

INTRODUCTION TO JAVA

Session Objectives

- ❑ History Of Java
- ❑ Java Vs C++
- ❑ Characteristics of Java
- ❑ Java Environment
- ❑ Program Structure
- ❑ Comment Styles
- ❑ Simple Program
- ❑ Command Line Arguments
- ❑ Types of Java Program

History of Java

- ❑ 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. It was named as 'OAK'
- ❑ When The World-Wide Web became popular in 1995 , Java came to life again. Renamed as Java
- ❑ 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

Java vs C++

- ❑ Java is pure object-oriented language because every statement in Java is written inside a class. In C++, the main() method is always written outside any class.
- ❑ In Java, all the data types except the primitive types are objects. Even the primitive data types can be encapsulated inside classes.

Java vs C++

- ❑ Java has two additional primitive data types byte and boolean.
- ❑ Boolean can store only true or false.
- ❑ Java omits pointers and structures that are available in C++.
- ❑ In Java, the char data type stores character in Unicode format unlike the 8-bit ASCII format in C++. Unicode is a 16-bit character format that can store Asian language alphabets.

Java vs C++

- ❑ Java does not support multiple inheritance but we can achieve by using interface.
- ❑ Java allows method overloading but does not support operator overloading.
- ❑ In C++, an array is a collection of elements where as in Java, arrays are real objects because you can allocate memory to an array using the new operator.
- ❑ In C++, the strings are end with null character but in java, Strings are real objects.

So, what is Java?

According to Sun's definition:

Java is a "simple, object-oriented, interpreted, robust, secure, architecture-neutral, portable, high-performance, multithreaded, and dynamic language."

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

Characteristics of Java

☒ Java Is Simple

Java is partially modeled on C++, but greatly simplified and improved. Some people refer to Java as "C++--" because it is like C++ but with more functionality and fewer negative aspects.

- ☐ 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

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

ChandraSekhar(CS) Baratham

Java is inherently object-oriented.

Although many object-oriented languages began strictly as procedural languages, Java was designed from the start to be object-oriented. Object-oriented programming (OOP) is a popular programming approach that is replacing traditional procedural programming techniques.

One of the central issues in software development is how to reuse code. Object-oriented programming provides great flexibility, modularity, clarity, and reusability through encapsulation, inheritance, and polymorphism.

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

Distributed computing involves several computers working together on a network. Java is designed to make distributed computing easy. Since networking capability is inherently integrated into Java, writing network programs is like sending and receiving data to and from a file.

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

You need an interpreter to run Java programs. The programs are compiled into the Java Virtual Machine code called bytecode. The bytecode is machine-independent and can run on any machine that has a Java interpreter, which is part of the Java Virtual Machine (JVM).

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

Java compilers can detect many problems that would first show up at execution time in other languages.

Java has eliminated certain types of error-prone programming constructs found in other languages.

Java has a runtime exception-handling feature to provide programming support for robustness.

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

Java implements several security mechanisms to protect your system against harm caused by stray programs.

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

Write once, run anywhere

With a Java Virtual Machine (JVM), you can write one program that will run on any platform.

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

Because Java is architecture neutral, Java programs are portable. They can be run on any platform without being recompiled.

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

Java's performance Because Java is architecture neutral, Java programs are portable. They can be run on any platform without being recompiled.

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

Multithread programming is smoothly integrated in Java, whereas in other languages you have to call procedures specific to the operating system to enable multithreading.

Characteristics of Java

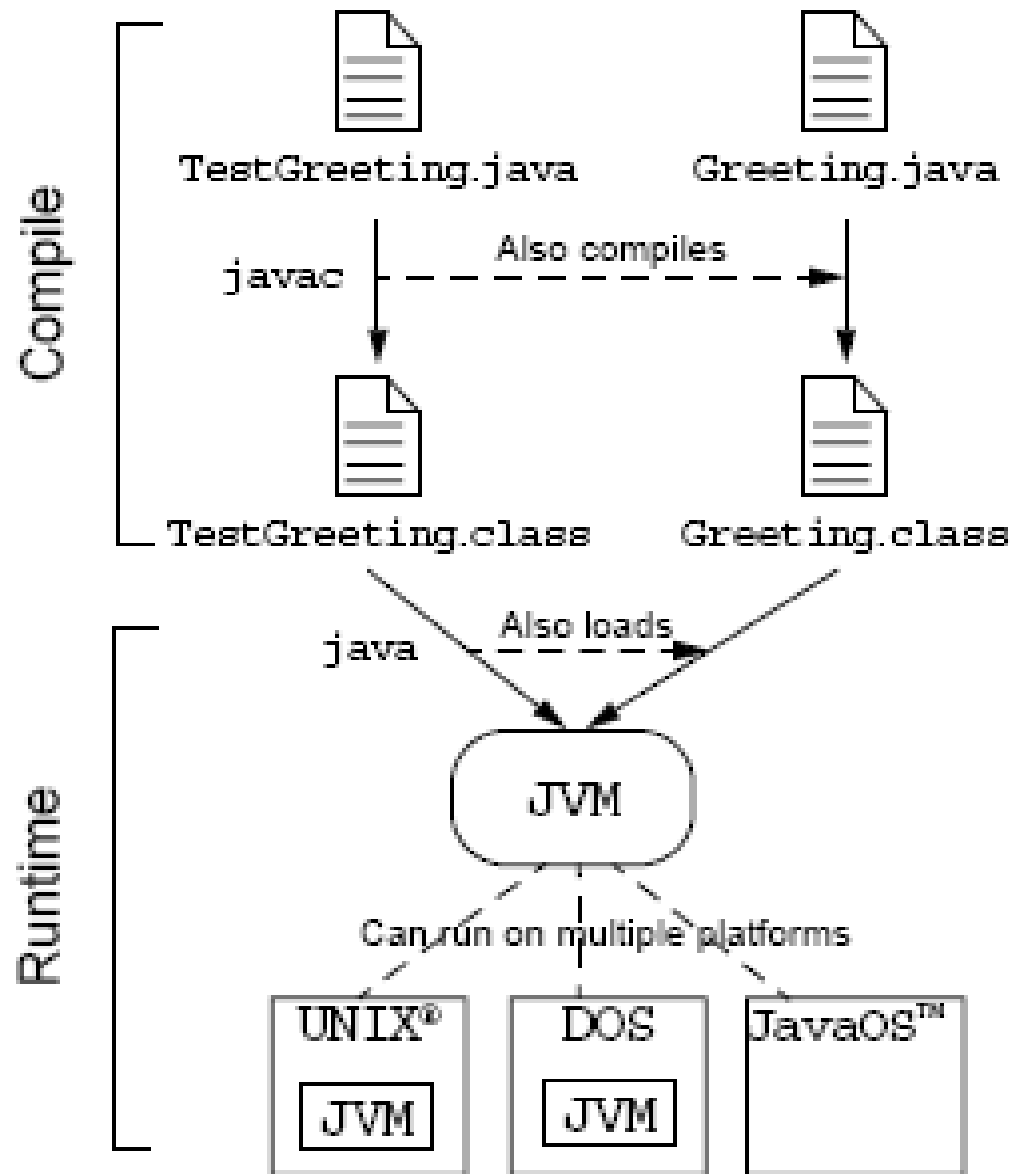
- ❑ Java Is Simple
- ❑ Java Is Object-Oriented
- ❑ Java Is Distributed
- ❑ Java Is Interpreted
- ❑ Java Is Robust
- ❑ Java Is Secure
- ❑ Java Is Architecture-Neutral
- ❑ Java Is Portable
- ❑ Java's Performance
- ❑ Java Is Multithreaded
- ❑ **Java Is Dynamic**

Java was designed to adapt to an evolving environment. New code can be loaded on the fly without recompilation. There is no need for developers to create, and for users to install, major new software versions. New features can be incorporated transparently as needed.

Java Environment



Java Technology Runtime Environment



Compiling/Running Java Programs

- ❑ To compile your program, use command:

c:\> javac MyProgramName.java

- ❑ To run your program, use command:

c:\> java MyProgramName

Program Structure

```
1 //the skeleton of a java application
2 package packagename;
3 import packagename.ClassName;
4 public class ProgramName
5 {
6     // Define program variables here.
7     // Define program methods here.
8     //Define the main method here.
9     public static main(String args[])
10    {
11        // Main method body
12    } //end of the main method.
13 } //End of class HelloWorld
```

Package & import
Statements are Optional

Main Class
& Main
Method
Mandatory

Notes

- Java is a case sensitive language
- Braces must occur on matching pairs
- Coding styles should be followed.

Comment Styles

- ❑ Comments are used to clear the logic of the code and to increase the readability if it.
- ❑ Comments are ignored by the compiler.
- ❑ Java uses three types of comments:
 - 📄 Single-line comment(`//`)
 - 📄 Multiple-line comment(`/*...*/`)
 - 📄 Documentation comment(`/**...*/`).

A Simple Java Program

Keyword

Main Class

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

Main Method

Trace a Program Execution

Entry Point

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

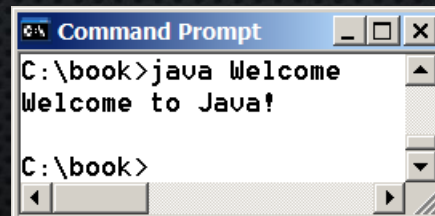

Trace a Program Execution

To Print a
Statement

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

Trace a Program Execution

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



```
Command Prompt  
C:\book>java Welcome  
Welcome to Java!  
C:\book>
```

print a message to the console

Trace a Program Execution

**Keyword & Accessing From
Out Side**

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

**If it is not a public then
Runtime : Main
method not found in
class Welcome**

Trace a Program Execution

**Keyword & static method can
run with out object**

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

**If it is not a static then
Runtime : Main
method is not static in
class Welcome**

Trace a Program Execution

Keyword & Return type

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

**Predefined Class in java.lang
Package (Default Package)**

**Command Line
Argument**

**public final class String extends Object
implements Serializable,
Comparable<String>, CharSequence**

Trace a Program Execution

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

**Predefined Class in java.lang
Package (Default Package)**

**public final
class System
extends
Object**

Command Line Argument

```
public class Welcome {  
    public static void main(String[] args) {  
        System.out.println("Input 1 : "+args[0]);  
        System.out.println("Input 2 : "+args[1]);  
    }  
}
```

```
C:\ Command Prompt  
D:\>javac Welcome.java  
D:\>java Welcome Hai Hello  
Input 1 :Hai  
Input 2 :Hello  
D:\>_
```

Input 2

Input 1

TYPES OF JAVA PROGRAM

- ❑ JAVA PROGRAMS FALL INTO TWO CATEGORIES,
 - ❑ APPLICATIONS PROGRAMS
 - ❑ APPLETS PROGRAMS

APPLICATIONS PROGRAMS

- ❑ EXECUTE FROM ANY OPERATING SYSTEM PROMPT
- ❑ THEY CAN BE EITHER WINDOW-BASED APPLICATIONS OR CONSOLE APPLICATIONS
- ❑ THEY RESIDE ON THE HARD DISK OF A LOCAL MACHINE

APPLETS PROGRAMS

- ☐ EXECUTE IN A JAVA-ENABLED BROWSER
- ☐ THEY ARE WINDOW-BASED
- ☐ THEY RESIDE ON A REMOTE MACHINE

???

WHAT WILL BE THE OUTPUT

```
PUBLIC CLASS VALAN {  
    PUBLIC STATIC VOID MAIN(STRING[] ARGS) {  
        SYSTEM.OUT.PRINTLN(10+20);  
        SYSTEM.OUT.PRINTLN("10"+20);  
        SYSTEM.OUT.PRINTLN("10"+20+30);  
    }  
}
```

30

1020

102030