

C# static

In C#, static is a keyword or modifier that belongs to the type not instance. So instance is not required to access the static members. In C#, static can be field, method, constructor, class, properties, operator and event.

Note: Indexers and destructors cannot be static.

Advantage of C# static keyword

Memory efficient: Now we don't need to create instance for accessing the static members, so it saves memory. Moreover, it belongs to the type, so it will not get memory each time when instance is created.

C# Static Field

A field which is declared as static, is called static field. Unlike instance field which gets memory each time whenever you create object, there is only one copy of static field created in the memory. It is shared to all the objects.

It is used to refer the common property of all objects such as `rateOfInterest` in case of `Account`, `companyName` in case of `Employee` etc.

C# static field example

Let's see the simple example of static field in C#.

```
using System;

public class Account
{
    public int accno;
    public String name;
    public static float rateOfInterest=8.8f;
    public Account(int accno, String name)
    {
        this.accno = accno;
        this.name = name;
    }

    public void display()
    {
        Console.WriteLine(accno + " " + name + " " + rateOfInterest);
    }
}

class TestAccount{
```

```
public static void Main(string[] args)
{
    Account a1 = new Account(101, "Sonoo");
    Account a2 = new Account(102, "Mahesh");
    a1.display();
    a2.display();

}
}
```

Output:

```
101 Sonoo 8.8
102 Mahesh 8.8
```