

System.Collections.Comparer Class

```
[ILASM]
.class public sealed serializable Comparer extends
System.Object implements System.Collections.IComparer

[C#]
public sealed class Comparer: IComparer
```

Assembly Info:

- *Name:* mscorlib
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00]
- *Version:* 1.0.x.x
- *Attributes:*
 - CLSCompliantAttribute(true)

Implements:

- **System.Collections.IComparer**

Summary

Provides the default implementation of the **System.Collections.IComparer** interface.

Inherits From: System.Object

Library: BCL

Thread Safety: All public static members of this type are safe for multithreaded operations. No instance members are guaranteed to be thread safe.

1 Comparer.Default Field

```
2 [ILASM]  
3 .field public static initOnly class  
4 System.Collections.Comparer Default  
  
5 [C#]  
6 public static readonly Comparer Default
```

7 Summary

8 Returns a new **System.Collections.Comparer** instance containing
9 the default implementation of the **System.Collections.IComparer**
10 interface.

11 Description

12 This field is read-only.

13

Comparer.Compare(System.Object, System.Object) Method

```
[ILASM]
.method public final hidebysig virtual int32 Compare(object
a, object b)

[C#]
public int Compare(object a, object b)
```

Summary

Returns the sort order of two **System.Object** instances.

Parameters

Parameter	Description
<i>a</i>	The first System.Object to compare.
<i>b</i>	The second System.Object to compare.

Return Value

A **System.Int32** containing a value that reflects the sort order of *a* as compared to *b*. The following table defines the conditions under which the returned value is a negative number, zero, or a positive number.

Value	Condition
Any negative number	$a < b$.
Zero	$a == b$.
Any positive number	$a > b$.

[Note: A null reference is considered to compare less than any other non-null object, and equal to any other null reference, independent of the underlying **System.Type** of either object.]

Description

The behavior of this method is as follows:

- If *a* implements the **System.IComparable** interface, returns *a.CompareTo(b)*.
- If *a* does not implement the **System.IComparable** interface but *b* does, returns *b.CompareTo(a)*.

- 1
- 2
- 3
- If *a* and *b* both are not **null** and do not implement the **System.IComparable** interface, **System.ArgumentException** is thrown.

4 **Exceptions**

5

6

Exception	Condition
System.ArgumentException	Both <i>a</i> and <i>b</i> are not null and do not implement the System.IComparable interface. -or- Both <i>a</i> and <i>b</i> are not null and are not assignment-compatible types.

7

8