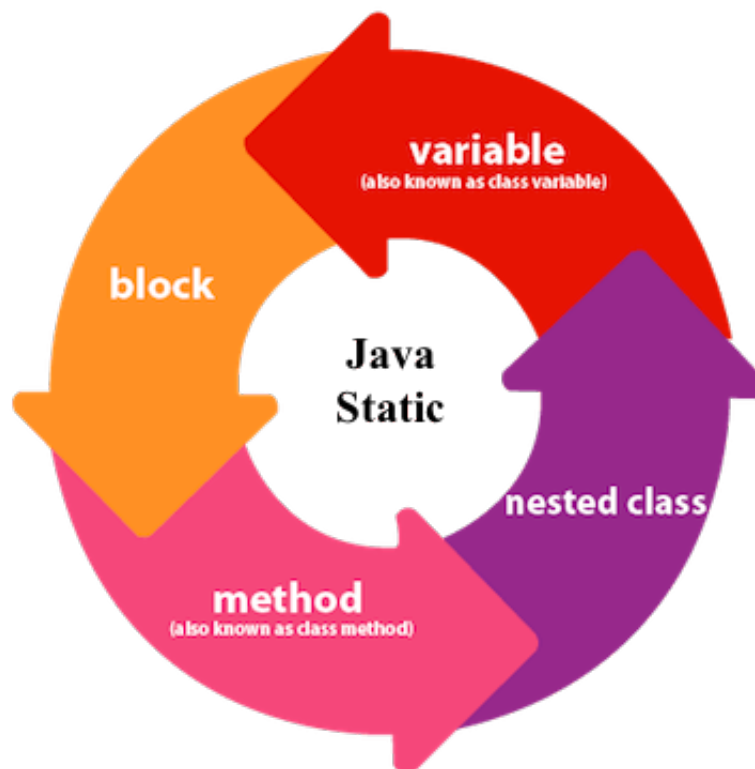# Java static keyword

The **static keyword** in [Java](#) is used for memory management mainly. We can apply static keyword with [variables](#), methods, blocks and [nested classes](#). The static keyword belongs to the class than an instance of the class.

The static can be:

1. Variable (also known as a class variable)
2. Method (also known as a class method)
3. Block
4. Nested class

1) Java static variable

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

- The static variable can be used to refer to the common property of all

objects (which is not unique for each object), for example, the company name of employees, college name of students, etc.

- The static variable gets memory only once in the class area at the time of class loading.

Advantages of static variable

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

Understanding the problem without static variable

1. **class** Student{
2.   **int** rollno;
3.   String name;
4.   String college="ITS";
5. }

Suppose there are 500 students in my college, now all instance data members will get memory each time when the object is created. All students have its unique rollno and name, so instance data member is good in such case. Here, "college" refers to the common property of all objects. If we make it static, this field will get the memory only once.

Java static property is shared to all objects.
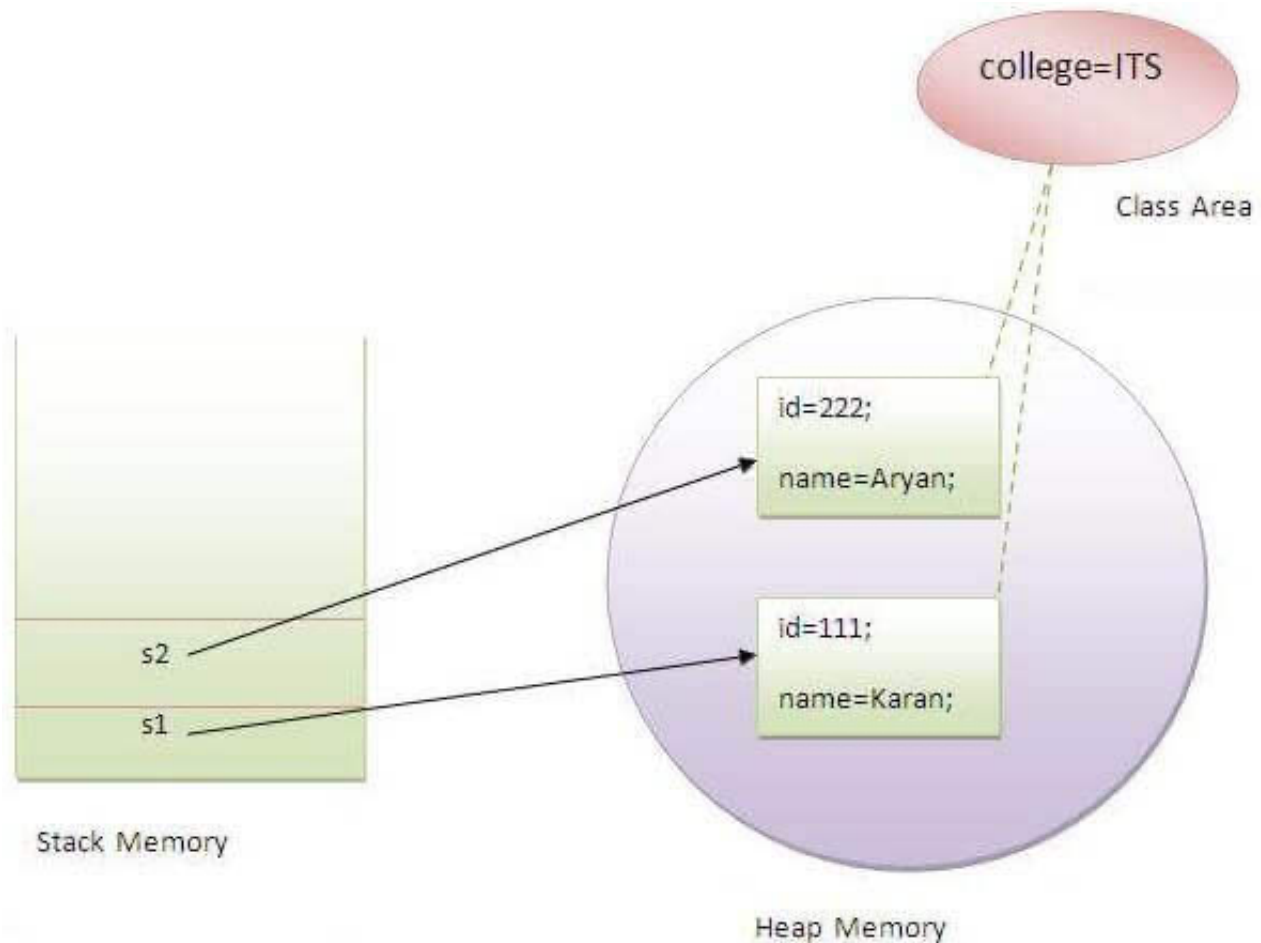
Example of static variable

1.
2. **class** Student{
3.   **int** rollno;
4.   String name;
5.   **static** String college ="ITS";
6.
7.   Student(**int** r, String n){

```
8.    rollno = r;
9.    name = n;
10.   }
11.
12.   void display (){System.out.println(rollno+" "+name+" "+college);}
13. }
14.
15. public class TestStaticVariable1{
16.   public static void main(String args[]){
17.   Student s1 = new Student(111,"Karan");
18.   Student s2 = new Student(222,"Aryan");
19.
20.
21.   s1.display();
22.   s2.display();
23.   }
24. }
```

Test it Now

Output:

```
111 Karan ITS
222 Aryan ITS
```

Program of the counter without static variable

In this example, we have created an instance variable named count which is incremented in the constructor. Since instance variable gets the memory at the time of object creation, each object will have the copy of the instance variable. If it is incremented, it won't reflect other objects. So each object will have the value 1 in the count variable.

1.
2.
3. **class** Counter{
4. **int** count=0;
5.
6. Counter(){
7. count++;
8. System.out.println(count);

9. }
10.
11. **public static void** main(String args[]){
12.
13. Counter c1=**new** Counter();
14. Counter c2=**new** Counter();
15. Counter c3=**new** Counter();
16. }
17. }

[Test it Now](#)

Output:

Program of counter by static variable

As we have mentioned above, static variable will get the memory only once, if any object changes the value of the static variable, it will retain its value.

1.
2.
3. **class** Counter2{
4. **static int** count=0;
5.
6. Counter2(){
7. count++;
8. System.out.println(count);
9. }
10.
11. **public static void** main(String args[]){
12.
13. Counter2 c1=**new** Counter2();

14.   Counter2 c2=**new** Counter2();
15.   Counter2 c3=**new** Counter2();
16.   }
17.   }

Test it Now

Output:

2) Java static method

If you apply static keyword with any method, it is known as static method.

- A static method belongs to the class rather than the object of a class.
- A static method can be invoked without the need for creating an instance of a class.
- A static method can access static data member and can change the value of it.

Example of static method

1.
2.   **class** Student{
3.   **int** rollno;
4.   String name;
5.   **static** String college = "ITS";
6.
7.   **static void** change(){
8.   college = "BBDIT";
9.   }
10.
11.   Student(**int** r, String n){
12.   rollno = r;

13.   name = n;
14.   }
15.
16.   **void** display(){System.out.println(rollno+" "+name+" "+college);}
17. }
18.
19. **public class** TestStaticMethod{
20.   **public static void** main(String args[]){
21.   Student.change();
22.
23.   Student s1 = **new** Student(111,"Karan");
24.   Student s2 = **new** Student(222,"Aryan");
25.   Student s3 = **new** Student(333,"Sonoo");
26.
27.   s1.display();
28.   s2.display();
29.   s3.display();
30.   }
31. }

Test it Now

```
Output:111 Karan BBDIT
       222 Aryan BBDIT
       333 Sonoo BBDIT
```

Another example of a static method that performs a normal calculation

1.
2.
3. **class** Calculate{
4.   **static int** cube(**int** x){

5.   **return** x*x*x;
6.   }
7.
8.   **public static void** main(String args[]){
9.   **int** result=Calculate.cube(5);
10.   System.out.println(result);
11.   }
12. }

[**Test it Now**](#)

Restrictions for the static method

There are two main restrictions for the static method. They are:

1. The static method can not use non static data member or call non-static method directly.
2. this and super cannot be used in static context.

1. **class** A{
2.   **int** a=40;
3.
4.   **public static void** main(String args[]){
5.   System.out.println(a);
6. }
7. }

[**Test it Now**](#)

```
Output:Compile Time Error
```

Q) Why is the Java main method static?

Ans) It is because the object is not required to call a static method. If it

were a non-static method, [JVM](#) creates an object first then call main() method that will lead the problem of extra memory allocation.

3) Java static block

- Is used to initialize the static data member.
- It is executed before the main method at the time of classloading.

Example of static block

1. **class** A2{
2.   **static**{System.out.println("static block is invoked");}
3.   **public static void** main(String args[]){
4.   System.out.println("Hello main");
5.   }
6. }

[Test it Now](#)

```
Output:static block is invoked
        Hello main
```

Q) Can we execute a program without main() method?

Ans) No, one of the ways was the static block, but it was possible till JDK 1.6. Since JDK 1.7, it is not possible to execute a Java class without the [main method](#).

1. **class** A3{
2.   **static**{
3.   System.out.println("static block is invoked");
4.   System.exit(0);
5.   }

6. }

Test it Now

Output:

Since JDK 1.7 and above, output would be:

```
Error: Main method not found in class A3, please define the ma
    public static void main(String[] args)
or a JavaFX application class must extend javafx.application./
```

Next Topic[this keyword in java](#)