**Static Class**

C# provides the important feature to create static classes, there are two main features of a static class, one is no object of static class can be created and another is, a static class must contain only static members, then it is important that what is the main benefit to create a static class, the main benefit of making static class, we do not need to make any instance of this class ,all members can be accessible with its own name.

**Declaration:**

A static class is created by using keyword 'Static' as shown here:

One more thing that is notable-within static class, all members must be explicitly specified as static, static class does not automatically make its members static. Static class can contain a collection of static methods.

**Example:**

static class Shape

{

    public static double GetArea(double Width, double height)

    {

        return Width \* Height;

    }

}

class Ractangle

{

    private void GetRactangleArea()

    {

        Double Area;

        Area = Shape.GetArea(10, 5);

    }

}

Shape is static class, it contain staic function GetArea.Ractangle is other class and with in GetArea function can be access without creating instace of Class Shape.

Although a static class cannot have an instance constructor, it can have a static constructor.

As is the case with all class types, the type information for a static class is loaded by the .NET Framework common language runtime (CLR) when the program that references the class is loaded. The program cannot specify exactly when the class is loaded. However, it is guaranteed to be loaded and to have its fields initialized and its static constructor called before the class is referenced for the first time in your program. **A static constructor is only called one time, and a static class remains in memory for the lifetime of the application domain in which your program resides.**

The following list provides the main features of a static class:

* Contains only static members.
* Cannot be instantiated.
* Is sealed.
* Cannot contain [Instance Constructors](https://msdn.microsoft.com/en-in/library/k6sa6h87.aspx).

Creating a static class is therefore basically the same as creating a class that contains only static members and a private constructor. **A private constructor prevents the class from being instantiated**. The **advantage** of using a static class is that **the compiler can check to make sure that no instance members are accidentally added**. The compiler will guarantee that instances of this class cannot be created

Static classes are **sealed** and therefore **cannot be inherited**. They cannot inherit from any class **except**[**Object**](https://msdn.microsoft.com/en-in/library/system.object.aspx). Static classes cannot contain an instance constructor; however, they can contain a static constructor. **Non-static classes should also define a static constructor if the class contains static members that require non-trivial initialization**

[Static Members](javascript:void(0))

**A non-static class can contain static methods, fields, properties, or events**. **The static member is callable on a class even when no instance of the class has been created.** The static member is always accessed by the class name, not the instance name. **Only one copy of a static member exists**, regardless of how many instances of the class are created. **Static methods and properties cannot access non-static fields and events in their containing type**, and they cannot access an instance variable of any object unless it is explicitly passed in a method parameter.

public class Automobile

{

public static int NumberOfWheels = 4;

public static int SizeOfGasTank

{

get

{

return 15;

}

}

public static void Drive() { }

public static event EventType RunOutOfGas;

// Other non-static fields and properties...

}

Static members are initialized before the static member is accessed for the first time and before the static constructor, if there is one, is called. To access a static class member, use the name of the class instead of a variable name to specify the location of the member, as shown in the following example:

Automobile.Drive();

int i = Automobile.NumberOfWheels;