Static Members: In Java, static members are those which belongs to the class and you can access these members without instantiating the class. The static keyword can be used with methods, fields, classes (inner/nested), blocks.

Static Methods − You can create a static method by using the keyword static. Static methods can access only static fields, methods. To access static methods there is no need to instantiate the class.

For Example:

```
class base
 {
static void sample()
   {
System.out.println("Hello");
   }
static void map()
   {
System.out.println("java program");
   }
static void by()
   {
System.out.println("core java program");
   }
public static void main(String args[])
   {
base.sample();
base.map();
base.by();
   }
}
```

Static Fields − You can create a static field by using the keyword static. The static fields have the same value in all the instances of the class. These are created and initialized when the class is loaded for the first time. Just like static methods you can access static fields using the class name (without instantiation).

For Example:

```
classMyClass
{
public static int d = 20;
public static void main(String args[])
   {
```

```
System.out.print("value of integer=");
System.out.println(MyClass.d);
   }
}
```

Static Blocks – These are a block of codes with a static keyword. In general, these are used to initialize the static members. JVM executes static blocks before the main method at the time of class loading.

For Example:

```
public class child {
static
   {
System.out.println("Hello this is a static block");
   }
public static void main(String args[])
   {
System.out.println(" ");
System.out.println("This is main method");
   }
}
```

Static class: We can declare a class static by using the static keyword. A class can be declared static only if it is a nested class. It does not require any reference of the outer class. The property of the static class is that it does not allows us to access the non-static members of the outer class.

To understand the concept of static class first we need to understand the concept of inner, outer, and nested class.

For Example:

```
class under
    {
  static class under1
{
    int a=12;
    int b=45;
int getvalue()
{
    return a+b;
}
}
```

```
    public static void main(String[] args)
    {
        under.under1 t=new under.under1();
        System.out.println("a+b="+t.getvalue());
        System.out.println("this is last block");
    }    }
```

Program:

```
public class last
    {
        static
        {
            System.out.println("hello");
        }
  static public class under1
{
     static
    {
System.out.println("hello-2");
    }
    int a=12;
    int b=45;
int getvalue()
{
    return a+b;
}
}
    public static void main(String[] args)
    {
        last.under1 t=new last.under1();
        System.out.println("a+b="+t.getvalue());
        System.out.println("this is last block");
          }


}
```