Chapter : 1

Introduction

1.0 OBJECTIVES

Dear Students,

After studying this chapter you will able to :

- discuss the background of development of Java
- explain with the Java buzzwords summed up by the Java committee.
- discuss what is meant by Object Oriented Programming.
- state the principles of Object Oriented Programming.

1.1 INTRODUCTION

Java was conceived by James Gosling, Patrick Noughton, Christ Warth, Ed Frank and Mike Sheridan at Sun Microsystems Inc. in 1991. The language was initially called **Oak** but later renamed as **Java** in 1995.

The primary need for the development of Java was the necessity of a platform independent language which could be used to create software for embedded systems. We know that languages like C and C++ can run on a variety of CPUs, however a compiler needs to be developed for every CPU on which the program is to be run. Compilers are expensive and time consuming to create. Therefore, during the development of Java the main emphasis was on creating a platform independent language which could run on a variety of CPUs under various environments. This led to the development of Java.

At around the same time the World Wide Web emerged and Java became a language of the Internet. The Internet consists of various types of computers operating under different operating systems and CPUs. It is necessary to run the same program on all these environments and Java had the capability to run on various platforms and thus offer portable programs. The Internet ultimately led to the phenomenal success of Java. Java has an immense effect on the Internet. In a network, two broad categories of objects are transmitted between the server and the client. The first category is the passive information and the second one is dynamic, self executing programs. Such a dynamic program is generated by the server, however it is active on the client computer. As such, these network programs pose serious difficulties in terms of security and portability.

We can create two types of programs in Java, **applications** and **applets**. An **application** is a program which runs on your computer under the operating system of your computer . An **applet** is an application which is developed to be transmitted over the Internet and executed by a Java compatible web browser. An **applet** is an intelligent program, which means it can react to the user input.

We know that everytime we try to download programs, we risk the possibility of infecting our systems with virus. Also other programs may exist which may try to gather private information such as credit card numbers, bank account numbers, passwords etc. by searching our computer files. Java provides a security against both these attacks by providing a **firewall** between the network application and the user's personal computer. This ability to maintain the security of the user's computer is considered to be one of the most important aspects of Java.

As we have already seen Java also satisfies the need of generating portable executable code. Both these aspects of security and portability are addressed by Java because the output generated by Java compiler is not executable code, but **bytecode**.

Bytecode is a set of instructions which are designed to be executed by the Java run time system called the **Java Virtual Machine**. Thus the **JVM** is the interpreter for the bytecode. Only the JVM needs to be implemented for each platform on which the Java programs are to be run. Although the details of JVM differ from machine to machine all of them interpret the same bytecode. This helps in creating portable programs which can run on a variety of CPUs. The execution of a Java program is under the control of the JVM, therefore it makes the program secure and ensures that no side effects outside the system are generated. Safety is also enhanced by the features provided in the language itself. The **bytecode** also makes the Java run time system execute programs faster.

1.1 Check Your Progress.

1. Fill in the blanks.

a) __________ is a set of instructions designed to be executed by the Java run time system.

b) The ___________ is the interpreter for bytecode.

c) Java is a ____________ independant language.

d) An applet is a dynamic program which is active on the _______ computer.

1.2 JAVA BUZZWORDS

This section describes briefly the list of buzzwords as summed up by the Java team. From among the buzzwords two viz. security and portability have already been addressed in the previous section. Here we shall see the remaining ones.

Simple : Java was designed to be easy to learn and effective to use. With prior programming experience, it is not difficult to master Java. Java provides a small number of clearly defined ways to accomplish a given task. Java inherits the syntax of C and C++ and many of the object oriented features of C++. Therefore users who have already learnt these languages find it easy to master Java with very little effort.

Object Oriented : Java is not designed to be source code compatible with any other language. The object model in Java is simple and easy to extend. The simple types, like integers are kept as high performance non objects.

Robust : The ability to create robust programs was given a very high priority during the design of Java. Java is a strictly typed language and restricts the programmer in a few key areas. It checks the program code at compile time as well as at run time. Java manages memory allocation and deallocation on its own unlike in C/C++ where the programmer has to manually allocate and free all dynamic memory. In Java, deallocation is completely automatic. Java provides object oriented exception handling.

Multithreaded : Java was designed to meet the real world requirement of creating interactive networked programs. Therefore to accomplish this, Java supports multithreading. Multithreaded programming allows you to write programs that can do many things simultaneously.

Architecture Neutral : Upgradation of operating systems, processor upgrades, and changes in the core system resources can all be the factors which may cause a program which was running successfully to malfunction at a later point of time. Java and the JVM has been designed in such a way so as to attempt to overcome this situation, which would enable the programs to run inspite of severe environmental changes.

Interpreted : We know now that the output of the Java compiler is the **bytecode** which is platform independent. This code can be interpreted by any system which provides a **Java Virtual Machine**.

Distributed : Java handles TCP/IP protocols. Accessing a resource using URL is similar to accessing a file. Java has a package called **Remote Method Invocation** (**RMI**) which brings a very high level of abstraction to client/server programming.

Dynamic : Java programs have the capability to carry a sufficient amount of run time type information. This information can be used to verify and resolve accesses to the objects at run time. This makes it possible to dynamically link code in a safe manner.

1.2 Check Your Progress.

1. Write True or False.

a) Java is a strictly typed language.

b) Multithreaded programming allows to write programs that do many things simultaneously.

c) Java does not check program code at run time.

1.3 OBJECT ORIENTED PRORAMMING

Object oriented programming is at the core of Java. All Java programs are object oriented. Let us therefore study the Object Oriented Programming Principles before commencing writing programs in Java.

We have already studied that all computer programs consists of two elements : code and data. A program can be organised either around its code or around its data. When a program is organised around the code we call it as a **process oriented** model where code acts on data. The process oriented model has been successfully implemented in procedural languages like C. However, as the programs grow in size they become more complex. In the second approach which is called **object oriented** programming, a program is organised around its data i.e **objects** and a set of interfaces to that data. Thus an object oriented program is characterized as data controlling the access to the code.

An important element of object oriented programming is **abstraction**. Abstraction allows you to use a complex system without being overwhelmed by its complexity. A powerful way of managing abstraction is through the use of hierarchical classification. The data from a traditional process oriented program is transformed into its component objects through abstraction. A sequence of process steps then becomes a collection of messages between these objects. Each object defines its own unique behaviour. These objects are treated as concrete entities. Your tell them to do something by sending messages. This forms the basis of object oriented programming.

1.3.1 OOP Principles :

Object oriented programming languages provide a mechanism which helps to implement the object oriented model. These are given herewith :

Encapsulation : This is the mechanism which binds the code and the data that it manipulates. It thus keeps both the code and the data safe from outside interference and misuse. Access to the code and data is possible only through well defined interfaces. With encapsulation everyone knows how to use the code regardless of the details of the implementation. The basis of encapsulation in Java is the **class**. The class defines the structure and behaviour i.e. data and code that will be shared by a set of objects. Each object of a given class contains the structure and behaviour defined by the class. The objects are referred to as an **instance** of a class. Thus a class is a logical construct and an object is a physical reality.

The code and the data that constitute a class are collectively called as **members** of the class. The data defined by the class are referred to as **member variables** or **instance variables**. The code which operates on the data are referred to as **member methods** or **methods**. The methods define how the member variables are used.

Inheritance : This is the process by which an object acquires the properties of another object. Inheritance supports the concept of **hierarchical classification**. With the use of inheritance, an object would need to define only those qualities that make it unique within its class. It can inherit the general attributes from its parent class. eg. if you wanted to describe animals in an abstract manner you would say they have the attributes like size, intelligence etc. They also have certain behavioural pattern like they eat, breathe, sleep. Such attributes and behaviour makes up the class definition of animals. A more specific class of animals could be mammals with some specific attributes like mammary glands. This class would be a subclass of animals and the animals class is the superclass. Mammals would inherit all the properties of the animal class and have its own specific attributes also. You can have more than one level of inheritance where a deeply inherited subclass will inherit all the attributes from each of its ancestors in the class hierarchy. If a given class encapsulates some attributes then any subclass will have the same attributes i.e it will inherit all the attributes of all its ancestors, as well as add its own specific characteristics.

Polymorphism : This is a feature which allows one interface to be used for a general class of actions. Polymorphism is expressed as "one interface, multiple methods". Thus it is possible to design a generic interface to a group of related activities. Polymorphism helps in reducing complexity by allowing the same interface to be used to specify a general class of action. The compiler would select the specific method as it applies to each situation. The programmer needs to know and use only the general interface.

Thus through the application of object oriented principles, the various parts of a complex program can be brought together to form a cohesive, robust and maintainable whole. Every Java program involves encapsulation, inheritance and polymorphism. This means that the student should bear in mind that every Java program is object oriented and study how this is applied in the chapters that follow.

1.3 Check Your Progress.

1. Match the following.

	Column A	**Column B**
a)	Polymorhphism	(i) is a logical construct
b)	class	(ii) One interface, multiple methods
c)	Inheritance	(iii) is a physical reality
d)	object	(iv) supports the concept of hierarchical classification

1.4 SUMMARY

Java was conceived by James Gosling, Patrick Noughton, Christ Warth, Ed Frank and Mike Sheridan at Sun Microsystems Inc. in 1991. The language was initially called Oak but later renamed as Java in 1995.

Java provides security by a firewall between the network application and the user's personal computer. This ability to maintain the security of the user's computer is considered to be one of the most important aspects of Java.

The output generated by Java compiler is not executable code, but bytecode. Bytecode is a set of instructions which are designed to be executed by the Java run time system called the Java Virtual Machine. Thus the JVM is the interpreter for the bytecode. Although the details of JVM differ from machine to machine all of them interpret the same bytecode.

Following is the list of buzzwords as summed up by the Java team.

Simple, Object Oriented, Robust, Multithreaded, Architecture Neutral, Interpreted, Distributed, Dynamic, Object oriented, OOP Principles, Encapsulation, Inheritance, Polymorphism

Source : *blog.ibeesolutions.com*

1.5 CHECK YOUR PROGRESS - ANSWERS

1.1 1. a) Bytecode
b) Java Virtual Machine
c) platform
d) client

1.2 1. a) True
b) True
c) False

1.3 1. a) - (ii)
b) - (i)
c) - (iv)
d) - (iii)

1.6 QUESTIONS FOR SELF - STUDY

1. Describe the evolution of the Java Programming language.
2. Write a short note on the Java Buzzwords
3. What is byte code? What do you understand by JVM?
4. Explain Object oriented programming. Describe in brief the OOP principles.

1.7 SUGGESTED READINGS

1. www.java.com
2. www.freejavaguide.com
3. www.java-made-easy.com
4. www.tutorialspoint.com
5. www.roseindia.net

❑ ❑ ❑

Notes