

# System.Char Structure

```
[ILAsm]
.class public sequential sealed serializable Char extends System.ValueType
implements System.IComparable, System.IComparable`1<char>,
System.IEquatable`1<char>

[C#]
public struct Char: IComparable, IComparable<Char>, IEquatable<Char>
```

## Assembly Info:

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

## Implements:

- **System.IComparable**
- **System.IComparable<System.Char>**
- **System.IEquatable<System.Char>**

## Summary

Represents a Unicode character.

## Inherits From: System.ValueType

**Library:** BCL

**Thread Safety:** This type is safe for multithreaded operations.

## Description

The `System.Char` value type represents Unicode characters, with code points ranging from 0 to 65,535.

[*Note:* The *code point* of a Unicode character is that character's 2-byte, encoded value.]

[*Note:* The `System.Globalization.UnicodeCategory` enumeration describes the categories that a Unicode character can be mapped to. For information on mapping specific Unicode characters to Unicode categories, see the `UnicodeData.txt` file in the Unicode Character Database at <http://www.unicode.org/Public/UNIDATA/UnicodeCharacterDatabase.html>. The `UnicodeData.txt` file format is described at <http://www.unicode.org/Public/3.1->

1 Update/UnicodeData-3.1.0.html.]  
2  
3

4

# Char.MaxValue Field

```
[ILAsm]  
.field public static literal valuetype System.Char MaxValue = (char)0xFFFF  
  
[C#]  
public const char MaxValue = (char)0xFFFF
```

## Summary

Contains the maximum code point for the `System.Char` type.

## Description

The numeric value of this constant is 65,535.

# Char.MinValue Field

```
[ILAsm]  
.field public static literal valuetype System.Char MinValue = (char)0x0  
  
[C#]  
public const char MinValue = (char)0x0
```

## Summary

Contains the minimum code point for the `System.Char` type.

## Description

The numeric value of this constant is 0.

# Char.CompareTo(System.Char) Method

```
[ILAsm]  
.method public final hidebysig virtual int32 CompareTo(char value)  
  
[C#]  
public int CompareTo(char value)
```

## Summary

Returns the sort order of the current instance compared to the specified `System.Char`.

## Parameters

Parameter	Description
<i>value</i>	The <code>System.Char</code> to compare to the current instance.

## Return Value

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

Return Value	Description
A negative number	Current instance < <i>value</i> .
Zero	Current instance == <i>value</i> .
A positive number	Current instance > <i>value</i> .

## Description

The comparison performed by this method is based on the code points of the current instance and *value*, not necessarily their lexicographical characteristics.

[*Note:* This method is implemented to support the `System.IComparable<Char>` interface.]

# Char.CompareTo(System.Object) Method

```
[ILAsm]  
.method public final hidebysig virtual int32 CompareTo(object value)  
  
[C#]  
public int CompareTo(object value)
```

## Summary

Returns the sort order of the current instance compared to the specified `System.Object`.

## Parameters

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

## Return Value

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

Return Value	Description
A negative number	Current instance < <i>value</i> .
Zero	Current instance == <i>value</i> .
A positive number	Current instance > <i>value</i> , or <i>value</i> is a null reference.

## Description

The comparison performed by this method is based on the code points of the current instance and *value*, not necessarily their lexicographical characteristics.

[*Note:* This method is implemented to support the `System.IComparable` interface.]

## Exceptions

Exception	Condition
<b>System.ArgumentException</b>	<i>value</i> is not a <code>System.Char</code> and is not a null reference.

1

2

# Char.Equals(System.Char) Method

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

## Summary

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

## Parameters

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

## Return Value

`true` if *obj* represents the same value as the current instance; otherwise, `false`.

## Description

The comparison performed by this method is based on the code points of the current instance and *obj*, not necessarily their lexicographical characteristics.

[*Note:* This method is implemented to support the `System.IEquatable<Char>` interface.]



# Char.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

`true` if *obj* represents the same type and value as the current instance. If *obj* is a null reference or is not an instance of `System.Char`, returns `false`.

## Description

The comparison performed by this method is based on the code points of the current instance and *obj*, not necessarily their lexicographical characteristics.

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

# Char.GetHashCode() Method

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

## Summary

Generates a hash code for the current instance.

## Return Value

A `System.Int32` value containing a hash code for the current instance.

## Description

The algorithm used to generate the hash code is unspecified.

[*Note:* This method overrides `System.Object.GetHashCode()`.]

**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## Char.GetNumericValue(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static float64 GetNumericValue(string s, int32  
index)  
  
[C#]  
public static double GetNumericValue(string s, int index)
```

### Summary

Returns the numeric value associated with the Unicode character at the specified position in the specified *System.String*.

### Parameters

Parameter	Description
<i>s</i>	A <i>System.String</i> .
<i>index</i>	A <i>System.Int32</i> that specifies the position of a character in <i>s</i> .

### Return Value

A *System.Double* representing the numeric value associated with the *System.Char* at position *index* in *s* if and only if that *System.Char* has an associated numeric value; otherwise, -1.0.

### Description

A character has an associated numeric value if and only if it is a member of one of the following categories in *System.Globalization.UnicodeCategory*:  
*System.Globalization.UnicodeCategory.DecimalDigitNumber*,  
*System.Globalization.UnicodeCategory.LetterNumber*, or  
*System.Globalization.UnicodeCategory.OtherNumber*.

### Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>s</i> is a null reference.

**System.ArgumentOutOfRangeException**

The value of *index* is less than zero, or greater than or equal to the length of *s*.

1

2

**The following member must be implemented if the ExtendedNumerics library is present in the implementation.**

## Char.GetNumericValue(System.Char) Method

```
[ILAsm]  
.method public hidebysig static float64 GetNumericValue(valuetype  
System.Char c)  
  
[C#]  
public static double GetNumericValue(char c)
```

### Summary

Returns the numeric value associated with the specified Unicode character.

### Parameters

Parameter	Description
<i>c</i>	A Unicode character.

### Return Value

A `System.Double` representing the numeric value associated with *c* if and only if *c* has an associated numeric value; otherwise, -1.0.

### Description

A character has an associated numeric value if and only if it is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.DecimalDigitNumber`,  
`System.Globalization.UnicodeCategory.LetterNumber`, or  
`System.Globalization.UnicodeCategory.OtherNumber`.

### Example

The following example demonstrates the `System.Char.GetNumericValue` method.

[C#]

```
using System;  
public class GetNumericValueExample {  
    public static void Main() {  
        Char[] cAry = {'8', 'V', Convert.ToChar(0X00BC)};  
        //Unicode U+00BC is the code point for the character  
        //representation of 1/4  
        foreach(Char c in cAry) {  
            Console.WriteLine("Numeric value of Unicode " +  
                              "character {0} ", c);  
        }  
    }  
}
```

```
1         Console.WriteLine(" is {0}",
2                             Char.GetNumericValue(c));
3     }
4 }
5 }
6 The output is
7
8 Numeric value of Unicode character 8 is 8
9
10
11 Numeric value of Unicode character V is -1
12
13
14 Numeric value of Unicode character 1/4 is 0.25
15
16
```

# Char.GetUnicodeCategory(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static valuetype  
System.Globalization.UnicodeCategory GetUnicodeCategory(string s, int32  
index)  
  
[C#]  
public static UnicodeCategory GetUnicodeCategory(string s, int index)
```

## Summary

Determines the `System.Globalization.UnicodeCategory` of the character at the specified position in the specified `System.String`.

## Parameters

Parameter	Description
<i>s</i>	A <code>System.String</code> .
<i>index</i>	A <code>System.Int32</code> that specifies the position of a character in <i>s</i> .

## Return Value

The `System.Globalization.UnicodeCategory` of the `System.Char` at position *index* in *s*.

## Description

[*Note:* For more information regarding Unicode categories, see `System.Globalization.UnicodeCategory`.]

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>s</i> is a null reference.
<b>System.ArgumentOutOfRangeException</b>	The value of <i>index</i> is less than zero, or greater than or equal to the length of <i>s</i> .





# Char.GetUnicodeCategory(System.Char)

## Method

```
[ILAsm]
.method public hidebysig static valuetype
System.Globalization.UnicodeCategory GetUnicodeCategory(valuetype
System.Char c)

[C#]
public static UnicodeCategory GetUnicodeCategory(char c)
```

### Summary

Determines the `System.Globalization.UnicodeCategory` of the specified Unicode character.

### Parameters

Parameter	Description
<i>c</i>	A Unicode character.

### Return Value

The `System.Globalization.UnicodeCategory` of *c*.

### Description

[*Note:* For more information regarding Unicode categories, see `System.Globalization.UnicodeCategory`.]

# Char.IsControl(System.String, System.Int32)

## Method

```
[ILAsm]  
.method public hidebysig static bool IsControl(string s, int32 index)  
  
[C#]  
public static bool IsControl(string s, int index)
```

### Summary

Determines whether the character at the specified position in the specified `System.String` is a control character.

### Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

### Return Value

true if the character at position `index` in `s` is a member of the following category in `System.Globalization.UnicodeCategory`: `System.Globalization.UnicodeCategory.Control`; otherwise, false.

### Exceptions

Exception	Condition
<b><code>System.ArgumentNullException</code></b>	<code>s</code> is a null reference.
<b><code>System.ArgumentOutOfRangeException</code></b>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .

# Char.IsControl(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsControl(valuetype System.Char c)  
  
[C#]  
public static bool IsControl(char c)
```

## Summary

Determines whether the specified Unicode character is a control character.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of the following category in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.Control`; otherwise, false.

# Char.IsDigit(System.String, System.Int32)

## Method

```
[ILAsm]  
.method public hidebysig static bool IsDigit(string s, int32 index)  
  
[C#]  
public static bool IsDigit(string s, int index)
```

### Summary

Determines whether the character at the specified position in the specified `System.String` is a decimal digit.

### Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

### Return Value

true if the character at position `index` in `s` is a member of the following category in `System.Globalization.UnicodeCategory`: `System.Globalization.UnicodeCategory.DecimalDigitNumber`; otherwise, false.

### Description

[*Note:* `System.Char.IsDigit` determines if a `System.Char` is a radix-10 digit. This contrasts with `System.Char.IsNumber`, which determines if a `System.Char` is of any numeric Unicode category.]

### Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<code>s</code> is a null reference.
<b>System.ArgumentOutOfRangeException</b>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .



# Char.IsDigit(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsDigit(valuetype System.Char c)  
  
[C#]  
public static bool IsDigit(char c)
```

## Summary

Determines whether a Unicode character is a decimal digit.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of the following category in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.DecimalDigitNumber`; otherwise, false.

## Description

[*Note:* `System.Char.IsDigit` determines if a `Char` is a radix-10 digit. This contrasts with `System.Char.IsNumber`, which determines if a `System.Char` is of any numeric Unicode category.]

# Char.IsLetter(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsLetter(valuetype System.Char c)  
  
[C#]  
public static bool IsLetter(char c)
```

## Summary

Determines whether the specified Unicode character is a letter.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.UppercaseLetter`,  
`System.Globalization.UnicodeCategory.LowercaseLetter`,  
`System.Globalization.UnicodeCategory.TitlecaseLetter`,  
`System.Globalization.UnicodeCategory.ModifierLetter`, or  
`System.Globalization.UnicodeCategory.OtherLetter`; otherwise, false.

# Char.IsLetter(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static bool IsLetter(string s, int32 index)  
  
[C#]  
public static bool IsLetter(string s, int index)
```

## Summary

Determines whether the character at the specified position in the specified `System.String` is a letter.

## Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

## Return Value

true if the character at position `index` in `s` is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.UppercaseLetter`,  
`System.Globalization.UnicodeCategory.LowercaseLetter`,  
`System.Globalization.UnicodeCategory.TitlecaseLetter`,  
`System.Globalization.UnicodeCategory.ModifierLetter`, or  
`System.Globalization.UnicodeCategory.OtherLetter`; otherwise, false.

## Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<code>s</code> is a null reference.
<code>System.ArgumentOutOfRangeException</code>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .



# Char.IsLetterOrDigit(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsLetterOrDigit(valuetype System.Char  
c)  
  
[C#]  
public static bool IsLetterOrDigit(char c)
```

## Summary

Determines whether the specified Unicode character is either a letter or a decimal digit.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.UppercaseLetter`,  
`System.Globalization.UnicodeCategory.LowercaseLetter`,  
`System.Globalization.UnicodeCategory.TitlecaseLetter`,  
`System.Globalization.UnicodeCategory.ModifierLetter`,  
`System.Globalization.UnicodeCategory.OtherLetter`, or  
`System.Globalization.UnicodeCategory.DecimalDigitNumber`; otherwise, false.

# Char.IsLetterOrDigit(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static bool IsLetterOrDigit(string s, int32  
index)
```

```
[C#]  
public static bool IsLetterOrDigit(string s, int index)
```

## Summary

Determines whether the character at the specified position in the specified System.String is either a letter or a decimal digit.

## Parameters

Parameter	Description
<i>s</i>	A System.String.
<i>index</i>	A System.Int32 that specifies a character position in <i>s</i> .

## Return Value

true if the character at position *index* in *s* is a member of one of the following categories in System.Globalization.UnicodeCategory: System.Globalization.UnicodeCategory.UppercaseLetter, System.Globalization.UnicodeCategory.LowercaseLetter, System.Globalization.UnicodeCategory.TitlecaseLetter, System.Globalization.UnicodeCategory.ModifierLetter, System.Globalization.UnicodeCategory.OtherLetter, or System.Globalization.UnicodeCategory.DecimalDigitNumber; otherwise, false.

## Exceptions

Exception	Condition
System.ArgumentNullException	<i>s</i> is a null reference.
System.ArgumentOutOfRangeException	The value of <i>index</i> is less than zero, or greater than or equal to the length of <i>s</i> .



# Char.IsLower(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsLower(valuetype System.Char c)  
  
[C#]  
public static bool IsLower(char c)
```

## Summary

Determines whether the specified Unicode character is a lowercase letter.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of the following category in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.LowercaseLetter`; otherwise, false.

# Char.IsLower(System.String, System.Int32)

## Method

```
[ILAsm]  
.method public hidebysig static bool IsLower(string s, int32 index)  
  
[C#]  
public static bool IsLower(string s, int index)
```

### Summary

Determines whether the character at the specified position in the specified `System.String` is a lowercase letter.

### Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

### Return Value

true if the character at position `index` in `s` is a member of the following category in `System.Globalization.UnicodeCategory`: `System.Globalization.UnicodeCategory.LowercaseLetter`; otherwise, false.

### Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<code>s</code> is a null reference.
<b>System.ArgumentOutOfRangeException</b>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .

# Char.IsNumber(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsNumber(valuetype System.Char c)  
  
[C#]  
public static bool IsNumber(char c)
```

## Summary

Determines whether the specified Unicode character is a number.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.DecimalDigitNumber`,  
`System.Globalization.UnicodeCategory.LetterNumber`, or  
`System.Globalization.UnicodeCategory.OtherNumber`; otherwise, false.

## Description

[*Note:* `System.Char.IsNumber` determines if a `System.Char` is of any numeric Unicode category. This contrasts with `System.Char.IsDigit`, which determines if a `System.Char` is a radix-10 digit.]

# Char.IsNumber(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static bool IsNumber(string s, int32 index)  
  
[C#]  
public static bool IsNumber(string s, int index)
```

## Summary

Determines whether the character at the specified position in the specified System.String is a number.

## Parameters

Parameter	Description
<i>s</i>	A System.String.
<i>index</i>	A System.Int32 that specifies a character position in <i>s</i> .

## Return Value

true if the character at position *index* in *s* is a member of one of the following categories in System.Globalization.UnicodeCategory: System.Globalization.UnicodeCategory.DecimalDigitNumber, System.Globalization.UnicodeCategory.LetterNumber, or System.Globalization.UnicodeCategory.OtherNumber; otherwise, false.

## Description

[*Note:* System.Char.IsNumber determines if a System.Char is of any numeric Unicode category. This contrasts with System.Char.IsDigit, which determines if a System.Char is a radix-10 digit.]

## Exceptions

Exception	Condition
System.ArgumentNullException	<i>s</i> is a null reference.
System.ArgumentOutOfRangeException	The value of <i>index</i> is less than zero, or greater

	than or equal to the length of $s$ .
--	--------------------------------------

1

2



# Char.IsPunctuation(System.Char) Method

```
[ILAsm]
.method public hidebysig static bool IsPunctuation(valuetype System.Char
c)

[C#]
public static bool IsPunctuation(char c)
```

## Summary

Determines whether the specified Unicode character is a punctuation mark.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.ConnectorPunctuation`,  
`System.Globalization.UnicodeCategory.DashPunctuation`,  
`System.Globalization.UnicodeCategory.OpenPunctuation`,  
`System.Globalization.UnicodeCategory.ClosePunctuation`,  
`System.Globalization.UnicodeCategory.InitialQuotePunctuation`,  
`System.Globalization.UnicodeCategory.FinalQuotePunctuation`, or  
`System.Globalization.UnicodeCategory.OtherPunctuation`; otherwise, false.

# Char.IsPunctuation(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static bool IsPunctuation(string s, int32 index)  
  
[C#]  
public static bool IsPunctuation(string s, int index)
```

## Summary

Determines whether the character at the specified position in the specified `System.String` is a punctuation mark.

## Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

## Return Value

true if the character at position `index` in `s` is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.ConnectorPunctuation`,  
`System.Globalization.UnicodeCategory.DashPunctuation`,  
`System.Globalization.UnicodeCategory.OpenPunctuation`,  
`System.Globalization.UnicodeCategory.ClosePunctuation`,  
`System.Globalization.UnicodeCategory.InitialQuotePunctuation`,  
`System.Globalization.UnicodeCategory.FinalQuotePunctuation`, or  
`System.Globalization.UnicodeCategory.OtherPunctuation`; otherwise, false.

## Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<code>s</code> is a null reference.
<code>System.ArgumentOutOfRangeException</code>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .

# Char.IsSeparator(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static bool IsSeparator(string s, int32 index)  
  
[C#]  
public static bool IsSeparator(string s, int index)
```

## Summary

Determines whether the character at the specified position in the specified `System.String` is a separator character.

## Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

## Return Value

`true` if the character at position `index` in `s` is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.SpaceSeparator`,  
`System.Globalization.UnicodeCategory.LineSeparator`, or  
`System.Globalization.UnicodeCategory.ParagraphSeparator`; otherwise, `false`.

## Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<code>s</code> is a null reference.
<code>System.ArgumentOutOfRangeException</code>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .

# Char.IsSeparator(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsSeparator(valuetype System.Char c)  
  
[C#]  
public static bool IsSeparator(char c)
```

## Summary

Determines whether the specified Unicode character is a separator character.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.SpaceSeparator`,  
`System.Globalization.UnicodeCategory.LineSeparator`, or  
`System.Globalization.UnicodeCategory.ParagraphSeparator`; otherwise, false.

# Char.IsSurrogate(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static bool IsSurrogate(string s, int32 index)  
  
[C#]  
public static bool IsSurrogate(string s, int index)
```

## Summary

Determines whether the character at the specified position in the specified `System.String` is a surrogate character.

## Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

## Return Value

true if the character at position `index` in `s` is a member of the following category in `System.Globalization.UnicodeCategory`: `System.Globalization.UnicodeCategory.Surrogate`; otherwise, false.

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<code>s</code> is a null reference.
<b>System.ArgumentOutOfRangeException</b>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .

# Char.IsSurrogate(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsSurrogate(valuetype System.Char c)  
  
[C#]  
public static bool IsSurrogate(char c)
```

## Summary

Determines whether the specified Unicode character is a surrogate character.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of the following category in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.Surrogate`; otherwise, false.

# Char.IsSymbol(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsSymbol(valuetype System.Char c)  
  
[C#]  
public static bool IsSymbol(char c)
```

## Summary

Determines whether the specified Unicode character is a symbol character.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.MathSymbol`,  
`System.Globalization.UnicodeCategory.CurrencySymbol`,  
`System.Globalization.UnicodeCategory.ModifierSymbol`, or  
`System.Globalization.UnicodeCategory.OtherSymbol`; otherwise, false.

# Char.IsSymbol(System.String, System.Int32)

## Method

```
[ILAsm]  
.method public hidebysig static bool IsSymbol(string s, int32 index)  
  
[C#]  
public static bool IsSymbol(string s, int index)
```

### Summary

Determines whether the character at the specified position in the specified `System.String` is a symbol character.

### Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

### Return Value

true if the character at position `index` in `s` is a member of one of the following categories in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.MathSymbol`,  
`System.Globalization.UnicodeCategory.CurrencySymbol`,  
`System.Globalization.UnicodeCategory.ModifierSymbol`, or  
`System.Globalization.UnicodeCategory.OtherSymbol`; otherwise, false.

### Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<code>s</code> is a null reference.
<code>System.ArgumentOutOfRangeException</code>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .



# Char.IsUpper(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static bool IsUpper(string s, int32 index)  
  
[C#]  
public static bool IsUpper(string s, int index)
```

## Summary

Determines whether the character at the specified position in the specified `System.String` is an uppercase letter.

## Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

## Return Value

`true` if the character at position `index` in `s` is a member of the following category in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.UppercaseLetter`; otherwise, `false`.

## Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<code>s</code> is a null reference.
<code>System.ArgumentOutOfRangeException</code>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .

# Char.IsUpper(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsUpper(valuetype System.Char c)  
  
[C#]  
public static bool IsUpper(char c)
```

## Summary

Determines whether the specified Unicode character is an uppercase letter.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* is a member of the following category in `System.Globalization.UnicodeCategory`:  
`System.Globalization.UnicodeCategory.UppercaseLetter`; otherwise, false.

# Char.IsWhiteSpace(System.String, System.Int32) Method

```
[ILAsm]  
.method public hidebysig static bool IsWhiteSpace(string s, int32 index)  
  
[C#]  
public static bool IsWhiteSpace(string s, int index)
```

## Summary

Determines whether the character at the specified position in the specified `System.String` is a whitespace character.

## Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> .
<code>index</code>	A <code>System.Int32</code> that specifies a character position in <code>s</code> .

## Return Value

true if the character at position `index` in `s` either has a code point of 0x0009, 0x000a, 0x000b, 0x000c, 0x000d, 0x0085, 0x2028, or 0x2029; or is a member of the following category in `System.Globalization.UnicodeCategory`: `System.Globalization.UnicodeCategory.SpaceSeparator`; otherwise, false.

## Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<code>s</code> is a null reference.
<code>System.ArgumentOutOfRangeException</code>	The value of <code>index</code> is less than zero, or greater than or equal to the length of <code>s</code> .

# Char.IsWhiteSpace(System.Char) Method

```
[ILAsm]  
.method public hidebysig static bool IsWhiteSpace(valuetype System.Char c)  
  
[C#]  
public static bool IsWhiteSpace(char c)
```

## Summary

Determines whether the specified Unicode character is a whitespace character.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

true if *c* either has a code point of 0x0009, 0x000a, 0x000b, 0x000c, 0x000d, 0x0085, 0x2028, or 0x2029; or is a member of the following category in System.Globalization.UnicodeCategory: System.Globalization.UnicodeCategory.SpaceSeparator; otherwise, false.

# Char.Parse(System.String) Method

```
[ILAsm]  
.method public hidebysig static valuetype System.Char Parse(string s)  
  
[C#]  
public static char Parse(string s)
```

## Summary

Returns the specified `System.String` converted to a `System.Char` value.

## Parameters

Parameter	Description
<code>s</code>	A <code>System.String</code> containing a single Unicode character.

## Return Value

The `System.Char` value obtained from `s`.

## Exceptions

Exception	Condition
<b><code>System.ArgumentNullException</code></b>	<code>s</code> is a null reference.
<b><code>System.FormatException</code></b>	<code>s</code> does not contain exactly one character.

# Char.ToLower(System.Char) Method

```
[ILAsm]
.method public hidebysig static valuetype System.Char ToLower(valuetype
System.Char c)

[C#]
public static char ToLower(char c)
```

## Summary

Converts a System.Char to its lowercase equivalent.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

The lowercase equivalent of *c*, or the value of *c* if and only if *c* is already lowercase or does not have a lowercase equivalent.

## Example

The following example demonstrates the System.Char.ToLower method.

```
[C#]

using System;
public class CharToLower {
    public static void Main() {
        Char[] cAry = {'A', 'c', '*'};
        foreach (Char c in cAry) {
            Console.WriteLine("Char '{0}' ToLower is ", c);
            Console.WriteLine("{0}", Char.ToLower(c));
        }
    }
}
```

The output is

Char 'A' ToLower is a

Char 'c' ToLower is c

```
1 Char '*' ToLower is *  
2  
3
```

# Char.ToString() Method

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

## Summary

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

## Return Value

A `System.String` representation of the current instance.

## Description

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



# Char.ToString(System.IFormatProvider)

## Method

```
[ILAsm]  
.method public final hidebysig virtual string ToString(class  
System.IFormatProvider provider)  
  
[C#]  
public string ToString(IFormatProvider provider)
```

### Summary

Converts the value of this instance to its equivalent `String` representation using the specified culture-specific format information.

### Parameters

Parameter	Description
<i>provider</i>	(Reserved) An <code>System.IFormatProvider</code> interface implementation that supplies culture-specific formatting information.

### Return Value

The `System.String` representation of the value of this instance as specified by *provider*.

### Description

*provider* is ignored; it does not participate in this operation.

# Char.ToUpper(System.Char) Method

```
[ILAsm]
.method public hidebysig static valuetype System.Char ToUpper(valuetype
System.Char c)

[C#]
public static char ToUpper(char c)
```

## Summary

Converts a System.Char to its uppercase equivalent.

## Parameters

Parameter	Description
<i>c</i>	A Unicode character.

## Return Value

The uppercase equivalent of *c*, or the value of *c* if and only if *c* is already uppercase or does not have an uppercase equivalent.

## Example

The following example demonstrates the System.Char.ToUpper method.

```
[C#]
using System;
public class CharToUpper {
    public static void Main() {
        Char[] cAry = {'A', 'c', '*'};
        foreach (Char c in cAry) {
            Console.Write("Char '{0}' ToUpper is {1}",
                c, Char.ToUpper(c));
            Console.WriteLine();
        }
    }
}
```

The output is

Char 'A' ToUpper is A

Char 'c' ToUpper is C

```
1 Char '*' ToUpper is *  
2  
3
```