Java Basics

What is Java

Java is a **programming language** and a **platform**.

Java is a high level, robust, secured and object-oriented programming language.

Platform: Any hardware or software environment in which a program runs, is known as a platform. Since Java has its own runtime environment (JRE) and API, it is called platform.

Where it is used?

According to Sun, 3 billion devices run java. There are many devices where java is currently used. Some of them are as follows:

1. Desktop Applications such as acrobat reader, media player, antivirus etc.
2. Web Applications such as irctc.co.in, javatpoint.com etc.
3. Enterprise Applications such as banking applications.
4. Mobile
5. Embedded System
6. Smart Card
7. Robotics
8. Games etc.

Types of Java Applications

There are mainly 4 type of applications that can be created using java programming:

1) Standalone Application

It is also known as desktop application or window-based application. An application that we need to install on every machine such as media player, antivirus etc. AWT and Swing are used in java for creating standalone applications.

2) Web Application

An application that runs on the server side and creates dynamic page, is called web application. Currently, servlet, jsp, struts, jsf etc. technologies are used for creating web applications in java.

3) Enterprise Application

An application that is distributed in nature, such as banking applications etc. It has the advantage of high level security, load balancing and clustering. In java, EJB is used for creating enterprise applications.

4) Mobile Application

An application that is created for mobile devices. Currently Android and Java ME are used for creating mobile applications.

Java Basics

History of Java

Java history is interesting to know. The history of java starts from Green Team. Java team members (also known as **Green Team**), initiated a revolutionary task to develop a language for digital devices such as set-top boxes, televisions etc.

For the green team members, it was an advance concept at that time. But, it was suited for internet programming. Later, Java technology as incorporated by Netscape.

James Gosling

Currently, Java is used in internet programming, mobile devices, games, e-business solutions etc. There are given the major points that describes the history of java.

1) **James Gosling**, **Mike Sheridan**, and **Patrick Naughton** initiated the Java language project in June 1991. The small team of sun engineers called **Green Team**.

2) Originally designed for small, embedded systems in electronic appliances like set-top boxes.

3) Firstly, it was called **"Greentalk"** by James Gosling and file extension was .gt.

4) After that, it was called **Oak** and was developed as a part of the Green project.

Why Oak name for java language?

5) **Why Oak?** Oak is a symbol of strength and choosen as a national tree of many countries like U.S.A., France, Germany, Romania etc.

6) In 1995, Oak was renamed as **"Java"** because it was already a trademark by Oak Technologies.

Why Java name for java language?

7) **Why they choosed java name for java language?** The team gathered to choose a new name. The suggested words were "dynamic", "revolutionary", "Silk", "jolt", "DNA" etc. They wanted something that reflected the essence of the technology: revolutionary, dynamic, lively, cool, unique, and easy to spell and fun to say.

According to James Gosling "Java was one of the top choices along with **Silk**". Since java was so unique, most of the team members preferred java.

8) Java is an island of Indonesia where first coffee was produced (called java coffee).

9) Notice that Java is just a name not an acronym.

10) Originally developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released in 1995.

11) In 1995, Time magazine called **Java one of the Ten Best Products of 1995**.

12) JDK 1.0 released in(January 23, 1996).

Java Version History

There are many java versions that has been released. Current stable release of Java is Java SE 8.

1. JDK Alpha and Beta (1995)
2. JDK 1.0 (23rd Jan, 1996)
3. JDK 1.1 (19th Feb, 1997)
4. J2SE 1.2 (8th Dec, 1998)
5. J2SE 1.3 (8th May, 2000)
6. J2SE 1.4 (6th Feb, 2002)
7. J2SE 5.0 (30th Sep, 2004)
8. Java SE 6 (11th Dec, 2006)
9. Java SE 7 (28th July, 2011)
10. Java SE 8 (18th March, 2014)

Features of Java

1. Simple
2. Object-Oriented
3. Platform independent
4. Secured
5. Robust
6. Architecture neutral
7. Portable
8. Dynamic
9. Interpreted
10. High Performance
11. Multithreaded
12. Distributed

Simple

According to Sun, Java language is simple because:

syntax is based on C++ (so easier for programmers to learn it after C++).

removed many confusing and/or rarely-used features e.g., explicit pointers, operator overloading etc.

No need to remove unreferenced objects because there is Automatic Garbage Collection in java.

Object-oriented

Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behaviour.

Object-oriented programming(OOPs) is a methodology that simplify software development and maintenance by providing some rules.

Basic concepts of OOPs are:

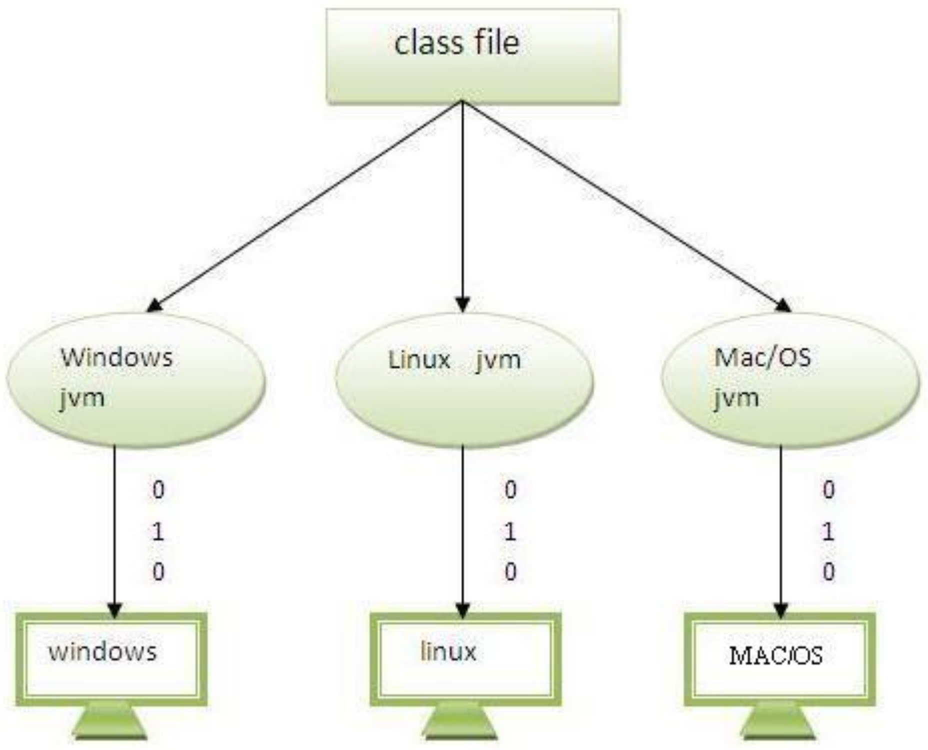
1. Object
2. Class
3. Inheritance
4. Polymorphism
5. Abstraction
6. Encapsulation

Platform Independent

A platform is the hardware or software environment in which a program runs. There are two types of platforms software-based and hardware-based. Java provides software-based platform. The Java platform differs from most other platforms in the sense that it's a software-based platform that runs on top of other hardware-based platforms.It has two components:

1. Runtime Environment

2. API(Application Programming Interface)

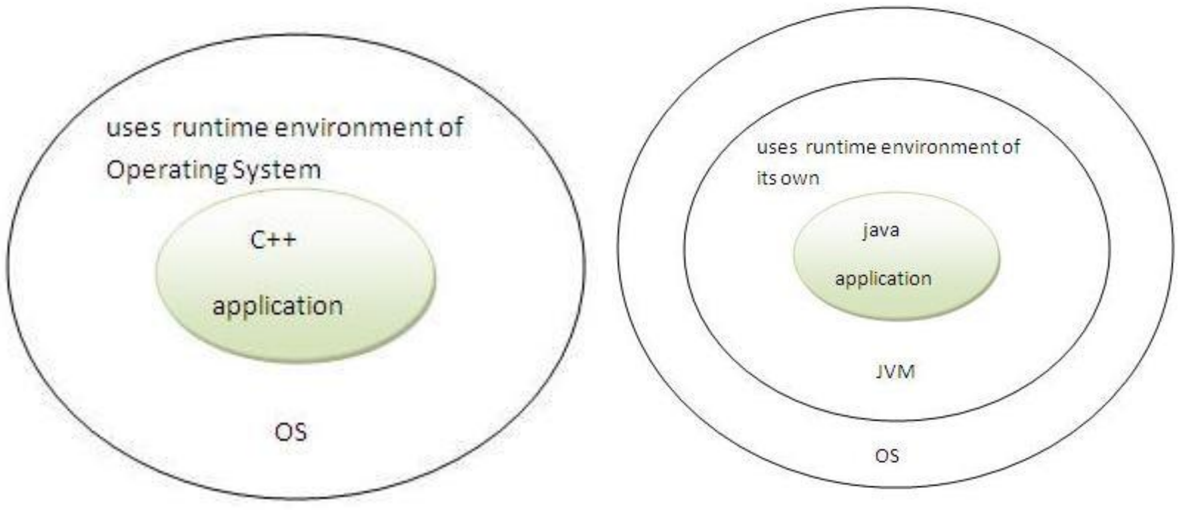


Java code can be run on multiple platforms e.g.Windows, Linux, SunSolaris, Mac/OS etc. Java code is compiled by the compiler and converted into bytecode.Thisbytecode is a platform independent code because it can be run on multiple platforms i.e. Write Once and Run Anywhere(WORA).

Secured

Java is secured because:

- No explicit pointer
- Programs run inside virtual machine sandbox.


uses runtime environment of Operating System
C++ application
OS
uses runtime environment of its own
java application
JVM
OS

- **Classloader-** adds security by separating the package for the classes of the local file system

from those that are imported from network sources.
- **Bytecode Verifier-** checks the code fragments for illegal code that can violate access right to objects.
- **Security Manager-** determines what resources a class can access such as reading and writing to the local disk.

These security are provided by java language. Some security can also be provided by application developer through SSL,JAAS,cryptography etc.

Robust

Robust simply means strong. Java uses strong memory management. There are lack of pointers that avoids security problem. There is automatic garbage collection in java. There is exception handling and type checking mechanism in java. All these points makes java robust.

Architecture-neutral

There is no implementation dependent features e.g. size of primitive types is set.

Portable

We may carry the java bytecode to any platform.

High-performance

Java is faster than traditional interpretation since byte code is "close" to native code still somewhat slower than a compiled language (e.g., C++)

Distributed

We can create distributed applications in java. RMI and EJB are used for creating distributed applications. We may access files by calling the methods from any machine on the internet.

Multi-threaded

A thread is like a separate program, executing concurrently. We can write Java programs that deal with many tasks at once by defining multiple threads. The main advantage of multi-threading is that it shares the same memory. Threads are important for multi-media, Web applications etc.

Simple Program of Java

To create a simple java program, you need to create a class that contains main method. Let's understand the requirement first.

Requirement for Hello Java Example

For executing any java program, you need to

- install the JDK if you don't have installed it, download the JDK and install it.
- set path of the jdk/bin directory. http://www.javatpoint.com/how-to-set-path-in-java
- create the java program
- compile and run the java program

Creating hello java example

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

save this file as Simple.java

To compile: Javac Simple.java

To execute: java Simple

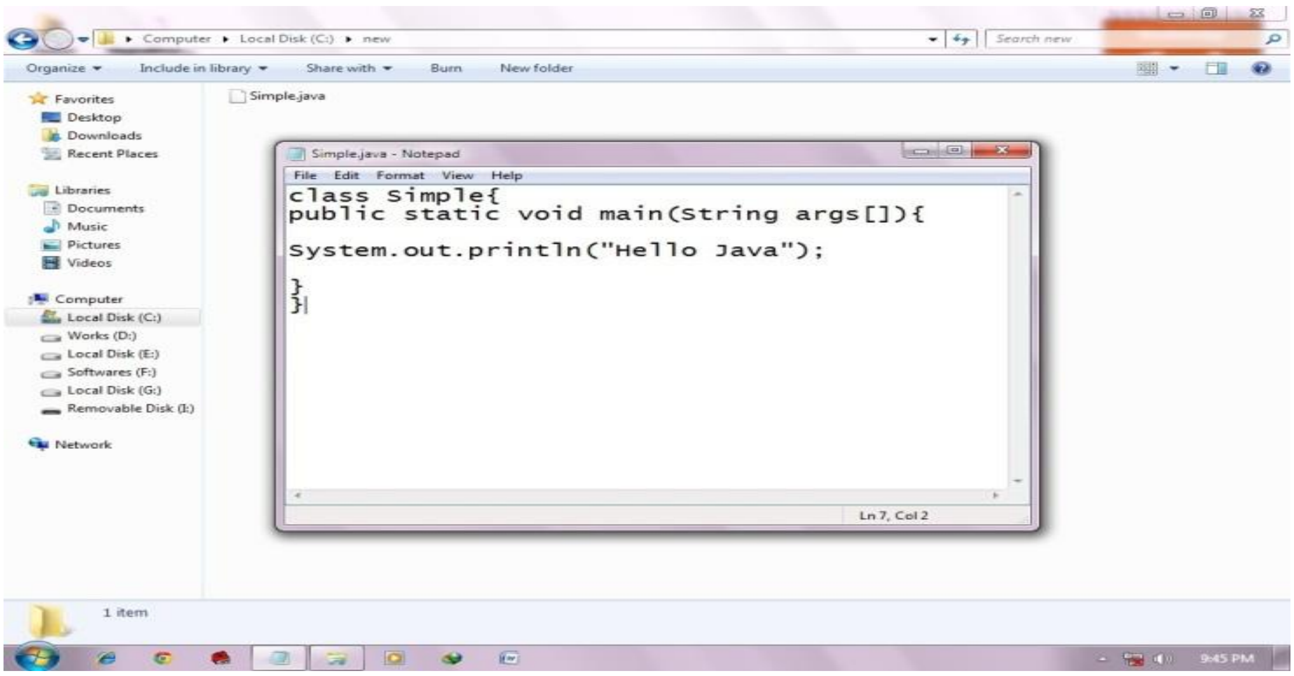
Output:`Hello Java`

Understanding first java program

Let's see what is the meaning of class, public, static, void, main, String[], System.out.println().

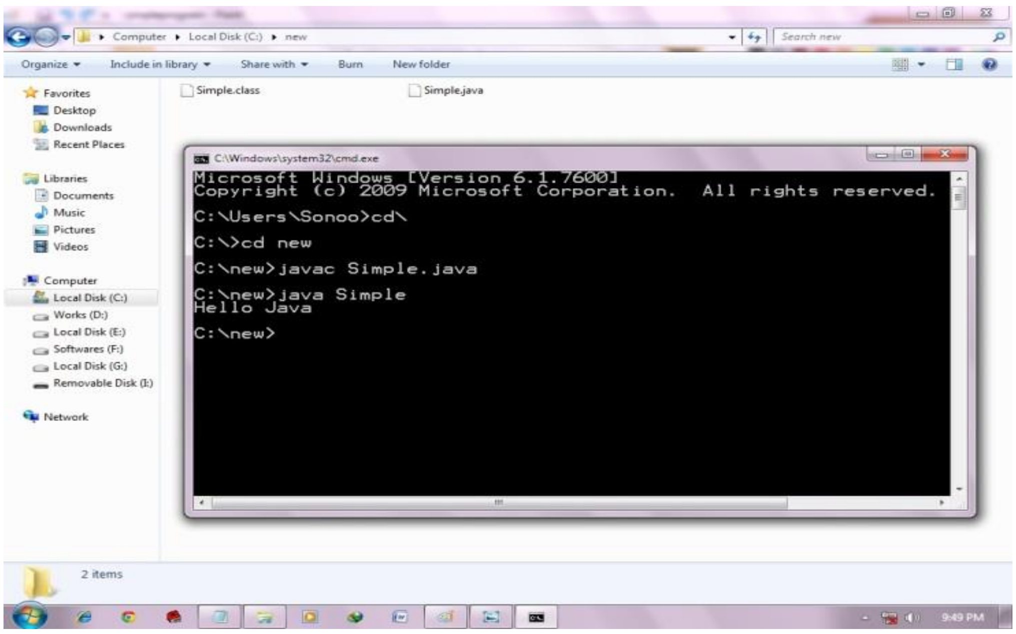
- **class** keyword is used to declare a class in java.
- **public** keyword is an access modifier which represents visibility, it means it is visible to all.
- **static** is a keyword, if we declare any method as static, it is known as static method. The core advantage of static method is that there is no need to create object to invoke the static method. The main method is executed by the JVM, so it doesn't require to create object to invoke the main method. So it saves memory.
- **void** is the return type of the method, it means it doesn't return any value.
- **main** represents startup of the program.
- **String[] args** is used for command line argument. We will learn it later.
- **System.out.println()** is used print statement. We will learn about the internal working of System.out.println statement later.

To write the simple program, open notepad by **start menu -> All Programs -> Accessories -> notepad** and write simple program as displayed below:



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

System.out.println("Hello Java");

}
}
```

As displayed in the above diagram, write the simple program of java in notepad and saved it as Simple.java. To compile and run this program, you need to open command prompt by **start menu -> All Programs -> Accessories -> command prompt**.



```
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Sonoo>cd\

C:\>cd new

C:\new>javac Simple.java

C:\new>java Simple
Hello Java

C:\new>
```

To compile and run the above program, go to your current directory first; my current directory is c:\new . Write here:

To compile: javac Simple.java

To execute: java Simple

Java Basics

How many ways can we write a java program

There are many ways to write a java program. The modifications that can be done in a java program are given below:

1) By changing sequence of the modifiers, method prototype is not changed.

Let's see the simple code of main method.

1. static public void main(String args[])

2) subscript notation in java array can be used after type, before variable or after variable.

Let's see the different codes to write the main method.

1. public static void main(String[] args)
2. public static void main(String []args)
3. public static void main(String args[])

3) You can provide var-args support to main method by passing 3 ellipses (dots)

Let's see the simple code of using var-args in main method. We will learn about var-args later in Java New Features chapter.

1. public static void main(String... args)

4) Having semicolon at the end of class in java is optional.

Let's see the simple code.

```
class A{
static public void main(String... args){
System.out.println("hello java4");
}
};
```

Valid java main method signature

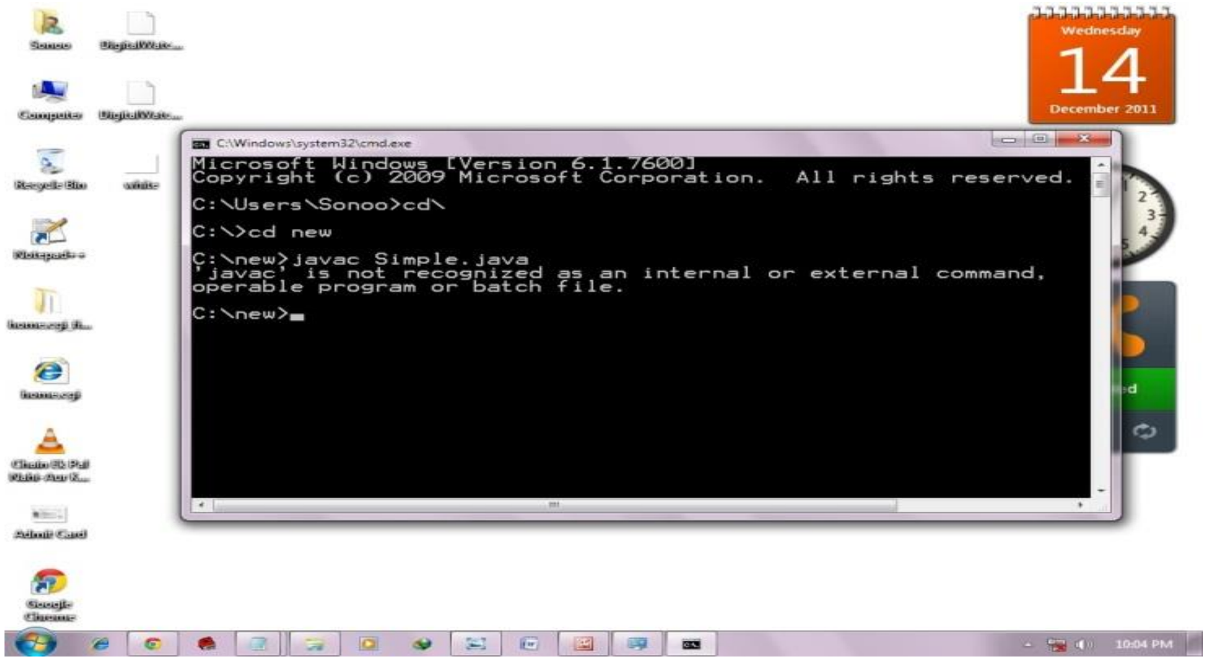
1. public static void main(String[] args)
2. public static void main(String []args)
3. public static void main(String args[])
4. public static void main(String... args)
5. static public void main(String[] args)
6. public static final void main(String[] args)
7. final public static void main(String[] args)
8. final strictfp public static void main(String[] args)

Invalid java main method signature

1. public void main(String[] args)
2. static void main(String[] args)
3. public void static main(String[] args)
4. abstract public static void main(String[] args)

Resolving an error "javac is not recognized as an internal or external command" ?

If there occurs a problem like displayed in the below figure, you need to set path. Since DOS doesn't know javac or java, we need to set path. Path is not required in such a case if you save your program inside the jdk/bin folder. But its good approach to set path. Click here for How to set path in java.



```
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Sonoo>cd\

C:\>cd new

C:\new>javac Simple.java
'javac' is not recognized as an internal or external command,
operable program or batch file.

C:\new>
```

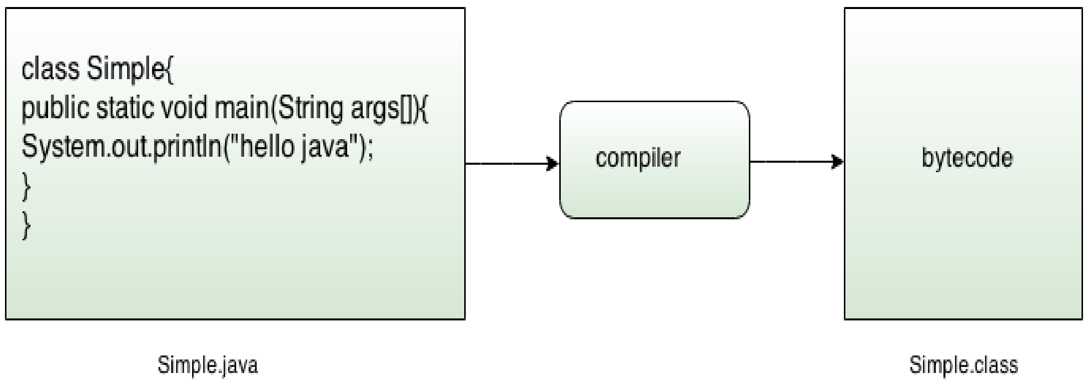
Internal Details of Hello Java Program

1. Internal Details of Hello Java

In the previous page, we have learned about the first program, how to compile and how to run the first java program. Here, we are going to learn, what happens while compiling and running the java program. Moreover, we will see some question based on the first program.

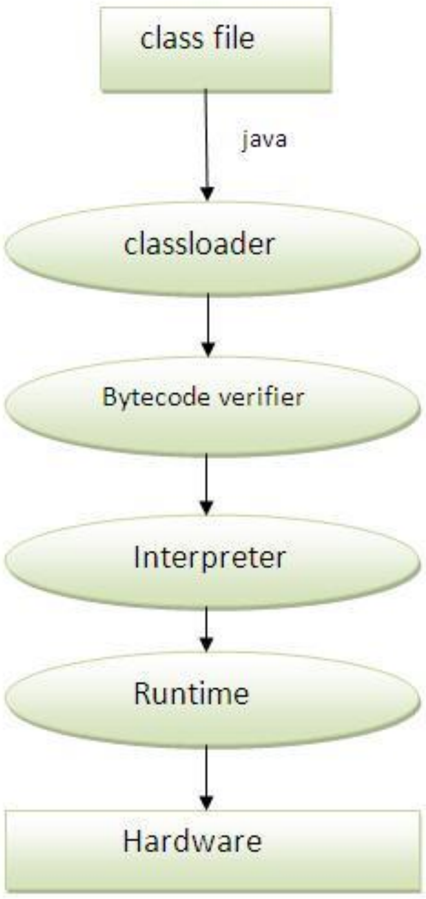
What happens at compile time?

At compile time, java file is compiled by Java Compiler (It does not interact with OS) and converts the java code into bytecode.


class Simple{
public static void main(String args[]){
System.out.println("hello java");
}
}
compiler
bytecode
Simple.java
Simple.class

What happens at runtime?

At runtime, following steps are performed:



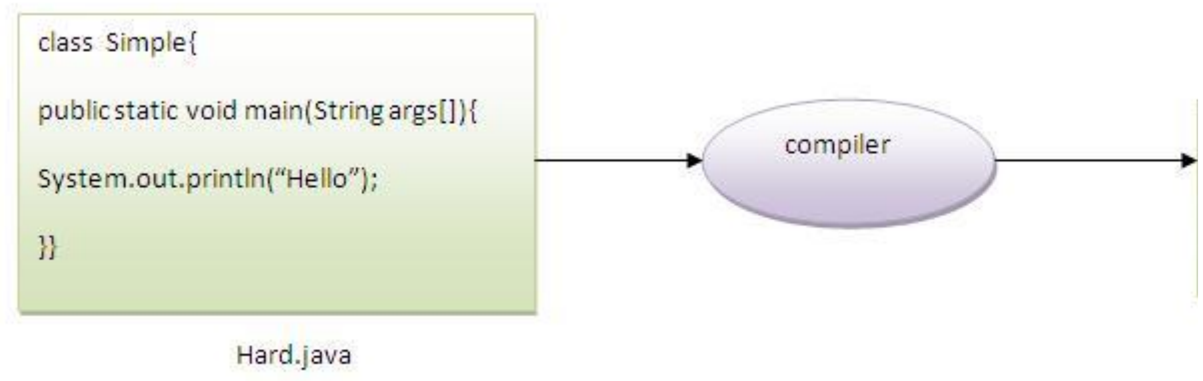
Classloader: is the subsystem of JVM that is used to load class files.

Bytecode Verifier: checks the code fragments for illegal code that can violate access right to objects.

Interpreter: read bytecode stream then execute the instructions.

Q)Can you save a java source file by other name than the class name?

Yes, if the class is not public. It is explained in the figure given below:



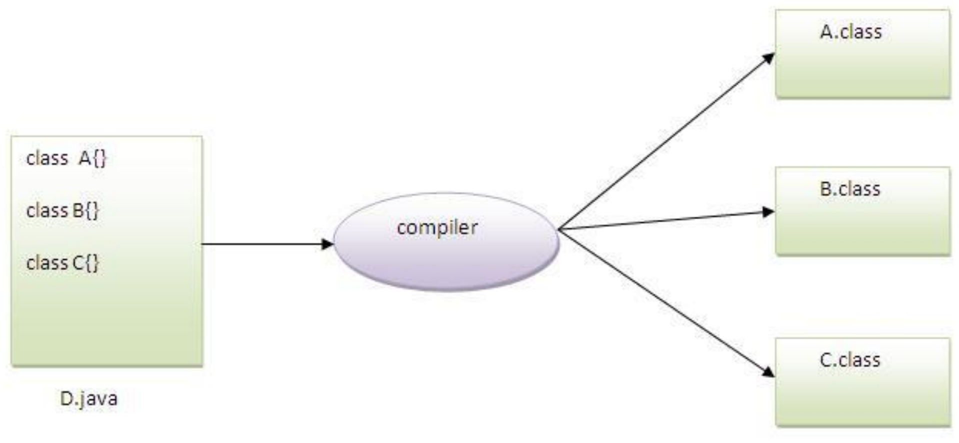
```
class  Simple{

public static void main(String args[]){

System.out.println("Hello");

}}
```

To compile: javac Hard.java

To execute: java Simple

Q)Can you have multiple classes in a java source file?

Yes, like the figure given below illustrates:



How to set path in Java

1. How to set path of JDK in Windows OS
    1. Setting Temporary Path of JDK
    2. Setting Permanent Path of JDK
2. How to set path of JDK in Linux OS

The path is required to be set for using tools such as javac, java etc.

If you are saving the java source file inside the jdk/bin directory, path is not required to be set because all the tools will be available in the current directory.

But If you are having your java file outside the jdk/bin folder, it is necessary to set path of JDK.

There are 2 ways to set java path:

1. temporary
2. permanent

1) How to set Temporary Path of JDK in Windows

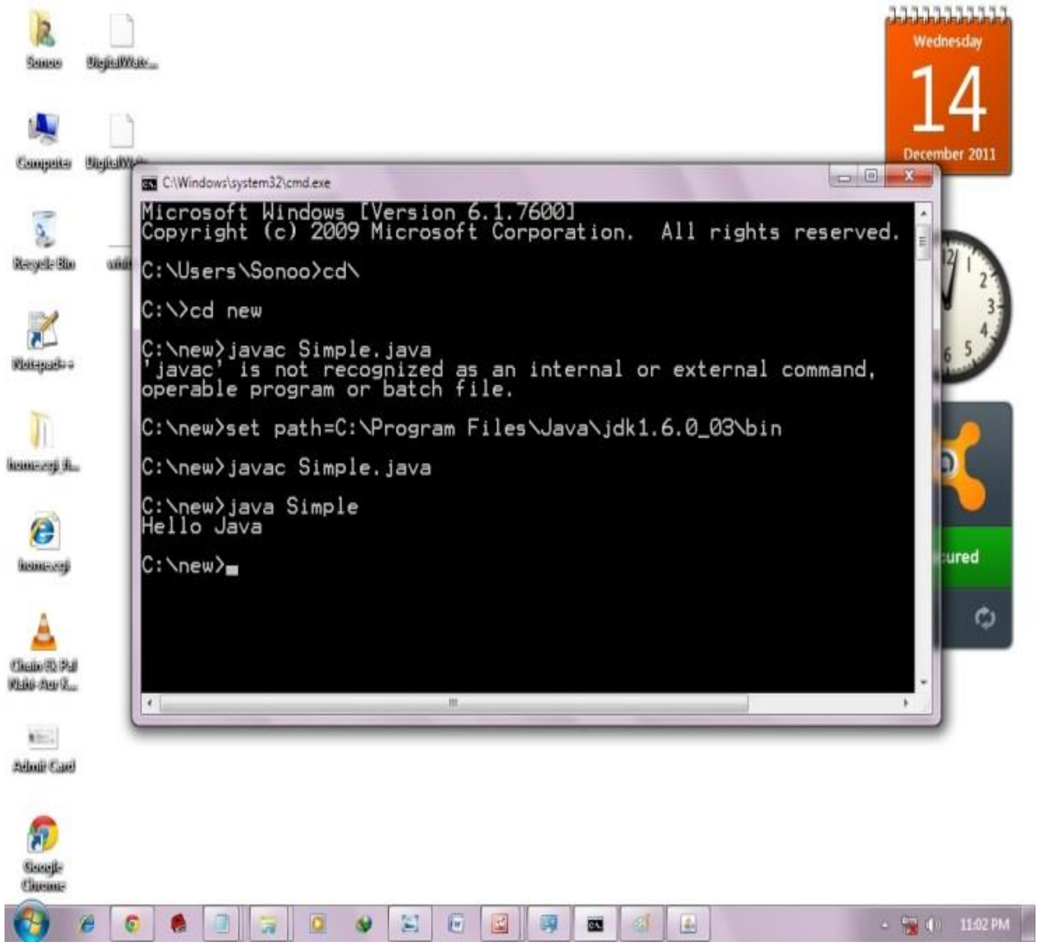
To set the temporary path of JDK, you need to follow following steps:

- Open command prompt
- copy the path of jdk/bin directory
- write in command prompt: set path=copied_path

For Example:

```
set path=C:\Program Files\Java\jdk1.6.0_23\bin
```

Let's see it in the figure given below:



```
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Sonoo>cd\

C:\>cd new

C:\new>javac Simple.java
'javac' is not recognized as an internal or external command,
operable program or batch file.

C:\new>set path=C:\Program Files\Java\jdk1.6.0_03\bin

C:\new>javac Simple.java

C:\new>java Simple
Hello Java

C:\new>
```

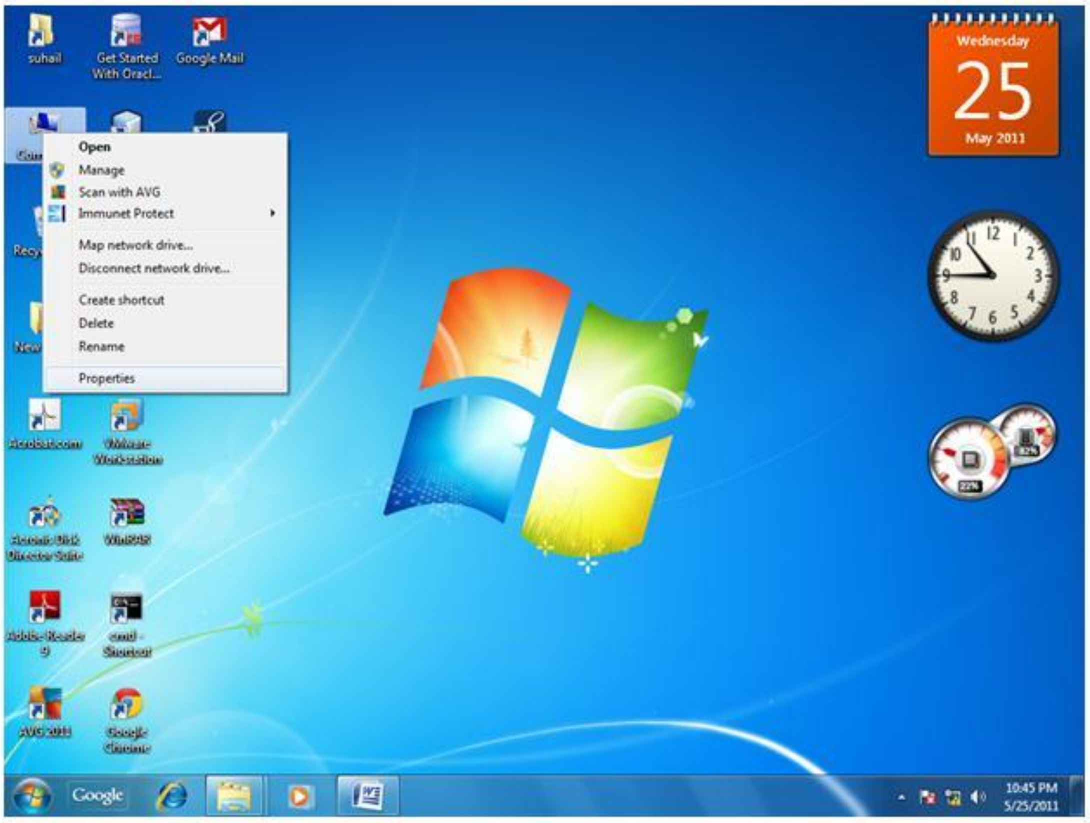
2) How to set Permanent Path of JDK in Windows

For setting the permanent path of JDK, you need to follow these steps:

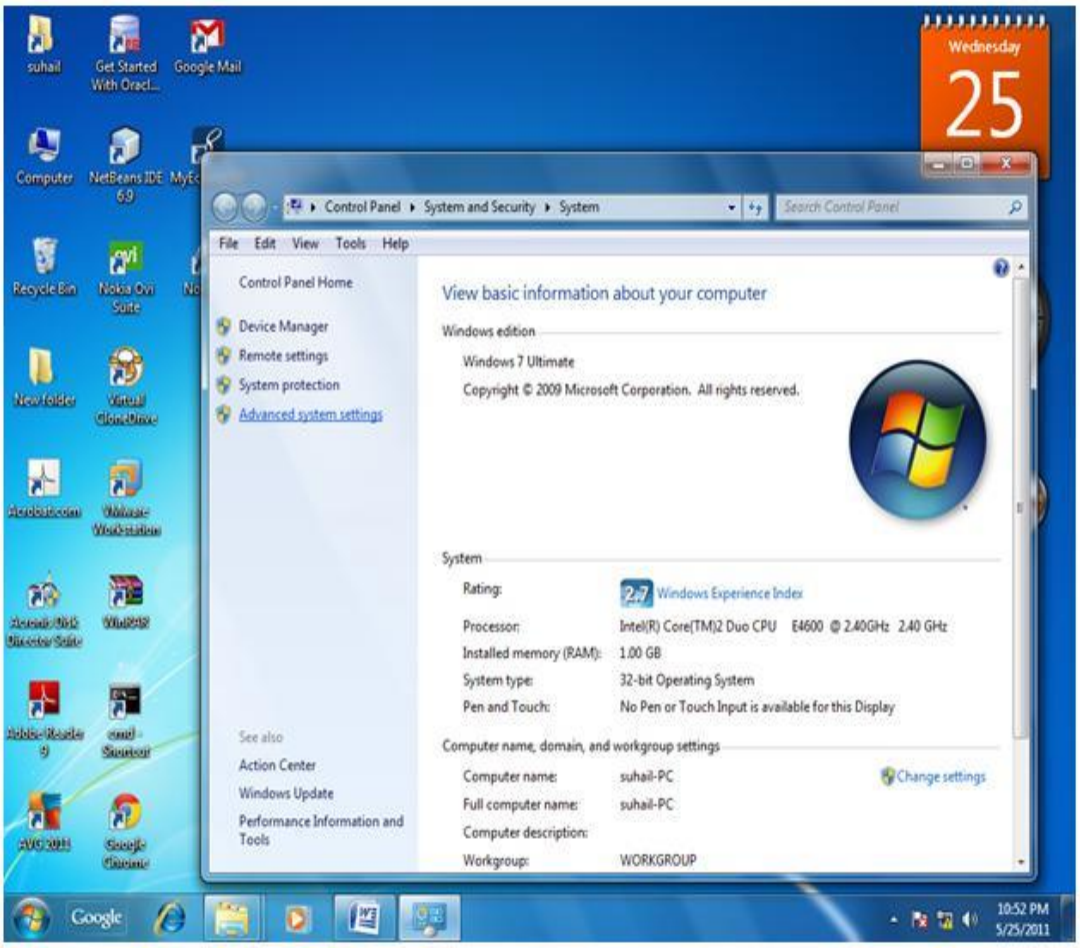
- Go to MyComputer properties -> advanced tab -> environment variables -> new tab of user variable -> write path in variable name -> write path of bin folder in variable value -> ok -> ok -> ok

For Example:

1)Go to MyComputer properties



2)click on advanced tab



View basic information about your computer

Windows edition

Windows 7 Ultimate

Copyright © 2009 Microsoft Corporation. All rights reserved.

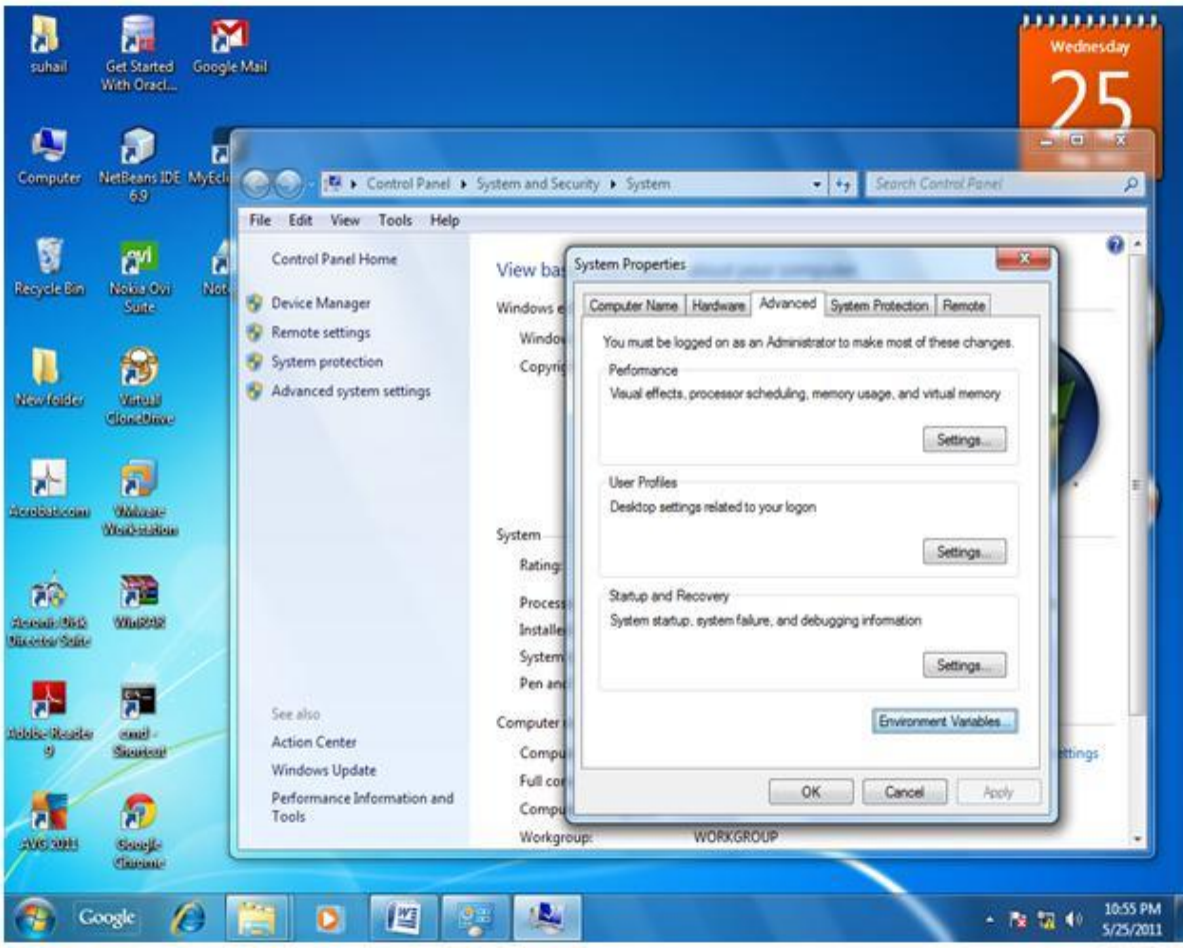
System

Rating:	2.7 Windows Experience Index
Processor:	Intel(R) Core(TM)2 Duo CPU E4600 @ 2.40GHz 2.40 GHz
Installed memory (RAM):	1.00 GB
System type:	32-bit Operating System
Pen and Touch:	No Pen or Touch Input is available for this Display

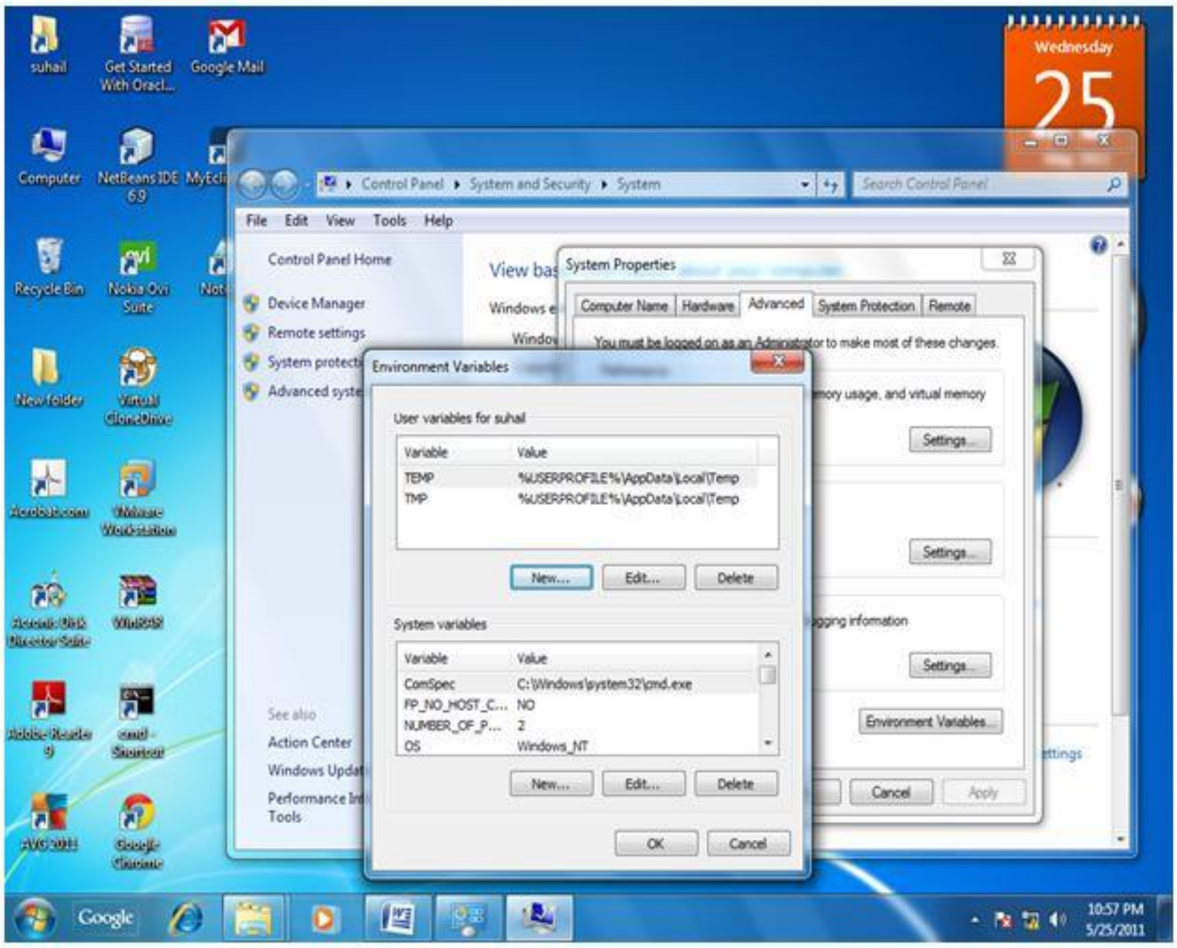
Computer name, domain, and workgroup settings

Computer name:	suhail-PC
Full computer name:	suhail-PC
Computer description:	
Workgroup:	WORKGROUP

3)click on environment variables


System Properties
Computer Name
Hardware
Advanced
System Protection
Remote
You must be logged on as an Administrator to make most of these changes.
Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...
User Profiles
Desktop settings related to your logon
Settings...
Startup and Recovery
System startup, system failure, and debugging information
Settings...
Environment Variables...
OK
Cancel
Apply

4)click on new tab of user variables



You must be logged on as an Administrator to make most of these changes.

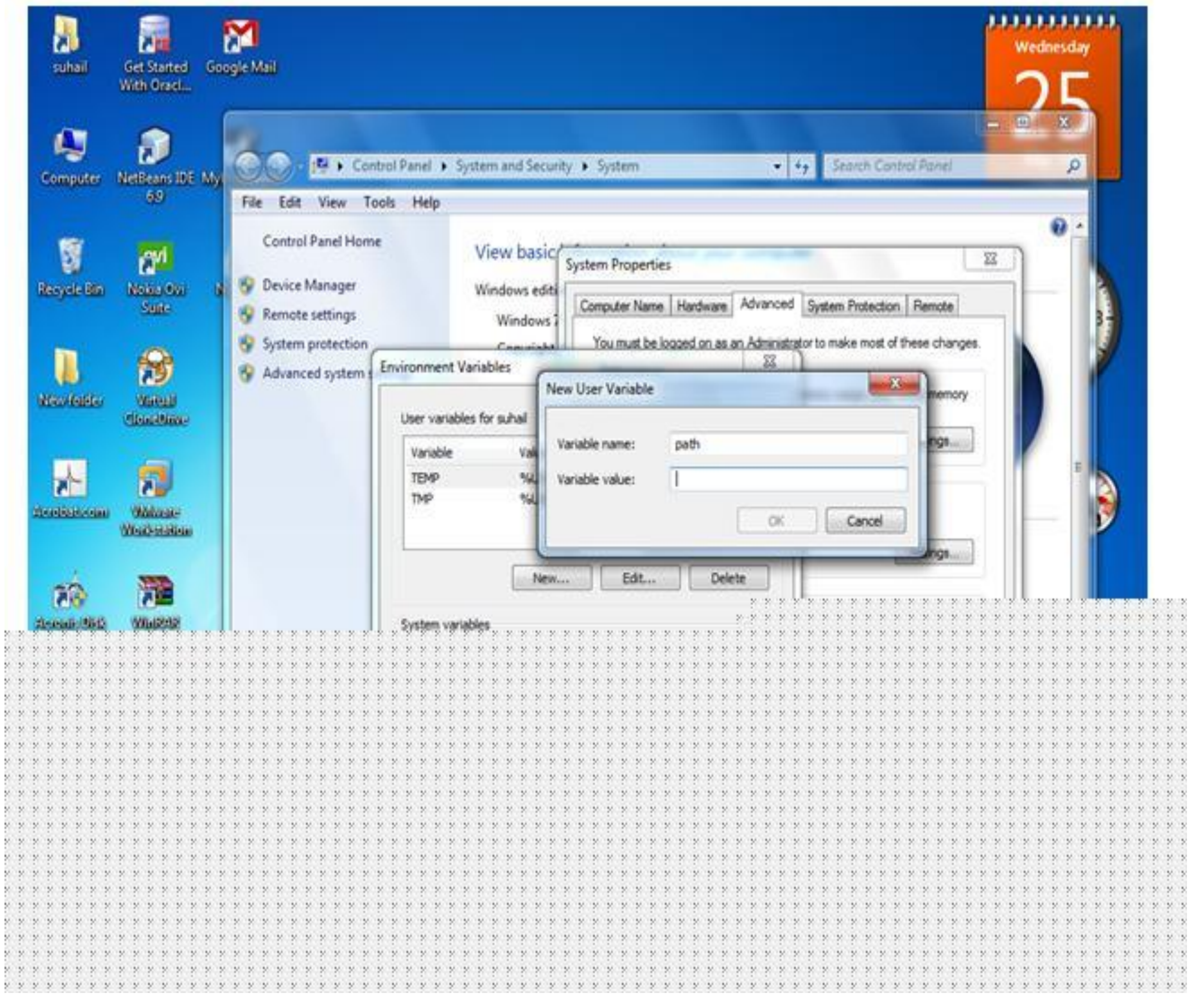
User variables for suhail

Variable	Value
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_P...	2
OS	Windows_NT

5)write path in variable name

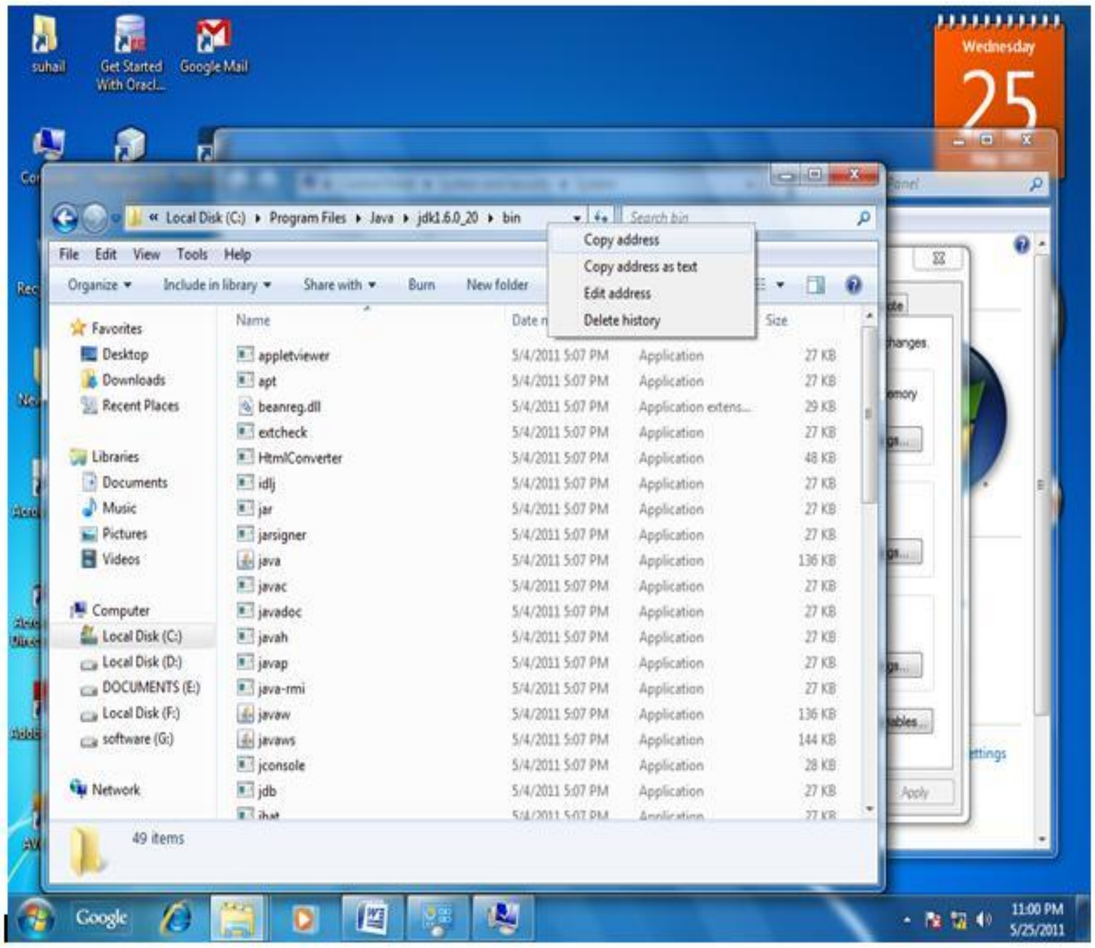


New User Variable

Variable name: path

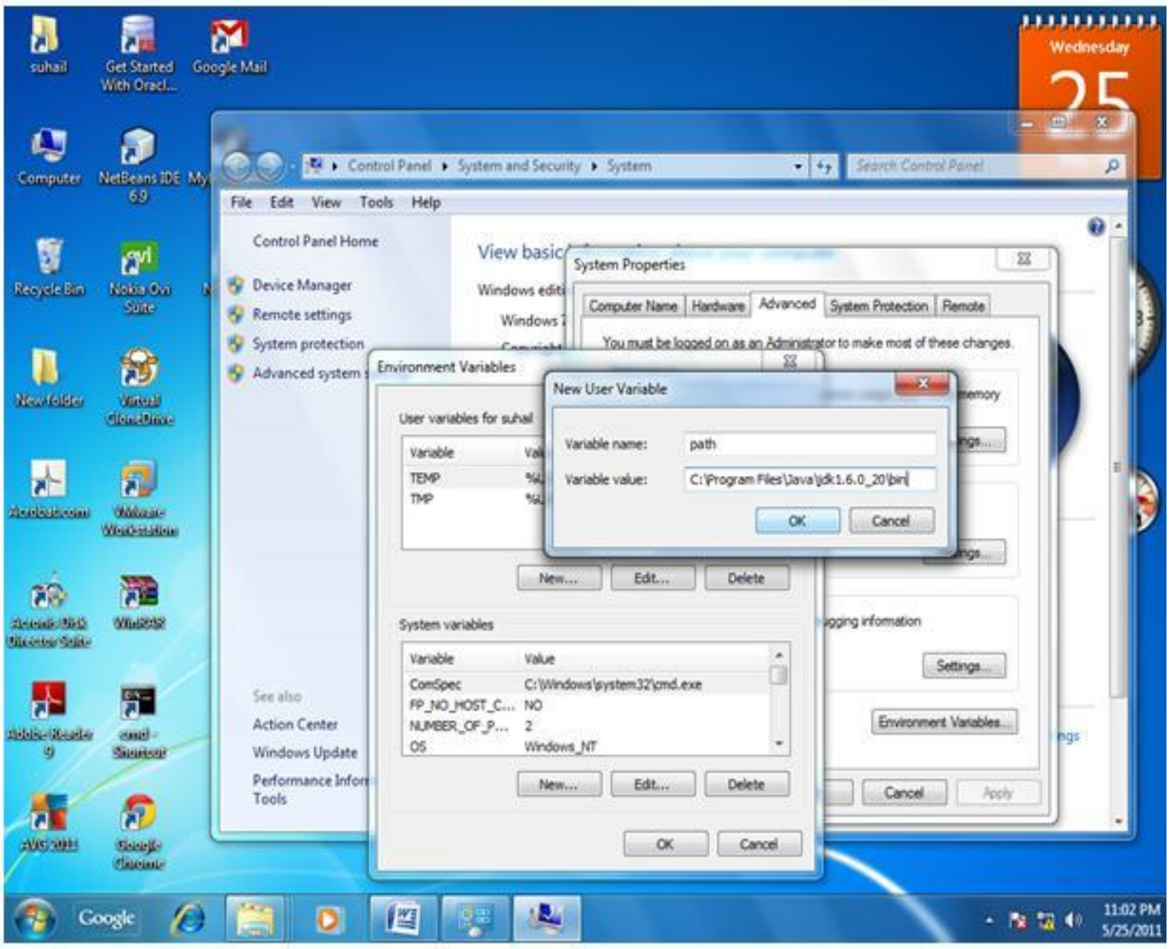
Variable value:

6)Copy the path of bin folder



Name	Date m		Size
appletviewer	5/4/2011 5:07 PM	Application	27 KB
apt	5/4/2011 5:07 PM	Application	27 KB
beanreg.dll	5/4/2011 5:07 PM	Application extens...	29 KB
extcheck	5/4/2011 5:07 PM	Application	27 KB
HtmlConverter	5/4/2011 5:07 PM	Application	48 KB
idlj	5/4/2011 5:07 PM	Application	27 KB
jar	5/4/2011 5:07 PM	Application	27 KB
jarsigner	5/4/2011 5:07 PM	Application	27 KB
java	5/4/2011 5:07 PM	Application	136 KB
javac	5/4/2011 5:07 PM	Application	27 KB
javadoc	5/4/2011 5:07 PM	Application	27 KB
javah	5/4/2011 5:07 PM	Application	27 KB
javap	5/4/2011 5:07 PM	Application	27 KB
java-rmi	5/4/2011 5:07 PM	Application	27 KB
javaw	5/4/2011 5:07 PM	Application	136 KB
javaws	5/4/2011 5:07 PM	Application	144 KB
jconsole	5/4/2011 5:07 PM	Application	28 KB
jdb	5/4/2011 5:07 PM	Application	27 KB
jhat	5/4/2011 5:07 PM	Application	27 KB

7)paste path of bin folder in variable value

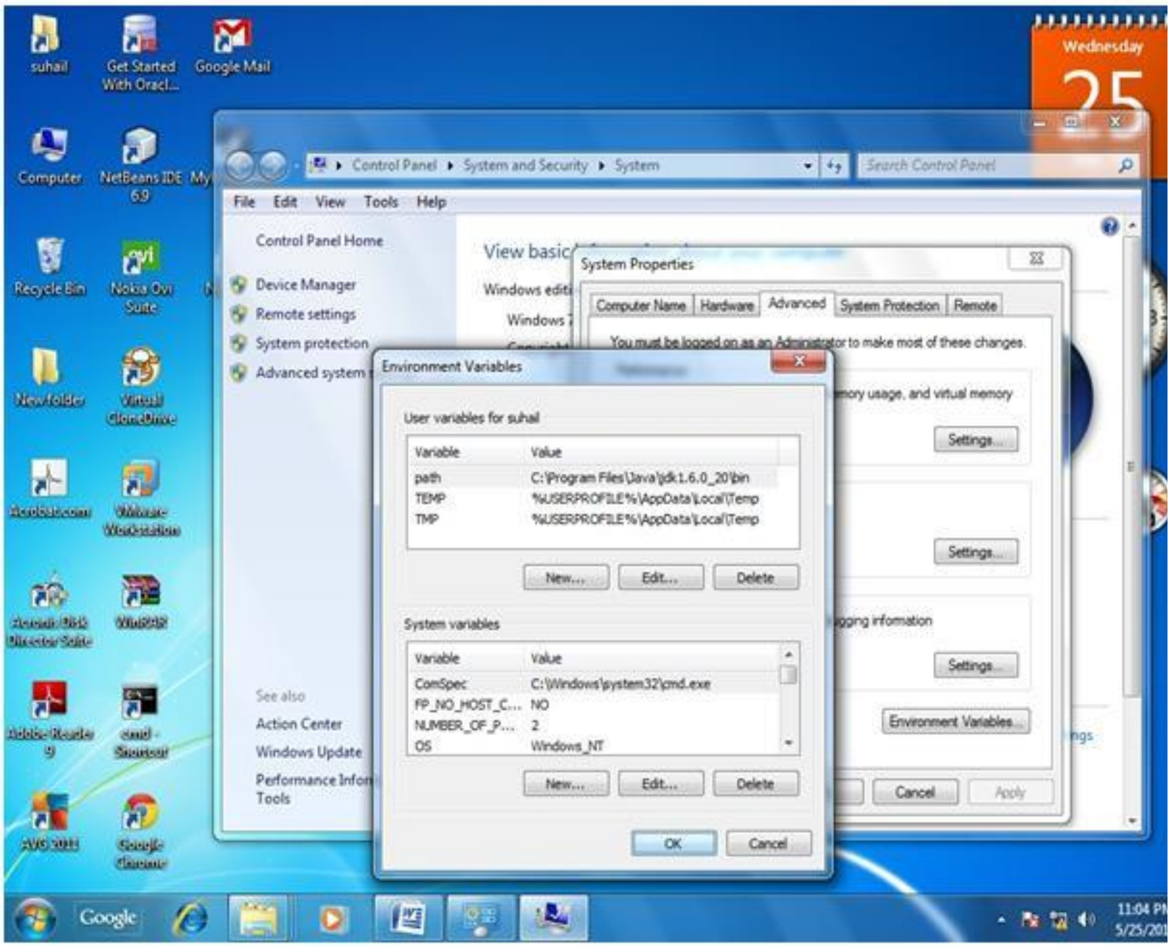


Variable name: path
Variable value: C:\Program Files\Java\jdk1.6.0_20\bin

Variable	Value
TEMP	%U
TMP	%U

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_P...	2
OS	Windows_NT

8)click on ok button



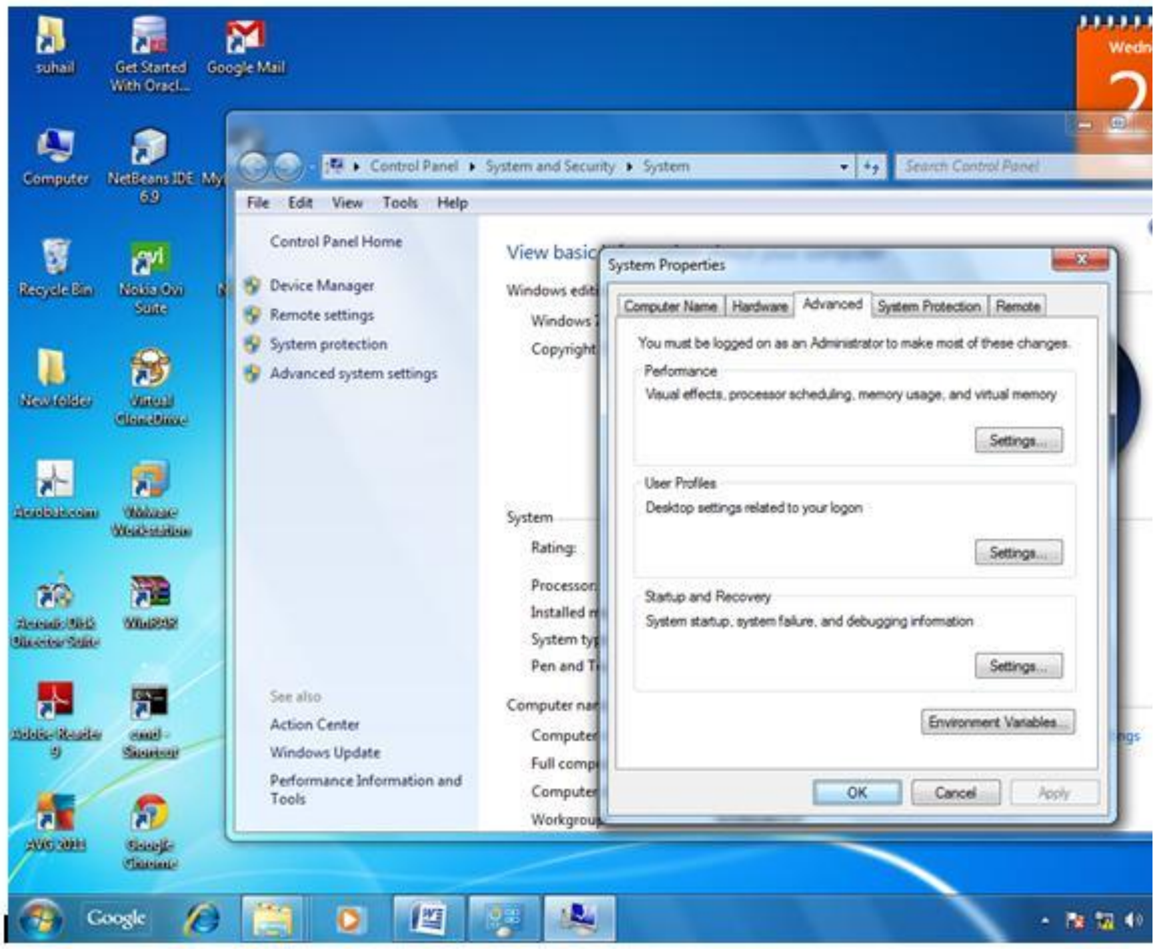
User variables for suhail

Variable	Value
path	C:\Program Files\Java\jdk1.6.0_20\bin
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_P...	2
OS	Windows_NT

9)click on ok button



You must be logged on as an Administrator to make most of these changes.

Visual effects, processor scheduling, memory usage, and virtual memory

Desktop settings related to your logon

System startup, system failure, and debugging information

Now your permanent path is set.You can now execute any program of java from any drive.

Setting Java Path in Linux OS

Setting the path in Linux OS is same as setting the path in the Windows OS. But here we use export tool rather than set. Let's see how to set path in Linux OS:

```
export PATH=$PATH:/home/jdk1.6.01/bin/
```

Here, we have installed the JDK in the home directory under Root (/home).

Difference between JDK, JRE and JVM

1. Brief summary of JVM
2. Java Runtime Environment (JRE)
3. Java Development Kit (JDK)

JVM

JVM (Java Virtual Machine) is an abstract machine. It is a specification that provides runtime environment in which java bytecode can be executed.

JVMs are available for many hardware and software platforms. JVM, JRE and JDK are platform dependent because configuration of each OS differs. But, Java is platform independent.

The JVM performs following main tasks:

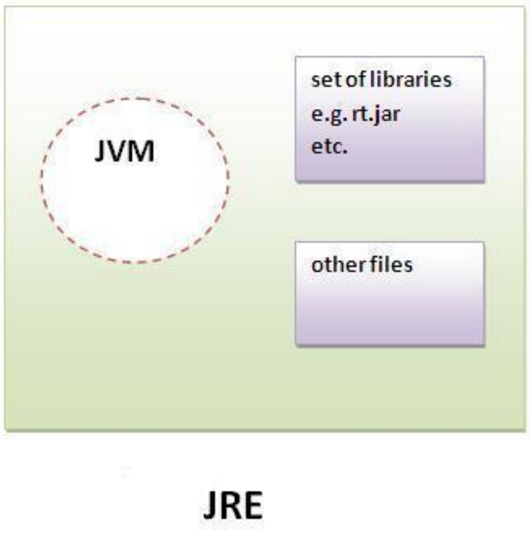
- Loads code
- Verifies code
- Executes code
- Provides runtime environment

-

JRE

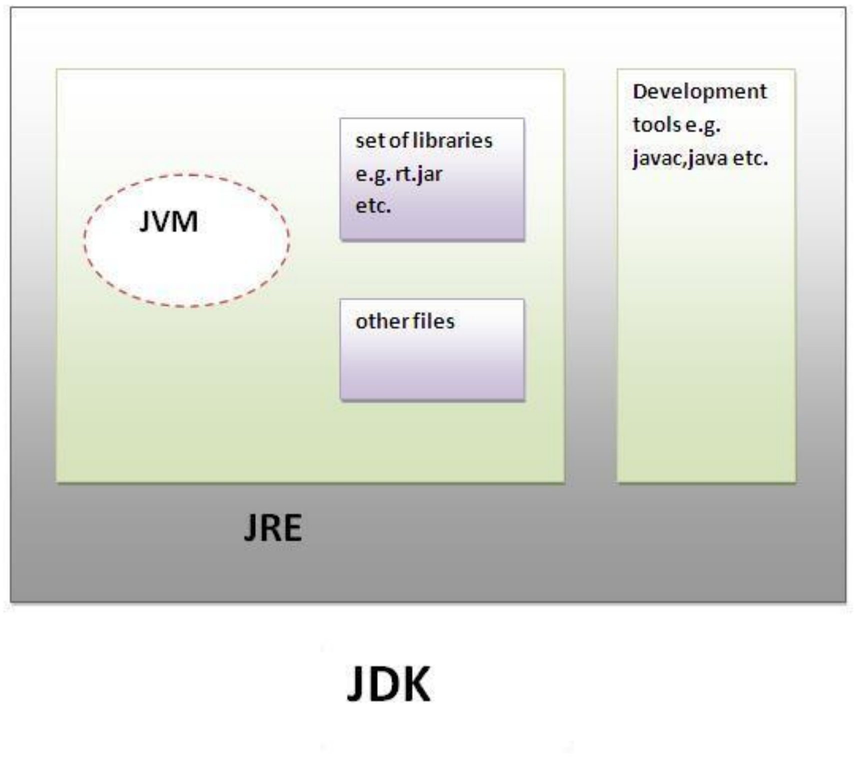
JRE is an acronym for Java Runtime Environment.It is used to provide runtime environment.It is the implementation of JVM.It physically exists.It contains set of libraries + other files that JVM uses at runtime.

Implementation of JVMs are also actively released by other companies besides Sun Micro Systems.



JDK

JDK is an acronym for Java Development Kit.It physically exists.It contains JRE + development tools.


set of libraries e.g. rt.jar etc.
JVM
other files
Development tools e.g. javac,java etc.
JRE
JDK

JVM (Java Virtual Machine)

1. Java Virtual Machine
2. Internal Architecture of JVM

JVM (Java Virtual Machine) is an abstract machine. It is a specification that provides runtime environment in which java bytecode can be executed.

JVMs are available for many hardware and software platforms (i.e.JVM is plateform dependent).

What is JVM?

It is:

1. **A specification** where working of Java Virtual Machine is specified. But implementation provider is independent to choose the algorithm. Its implementation has been provided by Sun and other companies.
2. **An implementation**Its implementation is known as JRE (Java Runtime Environment).

3. **Runtime Instance** Whenever you write java command on the command prompt to run the java class, and instance of JVM is created.

What it does?

The JVM performs following operation:

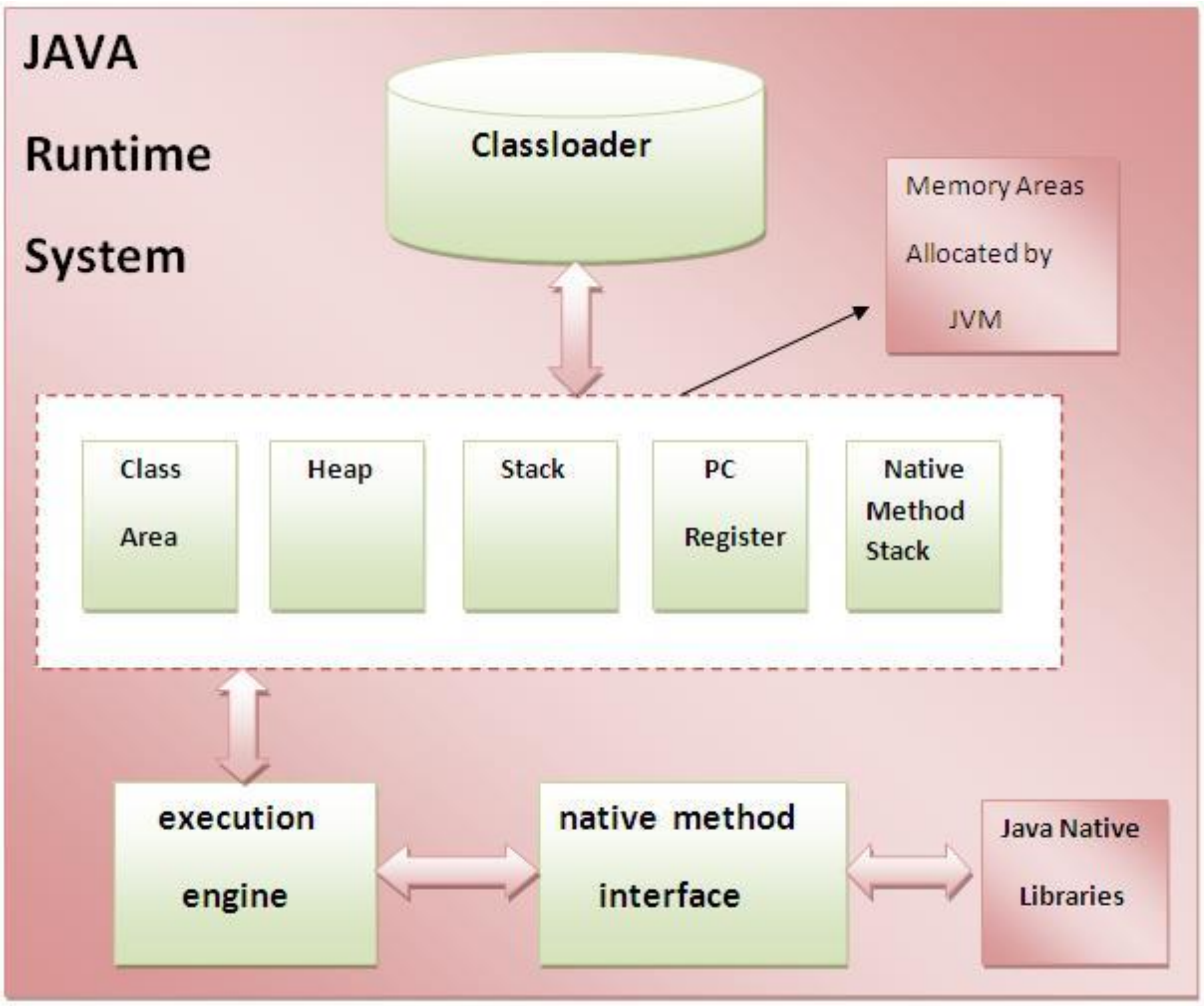
- Loads code
- Verifies code
- Executes code
- Provides runtime environment

JVM provides definitions for the:

- Memory area
- Class file format
- Register set
- Garbage-collected heap
- Fatal error reporting etc.

Internal Architecture of JVM

Let's understand the internal architecture of JVM. It contains classloader, memory area, execution engine etc.



1) Classloader:

Classloader is a subsystem of JVM that is used to load class files.

2) Class(Method) Area:

Class(Method) Area stores per-class structures such as the runtime constant pool, field and method data, the code for methods.

3) Heap:

It is the runtime data area in which objects are allocated.

4) Stack:

Java Stack stores frames.It holds local variables and partial results, and plays a part in method invocation and return.

Each thread has a private JVM stack, created at the same time as thread.

A new frame is created each time a method is invoked. A frame is destroyed when its method invocation completes.

5) Program Counter Register:

PC (program counter) register. It contains the address of the Java virtual machine instruction currently being executed.

6) Native Method Stack:

It contains all the native methods used in the application.

7) Execution Engine:

It contains:

1) A virtual processor

2) Interpreter: Readbytecode stream then execute the instructions.

3) Just-In-Time(JIT) compiler: It is used to improve the performance.JIT compiles parts of the byte code that have similar functionality at the same time, and hence reduces the amount of time needed for compilation.Here the term ?compiler? refers to a translator from the instruction set of a Java virtual machine (JVM) to the instruction set of a specific CPU.

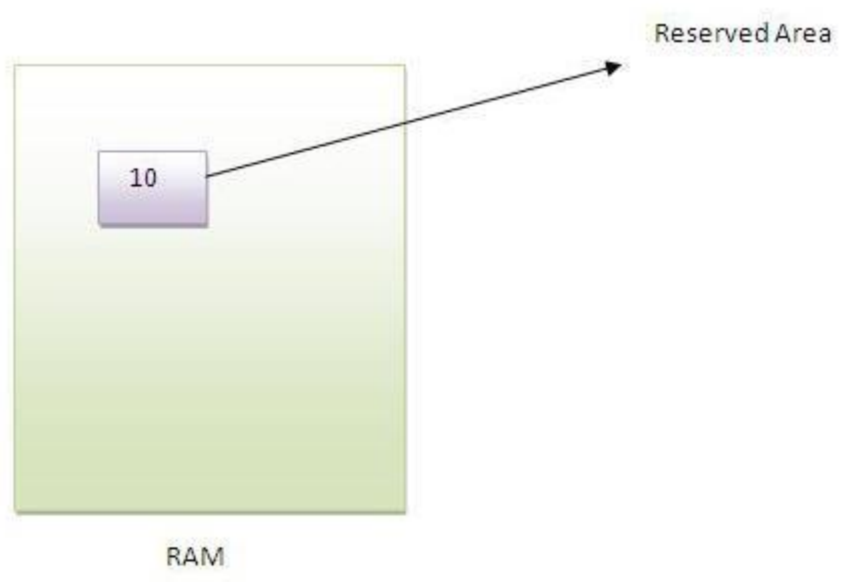
Variable and Datatype in Java

1. Variable
2. Types of Variable
3. Data Types in Java

Variable is a name of memory location. There are three types of variables: local, instance and static. There are two types of datatypes in java, primitive and non-primitive.

Variable

Variable is name of reserved area allocated in memory.

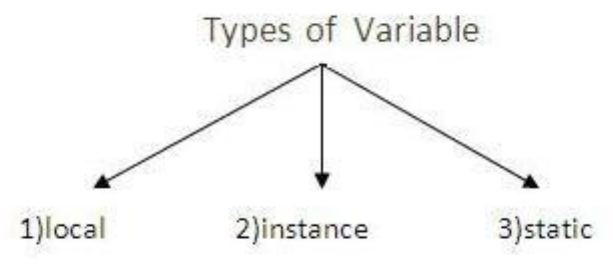


1. int data=50;//Here data is variable

Types of Variable

There are three types of variables in java

- local variable
- instance variable
- static variable



Local Variable

A variable that is declared inside the method is called local variable.

Instance Variable

A variable that is declared inside the class but outside the method is called instance variable . It is not declared as static.

Static variable

A variable that is declared as static is called static variable. It cannot be local.

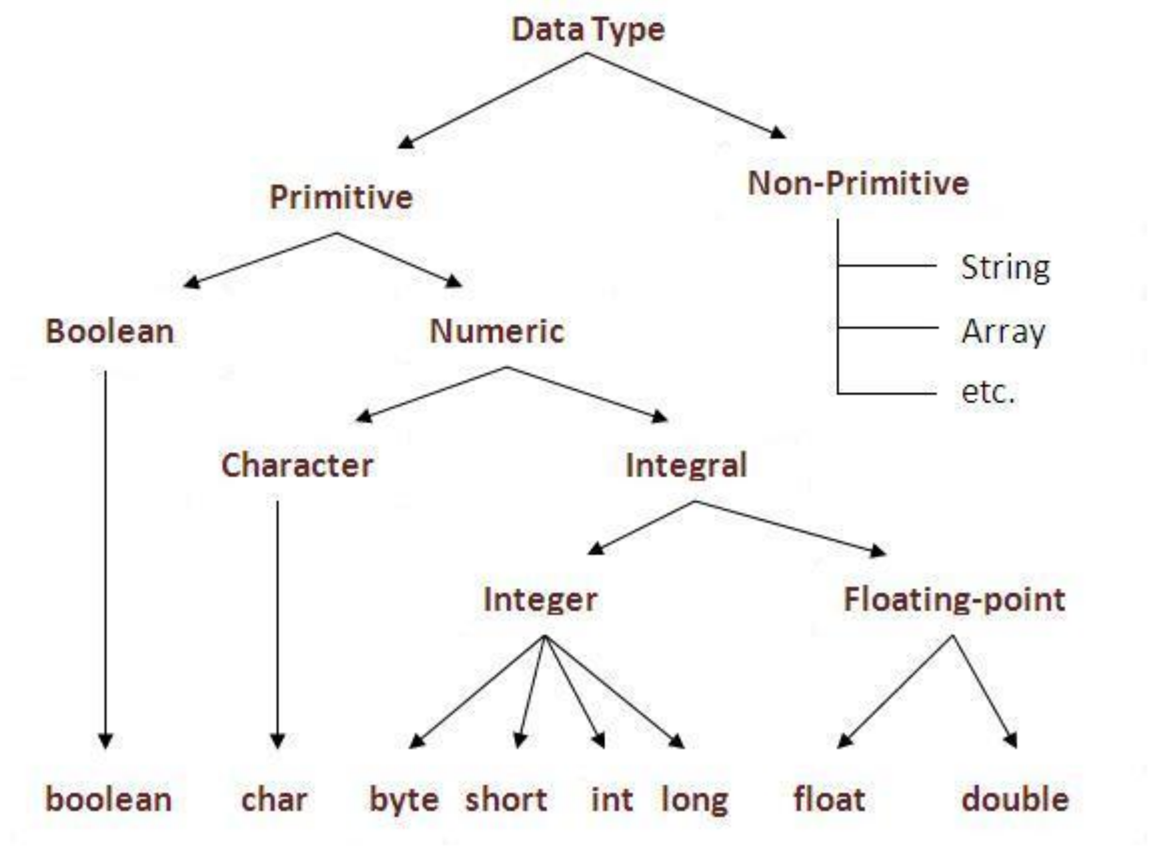
Example to understand the types of variables

```
class A{
int data=50;//instance variable
static int m=100;//static variable
void method(){
int n=90;//local variable
}
}//end of class
```

Data Types in Java

In java, there are two types of data types

- primitive data types
- non-primitive data types



Data Type	Default Value	Default size
boolean	false	1 bit
char	'\u0000'	2 byte
byte	0	1 byte
short	0	2 byte
int	0	4 byte
long	0L	8 byte
float	0.0f	4 byte
double	0.0d	8 byte

Why char uses 2 byte in java and what is \u0000 ?

because java uses unicode system rather than ASCII code system. \u0000 is the lowest range of unicodesystem.To get detail about Unicode see below.

Unicode System

Unicode is a universal international standard character encoding that is capable of representing most of the world's written languages.

Why java uses Unicode System?

Before Unicode, there were many language standards:

- **ASCII** (American Standard Code for Information Interchange) for the United States.
- **ISO 8859-1** for Western European Language.
- **KOI-8** for Russian.
- **GB18030 and BIG-5** for chinese, and so on.

This caused two problems:

1. A particular code value corresponds to different letters in the various language standards.
2. The encodings for languages with large character sets have variable length.Some common characters are encoded as single bytes, other require two or more byte.

To solve these problems, a new language standard was developed i.e. Unicode System.

In unicode, character holds 2 byte, so java also uses 2 byte for characters.

lowest value:\u0000

highest value:\uFFFF

Operators in java

Operator in java is a symbol that is used to perform operations. There are many types of operators in java such as unary operator, arithmetic operator, relational operator, shift operator, bitwise operator, ternary operator and assignment operator.

Operators	**Precedence**
postfix	`expr++ expr--`
unary	`++expr --expr +expr -expr~ !`
multiplicative	`* / %`
additive	`+ -`
shift	`<<>>>>>`
relational	`<><= >= instanceof`
equality	`== !=`
bitwise AND	`&`
bitwise exclusive OR	`^`
bitwise inclusive OR	`|`
logical AND	`&&`
logical OR	`||`
ternary	`? :`
assignment	`= += -= *= /= %= &= ^= |= <<= >>= >>>=`