

System.Nullable<T> Structure

```
[ILAsm]
.class public sequential sealed serializable Nullable`1< valuetype
(System.ValueType) T> extends System.ValueType

[C#]
public struct Nullable<T> where T: struct
```

Assembly Info:

- *Name:* mscorlib
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- *Version:* 2.0.x.x
- *Attributes:*
 - CLSCompliantAttribute(true)

Summary

Represents a nullable value type. An instance of `System.Nullable<T>` can contain a value of type `T` or an indication that the instance contains no value. Upon boxing, if it contains no value, it will be converted to the null reference; otherwise, it will be converted to a boxed `T`. [Note: Because of the constraint on the generic parameter, `T` cannot be of type `System.Nullable<U>` for any `U`. end note]

Inherits From: System.ValueType

Library: BCL

Thread Safety: This type is not guaranteed to be safe for multithreaded operations.

Description

The `System.Nullable<T>` value type represents a value of a given type `T` or an indication that the instance contains no value. Such a nullable type is useful in a variety of situations, such as in denoting nullable columns in a database table or optional attributes in an XML element. The runtime transforms `System.Nullable<T>` instances without values into true nulls when performing a box operation; instances with values are transformed into boxed `T`'s containing the `System.Nullable<T>`'s `Value`.

An instance of `System.Nullable<T>` has two properties, `System.Nullable<T>.HasValue` and `System.Nullable<T>.Value`. `System.Nullable<T>.HasValue` is used to determine whether the current instance currently has a value. It returns `true` or `false`, and never throws an exception. `System.Nullable<T>.Value` returns the current value of the instance, provided it has one (i.e., `System.Nullable<T>.HasValue` is `true`); otherwise, it throws an exception.

In addition to the above properties, there is a pair of methods, both overloads of `System.Nullable<T>.GetValueOrDefault`. The version taking no arguments returns the instance's current value, if it has one; otherwise, it returns the default value of type `T`. The version taking an argument of type `T` returns the instance's current value, if it has one; otherwise, it returns the default value argument passed to it.

Applying `System.Nullable<T>.HasValue` to an instance that has the default initial value, causes `false` to be returned.

Nullable<T>(T) Constructor

```
[ILAsm]  
public rtspecialname specialname instance void .ctor(!0 value)  
  
[C#]  
public Nullable(T value)
```

Summary

Constructs and initializes a new instance of `System.Nullable<T>` giving it the specified initial value.

Parameters

Parameter	Description
<i>value</i>	The initial value of the new instance.

Description

[*Note:* Once this constructor has executed, applying `System.Nullable<T>.HasValue` to the new instance returns `true`.]

Nullable<T>.Equals(System.Object) Method

```
[ILAsm]  
.method public hidebysig virtual bool Equals(object obj)  
  
[C#]  
public override bool Equals(object obj)
```

Summary

Determines whether the current instance and the specified `System.Object` represent the same type and value.

Parameters

Parameter	Description
<i>obj</i>	The <code>System.Object</code> to compare to the current instance.

Return Value

The following table defines the conditions under which the return value is true or false:

Returned Value	HasValue Condition	<code>obj.HasValue</code> Condition
false	The current instance and <i>obj</i> have different types.	The current instance and <i>obj</i> have different types.
true	false	<i>obj</i> is null.
false	true	<i>obj</i> is null.
true	false	false
false	false	true
false	true	false
<code>Value.Equals(obj.Value)</code>	true	true

1 **Description**

2 *[Note: This method overrides `System.Object.Equals.`]*

3

4

5

Nullable<T>.FromNullable(System.Nullable<T>) Method

```
[ILAsm]  
.method public hidebysig static !0 FromNullable(valuetype  
System.Nullable`1<!0> value)  
  
[C#]  
public static T FromNullable(Nullable<T> value)
```

Summary

Creates a T from a System.Nullable<T>.

Parameters

Parameter	Description
<i>value</i>	The System.Nullable<T> value to convert to type T.

Return Value

The value, if any, of the specified nullable value. Otherwise, a System.InvalidOperationException is thrown.

Description

[Note: This method corresponds to the Op_Explicit method.]

Exceptions

Exception	Condition
System.InvalidOperationException	System.Nullable<T>.HasValue is false.

1 **Nullable<T>.GetHashCode() Method**

```
2    [ILAsm]  
3    .method public hidebysig virtual int32 GetHashCode()  
  
4    [C#]  
5    public override int GetHashCode()
```

6 **Summary**

7 Generates a hash code for the current instance.

8 **Return Value**

9
10 If `System.Nullable<T>.HasValue` is true, a `System.Int32` containing the hash code for
11 the value of the current instance is returned; otherwise, 0 is returned.

12 **Description**

13 The algorithm used to generate the hash code is unspecified.

14
15 [*Note:* This method overrides `System.Object.GetHashCode.`]
16
17
18

1 **Nullable<T>.GetValueOrDefault() Method**

```
2    [ILAsm]  
3    .method public hidebysig !0 GetValueOrDefault()  
  
4    [C#]  
5    public T GetValueOrDefault()
```

6 **Summary**

7 Returns the value of the current instance, or if it has none, returns the default value for
8 the type T.

9 **Return Value**

11 A value of type T, which is either the value of the current instance, or if it has none, the
12 default value for the type T (i.e., all-bits-zero).

13 **Description**

14 [Note: System.Nullable<T>.GetValueOrDefault(T) allows a value other than the
15 default value to be returned if the current instance contains no value.]
16
17
18

1 **Nullable<T>.GetValueOrDefault(T) Method**

```
2    [ILAsm]  
3    .method public hidebysig !0 GetValueOrDefault(!0 alternateDefaultValue)  
  
4    [C#]  
5    public T GetValueOrDefault(T alternateDefaultValue)
```

6 **Summary**

7 Returns the value of the current instance, or if it has none, returns
8 *alternateDefaultValue*.

9 **Parameters**

Parameter	Description
<i>alternateDefaultValue</i>	The value to be returned if the current instance contains no value.

13 **Return Value**

15 A value of type *T*, which is either the value of the current instance, or if it has none, the
16 value of *alternateDefaultValue*.

17 **Description**

18 [Note: `System.Nullable<T>.GetValueOrDefault()` allows the default value for type *T*
19 to be returned if the current instance contains no value.]
20
21
22

Nullable<T>.op_Explicit(System.Nullable<T>) Method

```
[ILAsm]  
.method public hidebysig static specialname !0 op_Explicit(valuetype  
System.Nullable`1<!0> value)  
  
[C#]  
public static explicit operator T(Nullable<T> value)
```

Summary

Perform an explicit conversion of a `System.Nullable<T>` value to type `T`.

Parameters

Parameter	Description
<i>value</i>	The <code>System.Nullable<T></code> value to convert to type <code>T</code> .

Return Value

The value, if any, of the specified nullable value. Otherwise, a `System.InvalidOperationException` is thrown.

Description

[*Note:* The conversion implemented by this method corresponds exactly to obtaining the value of the `System.Nullable<T>.Value` property.]

Exceptions

Exception	Condition
System.InvalidOperationException	<code>System.Nullable<T>.HasValue</code> is false.

- 1
- 2
- 3

Nullable<T>.op_Implicit(T) Method

```
[ILAsm]  
.method public hidebysig static specialname valuetype Nullable`1<!0>  
op_Implicit(!0 value)  
  
[C#]  
public static implicit operator Nullable<T>(T value)
```

Summary

Perform an implicit conversion of a T value to System.Nullable<T>.

Parameters

Parameter	Description
<i>value</i>	The T value to convert to System.Nullable<T>.

Return Value

A System.Nullable<T> with the specified value.

Description

[*Note:* The conversion implemented by this method corresponds exactly to invoking the System.Nullable<T>(T) constructor.]

1 **Nullable<T>.ToNullable(T) Method**

```
2    [ILAsm]  
3    .method public hidebysig static valuetype Nullable`1<!0> ToNullable(!0  
4    value)  
  
5    [C#]  
6    public static Nullable<T> ToNullable(T value)
```

7 **Summary**

8 Creates a System.Nullable<T> from a T.

9 **Parameters**

Parameter	Description
<i>value</i>	The T value to convert to System.Nullable<T>.

12 **Return Value**

13 A System.Nullable<T> with the specified value.

14 **Description**

15 [Note: This method corresponds to the Op_Implicit method.]
16
17
18
19
20

1 **Nullable<T>.ToString() Method**

```
2    [ILAsm]  
3    .method public hidebysig virtual string ToString()  
  
4    [C#]  
5    public override string ToString()
```

6 **Summary**

7 Returns a `System.String` representation of the value of the current instance.

8 **Return Value**

9
10 If `System.Nullable<T>.HasValue` is `true`, `System.Nullable<T>.Value.ToString()` is
11 returned; otherwise, `System.String.Empty` is returned.

12 **Description**

13 [*Note:* This method overrides `System.Object.ToString`.]
14
15
16

1 **Nullable<T>.HasValue Property**

```
2    [ILAsm]  
3    .property bool HasValue { public hidebysig virtual abstract specialname  
4    bool get_HasValue() }  
  
5    [C#]  
6    public bool HasValue { get; }
```

7 **Summary**

8 Indicates whether the current instance contains a value.

9 **Property Value**

11 true if the current instance contains a value; otherwise false.

12 **Behaviors**

13 If System.Nullable<T>.HasValue is true, the instance contains a value, and
14 System.Nullable<T>.Value returns that value.

15
16 If System.Nullable<T>.HasValue is false, the instance contains no value, and an
17 attempt to read System.Nullable<T>.Value results in a
18 System.InvalidOperationException exception. A call to
19 System.Nullable<T>.Value.GetValueOrDefault returns the default value.

20
21 This property is read-only.

Nullable<T>.Value Property

```
[ILAsm]  
.property !0 Value { public hidebysig virtual abstract specialname !0  
get_Value() }  
  
[C#]  
public T Value { get; }
```

Summary

Gets the value, if any, of the current instance.

Property Value

The value of the current instance.

Behaviors

If `System.Nullable<T>.HasValue` is true, the instance contains a value, and `System.Nullable<T>.Value` returns that value.

If `System.Nullable<T>.HasValue` is false, the instance contains no value, and an attempt to read `System.Nullable<T>.Value` results in an exception.

This property is read-only.

Exceptions

Exception	Condition
<code>System.InvalidOperationException</code>	<code>System.Nullable<T>.HasValue</code> is false.