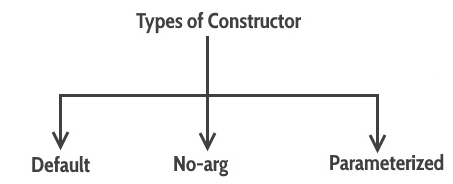
**Constructors :**

**Why construstor ?** if I create class student with data member int roll no, string name; default string name = null, default int roll no =0. Now created an object for 1 student, if there are 600 students, we need to create 600 objects of 600 students. By default, every student name will be null and roll no will be 0;

If after every obj creation, we initialize obj.rollNo=10, obj.name=”abc”; for 600 students 1200 lines of code will be generated, instead of this, we use constructor, while creating an object we initialize it with the help of constructor.

After creating an object, Constructors are used to initialize the object. It constructs the value i.e. provides data for the object. Constructors initialize the data members, i.e. provides a value to data members. Example in eclipse.

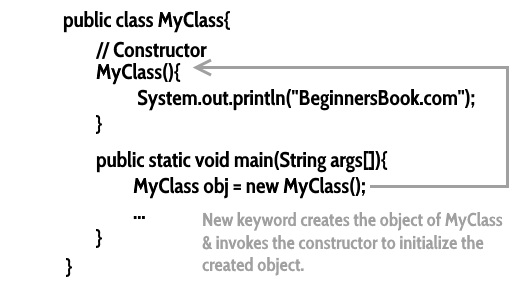
1. its name is the same as the class name.
2. Java constructors are invoked when their objects are created.
3. Every class has a constructor, when we don’t explicitly declare a constructor for any java class the compiler creates a default constructor for that class which **does not** have any **return** type.
4. **Constructor does not have any return type in Java.**
5. Constructor’s name must be similar to that of the class name inside.
6. Constructors are automatically called when an object is created. The constructor should be written inside the class.
7. Class can have **multiple** constructors (constructor overloading)
8. **Constructor Overriding and Inheritance not allowed** in Java
9. Only these **Modifiers allowed**: Public, default, protected, private (**NO** Static, final, abstract)



Default is a constructor provided by java by default. It only allocates memory; it does not assigns values to variables.

Parameterized : pass parameters to initialize data members of class.

Non-parameter : DO not provide any parameter



In above image, it shows a constructor flow, if we do not write a body for constructor, still we give a call to constructor while creating an object. And if we do not provide body to constructor, this will be provided by java.

NOTE : Constructor should be **inside** a class.

**Constructor overloading** : Java class can have multiple constructors with different parameters (i.e. constructor overloading) example :

Class Test

{

Test()

{

}

Test(int a)

{

}

Test(char c)

{

}

}

**Constructor Overriding and Inheritance** : Java class **Does not allow** constructor Overriding and constructor inheritance.