

Praktikum Pengenalan Java

Tujuan

- Mengetahui cara instalasi Java sebagai bahasa implementasi
- Dapat membenarkan kesalahan program

Cara Instalasi Java dan Menjalankan di Command prompt

Mengenai JDK

- Tahap pertama sebelum memulai pemrograman Java adalah mendapatkan JDK (Java Development Kit).
- Di JDK tsb telah tersedia kompiler Java(javac) dan interpreter Java(java).
- Dengan -javac menghasilkan bytecode selanjutnya dieksekusi menggunakan -java.
- Sun juga telah menyediakan JRE (Java Run-time Environment) yang merupakan subset dari JDK.
- Jika hanya mengeksekusi program Java, tidak membuat program Java, maka hanya perlu menggunakan JRE.
- Secara umum $JDK = JRE + \text{compiler Java} + \text{source code library Java}$.



Instalasi Java dan Menjalankan di Command Prompt

- File yang harus disiapkan :
 - JDK 6
- Langkah yang dilakukan :
 - Setting Path dan ClassPath
 - Mencoba hasil instal (optional)
 - Menulis file java menggunakan notepad
 - Mengkompile dan Menjalankan program java di Command prompt.

Setting Path dan ClassPath

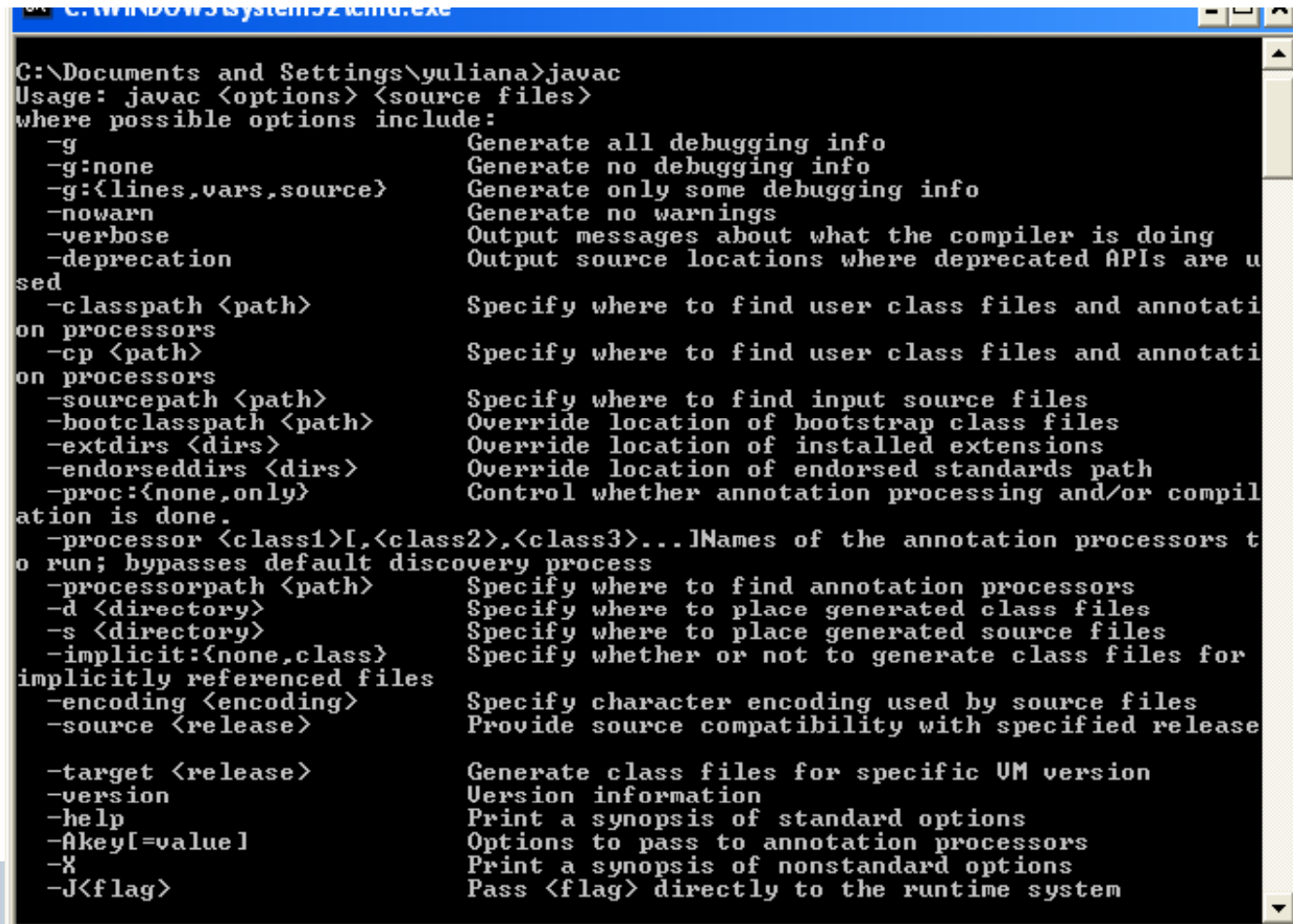
- Buka Control Panel – System
- Pilih tab : Advanced
- Pilih button: Environment Variables
- Di system variables lakukan setting:
 - PATH :
c:\nama_folder_tempat_instal\bin
 - CLASSPATH:
.;c:\nama_folder_tempat_instal\lib\tools.jar

Mencoba hasil instal

- Buka windows command prompt
- Ketikkan: `c:\javac`, Tekan enter, Bila keluar cara penggunaan berarti berhasil
- Ketikkan: `c:\java`, Tekan enter, Bila keluar cara penggunaan berarti berhasil

Pengecekan javac

- Ketikkan: `c:\....\javac`



```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\yuliana>javac
Usage: javac <options> <source files>
where possible options include:
  -g                    Generate all debugging info
  -g:none              Generate no debugging info
  -g:<lines,vars,source> Generate only some debugging info
  -nowarn              Generate no warnings
  -verbose             Output messages about what the compiler is doing
  -deprecation         Output source locations where deprecated APIs are used
  -classpath <path>   Specify where to find user class files and annotations processors
  -cp <path>          Specify where to find user class files and annotations processors
  -sourcepath <path>  Specify where to find input source files
  -bootclasspath <path> Override location of bootstrap class files
  -extdirs <dirs>     Override location of installed extensions
  -endorseddirs <dirs> Override location of endorsed standards path
  -proc:<none,only>   Control whether annotation processing and/or compilation is done.
  -processor <class1>[,<class2>,<class3>...] Names of the annotation processors to run; bypasses default discovery process
  -processorpath <path> Specify where to find annotation processors
  -d <directory>     Specify where to place generated class files
  -s <directory>     Specify where to place generated source files
  -implicit:<none,class> Specify whether or not to generate class files for implicitly referenced files
  -encoding <encoding> Specify character encoding used by source files
  -source <release>   Provide source compatibility with specified release

  -target <release>   Generate class files for specific VM version
  -version            Version information
  -help              Print a synopsis of standard options
  -Akey[=value]      Options to pass to annotation processors
  -X                 Print a synopsis of nonstandard options
  -J<flag>           Pass <flag> directly to the runtime system
```


- Ketikkan:
c:\....\javac

```

C:\WINDOWS\system32\cmd.exe
D:\>java
Usage: java [-options] class [args...]
        (to execute a class)
   or  java [-options] -jar jarfile [args...]
        (to execute a jar file)

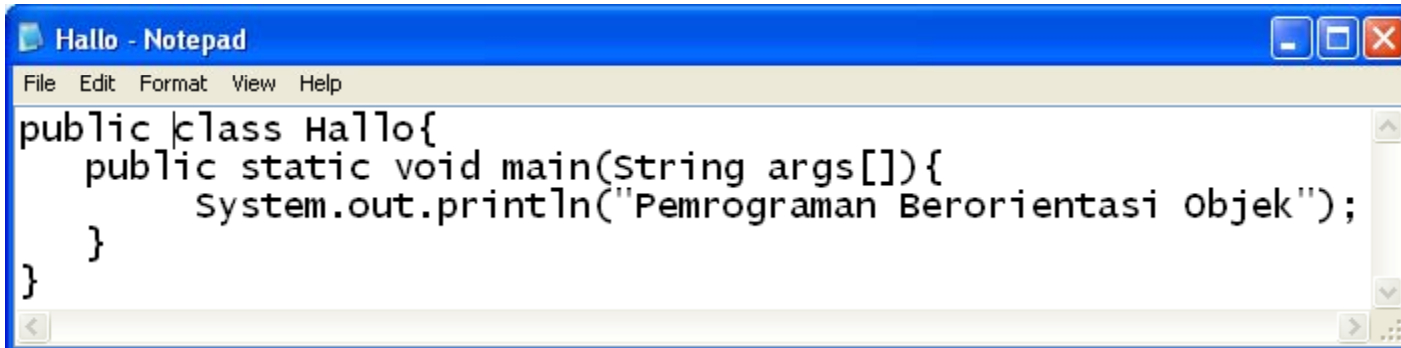
where options include:
   -client          to select the "client" VM
   -server          to select the "server" VM
   -hotspot         is a synonym for the "client" VM [deprecated]
                   The default VM is client.

   -cp <class search path of directories and zip/jar files>
   -classpath <class search path of directories and zip/jar files>
                   A ; separated list of directories, JAR archives,
                   and ZIP archives to search for class files.
   -D<name>=<value>
                   set a system property
   -verbose[:class!gc!jni]
                   enable verbose output
   -version         print product version and exit
   -version:<value>
                   require the specified version to run
   -showversion    print product version and continue
   -jre-restrict-search | -jre-no-restrict-search
                   include/exclude user private JREs in the version search
   -? -help        print this help message
   -X             print help on non-standard options
   -ea[:<packagename>...!:<classname>]
   -enableassertions[:<packagename>...!:<classname>]
                   enable assertions
   -da[:<packagename>...!:<classname>]
   -disableassertions[:<packagename>...!:<classname>]
                   disable assertions
   -esa | -enablesystemassertions
                   enable system assertions
   -dsa | -disablesystemassertions
                   disable system assertions
   -agentlib:<libname>[=<options>]
                   load native agent library <libname>, e.g. -agentlib:hprof
                   see also, -agentlib:jdwp=help and -agentlib:hprof=help
   -agentpath:<pathname>[=<options>]
                   load native agent library by full pathname
   -javaagent:<jarpath>[=<options>]
                   load Java programming language agent, see java.lang.instrument
   -splash:<imagepath>
                   show splash screen with specified image
    
```

Cara kompilasi dan menjalankan program java.

Instruksi:

1. Buka editor (notepad, ms word, dll).
2. Tulis program dibawah ini
3. Simpan dengan nama Hallo.java → nama file = nama class → case sensitive
(sebagai contoh simpan di d:\Hallo.java)



```
File Edit Format View Help
public class Hallo{
    public static void main(String args[]){
        System.out.println("Pemrograman Berorientasi Objek");
    }
}
```



Mengkompile dan Menjalankan program java di Command prompt.

- Masuk ke direktori aktif dan ketik:
 - d:/javac Hallo.java
 - d:/java Hallo

A screenshot of a Windows Command Prompt window. The title bar reads 'C:\WINDOWS\system32\cmd.exe'. The command history shows:

```
D:\>javac Hallo.java
D:\>java Hallo
Pemrograman Berorientasi Objek
D:\>_
```

The output of the 'java Hallo' command is 'Pemrograman Berorientasi Objek'. The cursor is currently on a new line after the prompt 'D:\>_'.

```
C:\WINDOWS\system32\cmd.exe
D:\>javac Hallo.java
D:\>java Hallo
Pemrograman Berorientasi Objek
D:\>_
```

Aturan Pendeklarasian File

- Dalam sebuah file hanya diperbolehkan **satu class** yang mempunyai **hak akses public**. Artinya dalam satu file boleh mempunyai lebih dari satu class tapi hanya satu class yang mempunyai hak akses public.
- Jika sebuah file tidak mempunyai public class, **nama file bisa diberi nama apapun** (tidak harus dari class-class yang ada di file tersebut).
- **Nama file harus sama dengan nama dari public class**. Sebagai contoh terdapat sebuah class yang dideklarasikan public class Dog { } maka file harus diberi nama dengan Dog.java.
- Jika class merupakan bagian dari sebuah package, maka package diletakkan pada baris pertama source code, selanjutnya diikuti dengan statement import (jika ada).
- Jika ada statement import, maka letaknya diantara statement package dan pendeklarasian class. Jika tidak ada statement package maka statement import harus diletakkan pada baris pertama source code. Jika tidak ada statement package ataupun import maka deklarasi class harus diletakkan pada baris pertama source code.
- Komentar bisa ditempatkan di mana saja pada file.



Praktikum 1. Ketik program dibawah ini, bila terjadi kesalahan waktu kompilasi dan runtime, betulkan! (tersimpan dalam file **Greeting.java**)

```
public class TestGreeting {  
    public static void main (String[] args) {  
        Greeting hello = new Greeting();  
        hello.greet();  
    }  
}  
  
class Greeting {  
    public void greet() {  
        System.out.println("hi");  
    }  
}
```

Praktikum 2: Test1.java

Why did it fail?

Fix it and recompile and run this program?

```
public class Test1 {  
    public static void main(String[] args) {  
        System.out.println("What's wrong with this program?");  
    }  
}
```

```
public class TestAnother1 {  
    public static void main(String[] args) {  
        System.out.println("What's wrong with this program?");  
    }  
}
```

Praktikum 3: Test2.java

Why did it fail?

Fix it and recompile and run this program?

```
public class Testing2 {  
    public static void main(String[] args) {  
        System.out.println("What's wrong with this program?");  
    }  
}
```

Praktikum 4: Test3.java

Why did it fail?

Fix it and recompile and run this program?

```
public class Test3 {  
    public static void main(String args) {  
        System.out.println("What's wrong with this program?");  
    }  
}
```


Praktikum 5: Test4.java

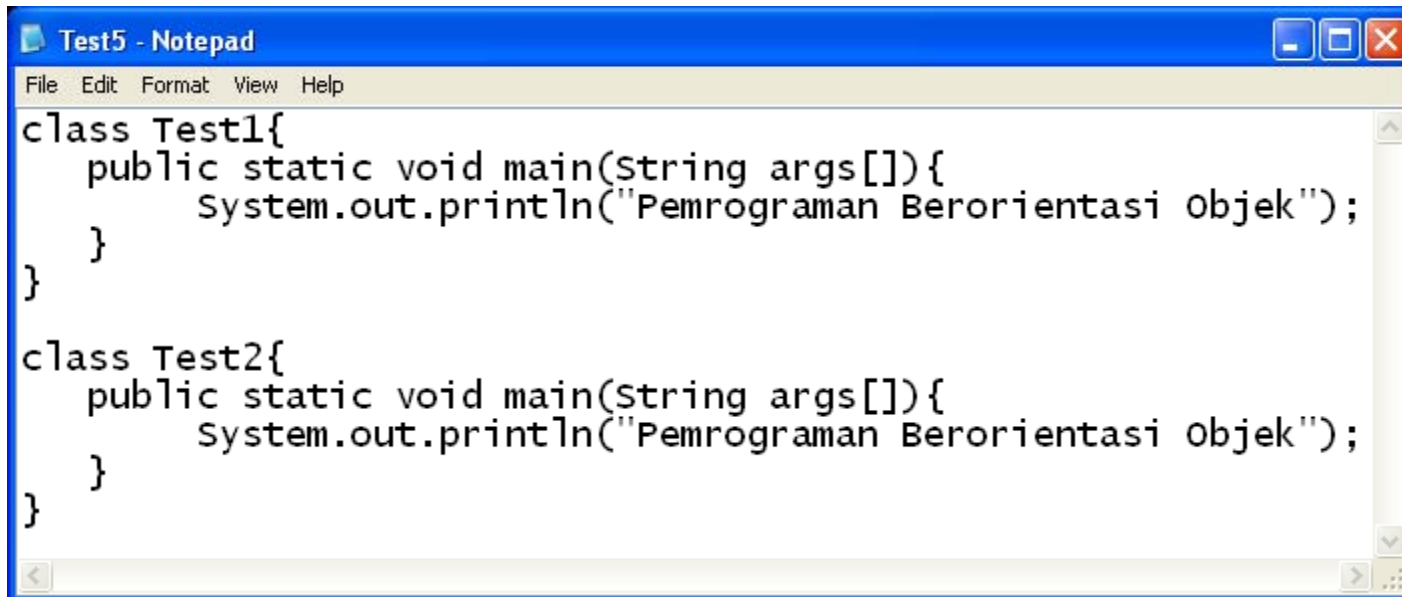
Why did it fail?

Fix it and recompile and run this program?

```
public class Test4 {  
    public void main(String[] args) {  
        System.out.println("What's wrong with this program?");  
    }  
}
```

Praktikum 6

- Beri nama file dengan Test5.java
- Errorkah program di bawah ini? Jelaskan !

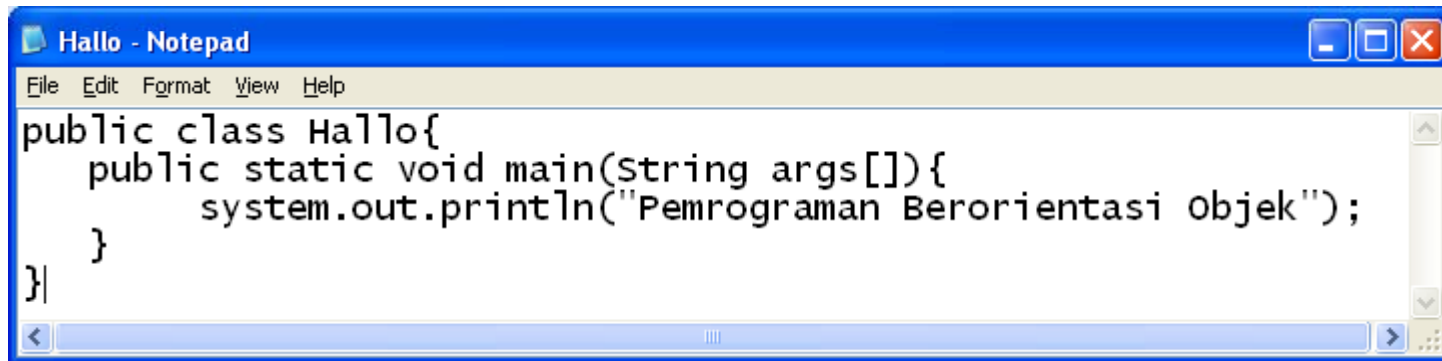


```
Test5 - Notepad
File Edit Format View Help
class Test1{
    public static void main(String args[]){
        System.out.println("Pemrograman Berorientasi Objek");
    }
}

class Test2{
    public static void main(String args[]){
        System.out.println("Pemrograman Berorientasi Objek");
    }
}
```

Praktikum 7

- Beri file dengan Hallo.java
- Errorkah program di bawah ini? Jelaskan !



```
Hallo - Notepad
File Edit Format View Help
public class Hallo{
    public static void main(String args[]){
        system.out.println("Pemrograman Berorientasi objek");
    }
}
```