

What is a class variable OR static field ?

- * It is a class level variable that means we can declare this variable at class level.
- * If we declare a variable inside the class with static modifier then it is called class variable OR static field.
- * The life of static field will be started at the time of LOADING THE .CLASS FILE INTO JVM MEMORY
- * It will be initialized with default value in the PREPARE PHASE at the time of loading the .class file.
- * In order to access static field we need not to create an Object, we can directly access with the help of class name.

Role of static field while creating an object :

- * Unlike non static field, a single copy of static field will be created and the same single copy will be sharable by all the objects.

```
package com.ravi.oop;

public class Demo
{
    static int x = 10;

    public static void main(String[] args)
    {
        Demo d1 = new Demo();
        Demo d2 = new Demo();

        ++d1.x;  ++d2.x;

        System.out.println(d1.x); //12
        System.out.println(d2.x); //12
    }
}
```

So, final conclusion is :

- non static field : Multiple copies will be created with each and every object.
- static field : single copy of variable will be created and sharable by all the objects.

When we should declare a variable as a static field and we should declare as a non static field ?

static field :

If the **value of the variable** is **common** for all the objects then we should declare static field.

Non static field :

If the **value of the variable** is **different** with respect to object then we should use non static field.

Example 1 :

```
public class Student
{
    int roll;
    String name;
    String address;
    static String collegeName = "NIT";
    static String courseName = "Java";
}
```

Example 2 :

```
public class Customer
{
    int customerId;
    String customerName;
    static String bankLocation = "Ameerpet";
    static String ifscCode = "SBIHYD00014";
}
```

So the conclusion is : static field we should use to SAVE THE MEMORY.

```
//PROGRAM :
package com.ravi.oop;

//BLC
public class Student
{
    int rollNumber;
    String studentName;
    String studentAddress;
    static String collegeName = "NIT";
    static String courseName = "Java";

    public void setStudentData(int roll, String name, String address)
    {
        rollNumber = roll;
        studentName = name;
        studentAddress = address;
    }

    public void getStudentData()
    {
        System.out.println("Student Roll number is :"+rollNumber);
        System.out.println("Student Name is :"+studentName);
        System.out.println("Student Address is :"+studentAddress);
        System.out.println("College Name is :"+collegeName);
        System.out.println("Course Name is :"+courseName);
    }
}

package com.ravi.oop;

public class VariableDeclaration
{
    public static void main(String[] args)
    {
        Student raj = new Student();
        raj.setStudentData(111, "Raj", "Ameerpet");
        raj.getStudentData();

        System.out.println(".....");
        Student priya = new Student();
        priya.setStudentData(222, "Priya", "S R Nagar");
        priya.getStudentData();
    }
}
```

*** What is Data Hiding in java ?

- * If we declare our non static field with **private access modifier** then it is called Data Hiding.
- * Once we declare our non static field with **private access modifier** then the data will not be visible to the outer world that means we can only access the data within the class.
- * **We can validate the outer world data which we are accepting through parameter Variable.**

```
package com.ravi.data_hiding;

public class Customer
{
    private double balance = 10000; //Data hiding

    public void deposit(double amount)
    {
        //Validation of Data
        if(amount <=0)
        {
            System.err.println("Amount cannot be deposited");
        }
        else
        {
            balance = balance + amount;
            System.out.println("After deposit balance is :"+balance);
        }
    }

    public void withdraw(double amount)
    {
        if(amount > balance)
        {
            System.err.println("Sorry!! Insufficient Balance");
        }
        else
        {
            balance = balance - amount;
            System.out.println("After withdraw balance is :"+balance);
        }
    }
}

package com.ravi.data_hiding;

public class BankApplication {

    public static void main(String[] args)
    {
        Customer raj = new Customer();
        raj.deposit(10000);
        raj.withdraw(15000);
    }
}
```

What is access modifier ? How many types of access modifiers we have in java ?

- * An access modifier defines the **accessibility level of a class and the member of the class** (Fields + Methods).
- * In java we have so many modifiers but we have only **4 types of access modifiers** which are as follows :
- a) private (Within the same class only)
- b) default (Within the same package)
- c) protected (Within the same package as well as from another package by using Inheritance)
- d) public (No restriction, can be accessible from everywhere)