Thoughts –

I believe, sharing knowledge is first step of learning. It’s really amazing when career and passion work together.

My passion is to teach and build the career of others as it reveals my inner desire and impart my knowledge. I tried to teach the best things what I mostly learnt. I will continuously learn more about what I am teaching. Meanwhile, I will also be learning a lot from my students.

Learning Continues….

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my family and friends for their support, guidance, and encouraged me to write this book.

My Special thanks goes to My Father, who has been a role model for me throughout my life and he has been a great inspiration for my teaching passion.

Finally, I would like to thank my wife, she is always there for cheering me up and stood by me all the time.

Disclaimer

Please mail me at clickraigarh@gmail.com for any query. Thank you.

Table of Contents

About Me

Hello all,

Myself Rakesh Acharya, working as a Project Manager in an IT company. I have 12 years of extensive work experience in JAVA/J2ee Technology as a core developer and now handling the role of project manager☺.

After completion of my engineering, I had joined in a small company and worked on struts framework and learnt a lot from various projects. As year passed and experience increased, I joined different organizations like India today Group, TCS, CSC and Cognizant. This was a great learning experience to learn different technologies and new frameworks with different projects in different organizations. I have worked on Struts, JSF, spring, Hibernate, Chordiant frameworks.

The day I have completed my 7 years of experience, I started giving in-house trainings within the organization and trained my team in different skill sets (JAVA/J2ee) and processes like Agile Scrum and Project Management.

This is my first book that I am sharing with you all. This book contains the basics of JAVA so that a biginner can start writing the java program in an easiest way with a minimal time. This book consists of majorly working examples that will help you to write and execute programs. This book is similar to my classroom training based on examples rather than theoritical.

I am sure you all will definitely learn the java basics from this book and help me to publish next version with advance topics.

I would like to say thanks to my wife because of her encouragment to write this book as she is in HR, so she has helped me to learn what actually IT industries are looking from technical guys.

"This is just the beginning but not the end ……………………………….."

https://www.linkedin.com/in/rakesh-acharya-46777318

INTRODUCTION TO JAVA

WHAT IS JAVA?
I heard the word java 1st in 2000 when I took admission in engineering. Word JAVA was a big thing for me at that time because I was not heard this word before. I started my college and 1st day during introduction I found most of the classmates those are from different schools and locations and they have already known about the JAVA and programming and I was only aware about c++☹. I thought let's start to know what is java? I started to discuss with classmates and my college professors and this was the time I came to know what is JAVA.

Let me share my findings about JAVA with you in short as I have already commited that I will not tell you much theory.

- Java first was developed in 1991 by James Gosling at Sun Microsystems to be used with interactive TV technology which had failed to find a market.
- When The World-Wide Web became popular in 1995, Java came to life again.
- Java is now considered as the native language of the Internet.
- Java is a C/C++ based language.
- Java is a Full object-oriented language

Note:- Now it's your turn to find out little bit more about java (Theory).

CHARACTERISTICS OF JAVA

- **JAVA IS SIMPLE**
- **JAVA IS OBJECT-ORIENTED**
- **JAVA IS DISTRIBUTED**
- **JAVA IS INTERPRETED**
- **JAVA IS ROBUST**
- **JAVA IS SECURE**
- **JAVA IS ARCHITECTURE-NEUTRAL**
- **JAVA IS PORTABLE**
- **JAVA'S PERFORMANCE**
- **JAVA IS MULTITHREADED**
- **JAVA IS DYNAMIC**

Hey guys, am I right with above points? Yes I am but how? Let’s discuss all the points one by one. Also, we will discuss these all points during our next topic and programming.

Simple – Java programs are easy to write and debug. Will see more details in our program.

Object Oriented – It uses the oops concept – inheritance, Encapsulation and polymorphism.

Distributed - The widely used protocols like HTTP and FTP are developed in java. Internet programmers can call functions on these protocols and can get access the files from any remote machine on the internet rather than writing codes on their local system.

Robust - Java has the strong memory allocation and automatic garbage collection mechanism. It provides the powerful exception handling and type checking mechanism as compare to other programming languages. Compiler checks the program whether there any error and interpreter checks any run time error and makes the system secure from crash. All of the above features makes the java language robust.

Performance: JVM uses some technologies like JIT compiler which converts byte code to optimize executable code.

Architecture Neutral: The goal was written once; run anywhere, anytime, forever. OS upgrade, processor upgrades and changes in core system will not affect the JVM to execute the code.

Portable -The feature Write-once-run-anywhere makes the java language portable provided that the system must have interpreter for the JVM.

Dynamic - While executing the java program the user can get the required files dynamically from a local drive or from a computer thousands of miles away from the user just by connecting with the Internet.

Secure

-No memory pointers

-Programs runs inside the virtual machine sandbox

-Array index limit checking

-Code pathologies reduced by

bytecode verifier - checks classes after loading

class loader - confines objects to unique namespaces. Prevents loading a hacked "java.lang.SecurityManager" class, for example.

Security manager - determines what resources a class can access such as reading and writing to the local disk.

Java is architecture-neutral – Means wrire your code once ; run anywhere, anytime, forever. OS upgrade, processor upgrades and changes in core system will not affect the JVM to execute the code.

JAVA IS PORTABLE - The feature Write-once-run-anywhere makes the java language portable provided that the system must have interpreter for the JVM

JAVA'S PERFORMANCE - JVM uses some technologies like JIT compiler which converts byte code to optimizes executable code

JAVA IS DYNAMIC - While executing the java program the user can get the required files dynamically from a local drive or from a computer thousands of miles away from the user just by connecting with the Internet.

JAVA IS MULTITHREADED/Multithreading - Lightweight processes, called threads, can easily be spun off to perform multiprocessing.

JDK VERSION

What does this mean JDK Version? Ohh I again missed to tell about the JVM and JRE? Are you aware about these two words? I am sure most of you all are already known but when I ran my 1st application, I was not aware about this and system told me JRE not available is your machine ☺could not create the Java Virtual Machine. Again it was a big problem for me and I had to find the JRE and JVM so that I can run my 1st java program.
I know very well what I was thinking at that time ☺ let me share with you.

I thought JAVA is very tough to learn. I am getting so many problems in initial stage, so what will happen in future with me if I will continue with Java Language. Let's plan for some other programming language☺.

Then I took one step back and discuss with my professor and classmates who know how to run java program. What is prerequisite required to run JAVA program.

Let's start with configuring your system to run java program.

- I am not going with system hardware configuration as we know very well that now a days desktop or laptop are coming with very good configuration.
- What is JVM ? :- JVM (Java Virtual Machine) is an abstract machine. It is a specification that provides runtime environment in which java byte code can be executed. JVMs are available for many hardware and software platforms .Note:- JVM is platform dependent.
  Let's Understand little bit in deep about JVM work. JVM Loads code, Verifies code , Executes code Provides runtime environment for java code to run.
  JVM provides definitions for the Memory area, Class file format, Register set, Garbage-collected heap, Fatal error reporting.
- What is JRE? JRE (JAVA Run Time Environment) is used to provide runtime environment. It is the implementation of JVM and physically exist in

computer drive. It contains set of libraries and other files that JVM uses at runtime.
- What is JDK? JDK(JAVA Development Kit) contains JRE and other development tools like JAVAC, JAVA

NOTE :-JVM, JRE and JDK are platform dependent because configuration of each OS differs. But, Java is platform independent.

JDK EDITION

In java world many editions are available I will discuss here about most popular used editions like –

Edition	**Release Year**
J2SE 1.4	**2002**
J2SE 5.0	**2005**
Java SE 6	**2006**
Java SE 7	**2011**
Java SE 8	**2014**

Note: - I am not going to cover any specific edition in our program. I will try to cover all the important things from each edition with comparison.

JAVA IDE

I was not aware about the different IDE (integrated development tools) at the time when I was beginner. I was writing my java program in notepad. But truly speaking it was very difficult to write in notepad and to find the error if we missed any syntax. IDE help us to write program with proper syntax.You can find many IDE for your program writing. I will use Eclipse here for our program writing.
Some IDE are as below.
- **Eclipse**
- J Developer
- Net Beans

GETTING STARTED WITH JAVA PROGRAMING

Uff I think we have talked a lot about Java history and prerequisites now. We have to step forward towardS programing and concept.

CREATE YOUR PROGRAM.

We are in the stage that we have to start programming and learn the java. So let's write one small program after that will discuss each line before execution.
I will write my first program in notepad and will discuss about that.

Example 1.1

Step 1:- open a notepad.

Step 2:- Write the java code as per below line .

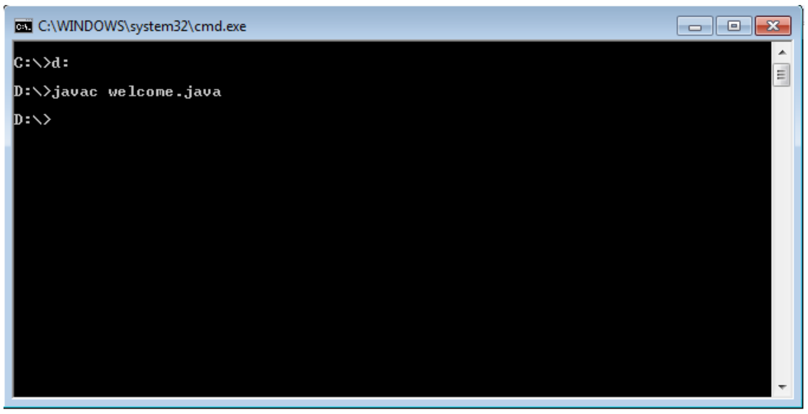
```
/*
        This application program prints Welcome
        to Java!
*/
package JavaWorld;
```

```
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome Rock!");
 }}
```

Step 3:- Save the file with the same name as class name with file extension as java. So, my saved file name is **Welcome.java**

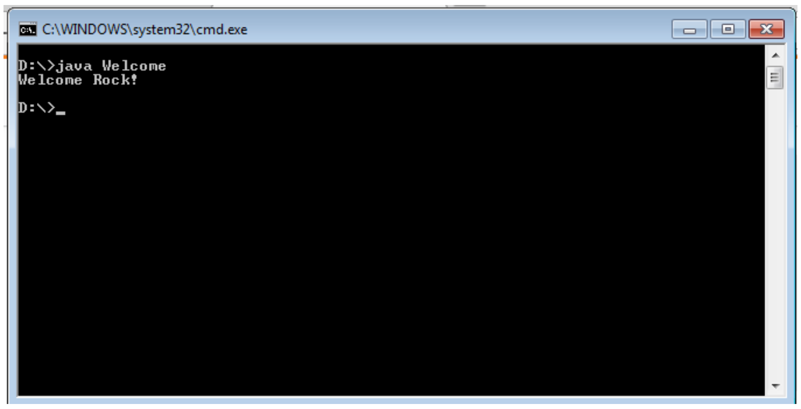
Ohh that's great...I have finally started to write my 1st Java program. Now let's run the program. To run the java program we have to compile java file first. I know you will ask me how to compile a java file? To compile a java file if you are not using any IDE. Simple steps are here.

Step 1:- Open command prompt or type cmd in run of your windows.
Step 2 :- Go to directory where you have saved your java file Welcome.java
Step 3:- Type Javac Welcome.java
This will take few seconds to compile your java file.



```
C:\>d:
D:\>javac welcome.java
D:\>
```

Step 4:- if compile will get successful you will get same line in command prompt and a new file will get create in same location with the name of command Welcome.class.
Step 5:- type java welcome



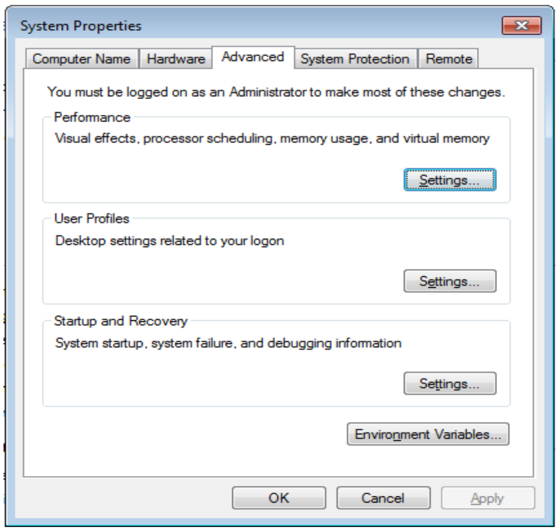
```
D:\>java Welcome
Welcome Rock!

D:\>_
```

You will get the output "Welcome Rock!"
Congrats. You have executed your 1st program successfully.
Hey!! What happened? Some of you are likethis☹. Not able to compile? Getting error like JAVAC not found or unknown command? Main function error like this? Again I forget to tell one thing that how to set environment variable in your system so that you can run your java program.

If you are running java program from command line, this is very important that you have to setup environment variable. Now how to set the environment variable and what are the variable required to set.
To execute any Java program we have to set 3 variables as an environment variable.

Variable Name	**Path**
JAVA_Home	Path of java home directory for me (C:\Program Files\Java)
Classpath	Path of lib folder in side java home(C:\Program Files\Java\jdk1.8.0_102\jre\lib)
Path	Path of bin folder in side java home (C:\Program Files\Java\jdk1.8.0_102\jre\bin)

To set these all variable click on my computer→ go to properties→Advance system setting→


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...

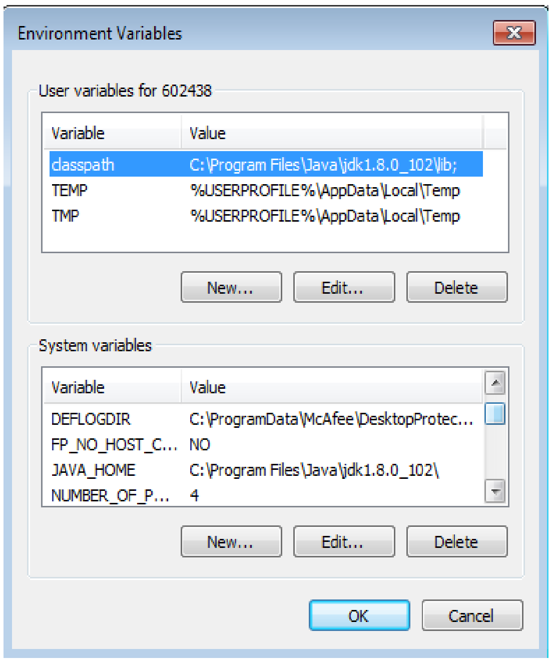
User Profiles
Desktop settings related to your logon
Settings...

Startup and Recovery
System startup, system failure, and debugging information
Settings...

Environment Variables...

OK | Cancel | Apply

Click on Environment Variable



Environment Variables

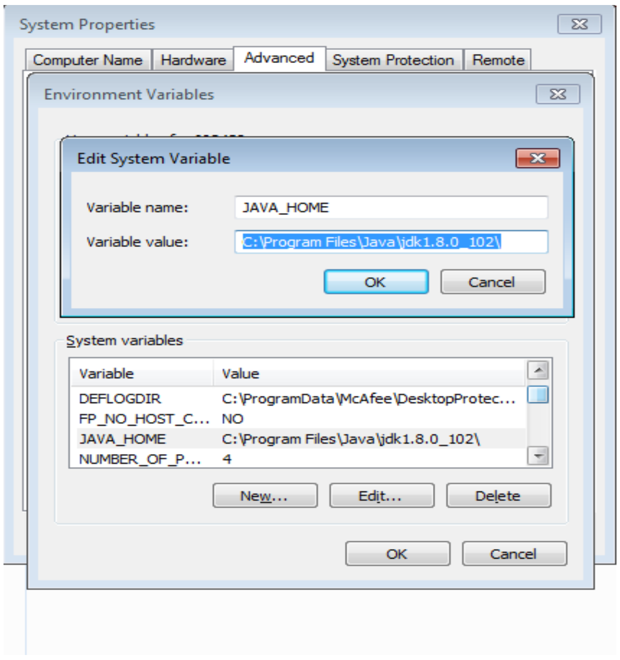
User variables for 602438

Variable	Value
classpath	C:\Program Files\Java\jdk1.8.0_102\lib;
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
DEFLOGDIR	C:\ProgramData\McAfee\DesktopProtec...
FP_NO_HOST_C...	NO
JAVA_HOME	C:\Program Files\Java\jdk1.8.0_102\
NUMBER_OF_P...	4

If your system already have these variables, then select the variable and click on **Edit.** If you didn't find the variable, click on **New** and enter the details on given field as below.



System Properties

Environment Variables

Edit System Variable

Variable name: JAVA_HOME

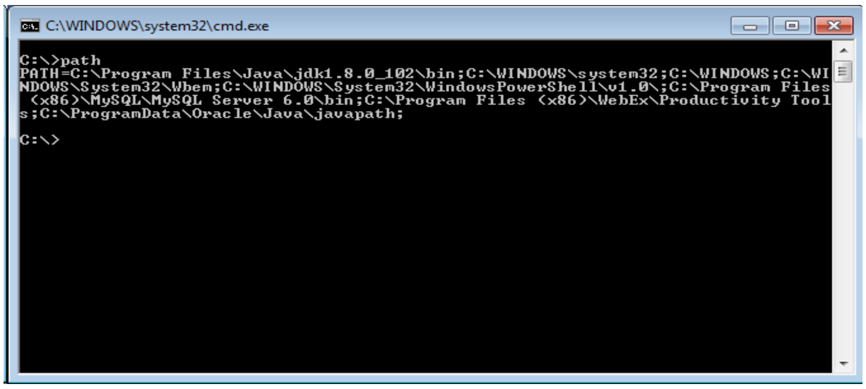
Variable value: C:\Program Files\Java\jdk1.8.0_102\

System variables

Variable	Value
DEFLOGDIR	C:\ProgramData\McAfee\DesktopProtec...
FP_NO_HOST_C...	NO
JAVA_HOME	C:\Program Files\Java\jdk1.8.0_102\
NUMBER_OF_P...	4

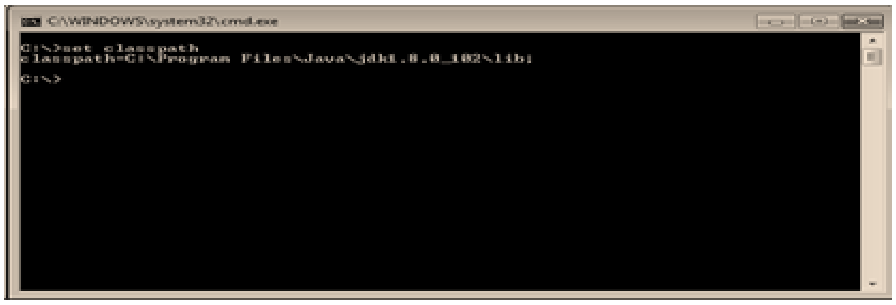
Once you done with all three variable, click on ok and close the window. To check these variables whether it has been set properly or not, go to command prompt and type the following command.

C:\> path



```
C:\>path
PATH=C:\Program Files\Java\jdk1.8.0_102\bin;C:\WINDOWS\system32;C:\WINDOWS;C:\WI
NDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\Program Files
 (x86)\MySQL\MySQL Server 6.0\bin;C:\Program Files (x86)\WebEx\Productivity Tool
s;C:\ProgramData\Oracle\Java\javapath;

C:\>
```

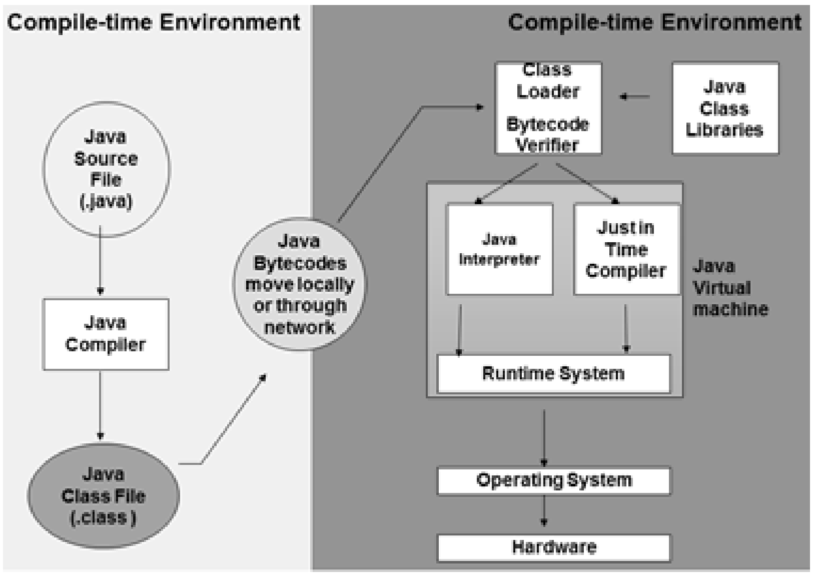
This will show the above details with java bin path
C:\> set classpath



```
C:\>set classpath
classpath=C:\Program Files\Java\jdk1.8.0_102\lib;

C:\>
```

This will show the above details with java lib path

How the above program works in JVM ?
Let's understand the flow of program; then we will discuss about the program and each line of program.



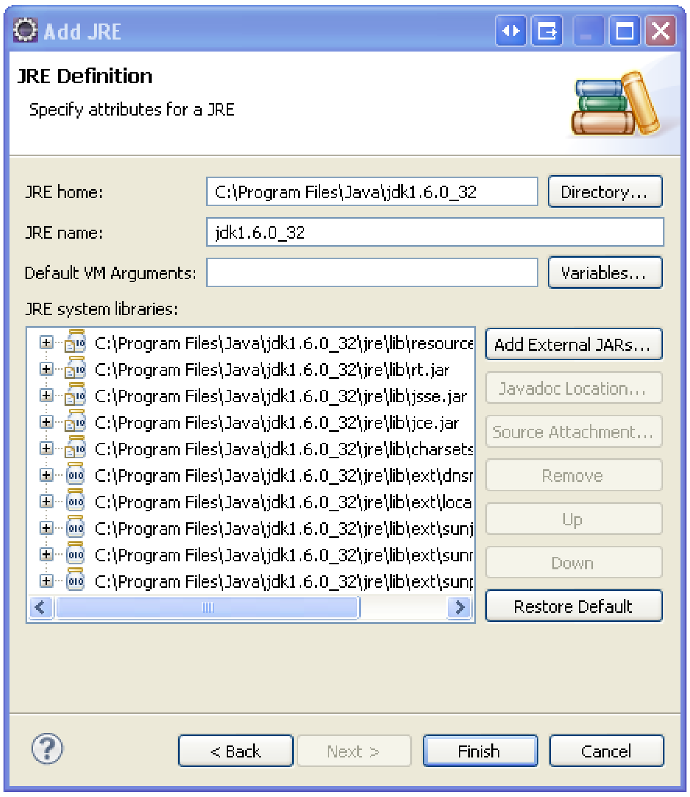
By looking above diagram, I got scared and went into a silent mode. I thought for 2 minutes, is it really a complex diagram? My answer was yes ☺. Then, I thought if this is really a complex flow then why people like java so much? Let's look again to diagram. I really found this is a very simple flow we have to just read this one by one. So, now it's your turn to read this.

Step 1:- You have written a JAVA source file.
Step 2:- You have complied your java file with java compiler. Using javac.
Step 3:- you will get a .class file with the same name of your java file.
Step 4:- class file is the form of bite code that can be loaded by class loader.
Step 5 Bytecode can be verify by Java Interpreter or JIT compiler.
Step 7:- To execute your class file JRE is required (Runtime system) and Java virtual machine.
Step 8 :- you will be able to see output in your OS.
So, now you understand how to run java program and how it's work internally.

Assignment for you
Write a JAVA program to print "Welcome to my world of JAVA".
Write a JAVA Program to print "It's My way to learn JAVA".
Write a JAVA Program to print "I have written my 1st JAVA Program ".
Write a JAVA Program to print "I am very happy that I have understood the basic flow".

SETUP AN IDE FOR YOUR PROGRAM WRITING

For programming, I will use Eclipse. Let's start to setup Eclipse IDE for your programming.
Step 1:- down load eclipse from google.
Step 2 :- once you finish the installation you can open the IDE.
Step 3:- This will ask you to select the workspace. Work space is nothing but the location in your hard drive where you can organize your all java project and java classes.
Step 4:- set the the Installed JREs


Add JRE

JRE Definition

Specify attributes for a JRE

JRE home: C:\Program Files\Java\jdk1.6.0_32

JRE name: jdk1.6.0_32

Default VM Arguments:

JRE system libraries:

```
C:\Program Files\Java\jdk1.6.0_32\jre\lib\resource
C:\Program Files\Java\jdk1.6.0_32\jre\lib\rt.jar
C:\Program Files\Java\jdk1.6.0_32\jre\lib\jsse.jar
C:\Program Files\Java\jdk1.6.0_32\jre\lib\jce.jar
C:\Program Files\Java\jdk1.6.0_32\jre\lib\charsets
C:\Program Files\Java\jdk1.6.0_32\jre\lib\ext\dnsr
C:\Program Files\Java\jdk1.6.0_32\jre\lib\ext\loca
C:\Program Files\Java\jdk1.6.0_32\jre\lib\ext\sunj
C:\Program Files\Java\jdk1.6.0_32\jre\lib\ext\sunr
C:\Program Files\Java\jdk1.6.0_32\jre\lib\ext\sunp
```

Step 5:- click Finish.

Anatomy Of A Java Program

While writing our Java program we will use so many words as I have written below. So let's understand the each one and will talk about those.

- **Comments**
- **Package**
- **Reserved words**
- **Modifiers**
- **Statements**
- **Blocks**
- **Classes**
- **Methods**
- **The main method**

Comments :-In java while writing the program, some time it's happened that we want to put some comments so that in future we can read the comments as a note. Also we use comments in program so that while execution line marked as comment can be ignored by compiler.
Java support following types of comments.
Single line comments :- Use two slashes (//) in front of a line. This will work as single line comments.

Multiline comments:- Enclosed your line between /* and */ in one or multiple lines. All the line written inside the /* */ will know as commented line and ignored by compiler to take any action.

```
/*                                                   Multiline comment
        This application program prints Welcome
        to Java!
*/
package chapter1;
public class Welcome {
  public static void main(String[] args) {
  // System.out.println ("This line will get comment!");      Single Line comments
   System.out.println ("Welcome to world of JAVA!");
  }
}
```

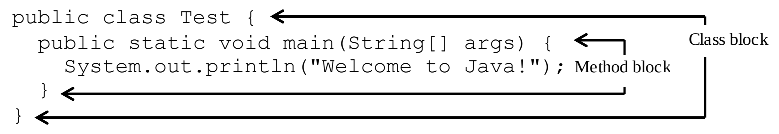
Package :- Packages in Java is a way to keep a group of classes, interfaces and sub packages in a same location in general language in a folder. It is easy to organize class files into packages so that we can group the classes. For example for a collage record I can make student package, Lecturer package, Department package. So if you ask me any information about student I can directly go to student package and get the info. We see more in details during programming.
Note:- Packages help use to organize the classes when we have a large number classes to work.

Reserved Words :- Reserved words or keywords are words that have a specific meaning to the compiler and cannot be used for other purposes in the program. For example, when the compiler sees the word class, it understands that the word after class is the name for the class.
Other reserved words in our above example are public, static, and void. Will discuss these all in next chapter during program.

Modifiers: - Java uses certain reserved words called modifiers that specify the properties of the data, methods, and classes and how they can be used.
Examples of modifiers are public and static. Other modifiers are private, final, abstract, and protected.
I will also explain these in programs by comparing with each other ☺

Statements:- A statement represents an action or a sequence of actions. The statement System.out.println("Welcome to my world of JAVA!") in the program in our example is a statement to display the greeting " Welcome to my world of JAVA!" Every statement in Java ends with a semicolon (;).

Blocks :- A pair of braces in a program forms a block that groups components of a program



```
public class Test {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Classes :- Concept of classes in java are same as a concept of OOP’s. A class can be defined as a template/blueprint that describes the behavior/state that the object of its type support.
Object is an instance of a class.

Methods :- A collection of statements that performs a sequence of operations.
Question:- What is main method in java ?

Some other Example Of Java Program

```
package JavaSample;
public class SampleProgram {
 public static void main(String[] args) {
        int num;  // declare he variable
        num = 100;  // assign the num value to 100
  System.out.println(“num is : “+ num);
        num = num * 2;
        System.out.print(“num * 2 is : “);
        System.out.println( num);
  }
}
```

Try to run above program. Wewill discuss all program line by line.
Let’s discuss the above program with line by line execution and meaning of each keyword.

```
package JavaSample;
public class SampleProgram {
 public static void main(String[] args) {
        int num;  // declare he variable
        num = 100;  // assign the num value to 100
  System.out.println(“num is : “+ num);
        num = num * 2;
        System.out.print(“num * 2 is : “);
        System.out.println( num);
  }
}
```

Package declaration as we discussed

We are using the public keyword with class SampleProgram. This means this class can be access from anywhere.

public : it is a access specifier that means it will be accessed by publically.
static : it is access modifier that means when the java program is load then it will create the space in memory automatically.
void : it is a return type i.e it does not return any value.
main() : it is a method or a function name.
string args[] : its a command line argument it is a collection of variables in the string format.

Access Modifier In Java

There are two types of modifiers in java access modifiers and non-access modifiers.
The access modifiers in java specifies accessibility (scope) of a data member, method, constructor or class.
There are 4 types of java access modifiers:

1. private
2. default
3. protected
4. public

There are many non-access modifiers such as static, abstract, synchronized, native, volatile, transient etc. Here, we will learn access modifiers.
Private, default, protected, public

Access Levels				
Modifier	**Class**	**Package**	**Subclass**	**World**
`Public`	Y	Y	Y	Y
`protected`	Y	Y	Y	N
no modifier	Y	Y	N	N
`Private`	Y	N	N	N

Primitive And Non- Primitive Data Types In Java

Primitive data type are predefined data type in java.There are 8: boolean, byte, char, short, int, long, float and double known as Primitive datatype. Primitive types are the most basic data types available within the Java language.

Type	Size in bits	Values
Boolean	8	true or false
Char	16	\u0000 - \uFFFF
Byte	8	-255
Short	16	-32,768 – 32,767
Int	32	-2,147,483,648 - +2,147,483,647
Long	64	-9,223,372,036,854,775,808 - +9,223,372,036,854,775,807
Float	32	0
Double	64	- 1.79769313486231570E+308 to - +1.79769313486231570E+308

UNDERSTANDING OF VARIABLE AND THEIR TYPE

Variables are used to represent the data that a program deals with. We have to define a variable before using it.

Declaration

type identifier = value;

Example:

Int is a data type and a,b,c are variable

```
 int a, b,c;
String name = "abc";
```

Types of Variable :- There are three types of variable

- **Local variables**
  - Local variables are declared in methods, constructors, or blocks.
  - It will be destroyed once it exits the method, constructor or block.
  - Access modifiers cannot be used for local variables.
  - Local variables are visible only within the declared method, constructor or block.
  - Local variables are implemented at stack level internally.
  - There is no default value for local so it has to be initialize

```
public class Test{
        public void pupAge(){
        int age = 0;
        age = age + 7;
        System.out.println("Puppy age is :
" + age)
        }
         public static void main(String args[]){
                 Test test = new Test();
                Test.pupAge();
        }
 }
```

Int age is defined inside the method pupAge() and we can't use this variable outside the method.

- **Instance variables**
  - Instance variables are declared in a class, but outside a method, constructor or any block.
  - Instance variable stored on heap
  - Instance variables are created when an object is created with the use of the key word 'new' and destroyed when the object is destroyed.
  - Instance variables hold values that must be referenced by more than one method, constructor or block, or essential parts of an object.s state that must be present through out the class.
  - Access modifiers can be given for instance variables.
  - The instance variables are visible for all methods, constructors and block in the class.

- Instance variables have default values.
- Instance variables can be accessed directly by calling the variable name inside the class

```
class Employee{
    public String name;  // this instance variable is visible for any child
class.
    private double salary; // salary variable is visible in Employee class
only.
    // The name variable is assigned in the constructor.
    public Employee (String empName){
    name = empName;
    }
    // The salary variable is assigned a value.
    public void setSalary(double empSal){
    salary = empSal;
    }
     // This method prints the employee details.
    public void printEmp(){
     System.out.println("name : " + name );
    System.out.println("salary :" + salary);
    }

    public static void main(String args[]){
    Employee empOne = new Employee("Ransika");
    empOne.setSalary(1000);
    empOne.printEmp();
    }
}
```

- **Class/static variables**
  - Class variables also known as static variables are declared with the static keyword, but outside a method, constructor or a block.
  - There would only be one copy of each class variable per class, regardless of how many objects are created from it.
  - Visibility is similar to instance variables. However, most static variables are declared public since they must be available for users of the class.
  - Default values are same as instance variables.
  - Static variables can be accessed by calling with the class name . ClassName.VariableName.

```
import java.io.*;
class Employee{
         // salary variable is a private static variable
        private static double salary;
        // DEPARTMENT is a constant
        public static final String DEPARTMENT = "Development";
        public static void main(String args[]){
        salary = 1000;
        System.out.println(DEPARTMENT+"average salary:"+salary);
        }
    }
```

CONVERSION & CASTING OF DATA TYPE

Type casting of datatype means changing an entity of one data type into another. For now understand the different type of casting. Will do some real casting in program and discuss.

- **Implicit Casting**

  A data type of lower size (occupying less memory) is assigned to a data type of higher size
  This is done implicitly by the JVM.
  Examples:
  int x = 10;   // occupies 4 bytes
  double y = x;   // occupies 8 bytes System.out.println(y);   // prints 10.0

  🤔 Hey guy's what happened? You didn't understand the logic? It's simple you can see above.
  int x occupy the 4 byte in memory and double y occupy the 8 byte in memory. So here x is occupying less memory than y so we can assign x in to y ☺

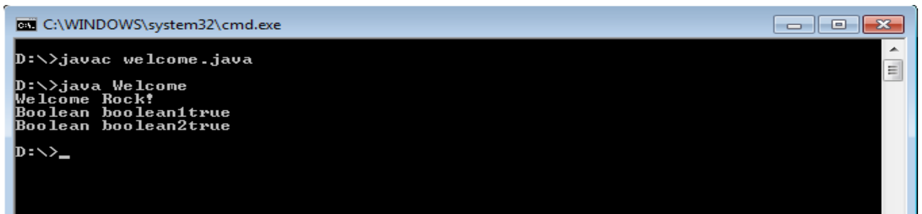
- **Explicit Casting**

  A data type of higher size (occupying less memory) is assigned to a data type of lower size
  This is not done implicitly by the JVM so requires Explicit Casting
  Examples:
  double x = 10.5;   // occupies 8 bytes
  int y = (int)x;   // occupies 4 bytes

- Boolean Casting

  As we know we are using Boolean for true/false condition. Sometimes, in programming this happen that we are getting true or false as a string value; at that time we require Boolean Casting. Let's see the example.

```
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome Rock!");
Boolean boolean1 = Boolean.valueOf("true");
boolean boolean2 = Boolean.parseBoolean("true");
System.out.println("Boolean boolean1"+boolean1);
System.out.println("Boolean boolean2"+boolean2);
 }
```



```
D:\>javac welcome.java

D:\>java Welcome
Welcome Rock!
Boolean boolean1true
Boolean boolean2true

D:\>_
```

```
        }
```

Assignment:- Try the same program without casting.

WHAT IS ARRAY AND USES OF ARRAYS

Array is a collection of elements of similar data type. An Array is created of fixed length this means once you define the array size will get fix and we cannot keep more element se the bellow example.
dataType[] Variable = new dataType[size];

```
public static void main(String[] args) {
      double[] myList = {1.9, 2.9, 3.4, 3.5};

      // Print all the array elements
      for (int i = 0; i < myList.length; i++) {
         System.out.println(myList[i] + " ");
      }

      // Summing all elements
      double total = 0;
      for (int i = 0; i < myList.length; i++) {
         total += myList[i];
      }
      System.out.println("Total is " + total);

      // Finding the largest element
      double max = myList[0];
      for (int i = 1; i < myList.length; i++) {
         if (myList[i] > max) max = myList[i];
      }
      System.out.println("Max is " + max);
   }
```

SOME EXAMPLE OF ARRAYS AND ASSIGNMENT

Assignment 1:- I have an array with 10 elements as student's name. Write a Java program to display element of Array.

Assignment 2: -Write the method available in Array. (note as you are using the eclipse you can take the help from eclipse. Once you declared the array you can use the"." With array object this will show you all available method).
Assignment 3:- Write a java program display the length of an array.

DIFFERENT OPERATORS AND THEIR USES

- ARITHMETIC OPERATIONS

Operation	Operator	Java Expression
Addition	+	a + 8
Subtraction	-	b – 7
Multiplication	*	p * 10
Division	/	c / 9
Modulus	%	b % 6

```
public class Numbers{
  public static void main(String[] args) {
        int a = 10;
        int b = 5;

        System.out.println("Add : "+  a + b);

                System.out.println("Sub : "+  a - b);

                System.out.println("Multiply :"+ a * b);

                System.out.println("Divide :"+  a / b);
        }

}
```

- DECISION MAKING OPERATIONS

Operation	Operator	Java Expression
Equal to	==	if (x == 1) …
Not equal to	!=	if (y != 5) …
Greater than	>	while (x > y) …
Less than	<	while (x < y) …
Greater than or equal	>=	if (X >= y) …
Less than or equal	<=	if (y <= z) …