CONSTRUCTOR

Constructor- Introduction

In Java, a Constructor is a block of codes similar to the method. It is called when an instance of the class is created. At the time of calling the constructor, memory for the object is allocated in the memory. It is a special type of method that is used to initialize the object. Every time an object is created using the new() keyword, at least one constructor is called.

Note: It is not necessary to write a constructor for a class. It is because the java compiler creates a default constructor (constructor with no arguments) if your class doesn't have any.

Need Of Constuctor

Think of a Box. If we talk about a box class then it will have some class variables (say length, breadth, and height). But when it comes to creating its object(i.e Box will now exist in the computer's memory), then can a box be there with no value defined for its dimensions? The answer is No.

Note: Constructors are used to assign values to the class variables at the time of object creation, either explicitly done by the programmer or by Java itself (Default Constructor).

When a Constructor is Called?

Each time an object is created using a new() keyword, at least one constructor (it could be the default constructor) is invoked to assign initial values to the Data Members(Fields) of the same class.

Rules for writing constructors are as follows:

- The constructor(s) of a class must have the same name as the class name in which it resides.
- A constructor in Java cannot be abstract, final, static, or Synchronized.
- Access modifiers can be used in constructor declaration to control its access i.e which other class can call the constructor.
- There are no "return value" statements in the constructor, but the constructor returns the current class instance. We can write 'return' inside a constructor.

Types of Constructor

Now is the correct time to discuss the types of the constructor, so primarily there are three types of constructors in Java are mentioned below:

- Default Constructor
- Parameterized Constructor
- Copy Constructor

1. Default Constructor in Java

A constructor that has no parameters is known as default the constructor. A default constructor is invisible. And if we write a constructor with no arguments, the compiler does not create a default constructor. It is taken out. It is being overloaded and called a parameterized constructor. The default constructor changed into the parameterized constructor. But Parameterized constructor can't change the default constructor. The default constructor can be implicit or explicit. If we don't define explicitly, we get an implicit default constructor. If we manually write a constructor, the implicit one is overridden.

Note: Default constructor provides the default values to the object like 0, null, etc. depending on the type.

2. Parameterized Constructor in Java
A constructor that has parameters is known as parameterized constructor. If we want to initialize fields of the class with our own values, then use a parameterized constructor.

Now the most important topic that comes into play is the strong incorporation of OOPS with constructors known as constructor overloading. Just like methods, we can overload constructors for creating objects in different ways. The compiler differentiates constructors on the basis of the number of parameters, types of parameters, and order of the parameters.

3. Copy Constructor in Java
Unlike other constructors copy constructor is passed with another object which copies the data available from the passed object to the newly created object.

In Java, there is no such inbuilt copy constructor available like in other programming languages such as C++, instead we can create our own copy constructor by passing the object of the same class to the other instance(object) of the class.

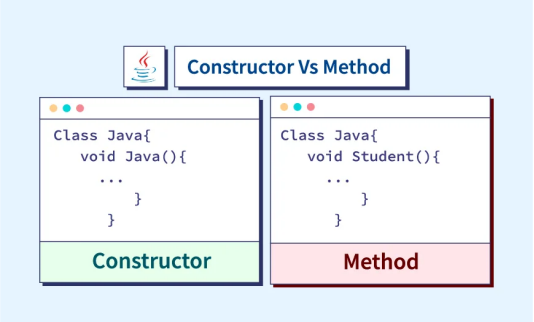
Constructor Overloading

Now the most important topic that comes into play is the strong incorporation of OOPS with constructors known as constructor overloading. Just like methods, we can overload constructors for creating objects in different ways. The compiler differentiates constructors on the basis of the number of parameters, types of parameters, and order of the parameters. Only those value will be initialised that are parameterised with the initiated value by us other values will be initiated by default.

Constructor VS Methods

Constructors must have the same name as the class within which it is defined it is not necessary for the method in Java.

- Constructors do not return any type while method(s) have the return type or void if does not return any value.
- Constructors are called only once at the time of Object Creation while method(s) can be called any number of times.



'this' Keyword

In Java, 'this' is a reference variable that refers to the current object or can be said "this" in Java is a keyword that refers to the current object instance. It can be used to call current class methods and fields, to pass an instance of the current class as a parameter, and to differentiate between the local and instance variables. Using "this" reference can improve code readability and reduce naming conflicts.

Following are the ways to use the ‘this’ keyword in Java mentioned below:

1. Using the ‘this’ keyword to refer to current class instance variables.
2. Using this() to invoke the current class constructor
3. Using ‘this’ keyword to return the current class instance
4. Using ‘this’ keyword as the method parameter
5. Using ‘this’ keyword to invoke the current class method
6. Using ‘this’ keyword as an argument in the constructor call
   this (without parentheses) refers to the current object, but in the case of this() it's specifically referring to a constructor call, not the object.

Note: this (without parentheses) refers to the current object, but in the case of this() it's specifically referring to a constructor call, not the object.