Static keyword in Java

Static is a keyword in Java which is used to declare static stuffs. It can be used to create variable, method block or a class.

Static variable or method can be accessed without instance of a class because it belongs to class. Static members are common for all the instances of the class but non-static members are different for each instance of the class. Let's study how it works with variables and methods.

Static Variables

Static variables defined as a class member can be accessed without object of that class. Static variable is initialized once and shared among different objects of the class. All the object of the class having static variable will have the same instance of static variable.

Note: Static variable is used to represent common property of a class. It saves memory.

Example:

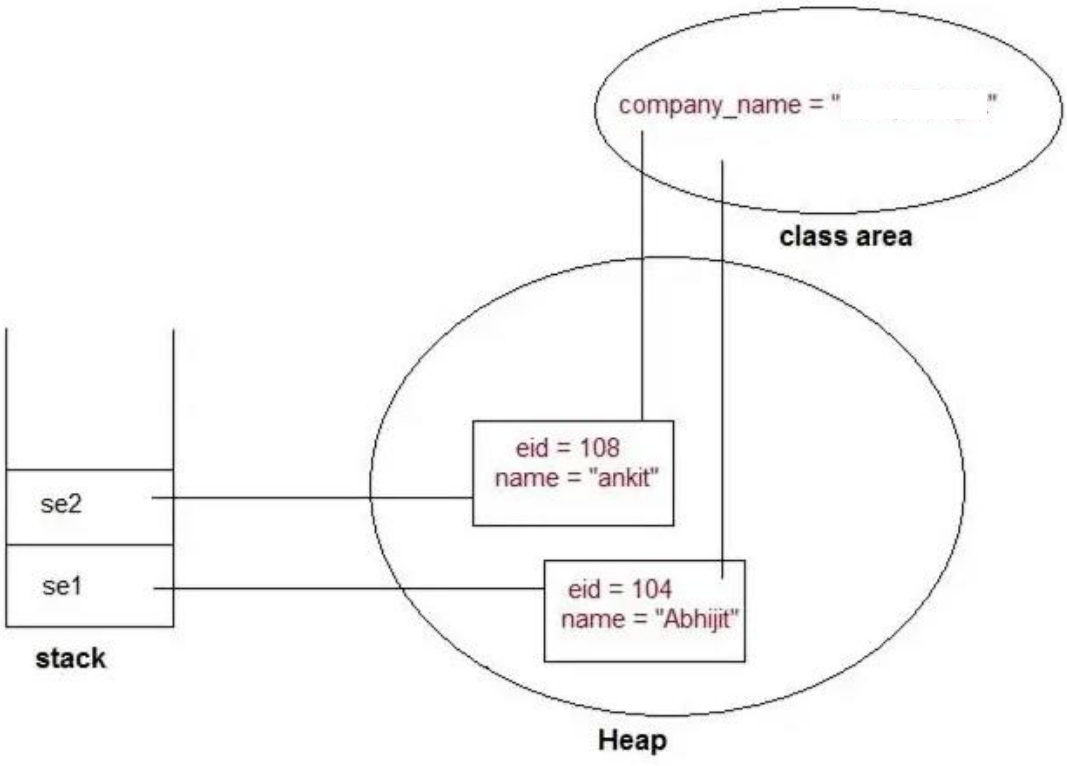
Suppose there are 100 employees in a company. All employees have their unique name and employee id but company name will be same for all 100 employee. Here company name is the common property. So, if you create a class to store employee detail, then declare company_name field as static

Below we have a simple class with one static variable, see the example.

```
class Employee{
        int eid;
        String name;
        static String company = "MyAnatomy";
        public void show()
        {
                System.out.println(eid + "-" + name + "-" + company);
        }
```

```
        public static void main( String[] args ){
                Employee se1 = new Employee();
                se1.eid = 104;
                se1.name = "Abhijit";
                se1.show();
                Employee se2 = new Employee();
                se2.eid = 108;
                se2.name = "ankit";
                se2.show();
                //System.out.println(company);
                //System.out.println(name);
        }
}
```

We can understand it using the below image that shows different memory areas used by the program and how static variable is shared among the objects of the class.



When we have a class variable like this, defined using the static keyword, then the variable is defined only once and is used by all the instances of the class. Hence if any of the class instance modifies it then it is changed for all the other instances of the class.

Static variable vs Instance variable

Here are some differences between static variable and instance variables. Instance variables are non-static variables that store into heap and accessed through object only.

Static variable	Instance Variable
Represent common property	Represent unique property
Accessed using class name (can be accessed using object name as well)	Accessed using object only
Allocated memory only once	Allocated new memory each time a new object is created

Example: static vs instance variables

Let's take an example and understand the difference.

```
public class Test
{
   static int x = 100;
   int y = 100;
   public void increment()
   {
```

```
    x++; y++;
  }
  public static void main( String[] args )
  {
    Test t1 = new Test();
    Test t2 = new Test();
    t1.increment();
    t2.increment();
    System.out.println(t2.y);
    System.out.println(Test.x);  //accessed without any instance of class.
  }
}
```

Output

101

102

See the difference in value of two variable. Static variable x shows the changes made to it by increment() method on the different object. While instance variable y show only the change made to it by increment() method on that particular instance.

Static Method in Java

A method can also be declared as static. Static methods do not need instance of its class for being accessed. main() method is the most common example of static method. main() method is declared as static because it is called before any object of the class is created.

Let's take an Example

```
class Test
{
```

```
    public static void square(int x)
    {
        System.out.println(x*x);
    }
    public static void main (String[] arg)
    {
        square(8);   //static method square () is called without any
instance of class.
    }
}
```

- **Static block in Java**

Static block is used to initialize static data members. Static block executes even before main() method.

It executes when the class is loaded in the memory. A class can have multiple Static blocks, which will execute in the same sequence in which they are programmed.

Example

Let’s see an example in which we declared a static block to initialize the static variable.

```
class ST_Employee
{
    int eid;
    String name;
    static String company_name;
    static {
        company_name ="MyAnatomy"; //static block invoked before
main() method
        //System.out.println("Hi");
```

```
	}
	public void show()
	{
		System.out.println(eid+" "+name+" "+company_name);
	}
	public static void main( String[] args )
	{
		ST_Employee se1 = new ST_Employee();
		se1.eid = 104;
		se1.name = "Abhijit";
		se1.show();
	}
}
```

Non-static (instance) variable cannot be referenced from a static context.

When we try to access a non-static variable from a static context like main method, java compiler throws an error message a non-static variable cannot be referenced from a static context. This is because non-static variables are related with instance of class(object) and they get created when instance of a class is created by using new operator. So, if we try to access a non-static variable without any instance, compiler will complain because those variables are not yet created and they don't have any existence until an instance is created and associated with it.

Example

Let's take an example of accessing non-static variable from a static context.

```
class Test
{
	int x;
```

```
    public static void main(String[] args)
    {
        x = 10;
    }
}
```

Output:

compiler error: non-static variable count cannot be referenced from a static context

Why main() method is static in java?

Because static methods can be called without any instance of a class and main() is called before any instance of a class is created.