

Introduction to Java

Dr. H. B. Prajapati

Associate Professor
Department of Information Technology
Dharmsinh Desai University

29 June '20

Core Java Technology

Table of contents

- 1 Characteristics of Java
- 2 Java Programming Tools
- 3 Developing Java Programs
- 4 Anatomy of Java Application Program

History of Java

- Java was developed by a team of **James Gosling** at Sun Microsystems (famous for Workstations).
- Java (originally called **Oak**) was designed for use in applications in embedded consumer electronics in **1991**.
- Later, Java was redesigned for Internet applications and was renamed as Java in **1995**.
- Java was designed as Object Oriented (**OO**) from the **beginning**.

Outline of Presentation

- 1 Characteristics of Java
- 2 Java Programming Tools
- 3 Developing Java Programs
- 4 Anatomy of Java Application Program

Java is Simple

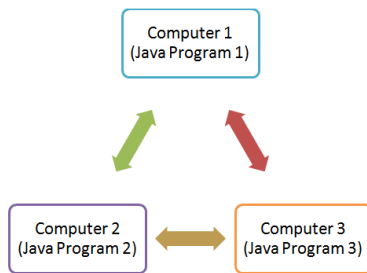
- Java is **simple** as compared to C++ (popular object programming language at that time)
- Java uses syntax and concepts of C++, but Java is **simplified** and **improved**.
- Java does not use **pointers** and **multiple inheritance**, which are complicated features of a language.
- C++ requires that the programmer allocates and deallocates memory. Java uses **automatic memory allocation** and **deallocation** (done by garbage collector—**GC**).

Java is Object Oriented

- The **real world** is object oriented. E.g., a student (object) reads a book (object); a person (object) drives a car (object).
- A Java program is object oriented:
 - Creates objects
 - Manipulates objects
 - Connects objects
- An object has **properties** and **behaviors**:
 - Properties are described by data. E.g., a car has some **color**
 - Behaviors are defined by methods. E.g., we can **drive** a car
- Objects are defined by a **class**. E.g., a car object is created from its model.
- Creating an object from a class is called **instantiating** an object
- Java provides many **pre-defined classes** and also allows us to write **our own classes**.
- Classes can be arranged in a hierarchy (**inheritance**)

Java is Distributed

- A Java application can have its classes distributed on **multiple computers** in a **network**.
- Networking capability is inherently integrated into Java



Java is Interpreted

- In a language supporting generating machine code, a program needs to be converted to machine code for each target machine.

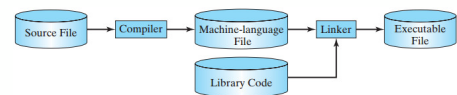


Figure : Steps to create machine dependent code, source ¹

- In a language supporting interpretation, a program is not required to be compiled for each target machine.



Figure : Steps to execute a program in Java

Java is Robust

- Robust = **Reliable**
- Java Compiler detects many problems (not just syntax errors) (e.g., using a variable without initialization)
- Java does not support pointers (overwriting memory or memory corruption is not possible)
- Java does many **early checking** before a program is run
- Java language has **exception handling** capability.

Java is Secure

- As Java is distributed, it allows to use code available at remote machine.
- However, Java provides **security for bytecode** accessed from a remote machine. E.g., applet security

Java is Architecture Neutral

- Architecture Neutral means Platform independent.
- Java's philosophy *Write Once Run Everywhere*
 - Java's bytecode is platform independent
 - Major OS vendors provide JVM to run bytecode
 - Web browser also supports Java to run applet

Java is Portable

- Program can run on any machine **without recompilation**.
- **Data size does not change** on different machines (e.g., int remains 4 bytes whether on Windows or Linux).
- Java **environment** is **portable** for new hardware and operating system.

Java is High Performance

- Generally, interpreted code is slow.
- Java uses many concepts to quickly generate machine code for bytecode at run time, thus making programs of high performance

Java is Multithreaded

- Multi-threading allows to run multiple **tasks simultaneously**. E.g., rendering GUI, networking task, data input, etc.
- Multi-threading is inbuilt in Java language

Java is Dynamic

- Java allows to **load classes** at run time.
- Java allows to instantiate objects at runtime.

Java Language Specification

- Java is standard-based
- Java Language Specification includes
 - ▀ Technical definition of language
 - ▀ Syntax
 - ▀ Structure
 - ▀ API
- Any vendor can provide implementation of Java language

Outline of Presentation

- 1 Characteristics of Java
- 2 **Java Programming Tools**
- 3 Developing Java Programs
- 4 Anatomy of Java Application Program

Program Development Tools

- For developing programs, we need **editor**
- For compiling programs, we need **compiler**
- For executing programs, we need **interpreter**
- Java provides JDK (Java Development Toolkit) including compiler (javac), interpreter (java), and other tools and libraries. Now, available from Oracle.
- We can also use Integrated Development Environment-**IDE** (includes code editor, compiler, interpreter, debugger, etc.). E.g., Netbeans, Eclipse, IntelliJ
- As a beginner, we should not use IDE. We will use Notepad/Notepad++ and command line tools (javac and java)

Preparing Environment for Java Programs

- If JDK is not installed, we need to download it from Oracle's website and need to install it (Note **JDK is platform dependent**)
- Check whether javac and java commands are accessible from command prompt (cmd on Windows or bash on Linux)

```

C:\Windows\system32\cmd.exe

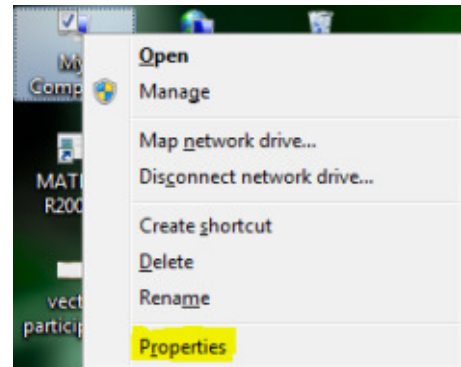
D:\programs>java -version
java version "1.8.0_45"
Java(TM) SE Runtime Environment (build 1.8.0_45-b14)
Java HotSpot(TM) Client VM (build 25.45-b02, mixed mode)

D:\programs>javac -version
javac 1.8.0_45
  
```

- If we get error "bad command", it means PATH is not set for these commands.

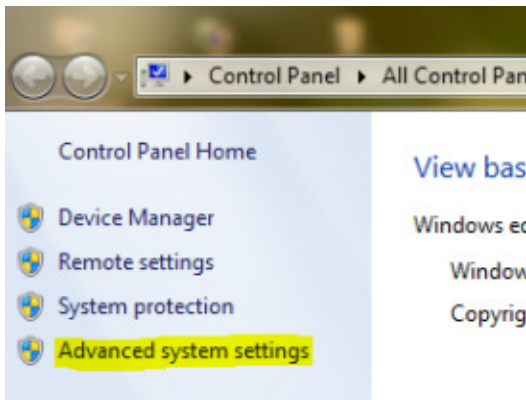
Preparing Environment for Java Programs, Step - I

On Windows machine, Right Click the "My Computer" icon, and click "Properties"



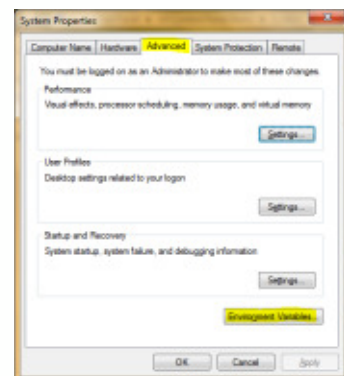
Preparing Environment for Java Programs, Step - II

Click "Advanced system settings"



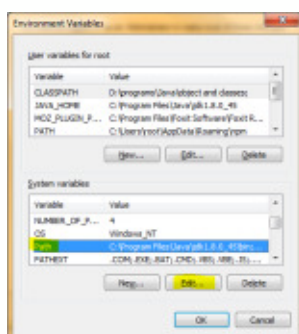
Preparing Environment for Java Programs, Step - III

In "Advanced" tab, click button "Environment Variables"



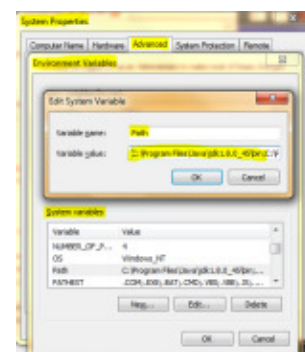
Preparing Environment for Java Programs, Step - IV

Edit "Path" System variable.



Preparing Environment for Java Programs, Step - V

Add path upto bin directory of JDK installation into PATH system variables, at the front, then close dialog boxes and start a **new command prompt**.



Outline of Presentation

- 1 Characteristics of Java
- 2 Java Programming Tools
- 3 Developing Java Programs
- 4 Anatomy of Java Application Program

Types of Java Programs

- There are two types of Java Programs
 - 1 Application (runs from command prompt)
 - 2 Applet (runs in Browser)
- Java Application can be further divided into two types:
 - 1 Console Based (CLI–Command Line Interface) allows use of **keyboard**
 - 2 Graphics Based (GUI–Graphical User Interface) allows use of **keyboard** and **mouse**

Steps of Java Program Development, Step - I

```

1 class HelloWorld {
2     public static void main(String[] args){
3         System.out.println("Welcome to Java
4             Programming");
5     }
6 }

```

Create a file HelloWorld.java using Notepad or Notepad++ editor:

Figure : Source code in Notepad Editor

Steps of Java Program Development, Step - II

Code in Notepad++ will look different

Figure : Source code in Notepad++ Editor

Steps of Java Program Development, Step - III

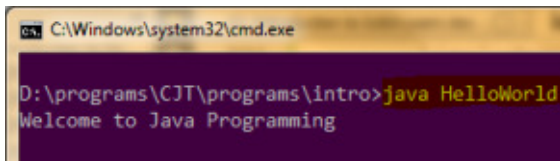
Start command prompt, make the program's directory as current working directory, and Compile the java program using the compiler (javac)

Steps of Java Program Development, Step - IV

If there is no error, .class file (bytecode) will be generated under the current working directory

Steps of Java Program Development, Step - V

Execute the .class file (bytecode) using the interpreter (java). We specify the name of java class (HelloWorld), i.e., name of .class file excluding the file extension (.class)



```
C:\Windows\system32\cmd.exe
D:\programs\CJT\programs\Intro>java HelloWorld
Welcome to Java Programming
```

Outline of Presentation

- 1 Characteristics of Java
- 2 Java Programming Tools
- 3 Developing Java Programs
- 4 Anatomy of Java Application Program

Components of Java Program

- Comments
- Reserved Words
- Modifiers
- Statements
- Blocks
- Classes
- Methods
- Variables
- The main() method

Comments

- Comments are for programmers to **document** the code or code fragments
- Comments are not processed by the compiler
- Three types of comments are available to a Java program
 - 1 Single line (E.g., `// This is a comment`)
 - 2 Multi line (E.g., `/* This is a multi-line comment */`)
 - 3 Javadoc comment (E.g., `/** This is Javadoc comment */`)

Reserved Words

- Reserved words are also called keywords
- Reserved words have special meaning to compiler and cannot be used for naming user-defined entity (class, variable, method, etc.)
- Some examples of reserved words are
 - class
 - new
 - private
 - public
 - protected
 - static
 - int, float, double, char, boolean

Modifiers

- Modifiers are reserved words that give properties to class, method, or data
- Examples of modifiers:
 - private, public, protected
 - final, abstract
 - static

Statements

- A statement represents an action or a sequence of actions.
- A statement is terminated by a semicolon sign (;)
- Examples of statement:
 - 1 `x = 5;`
 - 2 `z = x + y;`

Blocks

- Block structure groups statements together
- Blocks are used to identify components of the program
- Example:

```

1  {
2      int x;
3      x = 5;
4      x = x + 5;
5  }
```

Classes

- A class is an essential Java construct
- A class is a **template** or a **blueprint** for objects.
- Every Java program has at least one class.
- We cannot place `main()` method outside a class.
- A class can contain **data** declarations and **method** declarations or definitions.
- Each class in Java is compiled into a separate `.class` (bytecode) file.

Methods

- A method can contain a collection of statements that are executed in the sequence
- A method can do three essential things:
 - 1 take some input (through parameters),
 - 2 processes that input (method body),
 - 3 generates and returns output (using return statement),
- Earlier we used `System.out.println()`, in which `println()` is a method which takes one string as argument and returns nothing.

The `main()` method

- The `main()` method is a special user-defined method.
- It is an entry point of the Java program.
- All java classes are not required to have `main()` method
- But only an **executable class** needs the `main()` method.

Variables

- Variables are data holders
- Java is a **statically typed** language similar to C and C++, but unlike python (i.e., we need to specify data-type when we create a variable)
- Variables can be of two types:
 - Primitive
 - Object
- Example:

```

1  int age;
```

Summary of key terms

- History, James Gosling, 1991, 1995
- Object oriented, Distributed, Interpreted, Robust, Secure, Architecture Neutral, Portable, High Performance, Multi-threaded, dynamic, Language Specification
- Editor, Compiler, Interpreter, IDE, javac, java, Path, Java program development
- Comments, reserved words, modifiers, statements, blocks, classes, methods, The main method, variables

References

- An Introduction to Java Programming, Y. Daniel Liang, PHI
- An Introduction to Java Programming, Y. Daniel Liang, Eighth Edition, Prentice Hall