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 00]
- *Version:* 2.0.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.

Comparer.Default Field

[ILAsm]
```
.field public static initOnly class System.Collections.Comparer
Default
```

[C#]
```
public static readonly Comparer Default
```

Summary

Returns a new `System.Collections.Comparer` instance containing the default implementation of the `System.Collections.IComparer` interface.

Description

This field is read-only.

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
a	The first `System.Object` to compare.
b	The second `System.Object` to compare.

Return Value

The return value is a negative number, zero, or a positive number reflecting the sort order of *a* as compared to *b*. For non-zero return values, the exact value returned by this method is unspecified. The following table defines the return value:

Value	Condition
A negative number	*a* < *b*.
Zero	*a* == *b*.
A 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 the negated result of *b*.CompareTo(*a*).

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

Exceptions

Exception	Condition
System.ArgumentException	Both *a* and *b* are not `null` and do not implement the `System.IComparable` interface. -or- Both *a* and *b* are not `null` and are not assignment-compatible types.