Introduction

Page No.
Date

* Java features :-
  1. Compiled and interpreted
  2. Portable and platform independent.
  3. object oriented.
  4. Robust and secure.
  5. Distributed.
  6. Simple, small and familiar.
  7. High performance.
  8. Multithreaded and Interactive.
  9. Dynamic & extensible.

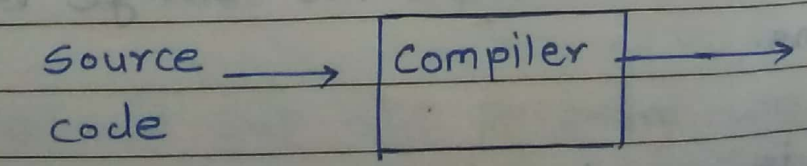
1. Compiled and Interpreted

- Basically all programing languages are either compiled or interpreted.

- Java supports feature of both (Compiled and interpreted)

- In Java source code is converted into byte code by Compiler. so, it is compiled language.

- Byte code is converted into machine code by interpreted & directly executed.



Scanned by CamScanner

byte code → [Interpreter] → machine code.

Portable and platform Independent

- Java supports the feature portability due to portability java programs move from one PC to another is quite easy.

- Java provides this type of portability in following two ways.

- First Java Compiler source code is converted into byte code.

- Created byte code can easily move from one machine to another machine and that can be executed on machine by using interpreter of that machine.

- So it is portable language.

Platform Independent

- The primitive data type does not depend upon the memory addressing of operating system (OS)

- Upgradation of OS processors and system resources never force the change in Java programs.

- Java is popular language for programming on internet where interconnection of different

Scanned by CamScanner

kinds of system is required.

3 object oriented

- Java is truly object-oriented language. Almost everything in Java is an object.

- Without creating object class cannot be representing.

- It is not purely object oriented language because it does not support goto statement, multiple inheritance, pointer, function overriding.

4 Multithreaded and interactive.

- Multithreaded :- It means handling multiple tasks simultaneously by supporting multithreaded programs.

- This means that there is no need to wait for the application to complete one task before beginning another.

- Therefore improving the interactive performance in graphics application.

5 Dynamic and Extensible

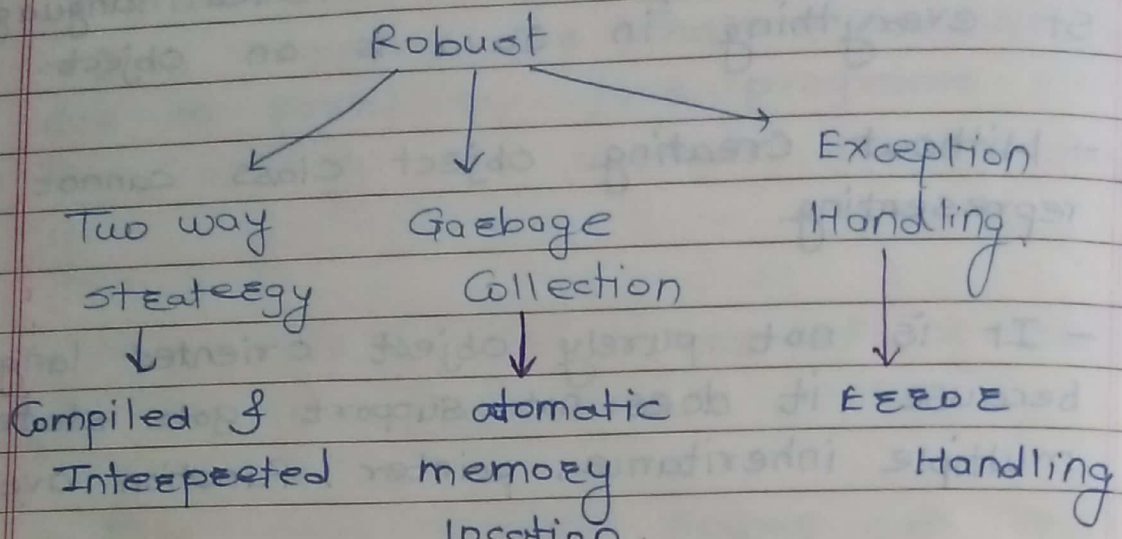
- Java is also a dynamic language.

- When we run the program we can add functions of other languages (c, c++) are supported by the java program.

Scanned by CamScanner

These methods are known as Native Method

6. Robust and Secuee :-



- Java is moee eobust language than othees as it has two times execution st-eateegy that is compilation & inteepeeted Theefoee, peovides eealiability.

- Gaebage Collection is one of the featuee of java which eeduce peogeammees task of memoey management.

- Java peovides exception Handling which caictes the seevee eeeoe & eeduce eisk of ceushing the system.

Secuee :→

Java Peogeamming system vaeify all memoey access and ensueed that vie- uses cannot communicate with an ~~emplete~~ applet The absent of pointee in java leads to peocess memoey access.

Scanned by CamScanner

7. familiaE, simple and small &→

• Java is veEy simple language which has many featuEes of C & C++ which aEE may be difficult & that aEE not paEt of java.

It does not use pointeE, pEE-pEocesSoE, headeE files, goto statement & many otheE

Java Jangouge also eliminates opeEatoE oVeEloading & multiple inheEitance.

8. High peEfoEmance &→

Java peEfoEmance is veEy EemaEk-able foE an inteEpEeted language mainly due to the use of inteEmediate Code Coate byte Code.

9. DisteIbuted &→

On diffeEent geogEaphical aEea we can shaEe disteIbute the java pEogEam by inteEnet.

Java is veEy populaE and used as disteIbuted Janguage foE CEeating applicat of netwoEks because of its ability to shaEe both data & pEogEam.

Java application are more robust. Hence can open & access remote places on internet as easily as on local system.

Scanned by CamScanner

- This features of java enables multiple program various remote location to work together on a single project.

Basic Concepts of object Oriented programming

① class :-

1. The objects are variable of class.
2. class is the collection of objects of similar data type.
3. classes are user defined datatypes & behave like build in types of a programming language.
4. ones class is created we can create number of objects.

2] objects

1. objects are basic run time entities in object oriented system.
2. objects may be created using class name.
3. objects are real time entities.
4. objects can communicate with each other without knowing the details of each other's data or code.
5. Syntax:

```
class_name object_name;
```

e.g:- Student S1, S2;

Scanned by CamScanner

Page No.
Date

3] Data abstraction :-

1. Data abstraction defines hiding complexity of Data from end users.
2. There are three levels of data abstraction
   ① Physical level
   ② logical level
   ③ View level

4] Data Encapsulation

1. Data encapsulation is one of the salient features of C++, which means data & function are wrapped into a single unit.
2. The data which is declared as private cannot directly accessible to the outside of the class & only the function which are wrapped in the class can access it.

5] Inheritance

1. Inheritance means deriving a new class from old class.
2. old class is known as base class and new class is also called derived class.

6] Polymorphism :-

1. Ability to the more than one form means known as polymorphism.
2. Their are two types :

Scanned by CamScanner

1. Compile time polymorphism :- When function is selected for execution at compile time is called compile time polymorphism.

There are two types -

1. Function overloading
2. Operator overloading

2. Run time polymorphism :- When function is select for execution at run time is called run time polymorphism.

- function overridding is type of run time polymorphism.

Message Passing :-

It is used for communication betn objects.

Dynamic binding :-

- It is defined as the linkings of a procedure call or function call to be executed in response to the call.

- It also known as late binding.

Scanned by CamScanner

Page No.
Date

Difference between c and Java

	c language	Java language.
1	c supports goto statement.	1. Java does not support goto statement.
2	c contains Header file.	2. Java does not contains header files.
3.	It is procedure oriented language	3. It is object oriented language.
4.	c does not support object oriented features.	4. Java support object oriented features.
5	c is less secure.	5. Java is more secure.
6	C contain pointer	6. It does not contain pointer
7	low performance	7. High performance
8.	It is not portable	8. It is portable
9.	It support either compiled & Interpreted	9. It supports both
10]	c contain struct & union datatype	10] java does not contain struct & union datatype

Scanned by CamScanner

	Java	C++
1	It is a true object oriented language.	1. It is basically C with object oriented extension.
2	It does not support operator overloading.	2. It support operator overloading.
3.	It does not have template classes.	3. It has template classes.
4.	Multiple inheritance cannot be supported.	4. Multiple inheritance can be supported.
5.	It does not support global variable	5. It support global variable.
6.	It does not use pointer.	6. ~~Their is~~ Pointers are used in C++.
7.	There are no header file in Java.	7. It include header files in program.
8	It is highly secure.	8. It is less secure.
9	It is highly performance.	9. It is low performance.

Scanned by CamScanner

JVM and Byte code

- There are 2 Types of languages
  A. platform dependant
  B. platform independant.

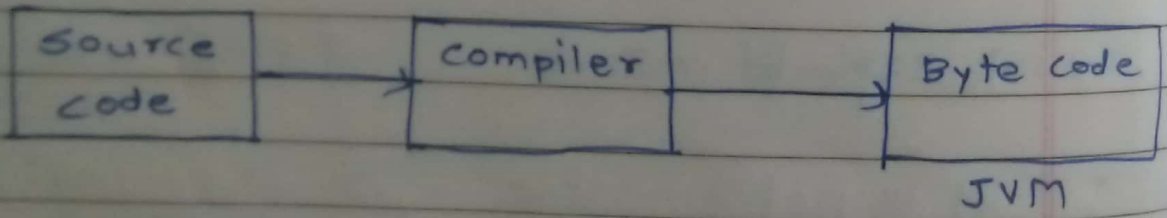
- In platform dependant (like c) the source code is directly converted into machine code by compiler which is directly executed.

- But it platform independant (like Java) the source code is converted into intermigate code by compiler.

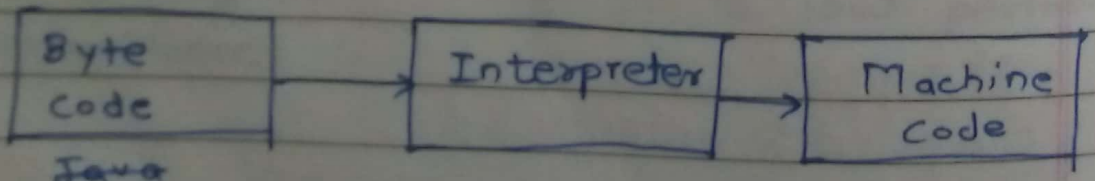
- And this intermigete is called byte code

- Byte code is created for a machine which is physically not present but virtually / logically present is memory as called as Java virtual Machine (JVM)

-

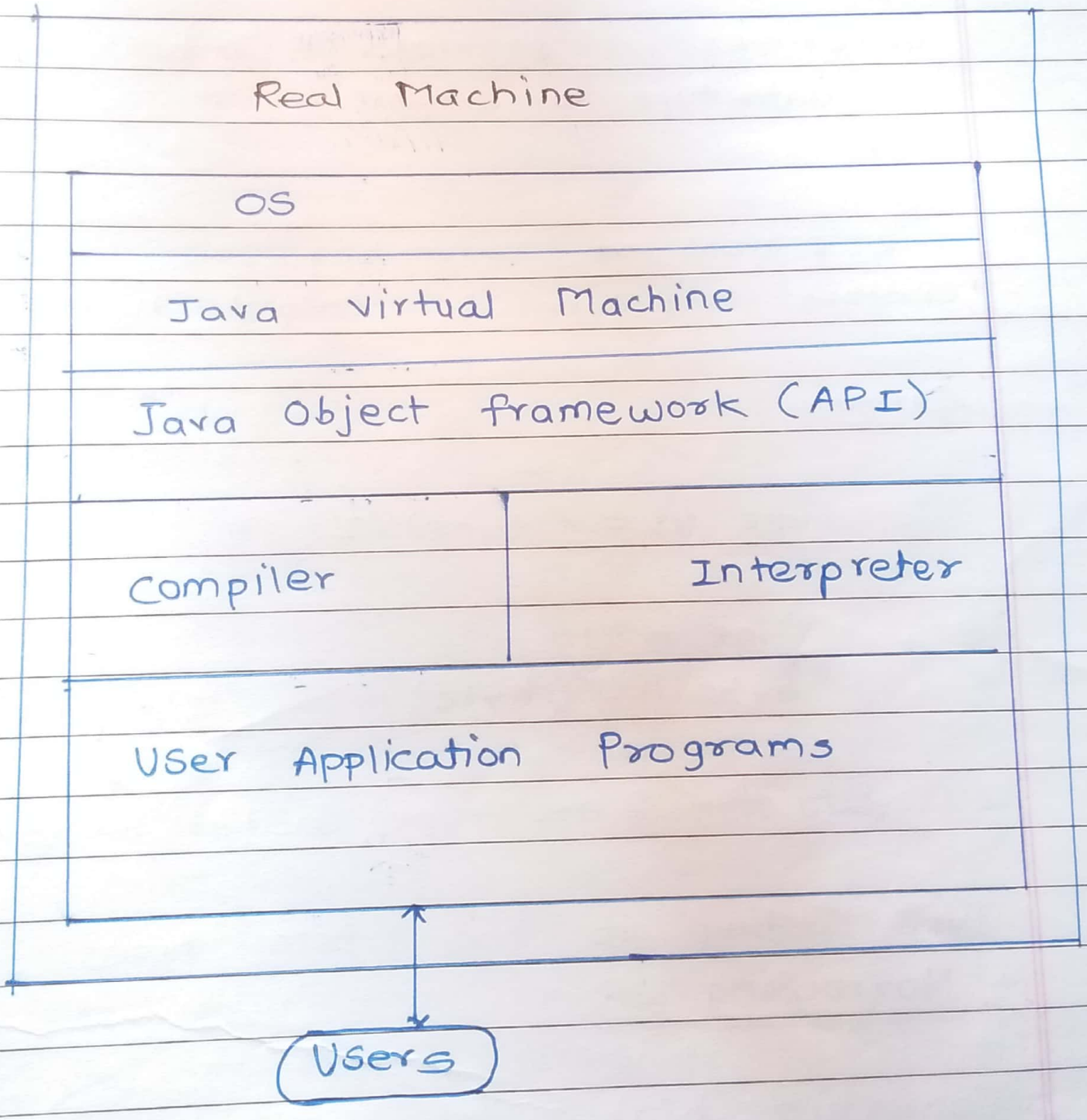


-



Scanned by CamScanner

layers of interactions for Java Programs



Scanned by CamScanner

Type Conversion and Casting

If you have previous programming experience, then you already know that it is fairly common to assign a value of one type to a variable of another type. If the two types are compatible, then Java will perform the conversion automatically. For example, it is always possible to assign an **int** value to a **long** variable. However, not all types are compatible, and thus, not all type conversions are implicitly allowed. For instance, there is no automatic conversion defined from **double** to **byte**. Fortunately, it is still possible to obtain a conversion between incompatible types. To do so, you must use a *cast*, which performs an explicit conversion between incompatible types. Let's look at both automatic type conversions and casting.

Java's Automatic Conversions

When one type of data is assigned to another type of variable, an *automatic type conversion* will take place if the following two conditions are met:

- The two types are compatible.
- The destination type is larger than the source type.

When these two conditions are met, a *widening conversion* takes place. For example, the **int** type is always large enough to hold all valid **byte** values, so no explicit cast statement is required.

For widening conversions, the numeric types, including integer and floating-point types, are compatible with each other. However, there are no automatic conversions from the numeric types to **char** or **boolean**. Also, **char** and **boolean** are not compatible with each other.

As mentioned earlier, Java also performs an automatic type conversion when storing a literal integer constant into variables of type **byte**, **short**, **long**, or **char**.

Casting Incompatible Types

Although the automatic type conversions are helpful, they will not fulfill all needs. For example, what if you want to assign an **int** value to a **byte** variable? This conversion will not be performed automatically, because a **byte** is smaller than an **int**. This kind of conversion is sometimes called a *narrowing conversion*, since you are explicitly making the value narrower so that it will fit into the target type.

To create a conversion between two incompatible types, you must use a cast. A *cast* is simply an explicit type conversion. It has this general form:

(target-type) value

Scanned by CamScanner

Here, *target-type* specifies the desired type to convert the specified value to. For example, the following fragment casts an **int** to a **byte**. If the integer's value is larger than the range of a **byte**, it will be reduced modulo (the remainder of an integer division by the) **byte**'s range.

```
int a;
byte b;
// ...
b = (byte) a;
```

A different type of conversion will occur when a floating-point value is assigned to an integer type: *truncation*. As you know, integers do not have fractional components. Thus, when a floating-point value is assigned to an integer type, the fractional component is lost. For example, if the value 1.23 is assigned to an integer, the resulting value will simply be 1. The 0.23 will have been truncated. Of course, if the size of the whole number component is too large to fit into the target integer type, then that value will be reduced modulo the target type's range.

The following program demonstrates some type conversions that require casts:

```
// Demonstrate casts.
class Conversion {
  public static void main(String args[]) {
    byte b;
    int i = 257;
    double d = 323.142;

    System.out.println("\nConversion of int to byte.");
    b = (byte) i;
    System.out.println("i and b " + i + " " + b);

    System.out.println("\nConversion of double to int.");
    i = (int) d;
    System.out.println("d and i " + d + " " + i);

    System.out.println("\nConversion of double to byte.");
    b = (byte) d;
    System.out.println("d and b " + d + " " + b);
  }
}
```

This program generates the following output:

```
Conversion of int to byte.
i and b 257 1

Conversion of double to int.
d and i 323.142 323

Conversion of double to byte.
d and b 323.142 67
```

Scanned by CamScanner

Let's look at each conversion. When the value 257 is cast into a **byte** variable, the result is the remainder of the division of 257 by 256 (the range of a **byte**), which is 1 in this case. When the **d** is converted to an **int**, its fractional component is lost. When **d** is converted to a **byte**, its fractional component is lost, *and* the value is reduced modulo 256, which in this case is 67.

Scanned by CamScanner

Date

Type casting

Sometimes we encounter a situation where one data type is converted into another data type

The process of converting one data type into another data type is called type casting.

Syntax :-

```
type v1 = (type) v2
```

e.g:-
```
int m = 10
byte n = (byte) m;
```

All data types are type casted except boolean

Type casting it has two types

1] Narrowing

Converting higher data type into lower data type called narrowing

It causes loss of data

e.g:- When float converting into integer it loss fractional part.

2] Widing.

Converting lower data type into higher data type called widing.

Scanned by CamScanner

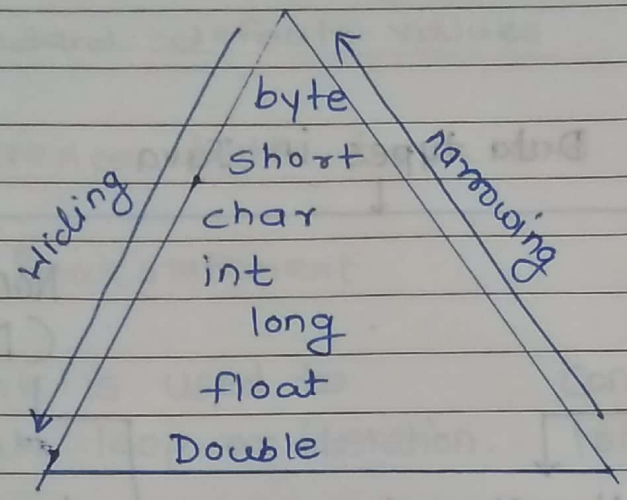


Table widing

byte → short char int long float double.

short → char int long float double.

char → int long float double

int → long float double

long → float double

float → double

Table Narrowing

Double → float long int Char short byte.

float → long int char short byte.

long → int char short byte.

int → char short byte.

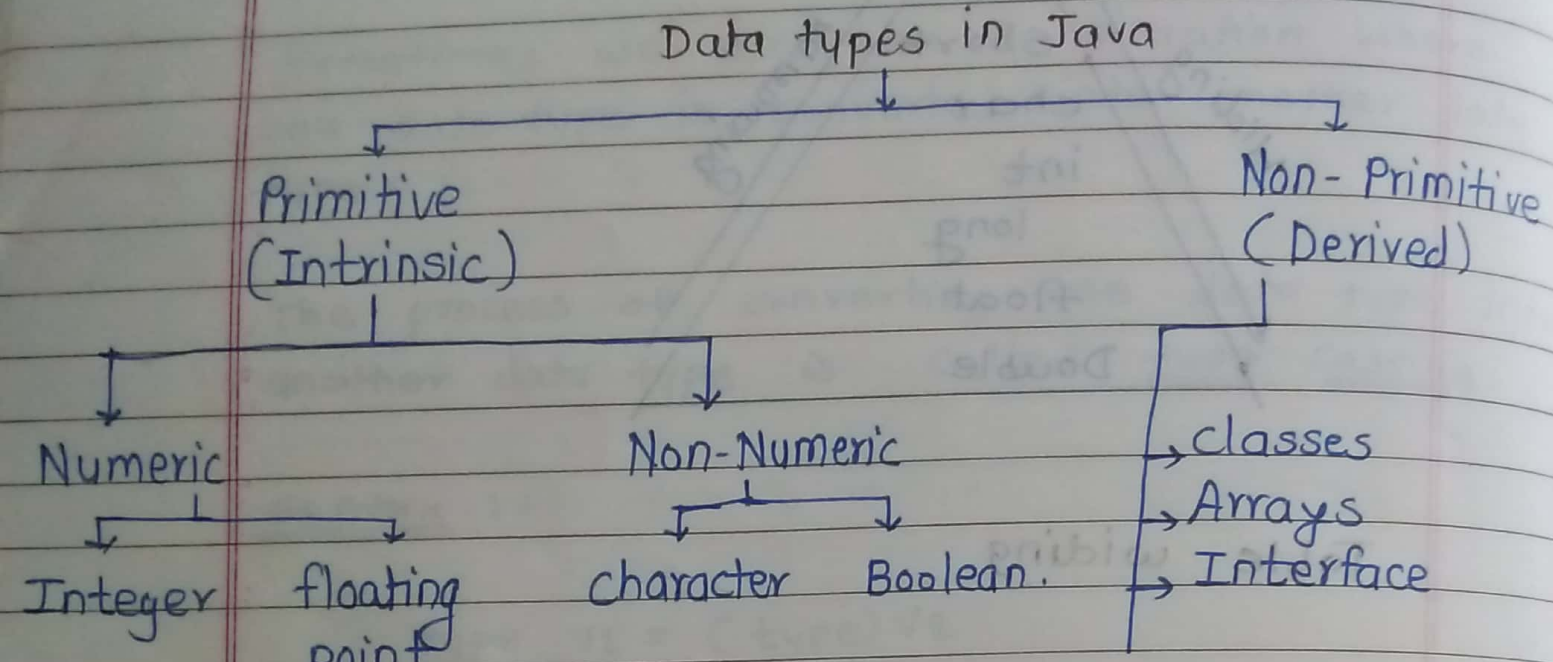
char → short byte.

short → byte.

Scanned by CamScanner

Page No.
Date

Data types in Java



Type	Size	Minimum value	Maximum value.
byte	1 byte	-128	127
Short	2 byte	-31,768	32,767
int	4 byte	-2,147,483,648	2,147,483,647.
long	8 byte	-9,223,372,036 854,775,808	9,223,372,036 854,775,807.

Scanned by CamScanner

Page No.
Date

Java and Internet

- Java is strongly associated with the Internet because of the fact that the first application program written in Java was HotJava, a Web Brower to run applets on Internet.

- Internet users can use java to create applets programs and run then locally using a "java.enabled Browser", such as HOTJAVA

- They can also use a Java-enabled Browser to download an applet located on a computer anywhere in the Internet and run it on his local computer.

- In fact, Java applets have made the Internet a true extension of the storage system of the local computer.

- Internet users can also setup their Websites containing Java applets that could be used by other remote users of Internet.

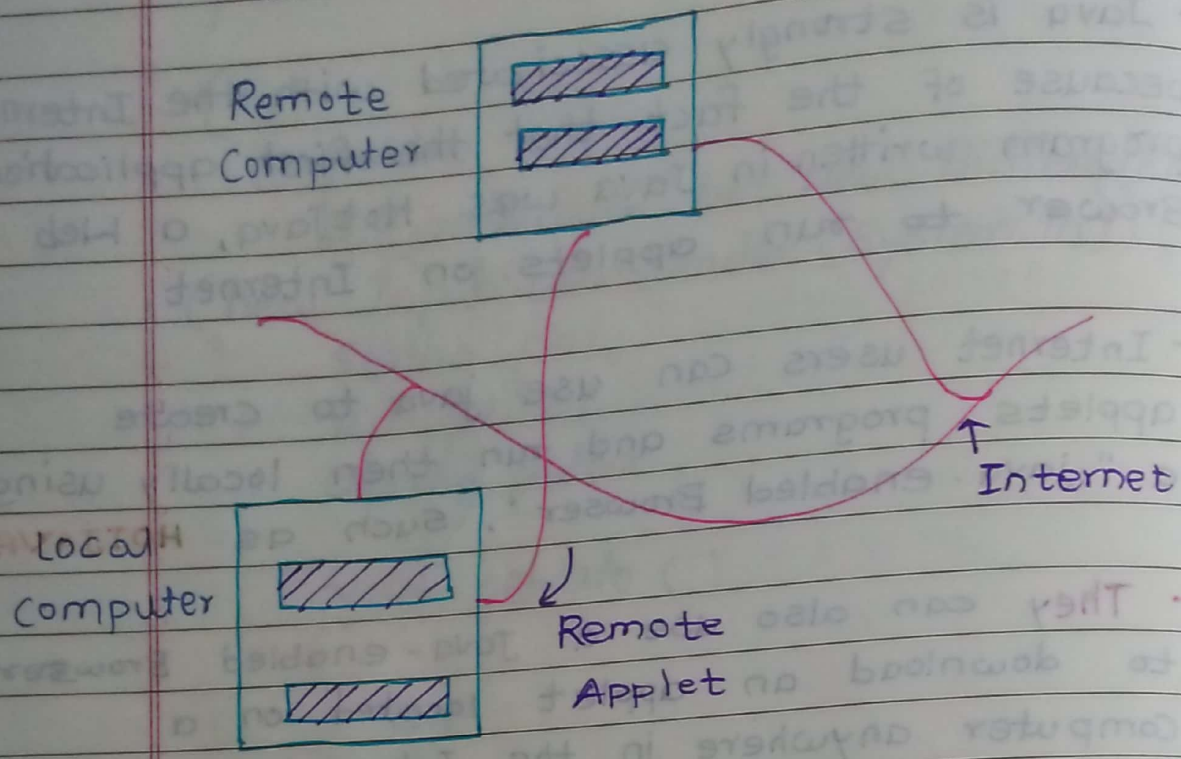
- The ability of Java applets to which a ride on the information Superhigh way has made Java a unique programming Language for the Internet.

- In fact, Java is popularly known as Internet language.

Scanned by CamScanner

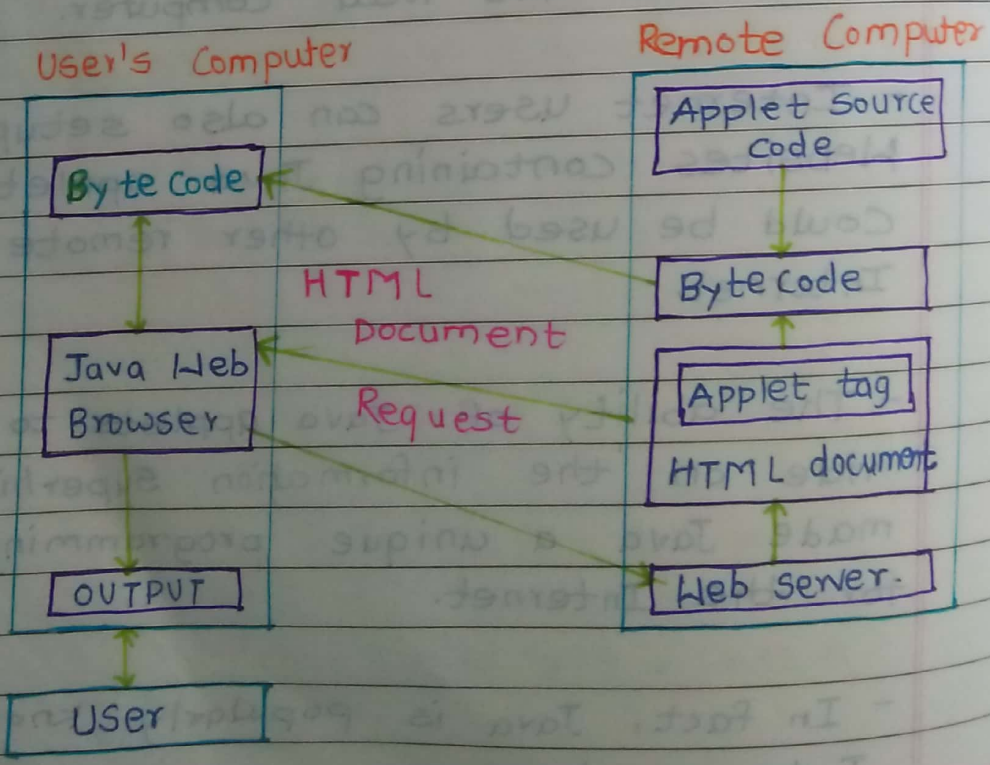
Date

* Downloading of applets via Internet.



* JAVA AND WORLD WIDE HEB :-

Java communicates with a web page through a special tag called <Applet>



Scanned by CamScanner

Page No.
Date

The fig. shows following communication steps.

1] The user sends the request for an HTML document to the remote computer's web server. The web server in program that accepts requests processes the request, & sends the required document.

2] The HTML document is returned to the user's browser. The document contains the Applet tag, which identified the applet.

3] The corresponding applet bytecode is transfered to user's Computer. This bytecode has been previously created by the Java compiler using the source code file for that applet.

4] The Java-enabled browser on the user's Computer interprets the bytecodes & provides o/p.

5] The user may have further interaction with the applet, but with no further downloading from the provider's web server. This is because the bytecode contains all the information necessary to interpret the applet.

Scanned by CamScanner