

# System.Collections.IComparer Interface

```
[ILASM]
.class interface public abstract IComparer

[C#]
public interface 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)

## Summary

Provides a mechanism to customize the sort ordering of a collection.

**Library:** BCL

## Description

The default implementation of this interface is **System.Collections.Comparer**.

[*Note:* **System.Collections.IComparer** contains the **System.Collections.IComparer.Compare** method. The consumer of an object should call this method when sorting members of a collection.]

# IComparer.Compare(System.Object, System.Object) Method

```
[ILASM]
.method public hidebysig virtual abstract int32
Compare(object x, object y)

[C#]
int Compare(object x, object y)
```

## Summary

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

## Parameters

Parameter	Description
x	First <b>System.Object</b> to compare.
y	Second <b>System.Object</b> to compare.

## Return Value

A **System.Int32** containing a value that reflects the sort order of x as compared to y. 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	$x < y$ .
Zero	$x == y$ .
Any positive number	$x > y$ .

## Description

## Behaviors

For any objects A, B, and C, the following are required to be true:

**System.Collections.IComparer.Compare** (A, A) is required to return zero.

If **System.Collections.IComparer.Compare**(A, B) returns zero, then **System.Collections.IComparer.Compare** (B, A) is required to return zero.

1  
2 If **System.Collections.IComparer.Compare**(A, B) returns zero and  
3 **System.Collections.IComparer.Compare**(B, C) returns zero then  
4 **System.Collections.IComparer.Compare** (A, C) is required to  
5 return zero.  
6

7 If **System.Collections.IComparer.Compare**(A, B) returns a value  
8 other than zero, then **System.Collections.IComparer.Compare** (B,  
9 A) is required to return a value of the opposite sign.  
10

11 If **System.Collections.IComparer.Compare**(A, B) returns a value x  
12 not equal to zero, and **System.Collections.IComparer.Compare**(B,  
13 C) returns a value y of the same sign as x, then  
14 **System.Collections.IComparer.Compare** (A, C) is required to  
15 return a value of the same sign as x and y.  
16

17 [*Note:* The exact ordering of this method is unspecified. The intent of  
18 the method is to provide a mechanism that orders instances of a class  
19 in a manner that is consistent with the mathematical definitions of the  
20 relational operators (<, >, and ==), without regard for class-specific  
21 definitions of the operators.]

## 22 Usage

23 This interface is used in conjunction with the **System.Array.Sort** and  
24 **System.Array.BinarySearch** methods.  
25