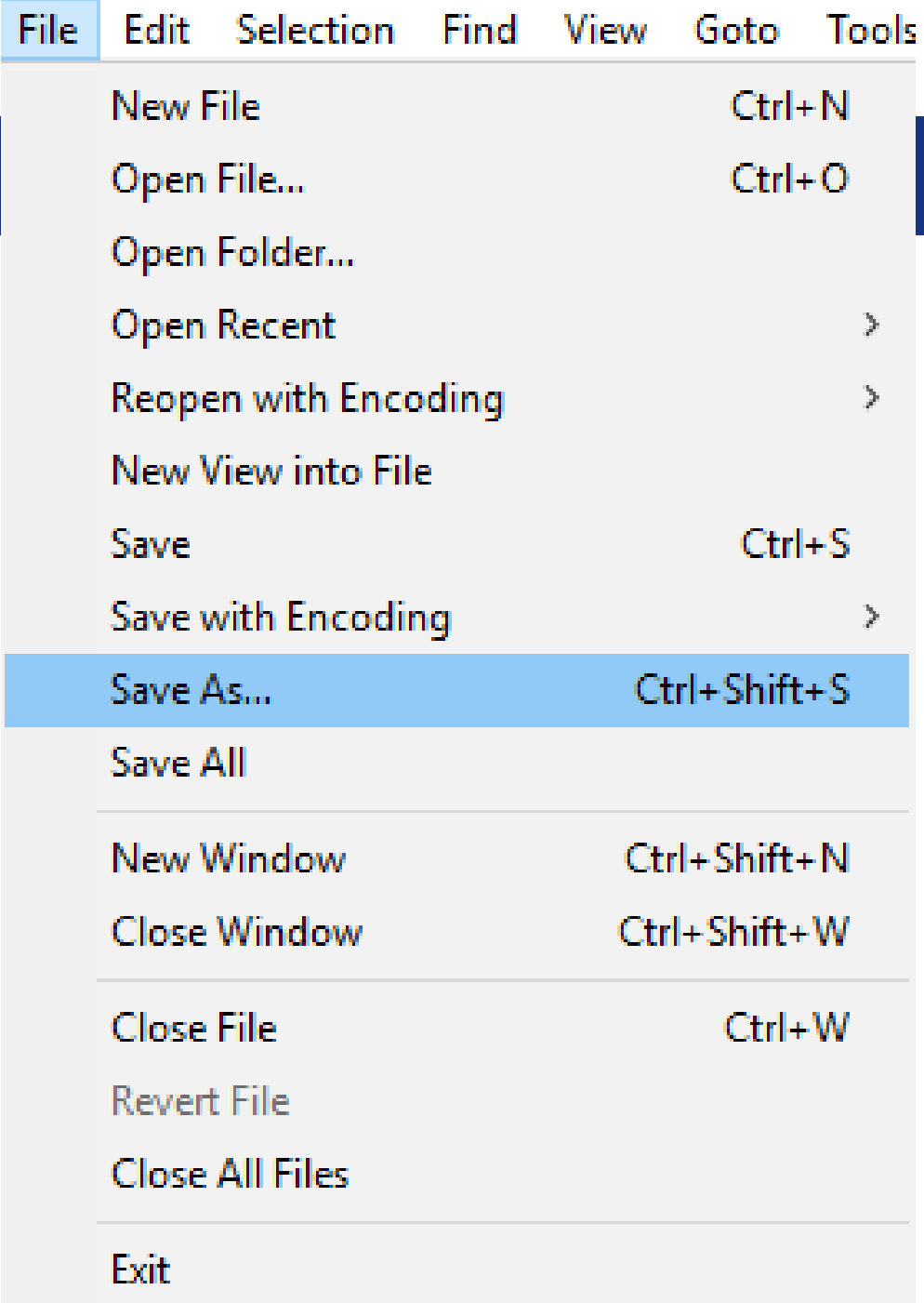
Close Window Ctrl+Shift+W

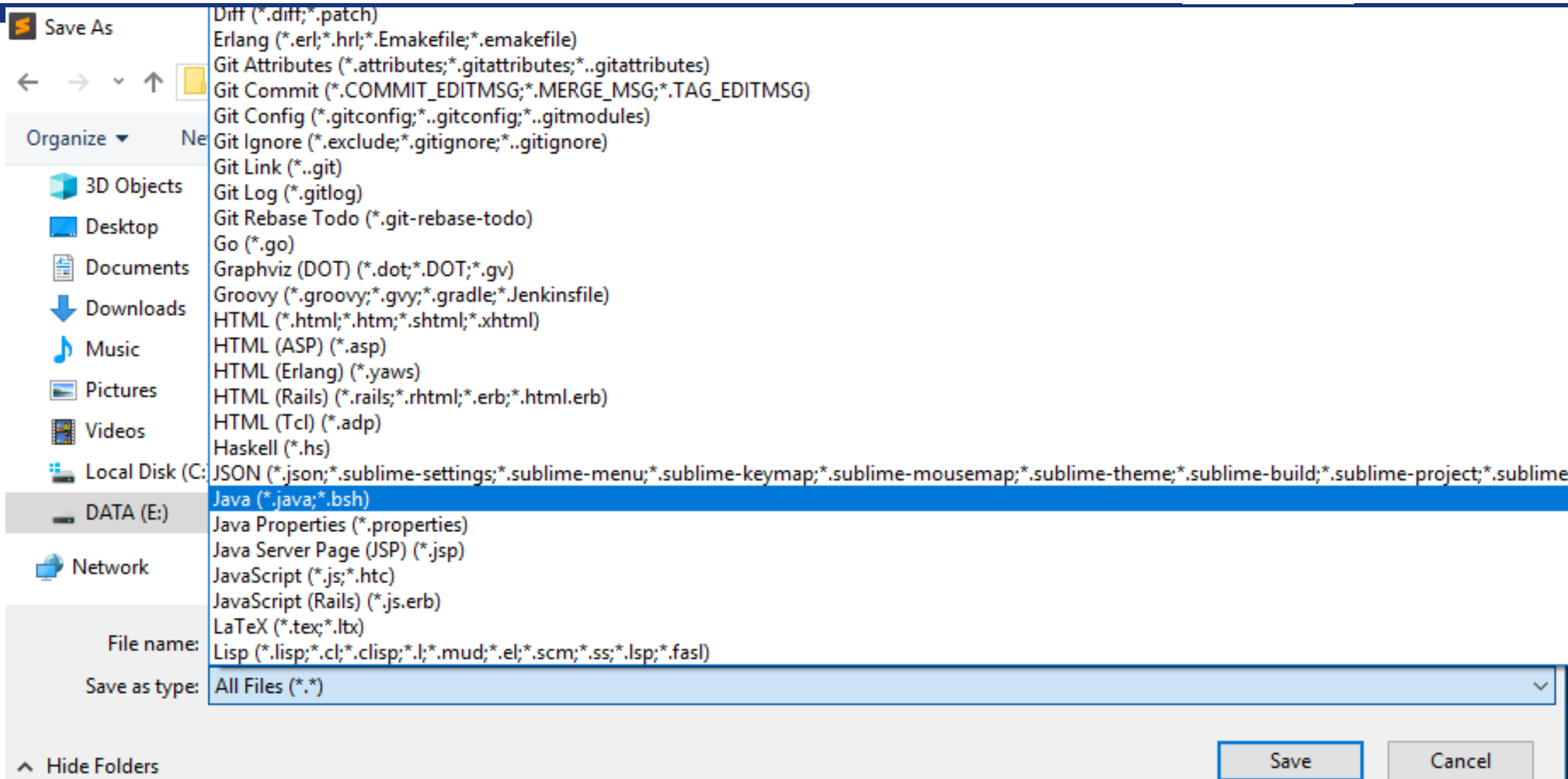
Close File Ctrl+W

Revert File

Close All Files

Exit





Save As

← → ↕ ↑ > This PC > DATA (E:) > MyClass

Search MyClass

Organize ▾ New folder

Desktop  
Documents  
Downloads  
Music  
Pictures  
Videos  
Local Disk (C:)  
DATA (E:)  
DVD RW Drive (F:  
Network

Name

Date modified

Type

Size

No items match your search.

File name: MyClass.java

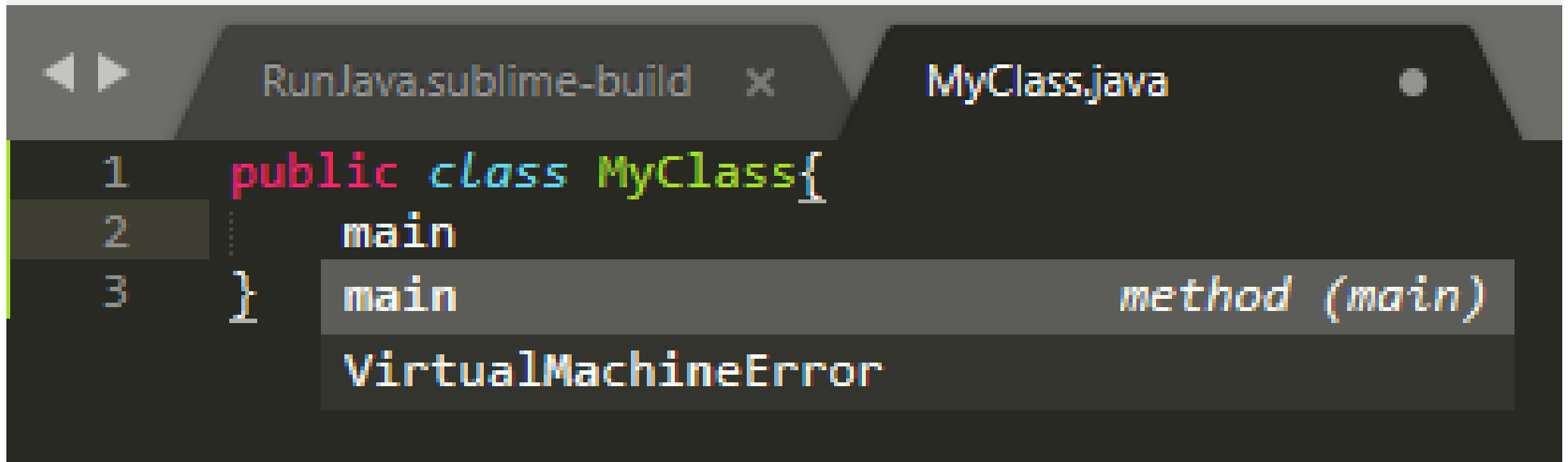
Save as type: Java (\*.java;\*.bsh)

^ Hide Folders

Save

Cancel

File Edit Selection Find View Goto Tools Project Preferenc



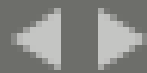
```
1 public class MyClass{
2     main
3 }
```

main method (main)

VirtualMachineError



File Edit Selection Find View Goto Tools Project Preferen

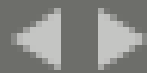


RunJava.sublime-build x

MyClass.java

```
1  public class MyClass{  
2      public static void main(String[] args) {  
3          |  
4      }  
5  }
```

File Edit Selection Find View Goto Tools Project Preferen

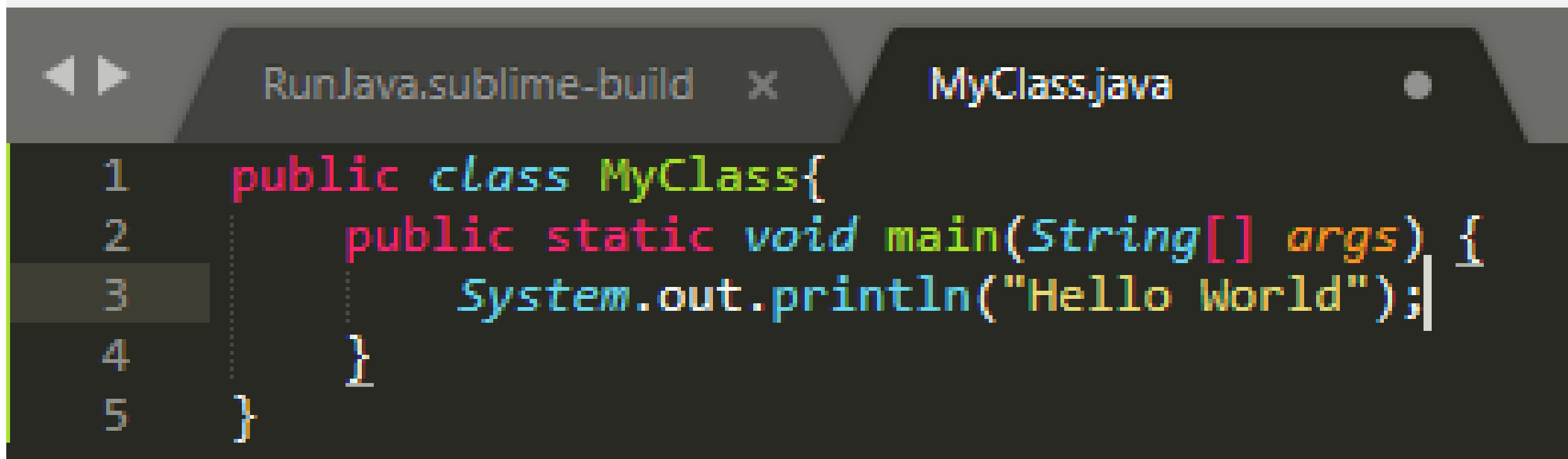


RunJava.sublime-build x

MyClass.java

```
1  public class MyClass{  
2      public static void main(String[] args) {  
3          |  
4      }  
5  }
```

File Edit Selection Find View Goto Tools Project Preferenc



```
1  public class MyClass{
2      public static void main(String[] args) {
3          System.out.println("Hello World");
4      }
5  }
```

```
RunJava.sublime-build x M
1 public class MyClass{
2     public static void mai
3         System.out.println
4     }
5 }
```

- Command Palette... Ctrl+ Shift+ P
- Snippets...
- Build System >
- Build Ctrl+ B
- Build With... Ctrl+ Shift+ B
- Cancel Build Ctrl+ Break
- Build Results >
- ✓ Save All on Build
- Record Macro Ctrl+ Q
- Playback Macro Ctrl+ Shift+ Q
- Save Macro...
- Macros >
- Developer >
- Install Package Control...

- Automatic
- ActionScript
- Ant
- C Single File
- C++ Single File
- Cargo
- D
- D dub
- Erlang
- Haskell
- ✓ JavaC
- Lua
- Make
- Python
- R
- Ruby
- RunJava
- Rust
- ShellScript
- Syntax Tests
- New Build System...

The screenshot shows the Sublime Text editor with a file named 'RunJava.sublime-build' open. The editor contains the following Java code:

```
1 public class MyClass{  
2     public static void main(String[] args){  
3         System.out.println("Hello World");  
4     }  
5 }
```

The 'Tools' menu is open, displaying the following options:

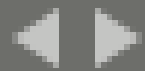
- Command Palette... (Ctrl+Shift+P)
- Snippets...
- Build System** (highlighted with a blue bar and a right arrow)
- Build (Ctrl+B)
- Build With... (Ctrl+Shift+B)
- Cancel Build (Ctrl+Break)
- Build Results (right arrow)
- ☒ Save All on Build
- Record Macro (Ctrl+Q)
- Playback Macro (Ctrl+Shift+Q)
- Save Macro...
- Macros (right arrow)
- Developer (right arrow)
- Install Package Control...

The 'Build System' submenu is open, showing the following options:

- Automatic
- ActionScript
- Ant
- C Single File
- C++ Single File
- Cargo
- D
- D dub
- Erlang
- Haskell
- ☒ JavaC
- Lua
- Make
- Python
- R
- Ruby
- RunJava** (highlighted with a blue bar)
- Rust
- ShellScript
- Syntax Tests
- New Build System...

**Ctrl + B**

File Edit Selection Find View Goto Tools Project Preferences



RunJava.sublime-build x

MyClass.java x

```
1 public class MyClass{
2     public static void main(String[] args) {
3         System.out.println("Hello World");
4     }
5 }
```

```
Hello World
[Finished in 1.0s]
```

# Sintaks Java

# Pelatihan

## Sintaks Java

kode berikut untuk mencetak "Hello World" ke layar:

```
public class MyClass {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

## Java Comment

## Pelatihan

- **Java Comments**

1. Komentar dapat digunakan untuk menjelaskan kode Java, dan membuatnya lebih mudah dibaca. Juga dapat digunakan untuk mencegah eksekusi ketika menguji kode alternatif.
2. Komentar baris tunggal dimulai dengan dua garis miring (//).
3. Teks apa pun antara // dan akhir baris diabaikan oleh Java (tidak akan dieksekusi).

- **Java Multi-line Comments**

1. Komentar banyak baris dimulai dengan / \* dan diakhiri dengan \* /.
2. Teks apa pun antara / \* dan \* / akan diabaikan oleh Java.

```
// This is a comment
```

```
/* The code below will print the words Hello World  
to the screen, and it is amazing */  
System.out.println("Hello World");
```



RunJava.sublime-build x

MyClass.java x

```
1  public class MyClass{
2      public static void main(String[] args) {
3          // This is a comment
4          /* The code below will print the words Hello World
5             to the screen, and it is amazing */
6          System.out.println("Hello World");
7      }
8  }
```

```
Hello World
[Finished in 1.1s]
```

## Konsep Variabel

## Pelatihan

- **Variabel Java**

Variabel adalah wadah untuk menyimpan nilai data.

Ada berbagai jenis variabel, misalnya:

1. String - menyimpan teks, seperti "Halo". Nilai string ditandai oleh tanda kutip ganda
2. int - Integer (bilangan bulat), tanpa desimal, seperti 123 atau -123
3. float - menyimpan angka floating point, dengan desimal, seperti 19,99 atau -19,99
4. char - menyimpan karakter tunggal, seperti 'a' atau 'B'. Nilai-nilai Char ditandai oleh tanda kutip tunggal
5. boolean - menyimpan nilai dengan dua status: benar atau salah

## Konsep Variabel

## Pelatihan

- **Mendeklarasikan (Membuat) Variabel**

1. Untuk membuat variabel, Anda harus menentukan tipe dan memberinya nilai:

Sintaksis: *type variable = value;*

2. Untuk membuat variabel yang harus menyimpan teks, lihat contoh berikut:  
Contoh Buat variabel bernama nama tipe String dan berikan nilai "John":

```
String name = "John";  
System.out.println(name);
```

Contoh Lain :

```
int myNum = 5;  
float myFloatNum = 5.99f;  
char myLetter = 'D';  
boolean myBool = true;  
String myText = "Hello"
```

```
public class MyClass{  
    public static void main(String[] args) {  
        // This is a comment  
        /* The code below will print the words Hello World  
        to the screen, and it is amazing */  
        System.out.println("Hello World");  
  
        String name = "john";  
        System.out.println(name);  
        int myNum = 5;  
        System.out.println(myNum);  
        float myFloatNum = 5.99f;  
        System.out.println(myFloatNum);  
        char myLetter = 'D';  
        System.out.println(myLetter);  
        boolean myBool = true;  
        System.out.println(myBool);  
        String myText = "Hello";  
        System.out.println(myText);  
    }  
}
```

```
Hello World
```

```
john
```

```
5
```

```
5.99
```

```
D
```

```
true
```

```
Hello
```

```
[Finished in 1.2s]
```

## Konsep Variabel

## Pelatihan

- **Menampilkan Variabel**

1. Metode `println ()` sering digunakan untuk menampilkan variabel.
2. Untuk menggabungkan teks dan variabel, gunakan karakter `+`:

```
String name = "John";  
System.out.println("Hello " + name);
```

```
String name = "john";  
System.out.println("Hello " + name);
```

```
Hello john
```

## Konsep Variabel

## Pelatihan

- Java Identifiers
  1. Semua variabel Java harus diidentifikasi dengan nama unik.
  2. Nama-nama unik ini disebut **identifiers**.
  3. **identifiers** dapat berupa nama pendek (seperti x dan y) atau nama yang lebih deskriptif (usia, jumlah, total Volume).

Aturan umum untuk membuat nama untuk variabel (unique identifiers) adalah:

1. Nama dapat berisi huruf, angka, garis bawah, dan tanda dolar
2. Nama harus dimulai dengan huruf
3. Nama juga dapat dimulai dengan \$ dan \_ (tetapi tidak dalam tutorial ini)
4. Case-sensitive ("myVar" dan "myvar" adalah variabel yang berbeda)
5. Nama harus dimulai dengan huruf kecil dan tidak boleh mengandung spasi
6. Kata-kata yang dicadangkan (seperti kata kunci Java, seperti int atau String) tidak dapat digunakan sebagai nama



## Konsep Variabel

## Pelatihan

- Tipe Data Java

Jenis Tipe data dibagi menjadi dua kelompok:

1. Tipe data primitif - termasuk byte, short, int, long, float, double, boolean dan char
2. Tipe data non-primitif - seperti String, Array, dan Class

- Jenis Tipe data Primitif

1. Tipe data primitif menentukan ukuran dan jenis nilai variabel, tidak memiliki metode tambahan.
2. Ada delapan tipe data primitif di Jawa:

# Konsep Variabel

# Pelatihan

Data Type	Size	Description
byte	1 byte	Stores whole numbers from -128 to 127
short	2 bytes	Stores whole numbers from -32,768 to 32,767
int	4 bytes	Stores whole numbers from -2,147,483,648 to 2,147,483,647
long	8 bytes	Stores whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	4 bytes	Stores fractional numbers from $3.4e-038$ to $3.4e+038$ . Sufficient for storing 6 to 7 decimal digits
double	8 bytes	Stores fractional numbers from $1.7e-308$ to $1.7e+038$ . Sufficient for storing 15 decimal digits
boolean	1 byte	Stores true or false values
char	2 bytes	Stores a single character/letter

# Konsep Variabel

# Pelatihan

Contoh :

```
byte myNum = 100;  
short myNum = 5000;  
int myNum = 100000;  
long myNum = 1500000000000L;  
float myNum = 5.75f;  
double myNum = 19.99d;  
boolean isJavaFun = true;  
boolean isFishTasty = false;  
char myGrade = 'B';  
String greeting = "Hello World";
```

## Konsep Variabel

## Pelatihan

- Tipe Data Non-Primitif

Tipe data non-primitif disebut tipe referensi karena merujuk pada objek.

Perbedaan utama antara tipe data primitif dan non-primitif adalah:

1. Tipe primitif sudah ditentukan sebelumnya (sudah ditentukan) di Java. Tipe non-primitif dibuat oleh programmer dan tidak didefinisikan oleh Java (kecuali untuk String).
2. Tipe non-primitif dapat digunakan untuk memanggil metode untuk melakukan operasi tertentu, sedangkan tipe primitif tidak bisa.
3. Tipe primitif selalu memiliki nilai, sedangkan tipe non-primitif bisa menjadi nol.
4. Tipe primitif dimulai dengan huruf kecil, sedangkan tipe non-primitif dimulai dengan huruf besar.
5. Ukuran tipe primitif tergantung pada tipe data, sedangkan tipe non-primitif memiliki semua ukuran yang sama.
6. Contoh tipe non-primitif adalah String, Array, Class, Interface, dll.

# Konsep Variabel

# Pelatihan

- Keyword final

Pengubah non-akses yang digunakan untuk kelas, atribut, dan metode, yang membuatnya tidak dapat diubah (tidak mungkin diwariskan atau ditimpa)

Final pada variable

```
class Lingkaran {  
    final double PI = 3.14;  
}
```

Final Pada Method

```
class Kampus {  
    final void tampil() {  
        System.out.println("I LOVE JAVA")  
    }  
}
```

Final pada class

```
class Kampus {  
    final void tampil() {  
        System.out.println("I LOVE JAVA")  
    }  
}
```

# Konsep Variabel

# Pelatihan

- Operator Java

Operator Aritmatika digunakan untuk melakukan operasi matematika umum.

Operator	Name	Description	Example
+	Addition	Adds together two values	$x + y$
-	Subtraction	Subtracts one value from another	$x - y$
*	Multiplication	Multiplies two values	$x * y$
/	Division	Divides one value from another	$x / y$
%	Modulus	Returns the division remainder	$x \% y$
++	Increment	Increases the value of a variable by 1	$++x$
--	Decrement	Decreases the value of a variable by 1	$--x$

```
int x = 1;  
int y = 2;  
int hasilTambah = x + y;  
int hasilKurang = x - y;  
int hasilKali = x * y;  
System.out.println("x + y = " + hasilTambah);  
System.out.println("x - y = " + hasilKurang);  
System.out.println("x * y = " + hasilKali);
```

$$x + y = 3$$

$$x - y = -1$$

$$x * y = 2$$

## Latihan

## Pelatihan

- Buatlah code untuk menghitung nilai akhir evaluasi pelatihan!
- Nilai akhir =  $(60\% \times \text{Ujian Praktek}) + (30\% \times \text{Ujian Pilihan Ganda}) + (5\% \times \text{Kehadiran}) + (5\% \times \text{Sikap})$
- Nilai Ujian Praktek, Ujian Pilihan Ganda, Kehadiran, dan Sikap berskala 100.



## Kondisi

## Pelatihan

- Kondisi Java dan Pernyataan `if`

Java mendukung kondisi logis yang biasa dari matematika:

1. Kurang dari: `a < b`
2. Kurang dari atau sama dengan: `a <= b`
3. Lebih besar dari: `a > b`
4. Lebih besar atau sama dengan: `a >= b`
5. Sama dengan `a == b`
6. Tidak sama dengan: `a != b`

Anda dapat menggunakan kondisi ini untuk melakukan tindakan berbeda untuk keputusan yang berbeda. Java memiliki pernyataan kondisional berikut:

1. Gunakan `if` untuk menentukan blok kode yang akan dieksekusi, jika kondisi yang ditentukan benar
2. Gunakan `else` untuk menentukan blok kode yang akan dieksekusi, jika kondisi yang sama salah
3. Gunakan `else if` menentukan kondisi baru untuk diuji, jika kondisi pertama salah
4. Gunakan `switch` menentukan banyak blok kode alternatif yang akan dieksekusi

## Kondisi

## Pelatihan

- Pernyataan if

```
int x = 20;  
int y = 18;  
if (x > y) {  
    System.out.println("x is greater than y");  
}
```

- Pernyataan else

```
int time = 20;  
if (time < 18) {  
    System.out.println("Good day.");  
} else {  
    System.out.println("Good evening.");  
}
```

## Kondisi

## Pelatihan

- Pernyataan else if

```
int time = 22;  
if (time < 10) {  
    System.out.println("Good morning.");  
} else if (time < 20) {  
    System.out.println("Good day.");  
} else {  
    System.out.println("Good evening.");  
}
```

## Latihan

- Buatlah code untuk menentukan bilangan ganjil atau genap!
- Buatlah code untuk membandingkan dua bilangan, apakah sama atau lebih besar salah satunya!

## Pelatihan

## Latihan

## Pelatihan

- Buatlah code untuk menentukan kriteria kelulusan pelatihan!

Kriteria Kelulusan	Rentang Nilai Akhir
Memuaskan	Nilai Akhir $\geq 95$
Baik Sekali	$85 \leq \text{Nilai Akhir} < 95$
Baik	$75 \leq \text{Nilai Akhir} < 85$
Cukup	$65 \leq \text{Nilai Akhir} < 75$
Tidak Lulus	Nilai Akhir $< 65$

## Kondisi

## Pelatihan

- Pernyataan Switch Java

Gunakan pernyataan `switch` untuk memilih salah satu dari banyak blok kode yang akan dieksekusi.

1. Ekspresi `switch` dievaluasi sekali.
2. Nilai ekspresi dibandingkan dengan nilai setiap kasus.
3. Jika ada kecocokan, blok kode terkait dijalankan.
4. `break` dan kata kunci `default` adalah opsional.

Kata kunci `break`

1. Ketika Java mencapai kata kunci `break`, maka keluar dari blok `switch`.
2. Ini akan menghentikan pelaksanaan lebih banyak kode dan pengujian kasus di dalam blok.
3. Ketika kecocokan ditemukan, dan pekerjaan selesai, saatnya `break`(dihentikan). Tidak perlu pengujian lebih lanjut.
4. `Break` dapat menghemat banyak waktu eksekusi karena "mengabaikan" eksekusi semua kode lainnya di blok `switch`.

- Kata Kunci `default`

Kata kunci `default` untuk dijalankan jika tidak ada kecocokan .

## Kondisi

## Pelatihan

Contoh :

```
int day = 4;
switch (day) {
    case 6:
        System.out.println("Today is Saturday");
        break;
    case 7:
        System.out.println("Today is Sunday");
        break;
    default:
        System.out.println("Looking forward to the
Weekend");
}
```

# Perulangan

# Pelatihan

- Perulangan `while`

Perulangan `while` dikerjakan selama kondisi yang ditentukan benar.

```
int i = 0;
while (i < 5) {
    System.out.println(i);
    i++;
}
```

- Perulangan `Do/While`

1. Do / while loop adalah varian dari perulangan `while`.
2. Perulangan ini akan mengeksekusi blok kode sekali, sebelum memeriksa apakah kondisinya benar, akan mengulang loop selama kondisinya benar.

```
int i = 0;
do {
    System.out.println(i);
    i++;
}
while (i < 5);
```



## Perulangan

## Pelatihan

- Perulangan For

```
for (int i = 0; i <= 10; i = i + 2) {  
    System.out.println(i);  
}
```

1. Pernyataan 1 dieksekusi (satu kali) sebelum eksekusi blok kode. Menetapkan variabel sebelum perulangan dimulai (`int i = 0`).
2. Pernyataan 2 mendefinisikan kondisi untuk mengeksekusi blok kode. Mendefinisikan kondisi untuk menjalankan perulangan ( harus kurang dari sama dengan 10). Jika kondisinya benar, perulangan akan memulai lagi, jika itu salah, perulangan akan berakhir.
3. Pernyataan 3 dieksekusi (setiap kali) setelah blok kode dieksekusi. Meningkatkan nilai (`i ++`) setiap kali blok kode dalam perulangan telah dieksekusi.

## Perulangan

## Pelatihan

- Perulangan For-Each

Perulangan **for-each**, digunakan secara eksklusif untuk mengulang elemen-elemen dalam array

```
String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
for (String i : cars) {
    System.out.println(i);
}
```

- Java break pada perulangan for

```
for (int i = 0; i < 10; i++) {
    if (i == 4) {
        break;
    }
    System.out.println(i);
}
```

## Latihan

- Buatlah code untuk menentukan akar bilangan bulat positif!

## Pelatihan

## Latihan

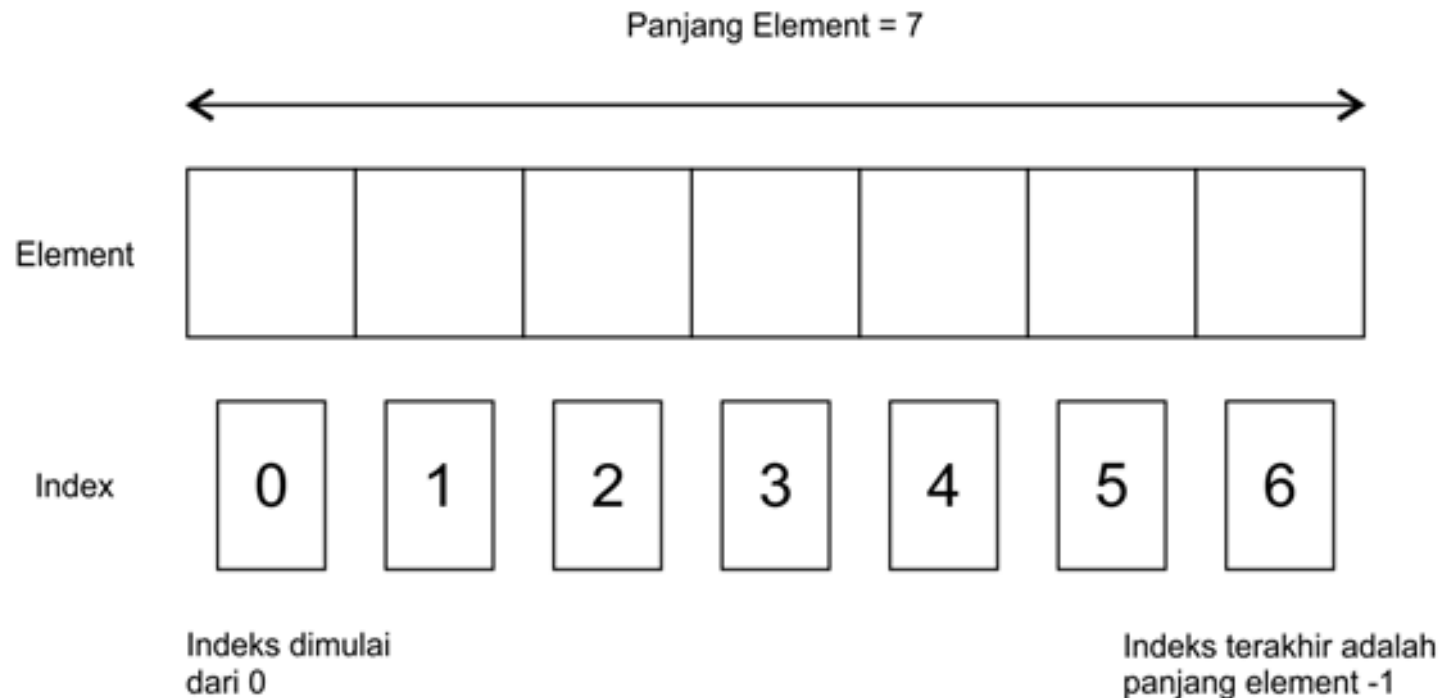
- Buatlah code untuk menampilkan deret bilangan prima, dengan batas bawah dan batas atas tertentu.
- Contoh:
- Batas bawah: 6
- Batas atas: 20
- Deret bilangan prima: 7, 11, 13, 17, 19

## Pelatihan

# Array

## Pelatihan

- Array adalah objek yang bisa digunakan untuk menyimpan kumpulan data lebih dari satu dengan tipe sama. Array memiliki jumlah data yang fixed (tetap).



# Array

# Pelatihan

- Cara penulisan array

```
// Cara pertama  
int[] arrA = new int[6];
```

```
// Cara kedua  
int arrB[] = new int[6];
```

# Array

# Pelatihan

- Inisiasi array

```
// Cara pertama
int[] arrA = new int[6];
arrA[0] = 1;
arrA[1] = 2;
arrA[2] = 3;
arrA[3] = 4;
arrA[4] = 5;
arrA[5] = 6;
System.out.println(arrA[0]);
System.out.println(arrA[1]);
System.out.println(arrA[2]);
System.out.println(arrA[3]);
System.out.println(arrA[4]);
System.out.println(arrA[5]);
```

# Array

# Pelatihan

- Inisiasi array

```
for (int i=0;i<6;i++) {  
    System.out.println(arrA[i]);  
}
```



# Array

# Pelatihan

- Inisiasi array

```
// Cara kedua  
int[] arrInt = {1, 2, 3, 4, 5, 6};  
for (int i=0;i<6;i++) {  
    System.out.println(arrInt[i]);  
}
```

# Pemrograman Berorientasi Objek

## Berorientasi Objek?



### Attribute:

Topi, Baju, Jaket,  
Tas Punggung,  
Tangan, Kaki, Mata

### Behavior:

Cara Jalan ke Depan  
Cara Jalan Mundur  
Cara Belok ke Kiri  
Cara Memanjat

## Berorientasi Objek?



### Attribute (State):

Ban, Stir, Pedal Rem, Pedal Gas,  
Warna, Tahun Produksi

### Behavior:

Cara Menghidupkan Mesin  
Cara Manjalankan Mobil  
Cara Memundurkan Mobil

Attribute → Variable(Member)

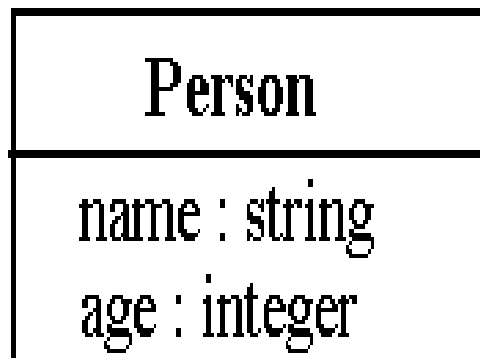
Behavior → Method(Fungsi)

## Perbedaan Class dan Object

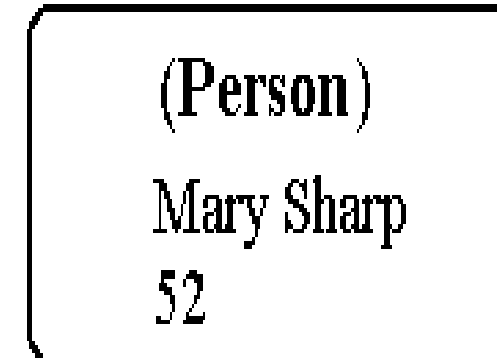
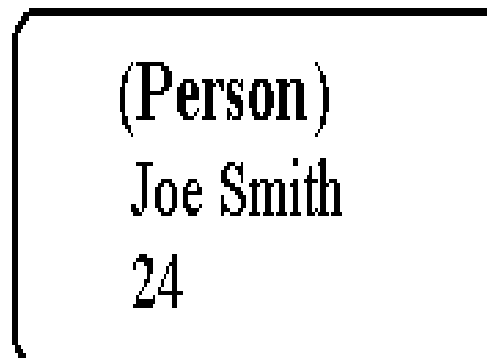
- Class: **konsep** dan **deskripsi** dari sesuatu
  - Class mendeklarasikan **method** yang dapat digunakan (dipanggil) oleh object
- Object: **instance dari class**, bentuk (contoh) nyata dari class
  - Object memiliki sifat **independen** dan dapat digunakan untuk memanggil method
- Contoh Class dan Object:
  - Class: **mobil**
  - Object: **mobilnya pak Joko, mobilku, mobil berwarna merah**

## Perbedaan Class dan Object

- Class seperti **cetakan kue**, dimana kue yg dihasilkan dari cetakan kue itu adalah **object**
- Warna kue bisa bermacam-macam meskipun berasal dari cetakan yang sama (**object memiliki sifat independen**)



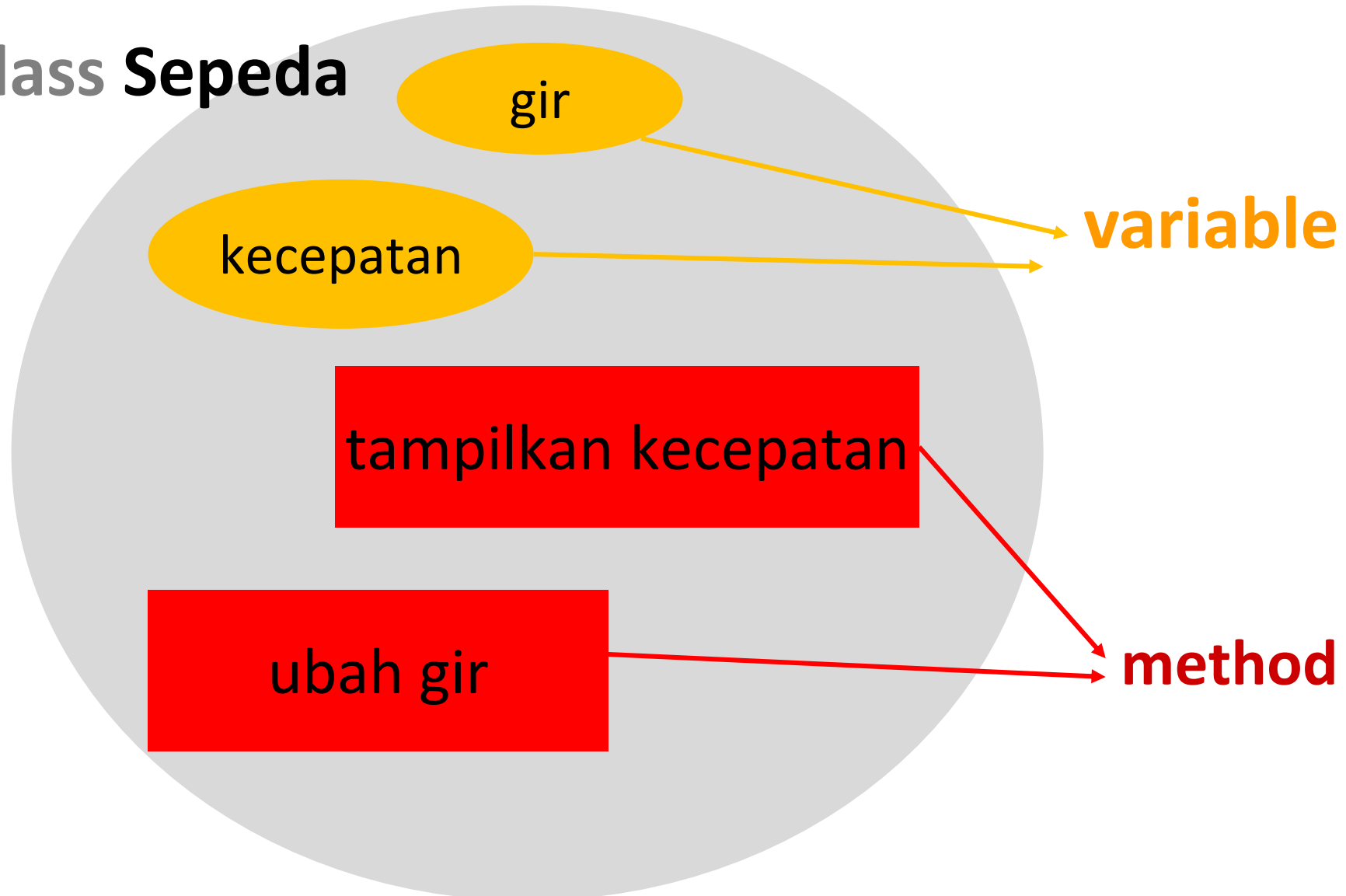
Class with Attributes



Objects with Values

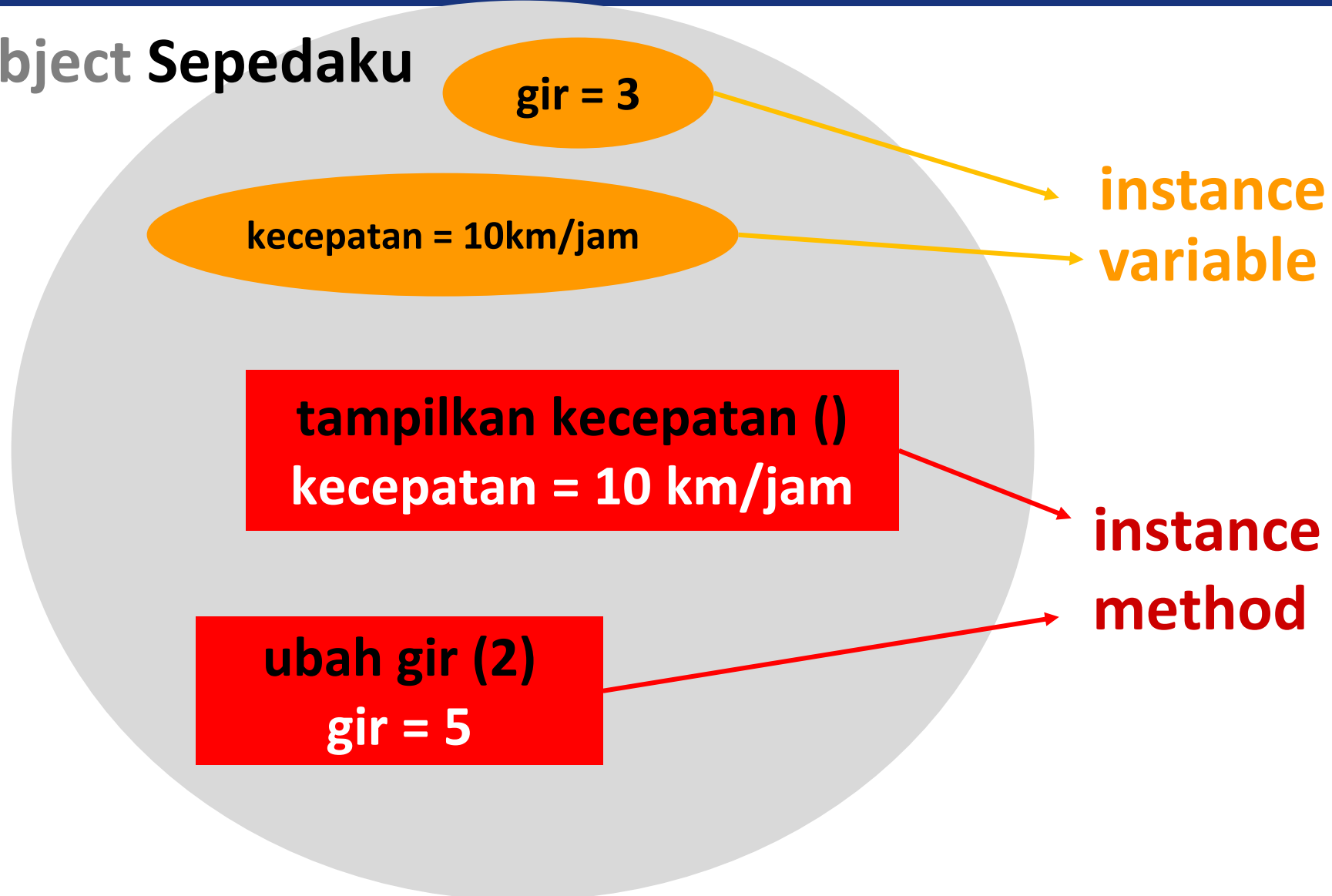
# Class = Method + Variable

## Class Sepeda



# Object = Method + Variable yg Memiliki Nilai

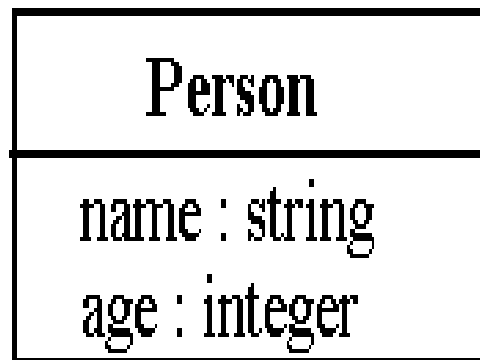
## Object Sepedaku



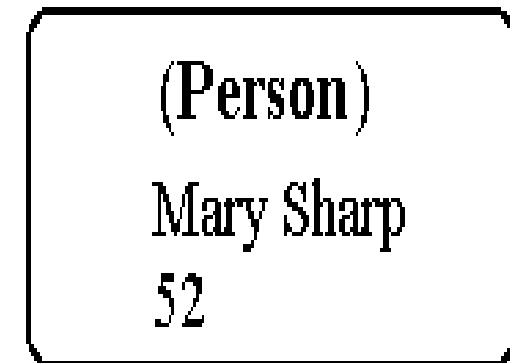
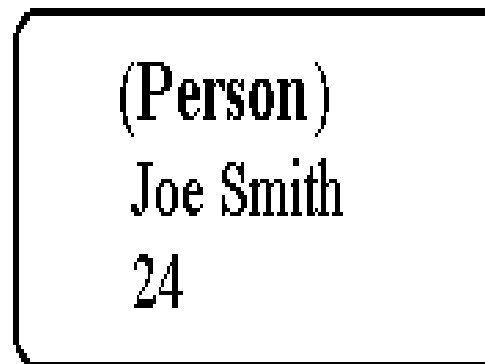


## Attribute

- **Variable** yang mengitari class, dengan **nilai datanya bisa ditentukan di object**
- Variable digunakan untuk **menyimpan nilai** yang nantinya akan digunakan pada program
- Variable memiliki **jenis (tipe), nama dan nilai**
- Name, age, dan weight adalah atribute (variabel) dari class Person



**Class with Attributes**



**Objects with Values**

```
public class Mobil{  
    String warna;  
    int tahunProduksi;  
}
```

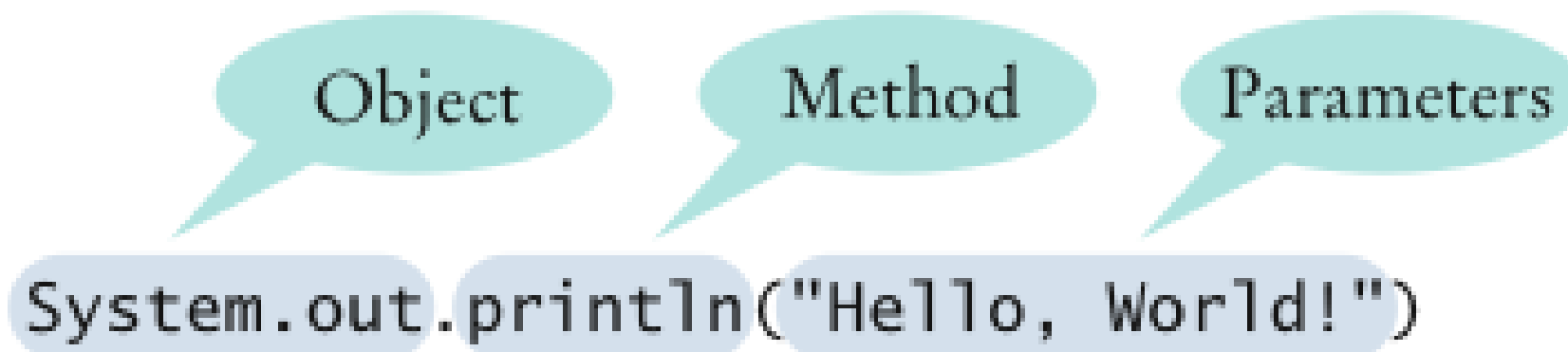
Mobil.java

```
public class MobilBeraksi{  
    public static void main(String[] args){  
        // Membuat object  
        Mobil mobilku = new Mobil();  
  
        /* memanggil atribut dan memberi nilai */  
        mobilku.warna = "Hitam";  
        mobilku.tahunProduksi = 2006;  
        System.out.println("Warna: " + mobilku.warna);  
        System.out.println("Tahun: " + mobilku.tahunProduksi);  
    }  
}
```

MobilBeraksi.java

## Method

- Method adalah **urutan instruksi** yang mengakses data dari object
- Method melakukan:
  1. **Manipulasi data**
  2. **Perhitungan matematika**
  3. **Memonitor kejadian** dari suatu event



# Method

**Syntax**    *object.methodName(parameters)*

**Example**

The method is  
invoked on this object.

This is the  
name of the method.

This parameter is  
passed to the method.

System.out.println("Hello")

Parameters are enclosed in parentheses.  
Multiple parameters are separated by commas.

# Membuat dan Memanggil Method

```
public class Mobil2{
```

```
    String warna;
```

```
    int tahunProduksi;
```

```
    void printMobil(){
```

```
        System.out.println("Warna: " + warna);
```

```
        System.out.println("Tahun: " + tahunProduksi);
```

```
    }
```

```
}
```

Mobil2.java

```
public class Mobil2Beraksi{
```

```
    public static void main(String[] args){
```

```
        Mobil2 mobilku = new Mobil2();
```

```
        mobilku.warna = "Hitam";
```

```
        mobilku.tahunProduksi = 2006;
```

```
        mobilku.printMobil();
```

```
    }
```

```
}
```

Mobil2Beraksi.java

# Membuat dan Memanggil Method

**Syntax**     *accessSpecifier returnType methodName(parameterType parameterName, . . . )*  
              {  
              *method body*  
              }

## Example

These methods  
are part of the  
public interface.

```
public void deposit(double amount)
{
    balance = balance + amount;
}
```

This method does  
not return a value.

A mutator method modifies  
an instance variable.

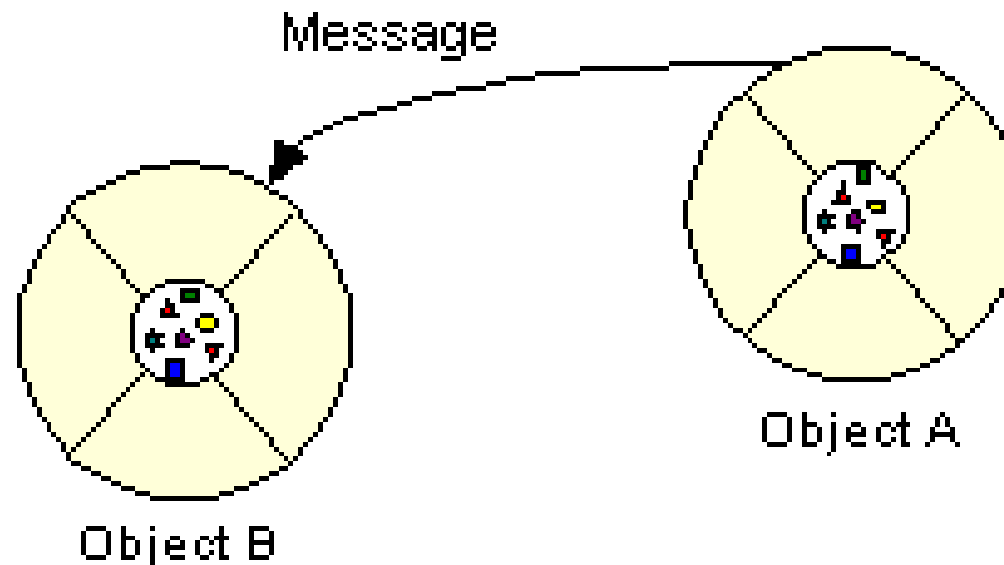
```
public double getBalance()
{
    return balance;
}
```

This method has  
no parameters.

An accessor method returns a value.

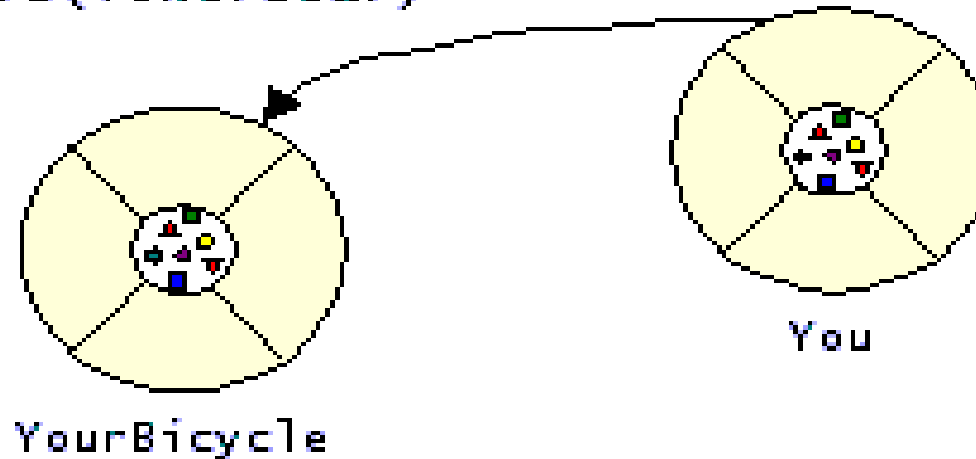
## Parameter

- Sepeda akan berguna apabila ada object lain **yang berinteraksi dengan sepeda tersebut**
- Object software berinteraksi dan berkomunikasi dengan object lain dengan cara mengirimkan **message atau pesan**
- Pesan adalah suatu method, dan informasi dalam pesan dikenal dengan nama **parameter**



# Pengiriman Pesan dan Parameter

`changeGears(lowerGear)`



1. **You** → object pengirim
2. **YourBicycle** → object penerima
3. **changeGears** → pesan berupa method yang dijalankan
4. **lowerGear** → parameter yang dibutuhkan method (pesan) untuk dijalankan



# Sepeda.java

```
public class Sepeda{  
    int gir;  
  
    // method (mutator) dengan parameter  
    void setGir(int pertambahanGir) {  
        gir= gir+ pertambahanGir;  
    }  
  
    // method (accessor)  
    int getGir() {  
        return gir;  
    }  
}
```

# SepedaBeraksi.java

```
public class SepedaBeraksi{  
    public static void main(String[] args) {  
        Sepeda sepedaku = new Sepeda();  
  
        sepedaku.setGir(1); // menset nilai gir = 1 (sebelumnya 0)  
        System.out.println("Gir saat ini: " + sepedaku.getGir());  
  
        sepedaku.setGir(3); // menambahkan 3 pada posisi gir saat ini (1)  
  
        System.out.println("Gir saat ini: " + sepedaku.getGir());  
    }  
}
```

## Latihan: Class Matematika dan Parameter

- Buat Class bernama **Matematika**, yang berisi method dengan **dua parameter**:
  - pertambahan(**int a**, **int b**)
  - pengurangan(**int a**, **int b**)
  - perkalian(**int a**, **int b**)
  - pembagian(**int a**, **int b**)
- Buat Class bernama **MatematikaBeraksi**, yang mengeksekusi method dan menampilkan:
  - Pertambahan:  $20 + 20 = 40$
  - Pengurangan:  $10 - 5 = 5$
  - Perkalian:  $10 * 20 = 200$
  - Pembagian:  $21 / 2 = 10.5$

## Konstruktor

- Method yang digunakan untuk memberi nilai awal pada saat object diciptakan
- Dipanggil secara otomatis ketika **new** digunakan untuk membuat instan class
- Sifat konstruktor:
  - Nama konstruktor **sama dengan nama class**
  - **Tidak memiliki nilai balik** dan tidak boleh ada kata kunci void

```
public class Mobil {  
    String warna;  
    int tahunProduksi;  
    public Mobil(String warna, int tahunProduksi){  
        this.warna = warna;  
        this.tahunProduksi = tahunProduksi;  
    }  
    public void info(){  
        System.out.println("Warna: " + warna);  
        System.out.println("Tahun: " + tahunProduksi);  
    }  
}
```

```
public class MobilKonstruktor{  
    public static void main(String[] args){  
        Mobil mobilku = new Mobil("Merah", 2003);  
        mobilku.info();  
    }  
}
```

## Kata Kunci this

Digunakan pada pembuatan class dan digunakan untuk  
**menyatakan object sekarang**

```
public class Mobil{  
    String warna;  
    int tahunProduksi;  
  
    void isiData(String aWarna,  
                  int aTahunProduksi){  
  
        warna = aWarna;  
        tahunProduksi = aTahunProduksi;  
    }  
}
```

```
public class Mobil{  
    String warna;  
    int tahunProduksi;  
  
    void isiData(String warna,  
                  int tahunProduksi){  
  
        this.warna = warna;  
        this.tahunProduksi = tahunProduksi;  
    }  
}
```

# Latihan

1. Buat class **Bank**
  - Buat konstruktor class Bank dengan parameter: **saldo**
  - Buat method: **simpanUang**, **ambilUang**, dan **getSaldo**
2. Buat class **BankBeraksi**, tetapkan saldo awal lewat konstruktor Rp. 100000, jalankan 3 method di atas, dan tampilkan proses sebagai berikut:

Selamat Datang di Bank ABC  
Saldo saat ini: Rp. 100000

Simpan uang: Rp. 500000  
Saldo saat ini: Rp. 600000

Ambil uang: Rp. 150000  
Saldo saat ini: Rp. 450000

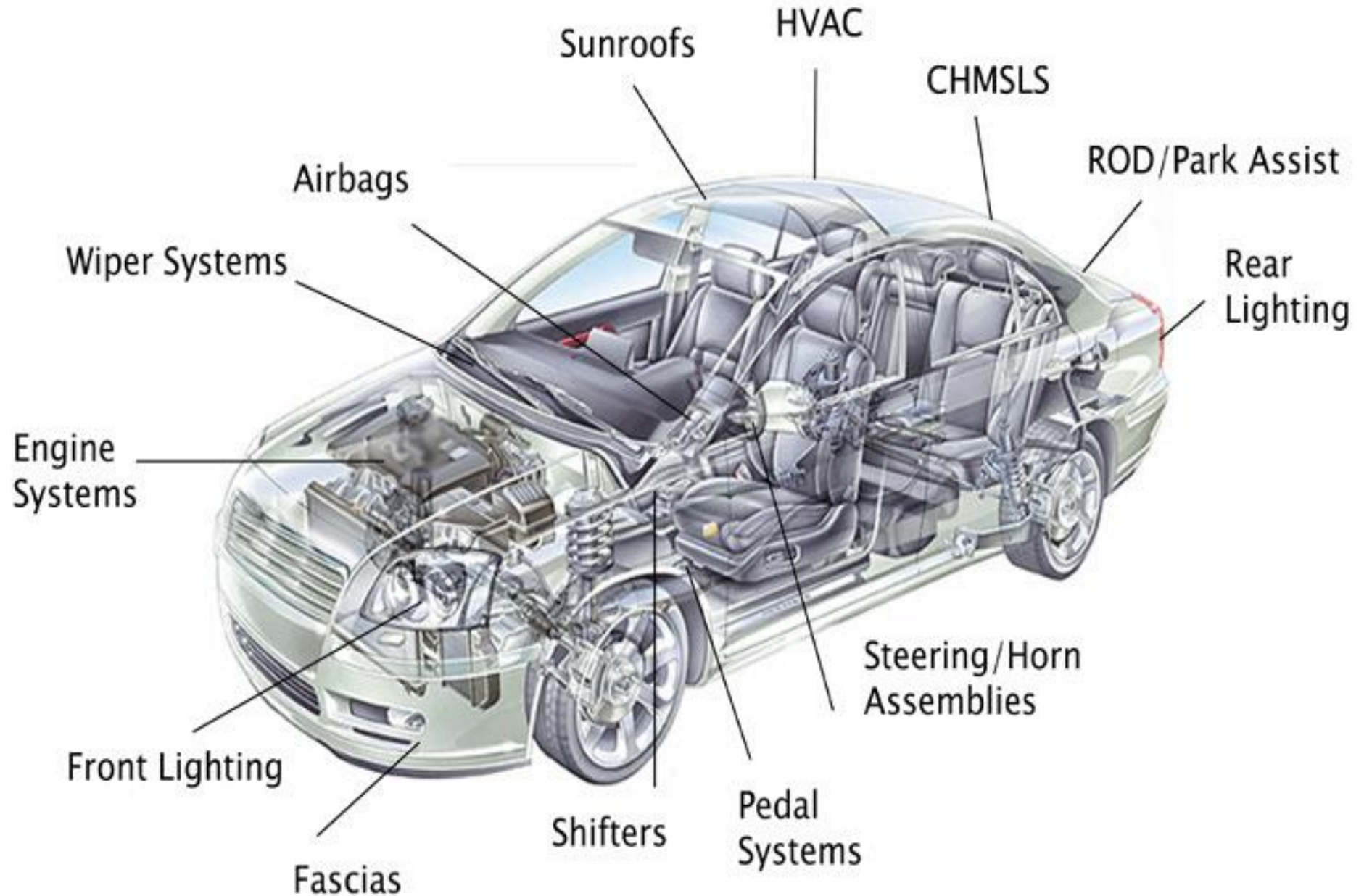
# Karakteristik Pemrograman Berorientasi Objek



# Abstraction

- Cara kita melihat suatu sistem dalam **bentuk yang lebih sederhana**, yaitu sebagai suatu kumpulan subsistem (object) yang saling berinteraksi.
  - Mobil adalah kumpulan sistem pengapian, sistem kemudi, sistem pengereman
- Alat meng-abstraksikan sesuatu adalah **class**
- Object bersifat **modularity**. Object dapat ditulis dan **dimaintain terpisah (independen)** dari object lain

# Abstraction



# Abstraction

## BICYCLE



# Encapsulation

- Mekanisme **menyembunyikan suatu proses dan data dalam sistem** untuk menghindari interferensi, dan menyederhanakan penggunaan proses itu sendiri
  - Tongkat transmisi (gigi) pada mobil
  - Tombol on/off/pengaturan suhu pada AC
- Class access level (public, protected, private) adalah **implementasi dari konsep encapsulation**
- Enkapsulasi data dapat dilakukan dengan cara:
  1. mendeklarasikan **instance variable** sebagai **private**
  2. mendeklarasikan **method** yang sifatnya **public** untuk mengakses variable tersebut

# Encapsulation

```
Class Lingkaran{  
    void buatLingkaran(){  
        for(){  
            Garis.buatGaris()  
        }  
    }  
}
```

```
class Garis{  
    private void  
    buatTitik(x, y){  
    }  
    public void buatGaris(tA, tB){  
  
    }  
}
```



# Encapsulation dan Access Modifier

Modifier	Dalam Class yang Sama	Dalam Package yang Sama	Dalam SubClass	Dalam Package Lain
private	✓			
tanpa tanda	✓	✓		
protected	✓	✓	✓	
public	✓	✓	✓	✓

# Encapsulation

- Enkapsulasi data juga dapat dilakukan dengan cara:
  1. mendeklarasikan **instance variable** sebagai **private**
  2. mendeklarasikan **method** yang sifatnya **public** untuk mengakses variable tersebut

*Syntax*

```
accessSpecifier class ClassName
{
    instance variables
    constructors
    methods
}
```

*Example*

```
public class Counter
{
    private int value;

    public Counter(double initialValue) { value = initialValue; }
    public void count() { value = value + 1; }
    public int getValue() { return value; }
}
```

Public interface

Private implementation

```
public class Sepeda{  
    int gir;  
  
    void setGir(int pertambahanGir) {  
        gir= gir+ pertambahanGir;  
    }  
  
    int getGir() {  
        return gir;  
    }  
}
```



# SepedaBeraksi.java

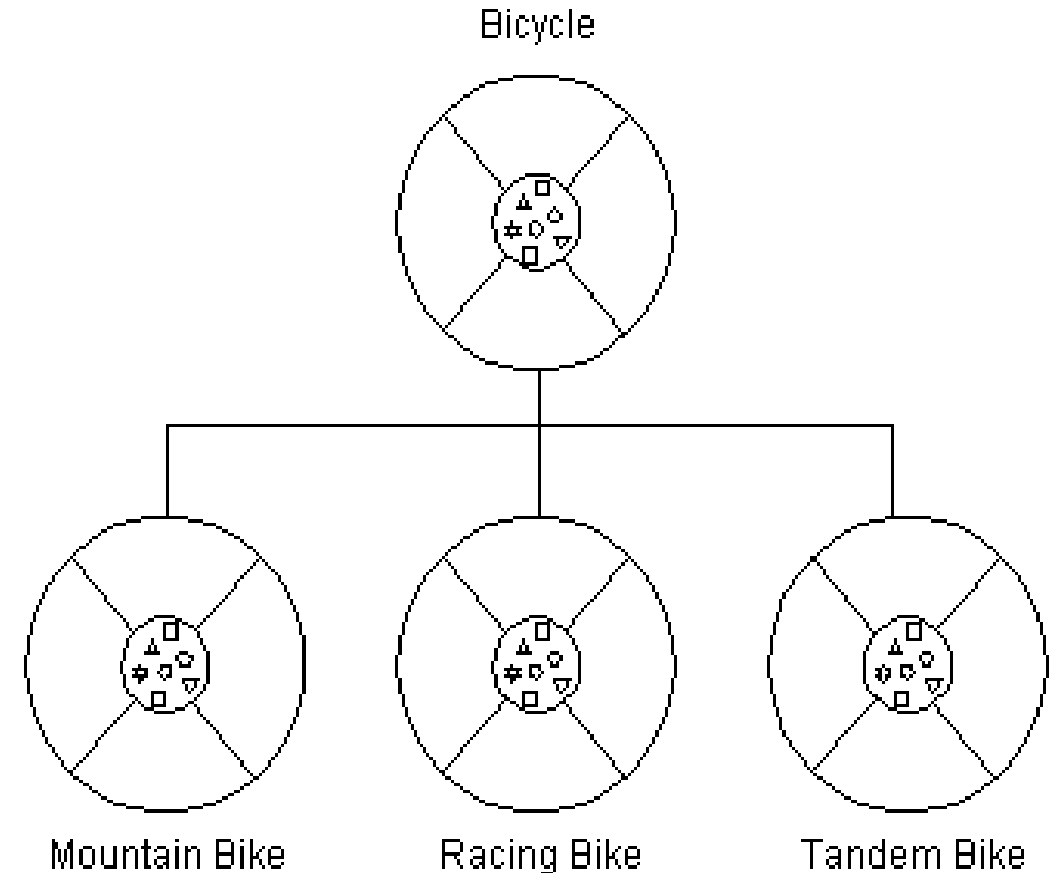
```
public class SepedaBeraksi{
    public static void main(String[] args) {
        Sepeda sepedaku = new Sepeda();

        sepedaku.setGir(1);
        /* Variabel bisa diubah atau tidak sengaja diubah.
           Hal ini berbahaya dan sering menimbulkan bug.
           Berikan access modifier private pada instance variable */
        sepedaku.gir = 3;
        System.out.println("Gir saat ini: " + sepedaku.getGir());
    }
}
```

```
public class Sepeda{  
    private int gir; // access modifier private pada instance variable  
  
    void setGir(int pertambahanGir) {  
        gir= gir+ pertambahanGir;  
    }  
  
    int getGir() {  
        return gir;  
    }  
}
```

# Inheritance (Pewarisan)

- Suatu class dapat **mewariskan atribut dan method** kepada class lain (subclass), serta membentuk class hierarchy
- Penting untuk **Reusability**
- Java Keyword: **extends**



```
public class Sepeda{  
    private int gir;  
  
    void setGir(int pertambahanGir) {  
        gir= gir+ pertambahanGir;  
    }  
  
    int getGir() {  
        return gir;  
    }  
}
```

# Class SepedaGunung Mewarisi Class Sepeda

```
public class SepedaGunung extends Sepeda{

    private int sadel;

    void setSadel (int jumlah) {
        sadel = getGir() - jumlah;
    }

    int getSadel(){
        return sadel;
    }
}
```

```
public class SepedaGunungBeraksi {
    public static void main(String[] args) {

        SepedaGunung sg=new SepedaGunung();

        sg.setGir(3);
        System.out.println(sg.getGir());

        sg.setSadel(1);
        System.out.println(sg.getSadel());
    }
}
```

## Latihan: Inheritance Matematika

1. Buat class **MatematikaCanggih** yang merupakan **inherit** dari class **Matematika**
  1. Tambahkan method **modulus(int a, int b)** yang menghitung modulus dari a dan b
  2. Operator modulus adalah %
2. Buat class **MatematikaCanggihBeraksi** yang memanggil method pertambahan, perkalian dan modulus

# Polymorphism

- Kemampuan untuk **memperlakukan object yang memiliki perilaku (bentuk) yang berbeda**
- Implementasi konsep polymorphism:
  1. **Overloading**: Kemampuan untuk menggunakan **nama yang sama** untuk beberapa **method yang berbeda parameter** (tipe dan atau jumlah)
  2. **Overriding**: Kemampuan subclass untuk **menimpa method** dari superclass, yaitu dengan cara menggunakan nama dan parameter yang sama pada method

# Polymorphism – Overloading

```
class Mobil {  
    String warna;  
    int tahunProduksi;  
  
    public Mobil(String warna, int tahunProduksi){  
        this.warna = warna;  
        this.tahunProduksi = tahunProduksi;  
    }  
  
    public Mobil(){  
    }  
  
    void info(){  
        System.out.println("Warna: " + warna);  
        System.out.println("Tahun: " + tahunProduksi);  
    }  
}
```

```
public class MobilKonstruktor{  
    public static void main(String[] args){  
        Mobil mobilku = new Mobil("Merah", 2003);  
        mobilku.info();  
  
        Mobil mobilmu = new Mobil();  
        mobilmu.info();  
    }  
}
```



# Polymorphism – Overloading

KEMENTERIAN  
REPUBLIC OF INDONESIA  
Manajemen Manaja

```
class Lingkaran{  
    void gambarLingkaran(){  
    }  
    void gambarLingkaran(int diameter){  
    ...  
    }  
    void gambarLingkaran(double diameter){  
    ...  
    }  
    void gambarLingkaran(int diameter, int x, int y){  
    ...  
    }  
    void gambarLingkaran(int diameter, int x, int y, int warna, String  
namaLingkaran){  
    ...  
    }  
}
```

# Polymorphism - Overriding

```
public class Sepeda{  
    private int gir;  
  
    void setGir(int pertambahanGir) {  
        gir= gir+ pertambahanGir;  
    }  
  
    int getGir() {  
        return gir;  
    }  
}
```

# Polymorphism - Overriding

KEMENTERIAN  
REPUBLIK INDONESIA  
Manajemen Masyarakat Informatika

```
public class SepedaGunung extends Sepeda{  
  
    void setGir(int pertambahanGir) {  
        super.setGir(pertambahanGir);  
        gir = 2*getGir();  
    }  
}
```

SepedaGunung.java

```
public class SepedaGunungBeraksi {  
    public static void main(String[] args) {  
  
        SepedaGunung sg=new SepedaGunung();  
  
        sg.setGir(2);  
        System.out.println(sg.getGir());  
  
        sg.setGir(3);  
        System.out.println(sg.getGir());  
    }  
}
```

SepedaGunungBeraksi.java

## Latihan: Overloading pada Matematika

1. Kembangkan class **Matematika**, **MatematikaCanggih** dan **MatematikaBeraksi**
2. Lakukan **overloading** pada **Method** yang ada (pertambahan, pengurangan, perkalian, pembagian, modulus)
3. Tambahkan method baru bertipe data **double** (pecahan) dan **memiliki 3 parameter**
4. Uji di kelas **MatematikaBeraksi** dengan parameter pecahan: 12.5, 28.7, 14.2
5. Uji konsep overloading dengan:  
pertambahan(12.5, 28.7, 14.2)      pertambahan(12, 28, 14)  
pertambahan(23, 34)                      pertambahan(3.4, 4.9)

# Matematika.java

KEMENTERIAN  
REPUBLIK INDONESIA  
Manajemen Masyarakat Informasi Indonesia

```
public class Matematika{  
    void pertambahan (int a, int b){  
        int hasil= a + b;  
        System.out.println("hasil:" + hasil);  
    }  
  
    void pertambahan (double a, double b, double c){  
        double hasil= a + b + c;  
        System.out.println("hasil:" + hasil);  
    }  
    ...  
}
```

# Konsep Layout

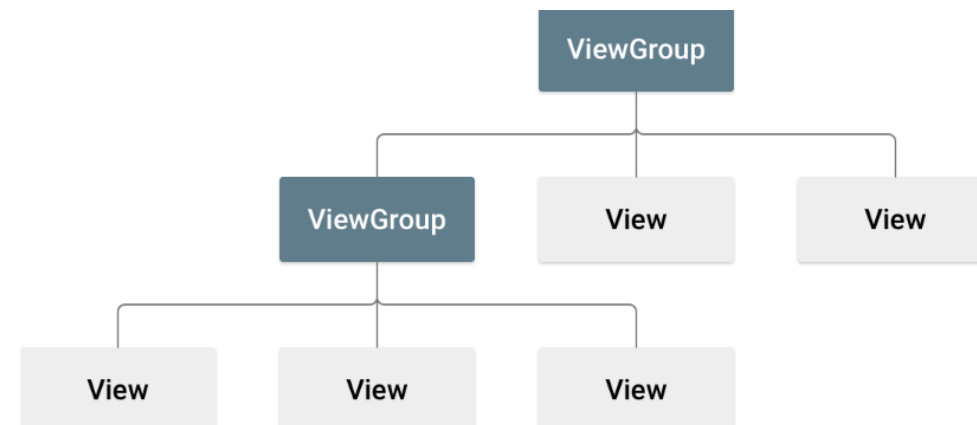
## Dasar Layout

## Pelatihan

- Layout

Layout mendefinisikan struktur visual untuk antarmuka pengguna, seperti UI sebuah aktivitas atau widget aplikasi Anda dapat mendeklarasikan layout dengan dua cara:

- 1. Deklarasikan elemen UI dalam XML.** Android menyediakan sebuah kosakata XML sederhana yang sesuai dengan kelas dan subkelas View, seperti halnya untuk widget dan layout.
- 2. Buat instance elemen layout saat runtime.** Dalam suatu aplikasi bisa membuat objek View dan ViewGroup (dan memanipulasi propertinya) lewat program.



## Deklarasikan elemen UI dalam XML

## Pelatihan

### Keuntungan mendeklarasikan UI dalam XML

1. Memungkinkan Anda memisahkan penampilan aplikasi dari kode yang mengontrol perilakunya dengan lebih baik.
2. Keterangan UI Anda bersifat eksternal bagi kode aplikasi Anda, yang berarti bahwa Anda bisa memodifikasi atau menyesuaikannya tanpa harus memodifikasi dan mengompilasi ulang kode sumber.
3. Misalnya, Anda bisa membuat layout XML untuk berbagai orientasi layar, berbagai ukuran layar perangkat, dan berbagai bahasa



## Layout - XML

## Pelatihan

File --> New --> Project...

Choose Android --> Android Application Project

Edit res/layout/activity\_main.xml, and replace everything with the following:

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

</LinearLayout>
```

## Layout - XML

## Pelatihan

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a Button" />
</LinearLayout>
```

## Layout - XML

## Pelatihan

- **Compile aplikasi**

1. Masing-masing file layout XML akan dikompilasi dalam sebuah sumber daya View.
2. Harus memuat sumber daya layout dari kode aplikasi, dalam implementasi callback Activity.onCreate().
3. Lakukan dengan memanggil setContentView(), dengan meneruskan acuan ke sumber daya layout berupa:  
`R.layout.layout_file_name`.

Misalnya, jika XML layout Anda disimpan sebagai `main_layout.xml`, Anda akan memuatnya untuk Activity seperti ini:

```
public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.main_layout);  
}
```

## Layout - XML

## Pelatihan

- Attributes

1. Setiap objek View dan ViewGroup mendukung variasi atribut XML-nya sendiri.
2. Sebagian atribut bersifat spesifik untuk objek View (misalnya, TextView mendukung atribut **textSize** ).
3. Sebagian atribut bersifat umum untuk semua objek View, karena diwarisi dari kelas Root View (seperti atribut **id**).
4. Atribut lain dianggap sebagai "parameter layout" yaitu atribut yang menjelaskan orientasi layout tertentu dari objek View, seperti yang didefinisikan oleh objek ViewGroup induk dari objek itu.

## Layout - XML

## Pelatihan

- ID

1. Objek View apa saja dapat memiliki ID integer yang dikaitkan dengannya, untuk mengidentifikasi secara unik `View` dalam pohon.
2. Bila aplikasi dikompilasi, ID ini akan diacu sebagai integer, namun ID biasanya ditetapkan dalam file XML layout sebagai string, dalam atribut `id`. Ini atribut XML yang umum untuk semua objek `View` (yang didefinisikan oleh kelas `View`) dan Anda akan sering sekali menggunakannya

```
android:id="@+id/my_button"
```

## Layout - XML

## Pelatihan

- **Parameter Layout**

Atribut layout XML bernama *layout\_something* mendefinisikan parameter layout View yang cocok untuk ViewGroup tempatnya berada.

Semua grup tampilan berisi lebar dan tinggi (*layout\_width* dan *layout\_height*), dan masing-masing tampilan harus mendefinisikannya. Banyak *LayoutParams* yang juga menyertakan *margin* dan *border* opsional.

Salah satu konstanta ini untuk mengatur lebar atau tinggi:

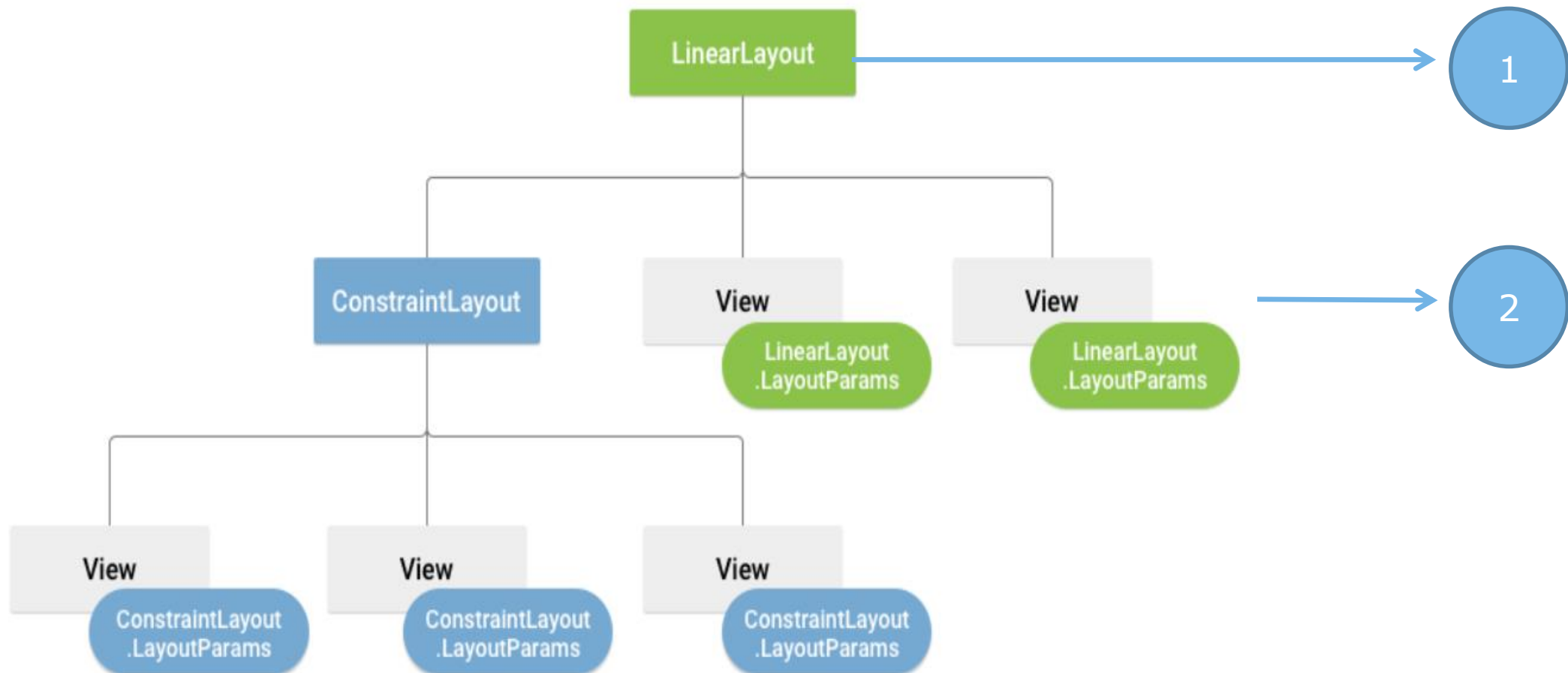
1. ***wrap\_content*** memberi tahu tampilan agar menyesuaikan sendiri ukurannya dengan dimensi yang dibutuhkan oleh materinya.
2. ***match\_parent*** memberi tahu tampilan agar menjadi sebesar yang akan diperbolehkan oleh kelompok tampilan induknya.

## Layout - Struktur

## Pelatihan

- **Parameter Layout**

1. Grup tampilan *root*.
2. Rangkaian tampilan anak dan grup tampilan pertama yang induknya

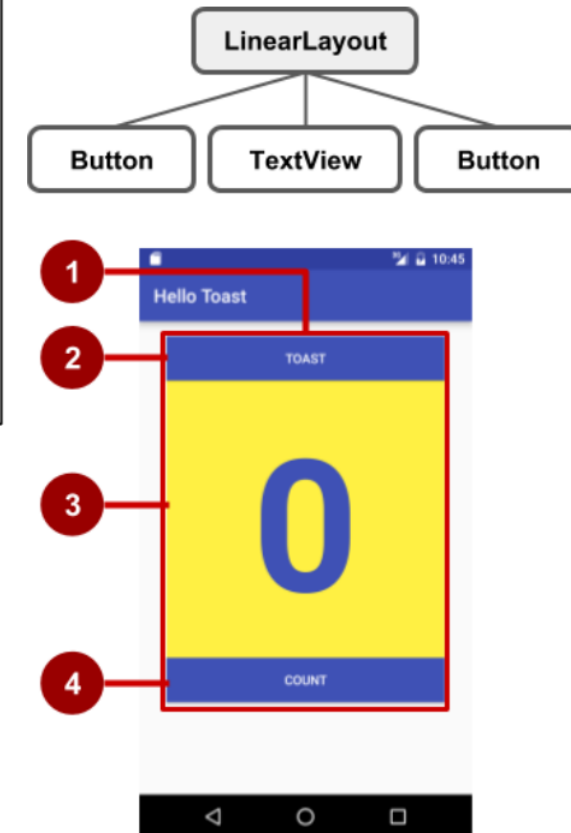
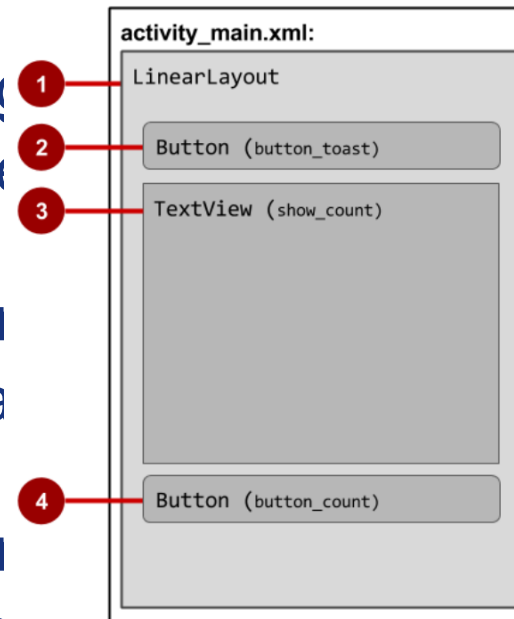


# LinearLayout

## Pelatihan

### Linear Layout

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.

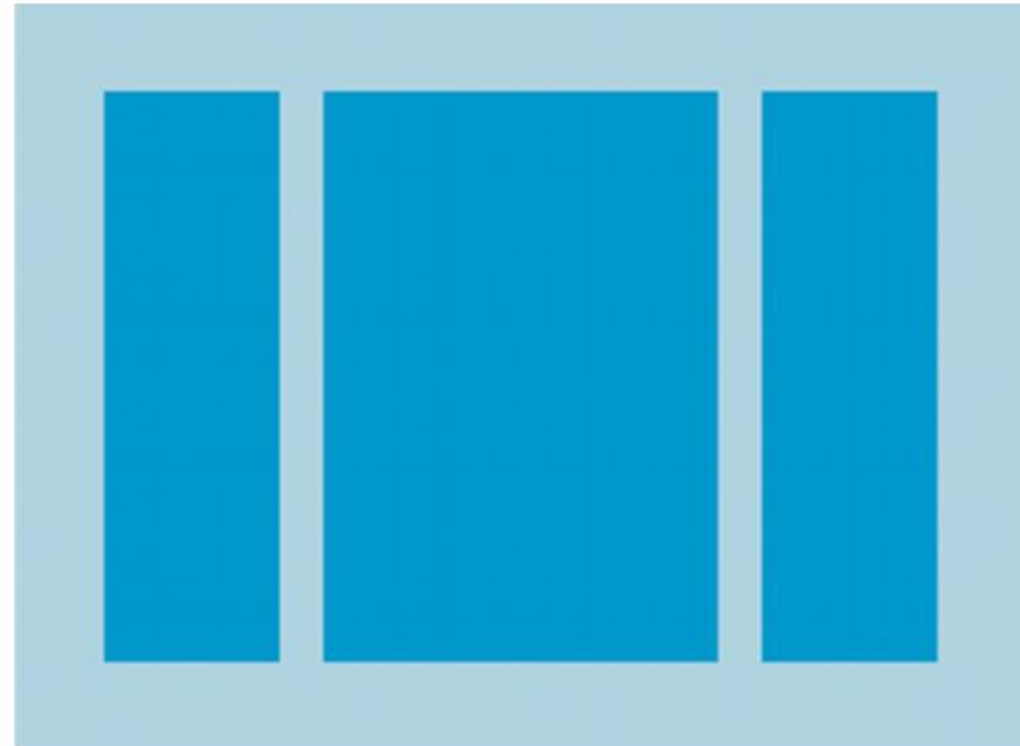




# Linearlayout

## Pelatihan

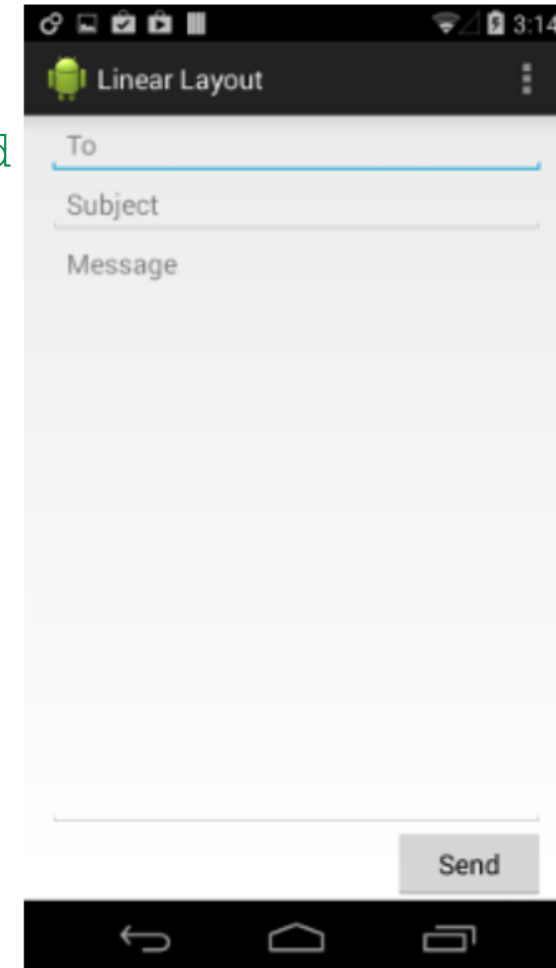
- Linear Layout
  1. LinearLayout adalah sekelompok tampilan yang menyejajarkan semua anak dalam satu arah, secara vertikal atau horizontal.
  2. Menetapkan arah layout dengan atribut `android:orientation`.



# Layout-Linier Layout

## Pelatihan

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:orientation="vertical" >
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/to" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/subject" />
```



# Layout-Linear Layout

## Pelatihan

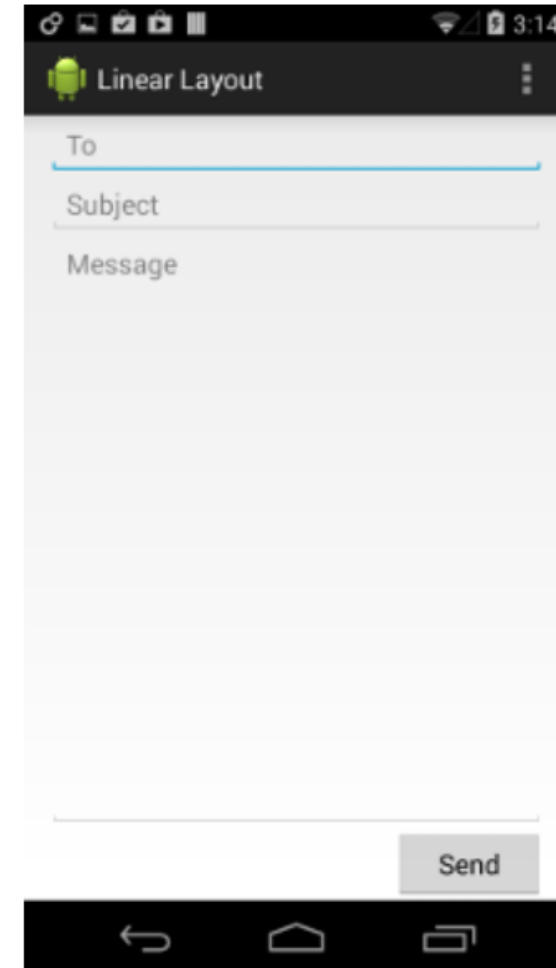
<EditText

```
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:gravity="top"  
    android:hint="@string/message" />
```

<Button

```
    android:layout_width="100dp"  
    android:layout_height="wrap_content"  
    android:layout_gravity="right"  
    android:text="@string/send" />
```

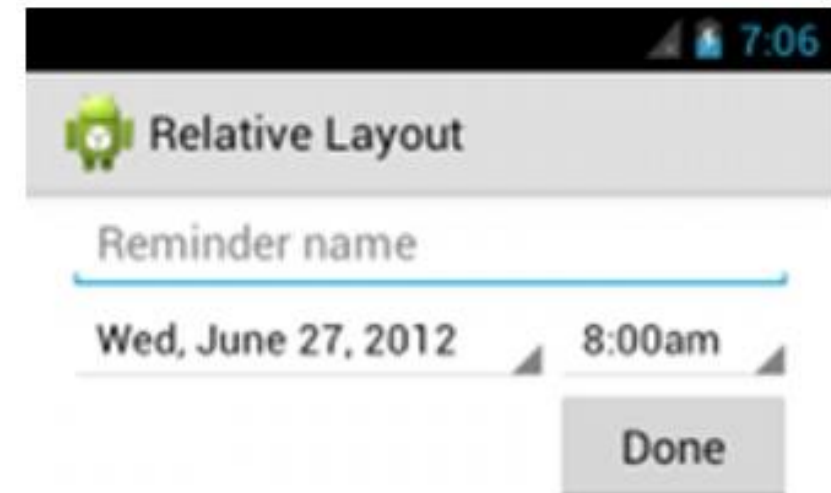
</LinearLayout>



## Relatif Layout

## Pelatihan

- Relatif Layout
  1. Grup tampilan anak yang setiap tampilannya diposisikan dan disejajarkan relatif terhadap tampilan dalam grup tampilan.
  2. Dengan kata lain, posisi tampilan anak bisa dijelaskan dalam hubungan satu sama lain atau dengan grup tampilan induk.



# Relatif Layout

# Pelatihan

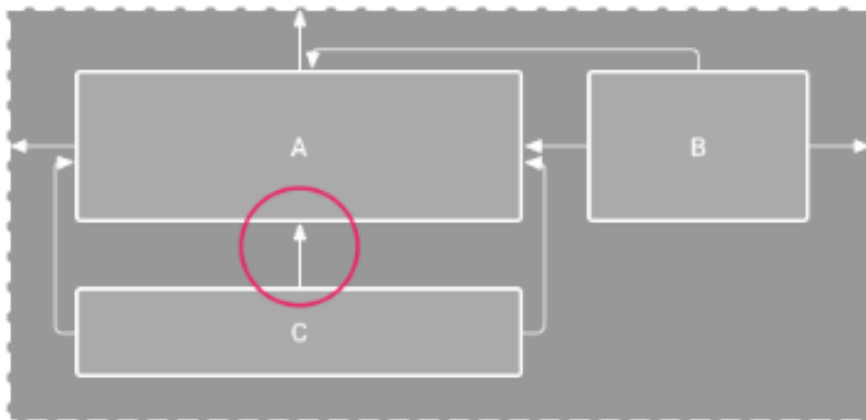
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingRight="16dp" >
    <EditText
        android:id="@+id/name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/reminder" />
    <Spinner
        android:id="@+id/dates"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_below="@id/name"
        android:layout_alignParentLeft="true"
        android:layout_toLeftOf="@+id/times" />
```

```
<Spinner
    android:id="@id/times"
    android:layout_width="96dp"
    android:layout_height="wrap_content"
    android:layout_below="@id/name"
    android:layout_alignParentRight="true"
/>
<Button
    android:layout_width="96dp"
    android:layout_height="wrap_content"
    android:layout_below="@id/times"
    android:layout_alignParentRight="true"
    android:text="@string/done"
/>
</RelativeLayout>
```

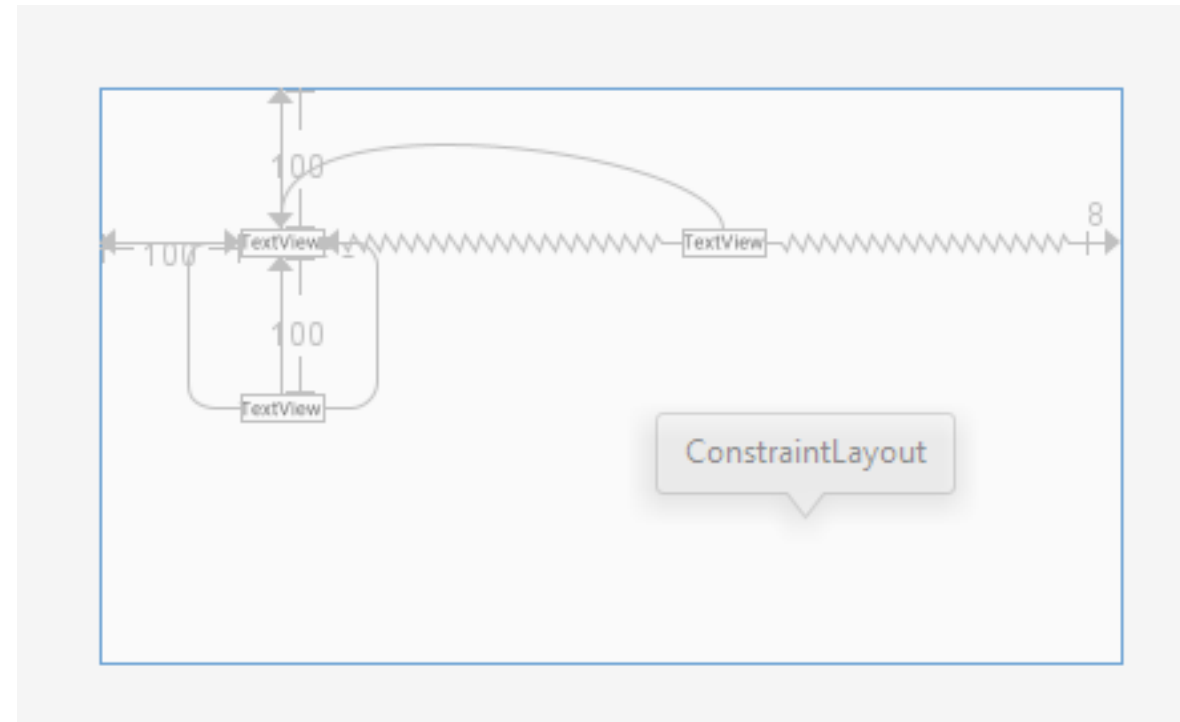
# Constraint Layout

## Pelatihan

1. Bangun tampilan yang responsive.
2. Mirip seperti layout relatif, dimana tampilan bergantung dengan view lain atau parent layoutnya.
3. Didukung oleh android studio, gunakan drag and drop daripada mengedit kode xml.



Contoh constraint layout



# Constraint Layout

# Pelatihan

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.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=".MainActivity">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="100dp"
        android:layout_marginLeft="100dp"
        android:layout_marginTop="100dp"
        android:text="TextView"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="100dp"
        android:text="TextView"
        app:layout_constraintStart_toEndOf="@+id/textView1"
        app:layout_constraintTop_toTopOf="@+id/textView1" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginRight="8dp"
        android:text="TextView"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.503"
        app:layout_constraintStart_toEndOf="@+id/textView1"
        app:layout_constraintTop_toTopOf="@+id/textView1" />
</android.support.constraint.ConstraintLayout>
```

## KESIMPULAN

1. Peserta mengetahui Kategori dan Jenis Bahasa Pemrograman
2. Peserta mengetahui cara mengatur Ruang Kerja
3. Peserta mengetahui Dasar Alur Pembuatan Software
4. Peserta mengetahui Konsep Variabel dan Konstanta
5. Peserta mengetahui Struktur Kondisi dan Perulangan
6. Peserta mengetahui Konsep Layout
7. Peserta bisa membuat sebuah aplikasi sederhana



# Referensi

1. THE WORLD'S LARGEST WEB DEVELOPER SITE, Java Tutorial, *di akses 27/04/2019*  
<https://www.w3schools.com/java/default.asp>
2. Developer Google-Android developer guides (Doc), User Interface & Navigation, *di akses 27/04/2019*  
<https://developer.android.com/guide/topics/ui>,
3. Tim Pelatihan Developer Google, Kursus Dasar-Dasar Developer Android-Konsep, Creative Commons Attribution-NonCommercial 4.0 International License, Desember 2016, *di akses 27/04/2019*, [https://google-developer-training.github.io/android-developer-fundamentals-course-concepts/idn/Unit%201/12\\_c\\_layouts\\_views\\_and\\_resources.html](https://google-developer-training.github.io/android-developer-fundamentals-course-concepts/idn/Unit%201/12_c_layouts_views_and_resources.html)

# Tim Penyusun:

- Alif Akbar Fitrawan, S.Pd, M. Kom (Politeknik Negeri Banyuwangi);
- Anwar, S.Si, MCs. (Politeknik Negeri Lhokseumawe);
- Eddo Fajar Nugroho (BPPTIK Cikarang);
- Eddy Tungadi, S.T., M.T. (Politeknik Negeri Ujung Pandang);
- Fitri Wibowo (Politeknik Negeri Pontianak);
- Ghifari Munawar (Politeknik Negeri Bandung);
- Hetty Meileni, S.Kom., M.T. (Politeknik Negeri Sriwijaya) ;
- I Wayan Candra Winetra, S.Kom., M.Kom (Politeknik Negeri Bali) ;
- Irkham Huda (Vokasi UGM) ;
- Josseano Amakora Koli Parera, S.Kom., M.T. (Politeknik Negeri Ambon) ;
- I Komang Sugiarta, S.Kom., MMSI (Universitas Gunadarma) ;
- Lucia Sri Istiyowati, M.Kom (Institut Perbanas) ;
- Maksy Sendiang, ST, MIT (Politeknik Negeri Manado) ;
- Medi Noviana (Universitas Gunadarma) ;
- Muhammad Nashrullah (Politeknik Negeri Batam) ;
- Nat. I Made Wiryana, S.Si., S.Kom., M.Sc. (Universitas Gunadarma) ;
- Rika Idmayanti, ST, M.Kom (Politeknik Negeri Padang) ;
- Rizky Yuniar Hakkun (Politeknik Elektronik Negeri Surabaya) ;
- Robinson A.Wadu, ST, MT (Politeknik Negeri Kupang) ;
- Roslina. M.IT (Politeknik Negeri Medan) ;
- Sukamto, SKom., MT. (Politeknik Negeri Semarang) ;
- Syamsi Dwi Cahya, M.Kom. (Politeknik Negeri Jakarta) ;
- Syamsul Arifin, S.Kom, M.Cs (Politeknik Negeri Jember) ;
- Usmanudin (Universitas Gunadarma) ;
- Wandy Alifha Saputra (Politeknik Negeri Banjarmasin) ;