

Pemograman dengan Java

Introduction

Apa itu Java?

- Java adalah bahasa yang “general purpose, concurrent, class-based, dan object-oriented” .
- Pembuat Java adalah James Gosling .
- Java merupakan sebuah bahasa yang bersifat WORA (write once, run anywhere) yang mana kita menuliskan sebuah kode program yang dapat berjalan di banyak platform.
- Java memiliki automatic garbage collector yang berfungsi untuk mengelola memori secara otomatis.

Mengapa Java?

- Merupakan sebuah bahasa pemograman yang portable
- Memiliki library yang paling lengkap
- Merupakan bahasa yang masih populer hingga kini
- Merupakan bahasa resmi membuat aplikasi android

Tools untuk Java!

- Java Development Kit (JDK) adalah tools yang digunakan seorang developer dalam proses developing, debugging, dan monitoring suatu aplikasi Java. Dalam JDK juga terdapat Java Runtime Environment yang digunakan untuk menjalankan suatu aplikasi Java.
- Develop adalah proses penyusunan baris per baris kode sampai menjadi satu aplikasi utuh yang memiliki suatu fungsi.
- Debug adalah proses mencari dan menghilangkan eror tersebut di dalam aplikasi untuk kemudian memperbaikinya.
- Monitor adalah proses untuk mengamati jalannya aplikasi dari yang kita buat.
- Integrated Development Environment (IDE) menjadi tools wajib untuk memaksimalkan produktivitas seorang developer dalam membuat suatu aplikasi. IntelliJ, adalah IDE

untuk Java development yang dikembangkan oleh JetBrains

referensi download JDK



<https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html>

referensi install IntelliJ



<https://www.jetbrains.com/idea/download/>

referensi IDE online



<https://glot.io/>

Project Pertama pada Java

```
/**
 * The HelloWorldApp class implements an application that
 * simply prints "Hello World!" to standard output.
 */
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!"); // Display the string.
    }
}
```

Kode java diatas terdiri dari Source Code Comments, The HelloWorldApp Class Definition, The Main Method.

Source Code Comments

Java memiliki 3 jenis komen

- `/* text */` ⇒ The compiler ignores everything from `/*` to `*/`.
- `/** documentation */` ⇒ This indicates a documentation comment (doc comment, for short). The compiler ignores this kind of comment, just like it ignores comments that use `/*` and `*/`. The javadoc tool uses doc comments when preparing automatically generated documentation. For more information on javadoc, see the Javadoc™ tool documentation .
- `// text` ⇒ The compiler ignores everything from `//` to the end of the line.

The HelloWorldApp Class Definition

```
class name {
    ...
}
```

HelloWorldApp merupakan nama dari class yang kita buat. Yang mana Class merupakan suatu blueprint atau cetakan untuk menciptakan suatu instant dari object.

The main Method

```
public static void main(String[] args) {  
    System.out.println("Hello World!"); // Display the string.  
}
```

Main Method adalah merupakan sebuah fungsi utama pada java. Yang mana merupakan blok program yang akan dieksekusi pertama kali.

Ekstensi pada java

Untuk membuat sebuah program java kita dapat menggunakan .class seperti diatas kita dapat membuat sebagai berikut `HelloWorldApp.class` .

Referensi

Java Resources for Students, Hobbyists and More | go.Java | Oracle

Java is at the heart of our digital lifestyle. It's the platform for launching careers, exploring human-to-digital interfaces, architecting the world's best applications, and unlocking innovation everywhere—from garages to global

 <https://go.java/>



The Java Language Environment

The Java Language Environment: Contents The Next Stage of the Known, Or a Completely New Paradigm? Taiichi Sakaiya-- The Knowledge-Value Revolution The Software Developer's Burden Imagine you're a software application developer. Your programming language of choice (or the language that's been foisted on you) is C or

 <https://www.oracle.com/java/technologies/introduction-to-java.html#943>

Modul 1 (Basic pada Java)

Package

Setiap class di program Java diletakkan dalam sebuah package, yakni mekanisme penempatan/penamaan class agar lebih terstruktur atau modular. Kegunaan package adalah untuk membedakan class dengan nama yang sama.

Langkah-langkah penamaan package

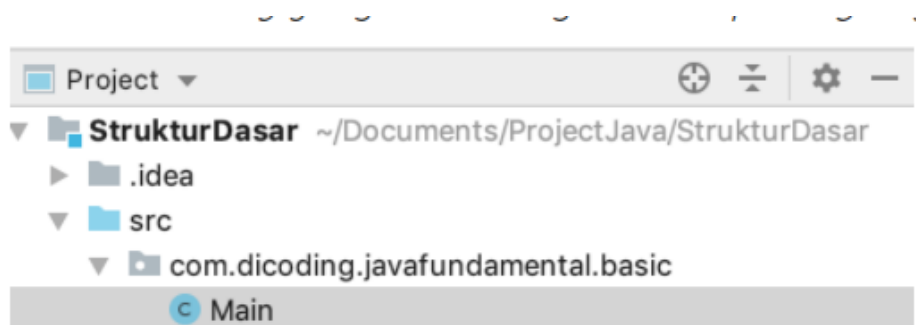
- Umumnya menggunakan nama domain institusi (dengan penulisan di balik) pembuat program tersebut.
- Sertakan nama program/aplikasi. Ikuti dengan nama modul-modulnya.
- Beri pemisah dengan tanda titik.

Contohnya :

Nama domain institusi	dicoding.com
Nama aplikasi	Java Fundamental
Nama modul	Basic
Maka nama <i>package</i> dapat kita tulis seperti ini	<code>com.dicoding.javafundamental.basic</code>

Perhatikan tabel di atas, nama package harus menggunakan huruf kecil tanpa spasi. Package akan sinkron dengan directory dari berkas (file) .java (source-code) dan juga hasil compile-nya yaitu file .class.

Contoh Penggunaan Package



Import

Import digunakan untuk menyederhanakan pemanggilan class yang berbeda package. Alhasil, Anda tak perlu menyebutkan fully qualified name dari class yang ingin digunakan.

Contoh Penggunaan import

```
1. import com.dicoding.javafundamental.basic.kendaraan.Kereta;
2. import com.dicoding.javafundamental.basic.kendaraan.Mobil;
3. import com.dicoding.javafundamental.basic.kendaraan.Motor;
4. import com.dicoding.javafundamental.basic.musik.Gitar;
```

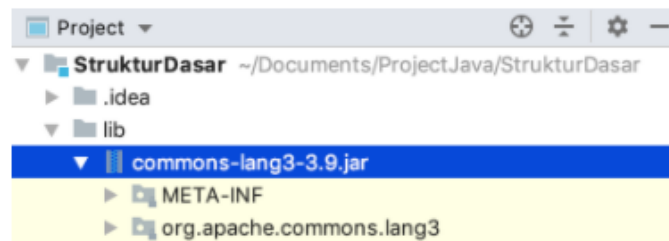
Menjadi:

```
1. import com.dicoding.javafundamental.basic.kendaraan.*;
2. import com.dicoding.javafundamental.basic.musik.Gitar;
```

ClassPath

Classpath adalah mekanisme di Java untuk menemukan class lain. Biasanya class lain tersebut berasal dari library yang berbeda atau bahkan JDK itu sendiri (kita sudah memakai class System). Jika kita tidak menemukan classnya maka akan muncul eror **ClassNotFoundException** atau **NoClassDefFoundError**.

Contoh Penggunaan ClassPath



Praktik yang kita lakukan barusan adalah menambahkan library (file jar) secara manual ke proyek. Cara ini sebenarnya kurang efektif jika dilakukan untuk proyek besar. Bayangkan jika kita menggunakan library A yang bergantung (dependencies) ke library B, C, D, lalu library B bergantung ke library X dan Y. Cukup bikin repot kan? Solusinya, gunakan tools seperti maven atau gradle.

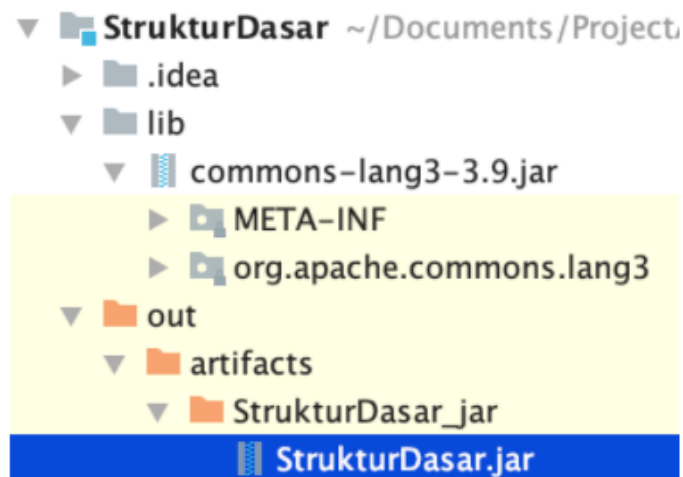
Pemaketan

Setiap kode Java yang ditulis akan dikompilasi menjadi bytecode (file dengan extension **.class**) lalu pada akhirnya akan dipaketkan untuk didistribusikan. Bentuk pemaketan yang sering digunakan adalah bentuk **JAR** (Java ARchive), ada juga bentuk lain misal **WAR** (Web ARchive) dan **EAR** (Enterprise ARchive). Dari nama pemaketan ini bisa ditebak

sebenarnya hanya berkas archive atau berkas zip. Isinya bisa diintip menggunakan winzip, winrar atau aplikasi sejenis.

Di dalam hasil pemaketan tersebut ada berkas metadata yang menjelaskan isi berkas JAR. File tersebut adalah manifest.mf yang diletakkan di directory META-INF.

Contoh Penggunaan Pemaketan



Referensi

Link source code Modul Struktur dasar

<https://github.com/dicodingacademy/BelajarJava/tree/master/Modul1Basic/StrukturDasar>

Input dan Output

Komponen dasar pemrograman dasar : **Input**, **Proses** dan **Output**.

- **Input** : Nilai yang dimasukkan
- **Proses** : Serangkaian langkah yang dilakukan untuk mengelola input yang diberikan
- **Output** : Menampilkan hasil olah data

Library input pada java : **BufferedReader** dan **Scanner**.

Library output pada java : **print** dan **println**.

Scanner

Scanner merupakan kelas yang menyediakan fungsi-fungsi untuk membaca dan mengambil input dari pengguna. **Scanner** memiliki kemudahan yang dapat membaca teks, baik yang memiliki tipe data primitif maupun string.

Contoh Penggunaan Scanner

```
1. package com.dicoding.javafundamental.inputouput;
2.
3. import java.util.Scanner;
4.
5. public class InputOutputFunction {
6.
7.     public static void main(String[] args) {
8.         Scanner scanner = new Scanner(System.in);
9.         System.out.println("Program penjumlahan sangat sederhana");
10.        System.out.print("Masukan Angka pertama : ");
11.        int value = scanner.nextInt();
12.        System.out.print("Masukan Angka kedua : ");
13.        int anotherValue = scanner.nextInt();
14.        int result = value + anotherValue;
15.        System.out.println("Hasilnya adalah : " + result);
16.    }
17. }
```

Pemanggilan Library Scanner

Inisialisasi Scanner

Menggunakan Scanner sebagai Input

Output

```
Run: InputOutputFunction x
/Library/Java/JavaVirtualMachines/jdk-12.0.1.jdk/Contents/Home/bin/java
"-javaagent:/Applications/IntelliJ IDEA CE.app/Contents/lib/idea_rt
.jar=53067:/Applications/IntelliJ IDEA CE.app/Contents/bin" -Dfile.encoding=UTF-8
-classpath /Users/gilangramadhan/Documents/ProjectJava/InputOuput/out/production
/InputOuput com.dicoding.javafundamental.inputouput.InputOutputFunction
Program penjumlahan sangat sederhana
Masukan Angka pertama : 10
Masukan Angka kedua : 5
Hasilnya adalah : 15
Process finished with exit code 0
```

Input

BufferedReader

BufferedReader dapat digunakan pada materi ini sebagai Basic Input karena sebenarnya kelas ini tidak hanya digunakan untuk membaca input dari keyboard saja, melainkan juga untuk mendapatkan input dari user. Dalam implementasinya BufferedReader tidak dapat berjalan sendiri. Untuk mendapatkan input dibutuhkan kelas InputStreamReader.

Contoh Penggunaan BufferedReader

Import Library BufferedReader

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
```

Inisialisasi BufferedReader

```
InputStreamReader streamReader = new InputStreamReader(System.in);
BufferedReader bufferedReader = new BufferedReader(streamReader);
```

Penggunaan input

```
try {
    System.out.print("Masukan Angka pertama : ");
    value = Integer.parseInt(bufferedReader.readLine());
    System.out.print("Masukan Angka kedua : ");
    anotherValue = Integer.parseInt(bufferedReader.readLine());
} catch (IOException e) {
    e.printStackTrace();
}
```

Input pada bufferedReader

try dan catch pada kode diatas berguna sebagai IO Exception atau lebih tepatnya sebagai penanganan error input.

Print dan Println

Contoh Penggunaannya

```

package com.dicoding.javafundamental.inputouput;

public class UserOutput {

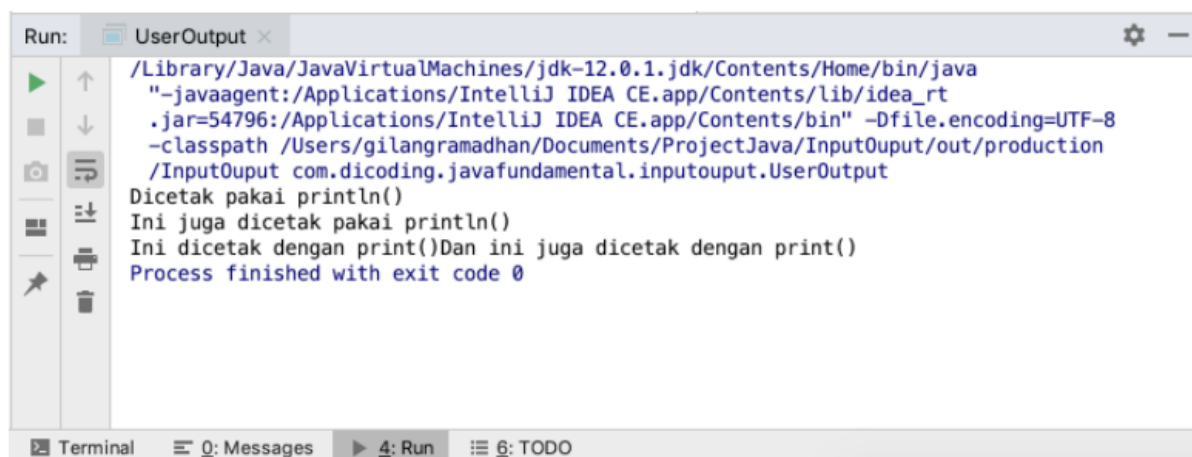
    public static void main(String[] args) {
        System.out.println("Dicetak pakai println()");
        System.out.println("Ini juga dicetak pakai println()");
        System.out.print("Ini dicetak dengan print()");
        System.out.print(" dan ini juga dicetak dengan print()");
    }
}

```

Penggunaan `println` akan menampilkan teks dan tambahan baris baru.

Sedangkan fungsi `print` menampilkan keluaran berupa teks sesuai dengan yang dimasukkan.

Output



```

Run: UserOutput x
/Library/Java/JavaVirtualMachines/jdk-12.0.1.jdk/Contents/Home/bin/java
"-javaagent:/Applications/IntelliJ IDEA CE.app/Contents/lib/idea_rt
.jar=54796:/Applications/IntelliJ IDEA CE.app/Contents/bin" -Dfile.encoding=UTF-8
-classpath /Users/gilangramadhan/Documents/ProjectJava/InputOuput/out/production
/InputOuput com.dicoding.javafundamental.inputouput.UserOutput
Dicetak pakai println()
Ini juga dicetak pakai println()
Ini dicetak dengan print()Dan ini juga dicetak dengan print()
Process finished with exit code 0

```

Referensi Source code modul input dan output

<https://github.com/dicodingacademy/BelajarJava/tree/master/Modul1Basic/InputOuput>

Tipe Data

Java bersifat statically typed, yang artinya setiap variabel harus dideklarasikan terlebih dahulu sebelum digunakan. Pada Java terdapat 2 jenis tipe data yaitu **primitive/primitif** dan **reference/referensi**.

Tipe Data Primitive/Primitif

Tipe data primitif adalah tipe data standar yang tidak diturunkan dari objek manapun. Terdapat 8 (delapan) tipe data primitif yang Java dukung, antara lain:

1. Byte

Tipe data integer 8 bit yang digunakan untuk menampung angka dalam range yang kecil. Nilai minimum dan maksimum dari tipe data ini adalah -128, dan 127. Tipe data byte memiliki nilai default 0. Ia digunakan untuk menghemat ruang dalam array yang besar, terutama pada bilangan bulat, karena 1 (satu) byte 4 (empat) kali lebih kecil dari tipe data int.

```
byte value = 10;  
byte anotherValue = -10;
```

2. Short

Merupakan tipe data integer 16 bit yang digunakan untuk menampung angka dalam range menengah, yaitu antara -32,768 sampai 32,767. Sama seperti tipe sebelumnya, nilai default-nya 0. Ia bisa digunakan untuk menghemat memori seperti tipe data byte namun hanya 2 (dua) kali lebih kecil dari tipe data int.

```
short value = 15000;  
short anotherValue = -20000;
```

3. Int

Merupakan tipe data integer 32 bit yang digunakan untuk menyimpan angka dalam range cukup besar, yakni antara -2,147,483,648 sampai 2,147,483,647 dengan nilai default 0. Jika kita tidak memperhatikan penggunaan memori, tipe data inilah yang biasanya dipakai.

```
int value = 150000;  
int anotherValue = -200000;
```

4. Long

Merupakan tipe data integer yang lebih besar jika dibandingkan dengan tipe data int. Ukurannya 64 bit dan bisa digunakan untuk menyimpan angka dengan range antara

-9,223,372,036,854,775,808 sampai 9,223,372,036,854,775,807. Tipe data long memiliki nilai default 0L.

```
long value = 150000L;  
long anotherValue = -200000L;
```

5. Float

Merupakan sebuah tipe data yang bisa digunakan untuk menampung angka desimal. Nilai default-nya 0.0f.

```
float value = 3.5f;
```

6. Double

Merupakan sebuah tipe data yang mirip seperti tipe data float, namun memiliki kapasitas yang lebih besar. Nilai default-nya 0.0d.

```
double value = 5.0;
```

7. Boolean

Merupakan sebuah tipe data yang memiliki 2 (dua) macam nilai, yaitu true dan false. Nilai default-nya false.

```
boolean value = true;  
boolean anotherValue = false;
```

8. Char

Merupakan sebuah tipe data yang bisa digunakan untuk menampung karakter. Nilai karakter tersebut dibungkus di dalam tanda ' '.

```
char item = 'A'
```

Tipe Data Reference

Tipe data reference merupakan sebuah tipe data yang merujuk ke sebuah objek atau instance dari sebuah class. Salah satu tipe data yang termasuk ke dalam tipe data reference adalah **string**. Tipe data lainnya dapat berupa sebuah kelas yang mana, kita bisa membuat variabel baru dengan tipe data class `User`.

```
User user = new User();
```

String

String adalah kumpulan beberapa karakter (char). String sendiri sebenarnya merupakan sebuah class yang terdapat dalam library Java dan digunakan untuk memanipulasi karakter

```
String greeting = "Hello World!";
```

Di dalam kelas String terdapat beberapa constructor yang memungkinkan kita untuk memberikan nilai awal untuk string dari sumber yang berbeda. Misalnya kita ingin membuat variabel String dari sebuah array character.

```
char[] dicodingChars = { 'd', 'i', 'c', 'o', 'd', 'i', 'n', 'g' };  
String dicodingString = new String(dicodingChars);  
System.out.println(dicodingString);
```

Mengetahui Panjang String

Kelas String memiliki sebuah method untuk mengetahui panjang dari sebuah string, yakni method `length()`. Fungsi tersebut akan mengembalikan/menghasilkan sejumlah karakter dari string, contohnya:

```
String dicoding = "dicoding";  
int length = dicoding.length();  
System.out.println(length);
```



Mengambil Karakter Dari Sebuah String

Kita juga bisa mengambil sebuah karakter secara spesifik dari sebuah String dengan menggunakan method `charAt(int index)` yang sudah tersedia dalam kelas String. Misalnya saat ingin mengambil sebuah karakter dari teks `dicoding`, kita bisa menggunakan kode berikut:

```
String dicoding = "dicoding";
char result = dicoding.charAt(7);
System.out.println(result);
```

Maka jika Anda menjalankan kode di atas, konsol akan menampilkan karakter urutan ke-8 dari teks tersebut, yaitu "g".

Beberapa Method lainnya yang bisa digunakan pada String

 Name	 Deskripsi
<u>length()</u>	Digunakan untuk mengetahui panjang atau jumlah karakter dalam string.
<u>charAt(int index)</u>	Digunakan untuk mengambil sebuah karakter berdasarkan indeks tertentu.
<u>format(String format, Object... args)</u>	Digunakan untuk memformat sebuah string.
<u>substring(int beginIndex)</u>	Mengembalikan/menghasilkan substring berdasarkan indeks yang diberikan.
<u>contains(CharSequence s)</u>	Mengembalikan/menghasilkan nilai true atau false setelah mencocokkan karakter.
<u>equals(Object object)</u>	Memeriksa apakah nilai objek sama dengan nilai string.
<u>isEmpty()</u>	Memeriksa apakah sebuah string itu kosong atau tidak.
<u>concat(String s)</u>	Mengkonsolidasikan sebuah string.
<u>replace(char a, char b)</u>	Mengganti suatu karakter di dalam string.
<u>indexOf(String a)</u>	Mengetahui indeks dari sebuah substring.
<u>toLowerCase()</u>	Mengubah format string menjadi huruf kecil semua.
<u>toUpperCase()</u>	Mengubah format string menjadi huruf kapital semua.
<u>trim()</u>	Menghapus spasi awal dan akhir dari string.
<u>valueOf(int value)</u>	Mengkonversi tipe yang diberikan menjadi sebuah string.
<u>compareTo()</u>	Membandingkan dua nilai

