JAVA

Java is a general-purpose, class-based, object-oriented programming language designed for having lesser implementation dependencies. It is a computing platform for application development. Java is fast, secure, and reliable, therefore. It is widely used for developing Java applications in laptops, data centers, game consoles, scientific supercomputers, cell phones, etc

Definition and Meaning

Java is a multi-platform, object-oriented, and network-centric language. It is among the most used programming language. Java is also used as a computing platform.
It is considered as one of the fast, secure, and reliable programming languages preferred by most organizations to build their projects.

Java Uses

Here are some important Java applications:

- It is used for developing Android Apps
- Helps you to create Enterprise Software
- Wide range of Mobile java Applications
- Scientific Computing Applications
- Use for Big Data Analytics
- Java Programming of Hardware devices
- Used for Server-Side Technologies like Apache, JBoss, GlassFish, etc.

History of Java Programming language

- The Java language was initially called OAK.
- Originally, it was developed for handling portable devices and set-top boxes. Oak was a massive failure.
- In 1995, Sun changed the name to "Java" and modified the language to take advantage of the burgeoning www (World Wide Web) development business.
- Later, in 2009, Oracle Corporation acquired Sun Microsystems and took ownership of three key Sun software assets: Java, MySQL, and Solaris.

Java Features

- It is one of the easy-to-use programming languages to learn.
- Write code once and run it on almost any computing platform.
- Java is platform-independent. Some programs developed in one machine can be executed in another machine.
- It is designed for building object-oriented applications.
- It is a multithreaded language with automatic memory management.
- It is created for the distributed environment of the Internet.
- Facilitates distributed computing as its network-centric.

Components Of Java Programming Language

A Java Programmer writes a program in a human-readable language called Source Code. Therefore, the CPU or Chips never understand the source code written in any programming language.
These computers or chips understand only one thing, which is called machine language or code. These machine codes run at the CPU level. Therefore, it would be different machine codes for other models of CPU.
run Java programs. It is possible to install more than one JDK version on the same computer.

Java Syntax

```
class HelloWorld
{
        public static void main(String[] args)
        {
                System.out.println ("Welcome to Hello World program");
        }
}
```

z/* Java Program Example - Java Basic Syntax */

- **Object** – Objects are known to possesses states and behaviors. For instance: A human being has states - color, name, and behavior and emotions. An object is also known as the instance of the class.
- **Class** – It can be basically defined as a template/blueprint that is used to describe the behavior/state that is being supported by the objects.
- **Methods** – A behavior of anything is simply known as a Method. Many methods can be contained in a class. The methods are responsible for the manipulation of written logics and execution of all the actions.
- **Instance Variables** – Every object has its own unique set of instance variables. And the state of an object is generally created by the values that are assigned to these instance variables.

Class and Objects

A Class groups variables and operations together in coherent modules. A class can have fields, constructors and methods.Objects are instances of classes. When you create an object, that object is of a certain class. The class is like a template (or blueprint) telling how objects of that class should look..

Variables and DataTypes

Computer programs, read data from input devices, processes it and write it to screen, file or network.In a Java program data is kept in variables. Your Java program first declares the variables, then read data into them, execute operations on the variables, and then write the variables somewhere again.
Each variable has a data type. The data type determines what kind of data the variable can contain, and what operations you can execute on it.

Fields

A field is a variable that belongs to a class or an object.

Operations

In Java, Operations are the instructions you can run to process the data in variables.Some operations read and write the values of variables, while other operations controls the flow of program.

Methods

A Java method is a collection of java statements that are grouped together to perform an operation. When you call a method, it actually executes several statements in order. Methods are

typically used when you need to group operations together, that you need to be able execute from several different places.

Constructors

Constructors are a special kind of java method that is executed when an object of that class is created. Constructors initializes the objects internal fields.

Interfaces

An interface is a programming structure that enforce certain properties on an object (class). For example, say we have a car class and a scooter class and a truck class. Each of these three classes should have a start_engine() action. How the "engine is started" for each vehicle is left to each particular class, but the fact that they must have a start_engine action is the domain of the interface.In its most common form, an interface is a group of related methods with empty bodies.

Packages

A package is a directory containing Java classes and interfaces. Packages groups related classes and interfaces, thus making modularization of your code easier.

Java Files

All Java code must reside inside a file with the extension .java . For instance, your first java class HelloWorld.java. An application can consist of many such .java files.

Java Comments

Comments are set of statements that are used to explain about java program components. It is provide information and explanation about variables,methods, class or any other statements. So that other developers can understand the code and flow of execution by reading the comments. It is not executed by compiler or interpreter.

Types of Comments

There are three types of comments

- Single-line comment
- Multi-line comment
- Documentation comment

Single-line comment

// – It is single line comment
Example:

```
Class Add
{
Int a;//declaration of variable
}
```

Multi-line comment

/*
*/ – Multi-line comment. It is also used for single line also.

Example:
Class Add
{
Int a;/*declaration of variable*/
/*
This is function for add two numbers
*/
Public void add();
}

Documentation comment

/**
*
*/- Documentation Comment
/**
* This is program for print hello statement
*/

Variables in Java

A variable is a name given to a memory location. It is the basic unit of storage in a program.

- The value stored in a variable can be changed during program execution.
- A variable is only a name given to a memory location, all the operations done on the variable effects that memory location.
- In Java, all the variables must be declared before use.

WE CAN DECLARE VARIABLES IN JAVA AS FOLLOWS:

type: Type of data that can be stored in this variable.
name: Name given to the variable.
In this way, a name can only be given to a memory location. It can be assigned values in two ways:

- Variable Initialization
- Assigning value by taking input

datatype: Type of data that can be stored in this variable.
variable_name: Name given to the variable.
value: It is the initial value stored in the variable.

Types of variables

There are three types of variables in Java:

- Local Variables
- Instance Variables
- Static Variables

Local Variables

- A variable defined within a block or method or constructor is called local variable.

- These variable are created when the block in entered or the function is called and destroyed after exiting from the block or when the call returns from the function.
- The scope of these variables exists only within the block in which the variable is declared. i.e. we can access these variable only within that block.
- Initilisation of Local Variable is Mandatory.

Instance Variables

- Instance variables are non-static variables and are declared in a class outside any method, constructor or block.
- As instance variables are declared in a class, these variables are created when an object of the class is created and destroyed when the object is destroyed.
- Unlike local variables, we may use access specifiers for instance variables. If we do not specify any access specifier then the default access specifier will be used.
- Initilisation of Instance Variable is not Mandatory. Its default value is 0
- Instance Variable can be accessed only by creating objects.

Static Variables

- Static variables are also known as Class variables.
- These variables are declared similarly as instance variables, the difference is that static variables are declared using the static keyword within a class outside any method constructor or block.
- Unlike instance variables, we can only have one copy of a static variable per class irrespective of how many objects we create.
- Static variables are created at the start of program execution and destroyed automatically when execution ends.
- Initilisation of Static Variable is not Mandatory. Its default value is 0
- If we access the static variable like Instance variable (through an object), the compiler will show the warning message and it won't halt the program. The compiler will replace the object name to class name automatically.
- If we access the static variable without the class name, Compiler will automatically append the class name.

Data types in Java

There are majorly two types of languages.

- First, one is **Statically typed language** where each variable and expression type is already known at compile time. Once a variable is declared to be of a certain data type, it cannot hold values of other data types.
- **Example:** C, C++, Java.
- The other is **Dynamically typed languages.** These languages can receive different data types over time.
- **Example:** Ruby, Python

Java is **statically typed and also a strongly typed language** because, in Java, each type of data (such as integer, character, hexadecimal, packed decimal, and so forth) is predefined as part of the programming language and all constants or variables defined for a given program must be described with one of the data types.

Data Types Are

- **Primitive Data Type:** such as boolean, char, int, short, byte, long, float, and double
- **Non-Primitive Data Type or Object Data type:** such as String, Array, etc.

Primitive Data Type

Boolean

Boolean data type represents only one bit of information **either true or false**, but the size of the boolean data type is **virtual machine-dependent**. Values of type boolean are not converted implicitly or explicitly (with casts) to any other type. But the programmer can easily write conversion code.

Byte

The byte data type is an 8-bit signed two's complement integer. The byte data type is useful for saving memory in large arrays.

Short

The short data type is a 16-bit signed two's complement integer. Similar to byte, use a short to save memory in large arrays, in situations where the memory savings actually matters.

Int

It is a 32-bit signed two's complement integer.

Long

The long data type is a 64-bit two's complement integer.

Float

The float data type is a single-precision 32-bit IEEE 754 floating-point. Use a float (instead of double) if you need to save memory in large arrays of floating-point numbers.

Double

The double data type is a double-precision 64-bit IEEE 754 floating-point. For decimal values, this data type is generally the default choice.

Char

The char data type is a single 16-bit Unicode character.

Non-Primitive Data Type or Reference Data Types

The **Reference Data Types** will contain a memory address of variable value because the reference types won't store the variable value directly in memory. They are **strings**, **objects**, arrays, etc.

String

- Strings are defined as an array of characters. The difference between a character array and a string in Java is, the string is designed to hold a sequence of characters in a single variable whereas, a character array is a collection of separate char type entities.
- Unlike C/C++, Java strings are not terminated with a null character.
- Below is the basic syntax for declaring a string in Java programming language.

Class

A class is a user-defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type. In general, class declarations can include these components, in order:

1. **Modifiers**: A class can be public or has default access (Refer this for details).
2. **Class name:** The name should begin with a initial letter (capitalized by convention).
3. **Superclass(if any):** The name of the class's parent (superclass), if any, preceded by the keyword extends. A class can only extend (subclass) one parent.
4. **Interfaces(if any):** A comma-separated list of interfaces implemented by the class, if any, preceded by the keyword implements. A class can implement more than one interface.
5. **Body:** The class body surrounded by braces, { }.

Object

It is a basic unit of Object-Oriented Programming and represents the real-life entities. A typical Java program creates many objects, which as you know, interact by invoking methods. An object consists of :

1. **State**: It is represented by attributes of an object. It also reflects the properties of an object.
2. **Behavior**: It is represented by methods of an object. It also reflects the response of an object with other objects.
3. **Identity**: It gives a unique name to an object and enables one object to interact with other objects.

Interface

Like a class, an interface can have methods and variables, but the methods declared in an interface are by default abstract (only method signature, nobody).

- Interfaces specify what a class must do and not how. It is the blueprint of the class.

- An Interface is about capabilities like a Player may be an interface and any class implementing Player must be able to (or must implement) move(). So it specifies a set of methods that the class has to implement.
- If a class implements an interface and does not provide method bodies for all functions specified in the interface, then the class must be declared abstract.
- A Java library example is, Comparator Interface. If a class implements this interface, then it can be used to sort a collection.

Array

An array is a group of like-typed variables that are referred to by a common name. Arrays in Java work differently than they do in C/C++. The following are some important points about Java arrays.

- In Java, all arrays are dynamically allocated. (discussed below)
-
- Since arrays are objects in Java, we can find their length using member length. This is different from C/C++ where we find length using size.
- A Java array variable can also be declared like other variables with [] after the data type.
- The variables in the array are ordered and each has an index beginning from 0.
- Java array can be also be used as a static field, a local variable or a method parameter.
- The **size** of an array must be specified by an int value and not long or short.
- The direct superclass of an array type is Object.

Strings in Java

Strings are an incredibly common type of data in computers. This page introduces the basics of Java strings: chars, **+**, length(), and substring().
A Java string is a series of characters gathered together, like the word "Hello", or the phrase "practice makes perfect". Create a string in the code by writing its chars out between double quotes.

It is probably the most commonly used class in java library. In java, every string that we create is actually an object of type **String**. One important thing to notice about string object is that string objects are **immutable** that means once a string object is created it cannot be changed.
The Java String class implements Serializable, Comparable and CharSequence interface a