Static

Compiled By:
Vinod Kumar(Asst. Prof.)

Java static keyword

The **static keyword** in java is used for memory management mainly. We can apply java static keyword with variables, methods, blocks and nested class. The static keyword belongs to the class than instance of the class.

- Java supports definition of global methods and variables that can be accessed without creating objects of a class. Such members are called Static members.
- Define a variable by marking with the static methods.
- This feature is useful when we want to create a variable common to all instances of a class.
- Note: Java creates only one copy for a static variable which can be used even if the class is never instantiated.

The static can be:

- variable (also known as class variable)
- method (also known as class method)
- block
- nested class

Java static variable

If you declare any variable as static, it is known static variable.

- The static variable can be used to refer the common property of all objects (that is not unique for each object) e.g. company name of employees, college name of students etc.
- The static variable gets memory only once in class area at the time of class loading.

Advantage of static variable

- It makes your program **memory efficient** (i.e. it saves memory).

```
class Student{
        int rollno;
        String name;
        String university="CURAJ";
}
```

Suppose there are 2000 students in our university, now all instance data members will get memory each time when object is created. All student have its unique rollno and name so instance data member is good . Here, university refers to the common property of all objects . If we make it static , this field will get memory only once.

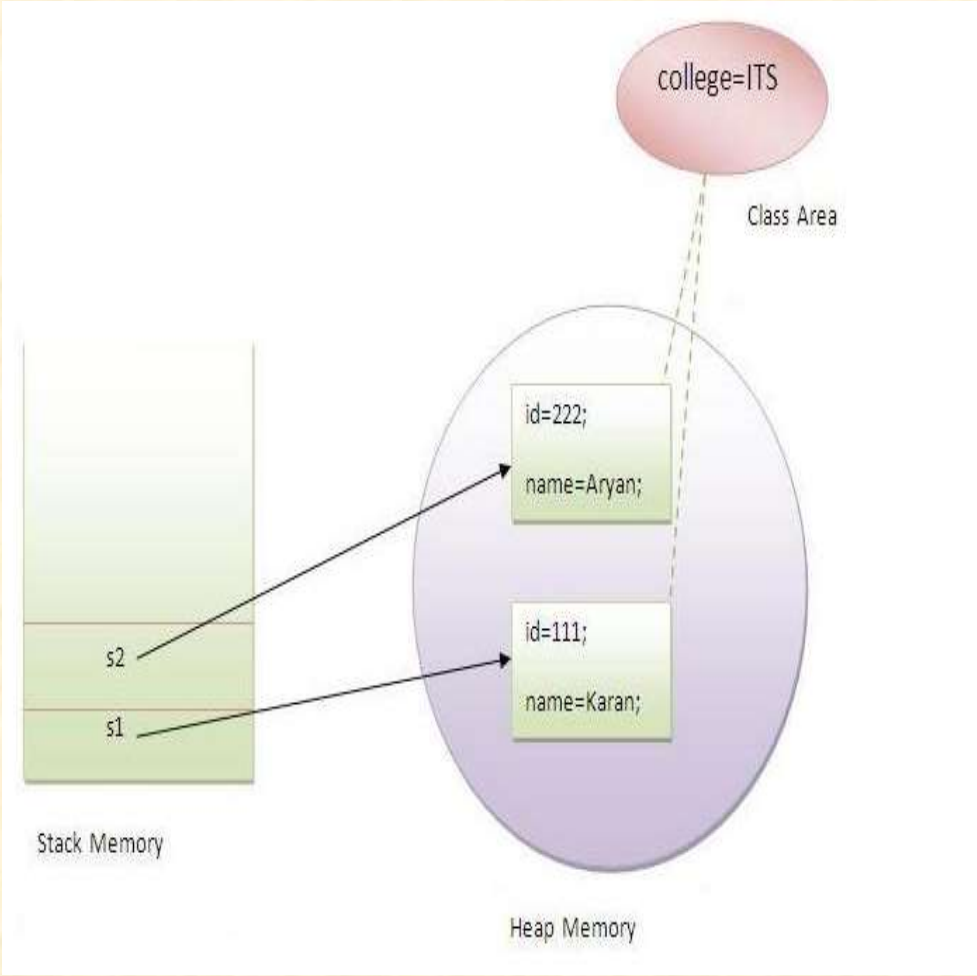
Example of static variable

```
class Student8{
  int rollno;
  String name;
  static String college ="ITS";

  Student8(int r,String n){
  rollno = r;
  name = n;
  }
 void display (){
System.out.println(rollno+" "+name+" "+college);
                    }

 public static void main(String args[]){
 Student8 s1 = new Student8(111,"Karan");
 Student8 s2 = new Student8(222,"Aryan");

 s1.display();
 s2.display();
 }
}
```



Program of counter without static variable and by static variable

without static variable

```
class Counter{
int count=0;/*will get memory when instance is c
reated */

Counter(){
count++;
System.out.println(count);
}

public static void main(String args[]){

Counter c1=new Counter();
Counter c2=new Counter();
Counter c3=new Counter();

 }
}
```

By static variable

```
class Counter2{
static int count=0;/*will get memory only once an
d retain its value */

Counter2(){
count++;
System.out.println(count);
}

public static void main(String args[]){

Counter2 c1=new Counter2();
Counter2 c2=new Counter2();
Counter2 c3=new Counter2();

 }
}
```

Static Methods

- A static method belongs to the class rather than object of a class.
- A class can have methods that are defined as static (e.g., main method).
- Static methods can be accessed without using objects. Also, there is NO need to create objects.
- static method can access static data member and can change the value of it.
- They are prefixed with keyword “static”
- Static methods are generally used to group related library functions that don’t depend on data members of its class. For example, Math library functions.

Example :1

```
class Student9{
    int rollno;
    String name;
    static String university = "ITS";

    static void change(){
    university = “Curaj";
    }

    Student9(int r, String n){
    rollno = r;
    name = n;
    }

    void display ()
{
System.out.println(rollno+" "+name+" "+university);
}

 public static void main(String args[]){
    Student9.change();

    Student9 s1 = new Student9 (111,"Karan");
    Student9 s2 = new Student9 (222,"Aryan");
    Student9 s3 = new Student9 (333,"Sonoo");

    s1.display();
    s2.display();
    s3.display();
    }
}
```

Output:-

111 Karan Curaj

222 Aryan Curaj

333 Sonoo Curaj

Example: 2

```
class Calculate{
  static int cube(int x){
  return x*x*x;
  }

  public static void main(String args[]){
  int result=Calculate.cube(5);
  System.out.println(result);
  }
}
```

Output :

125

Comparator class with Static methods

```
/* Comparator.java: A class with static data
      items comparison methods*/
class Comparator {
      public static int max(int a, int b)
      {
            if( a > b)
                  return a;
            else
                  return b;
      }

   public static String max(String a, String b)
      {
            if( a.compareTo (b) > 0)
                  return a;
            else
                  return b;
      }
}
```

```
class MyClass {
  public static void main(String args[])
  {
      String s1 = "Melbourne";
      String s2 = "Sydney";
      String s3  = "Adelaide";

      int a = 10;
      int b = 20;

      System.out.println(Comparator.max(a, b));
// which number is big
      System.out.println(Comparator.max(s1, s2));
// which city is big
      System.out.println(Comparator.max(s1, s3));
// which city is big
  }
}
```

Directly accessed using ClassName (NO Objects)

Static methods restrictions

- They can only call other static methods.
- They can only access static data.
- They cannot refer to “this” or “super” (more later) in anyway.

```
class A2{
  static{System.out.println("static block is invoked");}
  public static void main(String args[]){
   System.out.println("Hello main");
  }
}
```

Output:

```
static block is invoked
Hello main
```

Can we execute a program without main() method?

- Yes, one of the way is static block but in previous version of JDK not In JDK 1.7.

```
class A3{
  static{
  System.out.println("static block is invoked");
  System.exit(0);
  }
}
```