

1 System.Collections.Generic.Stack<T>.Enumer 2 ator Structure

```
3 [ILAsm]  
4 .class ansi serializable sealed nested public beforefieldinit  
5 Stack<T>.Enumerator extends System.ValueType implements  
6 System.Collections.Generic.IEnumerator`1<!0>, System.IDisposable,  
7 System.Collections.IEnumerator  
  
8 [C#]  
9 public struct Stack<T>.Enumerator:  
10 System.Collections.Generic.IEnumerator<T>
```

11 Assembly Info:

- 12 • *Name:* System
- 13 • *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00]
- 14 • *Version:* 4.0.0.0
- 15 • *Attributes:*
 - 16 ○ CLSCompliantAttribute(true)

17 Implements:

- 18 • System.Collections.Generic.IEnumerator<T>

19 Summary

20 Enumerates the elements of a Stack<T>

21 Inherits From: System.ValueType

22

23 **Library:** BCL

24 Usage

25 For a detailed description regarding the use of an enumerator, see
26 System.Collections.Generic.IEnumerator<T>.

27

1 Stack<T>.Enumerator.Dispose() Method

```
2 [ILAsm]  
3 .method public hidebysig newslot virtual final instance void Dispose() cil  
4 managed  
5 [C#]  
6 public void Dispose ()
```

7 Summary

8 Releases all resources used by the Stack(Of T).Enumerator.

9

1 Stack<T>.Enumerator.MoveNext() Method

```
2 [ILAsm]  
3 .method public hidebysig newslot virtual final instance bool MoveNext()  
4 cil managed  
  
5 [C#]  
6 public bool MoveNext ( )
```

7 Summary

8 Advances the enumerator to the next element of the Stack(Of T).

9 Return Value

10 A System.Boolean that is true if the enumerator was successfully advanced to the next
11 element; false if the enumerator has passed the end of the collection.

12 Description

13 After an enumerator is created, the enumerator is positioned before the first element in
14 the collection, and the first call to
15 System.Collections.Generic.Stack<T>.Enumerator.MoveNext advances the
16 enumerator to the first element of the collection.
17

18 If System.Collections.Generic.Stack<T>.Enumerator.MoveNext passes the end of
19 the collection, the enumerator is positioned after the last element in the collection and
20 System.Collections.Generic.Stack<T>.Enumerator.MoveNext returns false.
21 When the enumerator is at this position, subsequent calls to
22 System.Collections.Generic.Stack<T>.Enumerator.MoveNext also return false.
23

24 An enumerator remains valid as long as the collection remains unchanged. If changes
25 are made to the collection, such as adding, modifying, or deleting elements, the
26 enumerator is irrecoverably invalidated and its behavior is undefined.

27 Exceptions

Exception	Condition
System.InvalidOperationException	The collection was modified after the enumerator was created.

28

29

1

2 Stack<T>.Enumerator.System.Collections.IEnumerator.Reset() Method

```

4 [ILAsm]
5 .method private hidebysig newslot virtual final instance void
6 System.Collections.IEnumerator.Reset() cil managed
7
8 [C#]
9 void IEnumerator.Reset ()

```

9 Summary

10 Sets the enumerator to its initial position, which is before the first element in the
11 collection.

12 Description

13 An enumerator remains valid as long as the collection remains unchanged. If changes
14 are made to the collection, such as adding, modifying, or deleting elements, the
15 enumerator is irrecoverably invalidated and the next call to
16 System.Collections.IEnumerator.MoveNext or
17 System.Collections.IEnumerator.Reset throws an
18 System.InvalidOperationException.

19 Exceptions

Exception	Condition
System.InvalidOperationException	The collection was modified after the enumerator was created.

20

21

1 Stack<T>.Enumerator.Current Property

```
2 [ILAsm]  
3 .property instance !0 Current  
4 [C#]  
5 public T Current { get; }
```

6 Summary

7 Gets the element at the current position of the enumerator.

8 Property Value

9 Type: T, the element in the Stack(Of T) at the current position of the enumerator.

10 Description

11 Current is undefined under any of the following conditions:

12

13 1. The enumerator is positioned before the first element in the collection, immediately
14 after the enumerator is created. MoveNext must be called to advance the enumerator to
15 the first element of the collection before reading the value of Current.

16

17 2. The last call to MoveNext returned false, which indicates the end of the collection.

18

19 Current returns the same object until MoveNext is called. MoveNext sets Current to the
20 next element. If the collection is modified between MoveNext and Current, Current
21 returns the element that it is set to, even if the enumerator is already invalidated.

22

Stack<T>.Enumerator.System.Collections.IEnumerator.Current Property

```
[ILAsm]  
.property instance object System.Collections.IEnumerator.Current  
  
[C#]  
object System.Collections.IEnumerator.Current { get; }
```

Summary

Gets the element at the current position of the enumerator.

Property Value

The element in the collection at the current position of the enumerator.

Description

After an enumerator is created or after a `System.Collections.IEnumerator.Reset` is called, `System.Collections.IEnumerator.MoveNext` must be called to advance the enumerator to the first element of the collection before reading the value of `System.Collections.IEnumerator.Current`; otherwise, `System.Collections.IEnumerator.Current` is undefined.

`System.Collections.IEnumerator.Current` also throws an exception if the last call to `System.Collections.IEnumerator.MoveNext` returned `false`, which indicates the end of the collection.

`System.Collections.IEnumerator.Current` does not move the position of the enumerator, and consecutive calls to `System.Collections.IEnumerator.Current` return the same object until either `System.Collections.IEnumerator.MoveNext` or `System.Collections.IEnumerator.Reset` is called.

An enumerator remains valid as long as the collection remains unchanged. If changes are made to the collection, such as adding, modifying, or deleting elements, the enumerator is irrecoverably invalidated and the next call to `System.Collections.IEnumerator.MoveNext` or `System.Collections.IEnumerator.Reset` throws an `System.InvalidOperationException`. If the collection is modified between `System.Collections.IEnumerator.MoveNext` and `System.Collections.IEnumerator.Current`, `System.Collections.IEnumerator.Current` returns the element that it is set to, even if the enumerator is already invalidated.

Exceptions

Exception	Condition
-----------	-----------

System.InvalidOperationException

The enumerator is positioned before the first element of the collection or after the last element.

1

2