Constructor in Java

**Constructor in java** is a *special type of method* that is used to initialize the object.

Java constructor is *invoked at the time of object creation*. It constructs the values i.e. provides data for the object that is why it is known as constructor.

Rules for creating java constructor

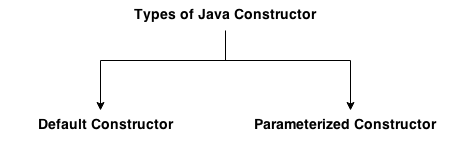
There are basically two rules defined for the constructor.

1. Constructor name must be same as its class name
2. Constructor must have no explicit return type

Types of java constructors

There are two types of constructors:

1. Default constructor (no-arg constructor)
2. Parameterized constructor



## Java Default Constructor

|  |
| --- |
| A constructor that have no parameter is known as default constructor. |

### Syntax of default constructor:

<class\_name>(){}

## Example of default constructor

|  |
| --- |
| In this example, we are creating the no-arg constructor in the Bike class. It will be invoked at the time of object creation. |

**class** Bike1{

Bike1()

{System.out.println("Bike is created");

}

**public** **static** **void** main(String args[])

{

Bike1 b=**new** Bike1();

}

}

Output:

Bike is created

### Java parameterized constructor

|  |
| --- |
| A constructor that have parameters is known as parameterized constructor. |

### Why use parameterized constructor?

|  |
| --- |
| Parameterized constructor is used to provide different values to the distinct objects. |

### Example of parameterized constructor

|  |
| --- |
| In this example, we have created the constructor of Student class that have two parameters. We can have any number of parameters in the constructor. |

1. **class** Student4{
2. **int** id;
3. String name;
5. Student4(**int** i,String n){
6. id = i;
7. name = n;
8. }
9. **void** display(){System.out.println(id+" "+name);}
11. **public** **static** **void** main(String args[]){
12. Student4 s1 = **new** Student4(111,"Karan");
13. Student4 s2 = **new** Student4(222,"Aryan");
14. s1.display();
15. s2.display();
16. }
17. }

[**Test it Now**](http://www.javatpoint.com/opr/test.jsp?filename=Student4)

Output:

111 Karan

222 Aryan

## Constructor Overloading in Java

|  |
| --- |
| Constructor overloading is a technique in Java in which a class can have any number of constructors that differ in parameter lists.The compiler differentiates these constructors by taking into account the number of parameters in the list and their type. |

### Example of Constructor Overloading

1. **class** Student5{
2. **int** id;
3. String name;
4. **int** age;
5. Student5(**int** i,String n){
6. id = i;
7. name = n;
8. }
9. Student5(**int** i,String n,**int** a){
10. id = i;
11. name = n;
12. age=a;
13. }
14. **void** display(){System.out.println(id+" "+name+" "+age);}
16. **public** **static** **void** main(String args[]){
17. Student5 s1 = **new** Student5(111,"Karan");
18. Student5 s2 = **new** Student5(222,"Aryan",25);
19. s1.display();
20. s2.display();
21. }
22. }

Output:

111 Karan 0

222 Aryan 25

## Difference between constructor and method in java

There are many differences between constructors and methods. They are given below.

|  |  |
| --- | --- |
| **Java Constructor** | **Java Method** |
| Constructor is used to initialize the state of an object. | Method is used to expose behaviour of an object. |
| Constructor must not have return type. | Method must have return type. |
| Constructor is invoked implicitly. | Method is invoked explicitly. |
| The java compiler provides a default constructor if you don't have any constructor. | Method is not provided by compiler in any case. |
| Constructor name must be same as the class name. | Method name may or may not be same as class name. |

|  |
| --- |
| Java Copy Constructor There is no copy constructor in java. But, we can copy the values of one object to another like copy constructor in C++.  There are many ways to copy the values of one object into another in java. They are:   * By constructor * By assigning the values of one object into another * By clone() method of Object class   In this example, we are going to copy the values of one object into another using java constructor. |

1. **class** Student6{
2. **int** id;
3. String name;
4. Student6(**int** i,String n){
5. id = i;
6. name = n;
7. }
9. Student6(Student6 s){
10. id = s.id;
11. name =s.name;
12. }
13. **void** display(){System.out.println(id+" "+name);}
15. **public** **static** **void** main(String args[]){
16. Student6 s1 = **new** Student6(111,"Karan");
17. Student6 s2 = **new** Student6(s1);
18. s1.display();
19. s2.display();
20. }
21. }

Output:

111 Karan

111 Karan