

Constructor :

- It is an initializer method , it is used for object instantiation and object initialization
- Object instantiation is nothing but creating an object , where as object initialization nothing but not only initializing the value for the fields and also resource allocation [Network Connections | Database Connection | Opening the File....]

What is difference between Instance methods and Constructors ?

- Instance methods are defined for one time and executed N.no.of times after object instantiation , But constructors are defined for one time but it can be invoked only once in order to creating an Object
- Instance methods required to call explicitly , where as constructors are invoked implicitly
- Instance methods required an object reference to call, where as constructors required keyword "new"

Can you create an Object for a class without a constructor ?

- No

Can you create an object for an empty class ?

- Yes, [possible by using compiled constructor]
- while compilation of java program , Then Java "compiler" will ensure the class is defined with any constructor or not .

- If the class is not defined with any constructors then java compiler will take the responsibility to create a constructor in the class "called" compiled constructor or default constructor.
- With the help of "compiled constructor" we can create an Object

Can You see the content of .class File ?

Yes, It is possible by using "Javap"

```
//A.java
```

```
class A
```

```
{  
}
```

```
>javac A.java
```

```
Eg: >javap A
```

```
Compiled from "A.java"
```

```
class A {
```

```
    A(); → "compiled constructor"  
}
```

```
>javap java.lang.String
```

```
>javap java.io.PrintStream
```

Rules Of the Constructor

- Constructor name and class name must be same
- Constructors can be parameterized
- Constructors can be overloaded

- Constructor can nested [constructor chaining]
- Execution order of the constructors always from top to bottom
- Constructor never returns any value even “void” type

Type of Constructors :

- Default constructor
- Parameterized Constructor
- Copy Constructor

Default constructor:

- It is a process of defining a constructor with out any arguments or 0 parameterized constructor or parameter less constructor

Example:

```
class Rect
```

```
{  
    float l,b;  
  
    Rect( )  
    { l=4.0f; b=4.0f; }  
  
    float findArea( )  
    { return (l*b); }  
  
    public static void main(String args[ ])  
    { Rect r=new Rect( );  
      float ar=r.findArea();  
      System.out.println("Area of Rect : "+ar);  
    }  
}
```

Parameterized Constructor:

- Process of defining the constructor with primitive type parameter
- If the class is defined only with parameterized constructor then it is mandatory to pass the parameter to that constructor in order initiation of an Object, otherwise JVM is trying to call

default constructor which is not defined in the class thus it will raise an Error

```
class Triangle
{
    float l,h; //instance fields

    Triangle(float l,float h) //l,h are formal parameter
    { this.l=l;          // acts as local variable
      this.h=h; }

    float findArea() //instance mtd
    { return (0.5f*l*h); }

    public static void main(String args[ ])
    { Triangle t=new Triangle(5.0f,5.0f);
      float at=t.findArea();
      System.out.println("Area of Triangle : "+at); }
}
```

Constructor Overloading :

- Process of defining more than one constructor for different purpose

```
class Demo
{
    Demo()
    { System.out.println("No-arg const "); }

    Demo(int x)
    { System.out.println("1-arg const : "+x); }

    Demo(int x,int y)
    {System.out.println("2-args const : "+x+" "+y); }

    public static void main(String args[])
    {
```

```
{ Demo d=new Demo( );  
  Demo d1=new Demo(123);  
  Demo d2=new Demo(10,20);  
}  
}
```

Constructor chaining :

- It is process of calling an existed constructor into another constructor of the same class
- It is possible using "this()"
- Advantages are reusability of code
- Call to this() must be the first statement in constructor
-

```
class Test  
{  
    Test( )  
    { System.out.println("Def const "); }  
  
    Test(int x)  
    { this();  
      System.out.println("1 para const : "+x); }  
  
    Test(int x,int y)  
    { this(123);  
      System.out.println("2 para const : "+x+" "+y); }  
  
    public static void main(String args[ ])  
    { Test t=new Test(10,20); }  
}
```

Copy Constructor :

- It is the process of defining constructor with reference type parameters
- The main motto of copy constructor is coping the data from one object to another
- In order to work with copy constructor , then corresponding class must be defined with either a default constructor or parameterized constructor . along with copy constructor

```
class Sample
{
    int x,y; //instance fields

    Sample( )
    { x=222; y=444; }

    Sample(Sample o) //o is formal are copies of actual
    { x=o.x; y=o.y; } //x,y are instance fields

    void getData()
    { System.out.println("x val is : "+x);
      System.out.println("y val is : "+y); }

    public static void main(String args[ ])
    { Sample s1=new Sample( );
      Sample s2=new Sample(s1) ;
        s1.getData( );
        System.out.println("Data From s2");
        s2.getData();
    }
}
```

Note: we should not declare static variable in the constructor , but we can use static variables in the constructor