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.

The static can be:

1. variable (also known as class variable)
2. method (also known as class method)
3. block
4. nested class

1) 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).

Understanding problem without static variable

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

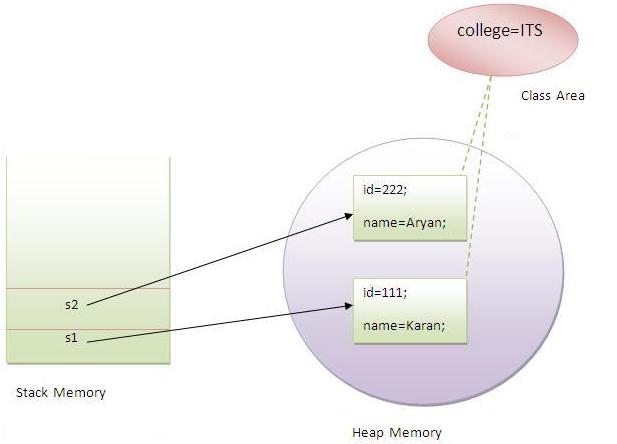
Suppose there are 500 students in my college, now all instance data members will get memory each time when object is created.All student have its unique rollno and name so instance data member is good.Here, college refers to the common property of all objects.If we make it static,this field will get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

//Program of static variable

1. **class** Student8{
2. **int** rollno;
3. String name;
4. **static** String college ="ITS";
6. Student8(**int** r,String n){
7. rollno = r;
8. name = n;
9. }
10. **void** display (){System.out.println(rollno+" "+name+" "+college);}
12. **public** **static** **void** main(String args[]){
13. Student8 s1 = **new** Student8(111,"Karan");
14. Student8 s2 = **new** Student8(222,"Aryan");
16. s1.display();
17. s2.display();
18. }
19. }

Output:111 Karan ITS

222 Aryan ITS



Program of 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 to other objects. So each objects will have the value 1 in the count variable.

1. **class** Counter{
2. **int** count=0;//will get memory when instance is created
4. Counter(){
5. count++;
6. System.out.println(count);
7. }
9. **public** **static** **void** main(String args[]){
11. Counter c1=**new** Counter();
12. Counter c2=**new** Counter();
13. Counter c3=**new** Counter();
15. }
16. }

Output:1

1

1

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. **class** Counter2{
2. **static** **int** count=0;//will get memory only once and retain its value
4. Counter2(){
5. count++;
6. System.out.println(count);
7. }
9. **public** **static** **void** main(String args[]){
11. Counter2 c1=**new** Counter2();
12. Counter2 c2=**new** Counter2();
13. Counter2 c3=**new** Counter2();
15. }
16. }

Output:1

2

3

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

Example of static method

1. //Program of changing the common property of all objects(static field).
3. **class** Student9{
4. **int** rollno;
5. String name;
6. **static** String college = "ITS";
8. **static** **void** change(){
9. college = "BBDIT";
10. }
12. Student9(**int** r, String n){
13. rollno = r;
14. name = n;
15. }
17. **void** display (){System.out.println(rollno+" "+name+" "+college);}
19. **public** **static** **void** main(String args[]){
20. Student9.change();
22. Student9 s1 = **new** Student9 (111,"Karan");
23. Student9 s2 = **new** Student9 (222,"Aryan");
24. Student9 s3 = **new** Student9 (333,"Sonoo");
26. s1.display();
27. s2.display();
28. s3.display();
29. }
30. }

Output:111 Karan BBDIT

222 Aryan BBDIT

333 Sonoo BBDIT

Restrictions for 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;//non static
4. **public** **static** **void** main(String args[]){
5. System.out.println(a);
6. }
7. }

Output:Compile Time Error

Q) why java main method is static?

|  |
| --- |
| Ans) because object is not required to call static method if it were non-static method, jvm create 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 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. }

Output:static block is invoked

Hello main

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

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

1. **class** A3{
2. **static**{
3. System.out.println("static block is invoked");
4. System.exit(0);
5. }
6. }

Output:static block is invoked (if not JDK7)

In JDK7 and above, output will be:

Output:Error: Main method not found in class A3, please define the main method as:

public static void main(String[] args)