Static Keyword

- static keyword is mainly used for *memory management*.
- It can be used with
  - Variables
  - Methods
  - Blocks
  - Nested classes
- Basically, static is used for a constant variable or a method that is same for every instance of a class.
- The static variable can be used to refer to the common property of all objects.
- The static variable gets memory only once in the class area at the time of class loading.
- It makes your program memory efficient.
- Syntax

```
static type variablename;
```

STATIC VARIABLE

```
class Student
{
    int rollno;
    String name;
    String college="GCET";
}
```

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 Program to demonstrate the use of static variable
class Student{
  int rollno;//instance variable
  String name;
  static String college ="GCET";//static variable
  //constructor
  Student(int r, String n){
  rollno = r;
  name = n;
  }
  //method to display the values
  void display (){System.out.println(rollno+" "+name+" "+college);}
}
//Test class to show the values of objects
public class TestStaticVariable1{
 public static void main(String args[]){
 Student s1 = new Student(111,"Karan");
 Student s2 = new Student(222,"Aryan");
 //we can change the college of all objects by the single line of code
 //Student.college="GU";
 s1.display();
 s2.display();
 }
}
```

In the line **Student s1 = new Student(111, "Karan");**, there are two main components to understand:

1. **Student()** - This represents a call to the constructor of the **Student** class. It's used to create a new instance (or object) of the **Student** class. Here, **Student(111, "Karan")** is creating a new **Student** object with **111** as an roll no. and **"Karan"** as a name.
2. **s1** - This is an object reference variable. **s1** holds the memory address (or reference) to the newly created **Student** object. It doesn't contain the actual object data itself; rather, it points to the location in memory where the object data (in this case, the roll no and name of the student) is stored.

s1 is the object reference, while **Student()** is used to instantiate (create) the actual object in memory.

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

```
//Java Program to demonstrate the use of a static method.
class Student{
    int rollno;
    String name;
    static String college = "GCET";
    //static method to change the value of static variable
    static void change(){
    college = "GU";
    }
    //constructor to initialize the variable
    Student(int r, String n){
    rollno = r;
    name = n;
    }
    //method to display values
    void display(){System.out.println(rollno+" "+name+" "+college);}
}
//Test class to create and display the values of object
public class TestStaticMethod{
   public static void main(String args[]){
   Student.change();//calling change method
   //creating objects
   Student s1 = new Student(111,"Karan");
   Student s2 = new Student(222,"Aryan");
   Student s3 = new Student(333,"Sonoo");
   //calling display method
   s1.display();
   s2.display();
   s3.display();
   }
}
```

Another example of a static method that performs a normal calculation

```
//Java Program to get the cube of a given number using the static method
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);
 }
}
```

Restrictions for the static method

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

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

```
class A{
 int a=40;//non static

 public static void main(String args[]){
  System.out.println(a);
 }
}
```

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 class loading.

Example of static block

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