UNIT II

Overview of mobile application development languages:

JAVA : Java Introduction

1. WHY JAVA?

Java is a computer language programming platform that was first released in 1995 by Sun Microsystems. Java is an Internet age-built programming language. Many applications and websites will not work unless you have Java installed, and more will be created every day. Java is fast, safe, and trustworthy. Java is anywhere and everywhere...from laptops to datacenters, game consoles to scientific supercomputers, Internet cell phones!

2. Introduction to Java Technology

Imagine that you are a developer of software applications. Your preferred programming language is either C or C++. For quite a while, you've been at this and your job doesn't seem to get any easier. You have seen multiple incompatible hardware architectures grow over the past few years, each supporting multiple incompatible operating systems, with each platform operating with one or more incompatible graphical user interfaces. Now you must handle all of this and make your applications work in a distributed client-server environment. The growth of the Internet, the World-Wide Web, and "electronic commerce" has introduced in the development process new dimensions of complexity. It does not seem that the tools you use to develop applications help you a lot. You are still facing the same old issues; the new object-oriented fashionable techniques seem to have added new problems without solving the old ones. You and your friends are saying, “There has to be a higher way"!

3. The Higher Way is Here Now

Now there's a higher way — Sun Microsystems ' Java programming language platform.

- Your programming language Java is object-oriented, but it's still very simple.
- Your development cycle is much faster due to the interpretation of Java technology. The cycle of compile-link-load-test-crash-debug is outdated — now you're just building and running.

- Your applications are portable across multiple platforms. Write your apps once, and you never have to port them — they run on multiple operating systems and hardware architectures without modification.
- Your applications are robust because your memory is managed by the Java runtime environment.
- The high performance of your interactive graphical applications is due to the multithreading built into the Java programming language and runtime platform supporting multiple concurrent threads of activity in your application.
- Because you can dynamically download code modules from anywhere on the network, your applications are adaptable to changing environments.
- Your end-users can trust your applications to be secure, even if they download code from all over the Internet; the Java runtime environment has a built-in virus and manipulation protection.

Java is one of the most popular computer languages in the world. Java is an object-oriented, interpreted, robust, secure, architecture-neutral, portable, high-performance, the multi-threaded language of the computer. It is intended to allow developers of applications to "write once, run anywhere" (WORA), meaning code running on one platform does not need to be recompiled to run on another.

Java technology is a language of programming as well as a platform.

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

In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes — the machine language of the Java Virtual Machine1 (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.

Because the Java VM is available on many different operating systems, the same .class files are capable of running on Microsoft Windows, the Solaris™ Operating System (Solaris OS), Linux, or Mac OS.

Through the Java VM, the same application is capable of running on multiple platforms.

4. The Java Platform

A platform is the hardware or software environment in which a program runs. Some of the most popular platforms are Microsoft Windows, Linux, Solaris OS, and Mac OS. Most platforms can be described as a combination of the operating system and underlying hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The Java Virtual Machine
- The Java Application Programming Interface (API)

The API and Java Virtual Machine insulate the program from the underlying hardware.

5. Where is Java used?

There are many devices where Java is currently used. Some of them are as follows:

- Desktop Applications
- Web Applications
- Mobile
- Embedded System
- Robotics
- Games etc.

6. Types of Java Applications

- **Application programs**

Application programs are stand-alone programs that are written to carry out certain tasks on local computer such as solving equations, reading and writing files etc.

- **Applet Programs**

Applets are small Java programs developed for Internet applications. An applet located in a distant computer can be downloaded via the Internet and executed on a local computer using Java-capable browser.

7. How Will Java Technology Change Your Life?

- **Get started quickly**: Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code**: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program written in C++.
- **Write better code**: The Java programming language encourages good coding practices, and automatic garbage collection helps you avoid memory leaks. Its object orientation, JavaBeans™ component architecture, and its wide-ranging, easily extendible API lets you reuse existing, tested code and introduce fewer bugs.
- **Develop programs more quickly**: The Java programming language is simpler than C++, and as such, your development time could be up to twice as fast when writing in it. Your programs will also require fewer lines of code.
- **Avoid platform dependencies**: You can keep your program portable by avoiding the use of libraries written in other languages.
- **Write once, run anywhere**: Because applications written in the Java programming language are compiled into machine-independent bytecodes, they run consistently on any Java platform.
- **Distribute software more easily**: With Java Web Start software, users will be able to launch your applications with a single click of the mouse. An automatic version check at startup ensures that users are always up to date with the latest version of your software. If an update is available, the Java Web Start software will automatically update their installation.

History of Java

Java, having been developed in 1991, is a relatively new programming language. At that time, James Gosling from Sun Microsystems and his team began designing the first version of Java aimed at programming home appliances which are controlled by a wide variety of computer processors.

Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. The language was initially called Oak after an oak tree that stood outside Gosling's office. Later the project went by the name Green and was finally renamed Java, from Java coffee. Gosling designed Java with a C/C++-style syntax that system and application programmers would find familiar.

In 1994, Gosling realized that such a language would be ideal for use with web browsers and Java's connection to the internet began. Sun Microsystems released the first public implementation as Java 1.0 in 1995. It promised "Write Once, Run Anywhere" (WORA), providing no-cost run-times on popular platforms. Fairly secure and featuring configurable security, it allowed network- and file-access restrictions. Major web browsers soon incorporated the ability to run Java applets within web pages, and Java quickly became popular.

Features of Java

- **Object-Oriented**

Java is purely an object-oriented programming language because without class and object it is impossible to write any Java program. Java is not a pure object-oriented programming language. because it supports non-primitive data types like int, float, boolean, double, long etc.

- **Platform Independent**

Platform independent language means once compiled you can execute the program on any platform (OS). Java is considered to be platform independent, because the Java compiler converts the source code to bytecode, which is an intermediate language. Bytecode can be executed on any platform (OS) using JVM (Java Virtual Machine).

-

- **Simple**

Java is very easy to learn, and its syntax is simple, clean and easy to understand. According to Sun, Java language is a simple programming language because:

1. Java syntax is based on C++ (so it is easier for programmers to learn it after C++).
2. Java has removed many complicated and rarely-used features, for example, explicit pointers, operator overloading, etc.
3. There is no need to remove unreferenced objects because there is an Automatic Garbage Collection in Java.

- **Secure**

Java compiles as a bytecode which then runs inside a Virtual machine, it cannot access the computer it runs on like a natively compiled program can. The general reason why Java is considered to be more secure than, say C, is because it handles memory management for you. So in that respect, it is more secure.

- **Architecture-neutral**

Java is considered to be architecture-neutral because it can run on many different generations of Windows. it needs only be compiled once and can run on many different CPUs.

- **Portable**

Java is a portable language due to its JVM machine concept. Java source code is first compiled by the compiler and converted into bytecodes; this bytecode is portable.

- **Robust**

Java is robust because it is a highly supported language. It is portable for different Operating systems. Feature of Java is the Automatic memory management and garbage collection. Strong type checking mechanism of Java also helps in making java robust.

- **Multithreaded**

Multithreading is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU. Each part of such a program is

called a thread. So, threads are light-weight processes within a process. We can create a class that extends the java.lang.Thread class.

- **High Performance**

Java uses Just-In-Time compilers, this enables high performance.

- **Distributed**

Java is distributed because it facilitates users to create distributed applications in Java. This feature of Java makes us able to access files by calling the methods from any machine on the internet.

- **Dynamic**

Java is considered as Dynamic because of Bytecode. A source code is written in one platform; the same code can be executed in any platform. It also loads the class files at runtime. Anything that happens at runtime is considered as dynamic

Difference Between C++ and Java

Criterion	Java	C++
Relationship	The strict relationship is enforced, e.g. in PayRoll.java must be the source code for the PayRoll class.	No strict relation between filenames and names of classes. Header files and implementation files are used in C++ for specific classes.
Input mechanism	I / O input mechanism is quite complex as it reads one byte at a time (System.in), e.g. System.out.println(x) is easy to execute.	Use in and out of I / O statements, e.g. in » x;cout « y ;
Input mechanism	I / O input mechanism is quite complex as it reads one byte at a time (System.in), e.g. System.out.println(x) is easy to execute.	Use in and out of I / O statements, e.g. in » x;cout « y ;

Criterion	Java	C++
Complier and Interpreter	Java supports compiler as well as interpreter.	C++ supports compiler only.
Compatibility with other languages	No backward compatibility with any preceding language. C / C++ affects the syntax.	Source code compatible with C, except in some exceptional cases.
Concept	Write once run anywhere.	Write once compile anywhere
Support for programming type	Support the model of object-oriented programming.	Allows both the programming of procedures and the programming of objects.
Interface	Just call the Java Native Interface and Java Native Access recently.	Allows direct calls to libraries of the native system.
Memory management	Controlled by System.	Accessible to programmer
Root hierarchy	Java is a language of programming that is purely object-oriented. That's why it follows the hierarchy of a single root.	There is no root hierarchy like this in C++. C++ supports both procedural and object-oriented programming; hence a hybrid language is called.
Best features	Java supports automatic collection of garbage. As C++ does, it does not support destructors.	C++ supports Procedural programming features for object-oriented purposes.

Criterion	Java	C++
Goto Statement	Java has no declaration of goto. Keywords have been given, and const/goto is reserved even if not used.	C++ has a declaration of goto. Although, using a goto statement is not ideal.
Multiple inheritances	Java does not support multiple inheritances.	C++ provide . The keyword virtual is used to resolve problems during multiple inheritances if there is any.
Runtime error detection	It is the responsibility of the system to check the program's error.	It is the responsibility of the programmer to check the errors in C++.
Pointer	Java only offers limited pointer support.	C++ support pointers.
Structure	It has no structural support.	It supports structures.
Unions	Java is not in favour of unions.	C++ supports unions.
Object management	Java is heavily dependent on collecting automatic garbage. It does not support destroyers.	C++ supports manual object management with the help of new and delete keywords.
Libraries	Massive, high-level services classes.	Predominantly low-level functionality
Runtime error detection	Responsibility of System.	Programmer responsibility.

Criterion	Java	C++
Supports	Threads and interfaces.	Points, structure, and union.
Functions & Data	Classes contain all functions and data; the package scope is available.	Outside of any class, global and namespace scopes, functions and data may exist.
Platform	Java programs are independent of the platform. For Java Virtual Machine (JVM), Java programs are written. It will run without recompiling.	C++ programs depend on the platform. For a specific platform, they need to be compiled.

Conclusion:

- C++ and Java both are object-oriented programming (OOP) languages
- Java is a programming language developed by Sun Microsystems
- C++ was developed by Bjarne Stroustrup at AT&T Bell Laboratories in Murray Hill.
- The Java language was initially called OAK. Originally, it was developed for handling devices and set-top boxes.
- No strict relationship between class names and filenames. In C++, header files and implementation files are used for a specific class.
- The strict relationship is enforced, e.g., the source code for class PayRoll has to be in PayRoll.java.