Static classes and class members are used to create data and functions that can be accessed without creating an instance of the class. Static class members can be used to separate data and behavior that is independent of any object identity: the data and functions do not change regardless of what happens to the object. Static classes can be used when there is no data or behavior in the class that depends on object identity.

**Static Classes**

A class can be declared [static](https://msdn.microsoft.com/en-us/library/98f28cdx%28v=vs.80%29.aspx), indicating that it contains only static members. It is not possible to create instances of a static class using the [new](https://msdn.microsoft.com/en-us/library/51y09td4%28v=vs.80%29.aspx) keyword. Static classes are loaded automatically by the .NET Framework common language runtime (CLR) when the program or namespace containing the class is loaded.

Use a static class to contain methods that are not associated with a particular object. For example, it is a common requirement to create a set of methods that do not act on instance data and are not associated to a specific object in your code. You could use a static class to hold those methods.

The main features of a static class are:

* They only contain static members.
* They cannot be instantiated.
* They are sealed.
* They cannot contain [Instance Constructors (C# Programming Guide)](https://msdn.microsoft.com/en-us/library/k6sa6h87%28v=vs.80%29.aspx).

Creating a static class is therefore much 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. Static classes cannot contain a constructor, although it is still possible to declare a static constructor to assign initial values or set up some static state. For more information, see [Static Constructors (C# Programming Guide)](https://msdn.microsoft.com/en-us/library/k9x6w0hc%28v=vs.80%29.aspx).