Stack Class

Namespace: System.Collections

Assemblies: System.Collections.NonGeneric.dll, mscorlib.dll, netstandard.dll

Represents a simple last-in-first-out (LIFO) non-generic collection of objects.

In this article

C# Copy

```
[System.Runtime.InteropServices.ComVisible(true)]
[System.Serializable]
public class Stack : ICloneable, System.Collections.ICollection
```

Inheritance Object → Stack

Attributes ComVisibleAttribute, SerializableAttribute

Implements ICollection, IEnumerable, ICloneable

Examples

The following example shows how to create and add values to a Stack and how to display its values.

C# Copy

```
using System;
using System.Collections;
public class SamplesStack  {

   public static void Main()  {

      // Creates and initializes a new Stack.
      Stack myStack = new Stack();
      myStack.Push("Hello");
      myStack.Push("World");
      myStack.Push("!");

      // Displays the properties and values of the Stack.
      Console.WriteLine( "myStack" );
      Console.WriteLine( "\tCount:    {0}", myStack.Count );
      Console.Write( "\tValues:" );
      PrintValues( myStack );
   }

   public static void PrintValues( IEnumerable myCollection )  {
      foreach ( Object obj in myCollection )
         Console.Write( "    {0}", obj );
      Console.WriteLine();
   }

}


/*
This code produces the following output.

myStack
    Count:    3
    Values:    !    World    Hello
*/
```

Remarks

The capacity of a Stack is the number of elements the Stack can hold. As elements are added to a Stack, the capacity is automatically increased as required through reallocation.

> **ⓘ Important**
>
> We don't recommend that you use the `Stack` class for new development. Instead, we recommend that you use the generic System.Collections.Generic.Stack<T> class. For

more information, see **Non-generic collections shouldn't be used** on GitHub.

If Count is less than the capacity of the stack, Push is an O(1) operation. If the capacity needs to be increased to accommodate the new element, Push becomes an O(`n`) operation, where `n` is Count. Pop is an O(1) operation.

Stack accepts `null` as a valid value and allows duplicate elements.

Constructors

Stack()	Initializes a new instance of the Stack class that is empty and has the default initial capacity.
Stack(ICollection)	Initializes a new instance of the Stack class that contains elements copied from the specified collection and has the same initial capacity as the number of elements copied.
Stack(Int32)	Initializes a new instance of the Stack class that is empty and has the specified initial capacity or the default initial capacity, whichever is greater.

Properties

Count	Gets the number of elements contained in the Stack.
IsSynchronized	Gets a value indicating whether access to the Stack is synchronized (thread safe).
SyncRoot	Gets an object that can be used to synchronize access to the Stack.

Methods

Clear()	Removes all objects from the Stack.
Clone()	Creates a shallow copy of the Stack.

Contains(Object)	Determines whether an element is in the Stack.
CopyTo(Array, Int32)	Copies the Stack to an existing one-dimensional Array, starting at the specified array index.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetEnumerator()	Returns an IEnumerator for the Stack.
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Peek()	Returns the object at the top of the Stack without removing it.
Pop()	Removes and returns the object at the top of the Stack.
Push(Object)	Inserts an object at the top of the Stack.
Synchronized(Stack)	Returns a synchronized (thread safe) wrapper for the Stack.
ToArray()	Copies the Stack to a new array.
ToString()	Returns a string that represents the current object. (Inherited from Object)

Extension Methods

Cast<TResult> (IEnumerable)	Casts the elements of an IEnumerable to the specified type.
OfType<TResult> (IEnumerable)	Filters the elements of an IEnumerable based on a specified type.

AsParallel(IEnumerable)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.

Applies to

.NET Core

3.0 Preview 2, 2.2, 2.1, 2.0, 1.1, 1.0

.NET Framework

4.8, 4.7.2, 4.7.1, 4.7, 4.6.2, 4.6.1, 4.6, 4.5.2, 4.5.1, 4.5, 4.0, 3.5, 3.0, 2.0, 1.1

.NET Standard

2.0

UWP

10.0

Xamarin.Android

7.1

Xamarin.iOS

10.8

Xamarin.Mac

3.0

Thread Safety

Public static (`Shared` in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

To guarantee the thread safety of the Stack, all operations must be done through the wrapper returned by the Synchronized(Stack) method.

Enumerating through a collection is intrinsically not a thread-safe procedure. Even when a collection is synchronized, other threads can still modify the collection, which causes the enumerator to throw an exception. To guarantee thread safety during enumeration, you can either lock the collection during the entire enumeration or catch the exceptions resulting from changes made by other threads.

See also

- Stack<T>