

1 System.Char Structure

```
2 [ILAsm]
3 .class public sequential sealed serializable Char extends System.ValueType
4 implements System.IComparable, System.IComparable`1<char>,
5 System.IEquatable`1<char>
6
7 [C#]
8 public struct Char: IComparable, IComparable<Char>, IEquatable<Char>
```

8 Assembly Info:

- 9 • *Name:* mscorlib
- 10 • *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00]
- 11 • *Version:* 2.0.x.x
- 12 • *Attributes:*
 - 13 ○ CLSCompliantAttribute(true)

14 Implements:

- 15 • **System.IComparable**
- 16 • **System.IComparable<System.Char>**
- 17 • **System.IEquatable<System.Char>**

18 Summary

19 Represents a Unicode character.

20 Inherits From: System.ValueType

21 **Library:** BCL

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

26 Description

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

29
30 [*Note:* The *code point* of a Unicode character is that character's 2-byte, encoded value.]

31
32
33
34 [*Note:* The `System.Globalization.UnicodeCategory` enumeration describes the
35 categories that a Unicode character can be mapped to. For information on mapping
36 specific Unicode characters to Unicode categories, see the `UnicodeData.txt` file in the
37 Unicode Character Database at
38 <http://www.unicode.org/Public/UNIDATA/UnicodeCharacterDatabase.html>. The
39 `UnicodeData.txt` file format is described at [1](http://www.unicode.org/Public/3.1-</p></div><div data-bbox=)

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

4

1 Char.MaxValue Field

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

6 Summary

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

8 Description

9 The numeric value of this constant is 65,535.

10

1 Char.MinValue Field

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

6 Summary

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

8 Description

9 The numeric value of this constant is 0.

10

1 Char.CompareTo(System.Char) Method

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

6 Summary

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

8 Parameters

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

9 Return Value

10 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> .

15 Description

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

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

1 Char.CompareTo(System.Object) Method

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

6 Summary

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

8 Parameters

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

9 Return Value

10 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.

15 Description

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

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

23 Exceptions

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

1

2

1 Char.Equals(System.Char) Method

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

6 Summary

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

9 Parameters

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

10

11 Return Value

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

13 Description

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

16

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

19

20

21

1 Char.Equals(System.Object) Method

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

6 Summary

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

9 Parameters

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

10

11 Return Value

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

14 Description

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

17
18 [Note: This method overrides `System.Object.Equals`.]
19
20

21

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

10 Description

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

12

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

14

15

16

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 <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

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
System.ArgumentNullException	<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
```

1 Char.GetUnicodeCategory(System.String, 2 System.Int32) Method

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

9 Summary

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

12 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> .

13

14 Return Value

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

17 Description

18 [*Note:* For more information regarding Unicode categories, see
19 `System.Globalization.UnicodeCategory`.]
20
21

22 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> .

23

1 Char.GetUnicodeCategory(System.Char)

2 Method

```
3 [ILAsm]  
4 .method public hidebysig static valuetype  
5 System.Globalization.UnicodeCategory GetUnicodeCategory(valuetype  
6 System.Char c)  
  
7 [C#]  
8 public static UnicodeCategory GetUnicodeCategory(char c)
```

9 Summary

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

12 Parameters

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

13 Return Value

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

16 Description

17 [*Note:* For more information regarding Unicode categories, see
18 `System.Globalization.UnicodeCategory`.]
19
20

21

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

2 Method

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

7 Summary

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

10 Parameters

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

11 Return Value

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

16 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> .

17
18

1 Char.IsControl(System.Char) Method

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

6 Summary

7 Determines whether the specified Unicode character is a control character.

8 Parameters

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

9

10 Return Value

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

14

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

2 Method

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

7 Summary

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

10 Parameters

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

11 Return Value

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

16 Description

17 [Note: System.Char.IsDigit determines if a System.Char is a radix-10 digit. This
18 contrasts with System.Char.IsNumber, which determines if a System.Char is of any
19 numeric Unicode category.]
20
21

22 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> .

1 Char.IsDigit(System.Char) Method

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

6 Summary

7 Determines whether a Unicode character is a decimal digit.

8 Parameters

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

9

10 Return Value

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

14 Description

15 [Note: System.Char.IsDigit determines if a Char is a radix-10 digit. This contrasts
16 with System.Char.IsNumber, which determines if a System.Char is of any numeric
17 Unicode category.]
18
19

20

1 Char.IsLetter(System.Char) Method

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

6 Summary

7 Determines whether the specified Unicode character is a letter.

8 Parameters

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

9

10 Return Value

11 true if *c* is a member of one of the following categories in
12 System.Globalization.UnicodeCategory:
13 System.Globalization.UnicodeCategory.UppercaseLetter,
14 System.Globalization.UnicodeCategory.LowercaseLetter,
15 System.Globalization.UnicodeCategory.TitlecaseLetter,
16 System.Globalization.UnicodeCategory.ModifierLetter, or
17 System.Globalization.UnicodeCategory.OtherLetter; otherwise, false.

18

1 Char.IsLetter(System.String, System.Int32)

2 Method

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

7 Summary

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

10 Parameters

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

11 Return Value

13 true if the character at position *index* in *s* is a member of one of the following
14 categories in System.Globalization.UnicodeCategory:
15 System.Globalization.UnicodeCategory.UppercaseLetter,
16 System.Globalization.UnicodeCategory.LowercaseLetter,
17 System.Globalization.UnicodeCategory.TitlecaseLetter,
18 System.Globalization.UnicodeCategory.ModifierLetter, OR
19 System.Globalization.UnicodeCategory.OtherLetter; otherwise, false.

20 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> .

21
22

1 Char.IsLetterOrDigit(System.Char) Method

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

7 Summary

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

9 Parameters

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

10

11 Return Value

12 true if *c* is a member of one of the following categories in
13 System.Globalization.UnicodeCategory:
14 System.Globalization.UnicodeCategory.UppercaseLetter,
15 System.Globalization.UnicodeCategory.LowercaseLetter,
16 System.Globalization.UnicodeCategory.TitlecaseLetter,
17 System.Globalization.UnicodeCategory.ModifierLetter,
18 System.Globalization.UnicodeCategory.OtherLetter, OR
19 System.Globalization.UnicodeCategory.DecimalDigitNumber; otherwise, false.

20

1 Char.IsLetterOrDigit(System.String, 2 System.Int32) Method

```
3 [ILAsm]  
4 .method public hidebysig static bool IsLetterOrDigit(string s, int32  
5 index)  
  
6 [C#]  
7 public static bool IsLetterOrDigit(string s, int index)
```

8 Summary

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

11 Parameters

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

12 13 Return Value

14 true if the character at position *index* in *s* is a member of one of the following
15 categories in System.Globalization.UnicodeCategory:
16 System.Globalization.UnicodeCategory.UppercaseLetter,
17 System.Globalization.UnicodeCategory.LowercaseLetter,
18 System.Globalization.UnicodeCategory.TitlecaseLetter,
19 System.Globalization.UnicodeCategory.ModifierLetter,
20 System.Globalization.UnicodeCategory.OtherLetter, OR
21 System.Globalization.UnicodeCategory.DecimalDigitNumber; otherwise, false.
22
23

24 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> .

1 Char.IsLower(System.Char) Method

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

6 Summary

7 Determines whether the specified Unicode character is a lowercase letter.

8 Parameters

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

9

10 Return Value

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

14

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

2 Method

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

7 Summary

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

10 Parameters

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

11 Return Value

13 true if the character at position *index* in *s* is a member of the following category in
14 System.Globalization.UnicodeCategory: System.Globalization.UnicodeCategory.L
15 owercaseLetter; otherwise, false.

16 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> .

17
18

1 Char.IsNumber(System.Char) Method

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

6 Summary

7 Determines whether the specified Unicode character is a number.

8 Parameters

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

9

10 Return Value

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

16 Description

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

20

21

22

1 Char.IsNumber(System.String, 2 System.Int32) Method

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

7 Summary

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

10 Parameters

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

11 12 Return Value

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

18 Description

19 [Note: System.Char.IsNumber determines if a System.Char is of any numeric Unicode
20 category. This contrasts with System.Char.IsDigit, which determines if a System.Char
21 is a radix-10 digit.]
22
23

24 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

1 Char.IsPunctuation(System.Char) Method

```
2 [ILAsm]  
3 .method public hidebysig static bool IsPunctuation(valuetype System.Char  
4 c)  
5 [C#]  
6 public static bool IsPunctuation(char c)
```

7 Summary

8 Determines whether the specified Unicode character is a punctuation mark.

9 Parameters

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

10

11 Return Value

12 true if *c* is a member of one of the following categories in
13 System.Globalization.UnicodeCategory:
14 System.Globalization.UnicodeCategory.ConnectorPunctuation,
15 System.Globalization.UnicodeCategory.DashPunctuation,
16 System.Globalization.UnicodeCategory.OpenPunctuation,
17 System.Globalization.UnicodeCategory.ClosePunctuation,
18 System.Globalization.UnicodeCategory.InitialQuotePunctuation,
19 System.Globalization.UnicodeCategory.FinalQuotePunctuation, OR
20 System.Globalization.UnicodeCategory.OtherPunctuation; otherwise, false.

21

1 Char.IsPunctuation(System.String, 2 System.Int32) Method

```
3 [ILAsm]  
4 .method public hidebysig static bool IsPunctuation(string s, int32 index)  
5  
6 [C#]  
7 public static bool IsPunctuation(string s, int index)
```

7 Summary

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

10 Parameters

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

11 12 Return Value

13 true if the character at position *index* in *s* is a member of one of the following
14 categories in System.Globalization.UnicodeCategory:
15 System.Globalization.UnicodeCategory.ConnectorPunctuation,
16 System.Globalization.UnicodeCategory.DashPunctuation,
17 System.Globalization.UnicodeCategory.OpenPunctuation,
18 System.Globalization.UnicodeCategory.ClosePunctuation,
19 System.Globalization.UnicodeCategory.InitialQuotePunctuation,
20 System.Globalization.UnicodeCategory.FinalQuotePunctuation, OR
21 System.Globalization.UnicodeCategory.OtherPunctuation; otherwise, false.

22 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> .

23

24

1 Char.IsSeparator(System.String, 2 System.Int32) Method

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

7 Summary

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

10 Parameters

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

11

12 Return Value

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

18 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> .

19

20

1 Char.IsSeparator(System.Char) Method

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

6 Summary

7 Determines whether the specified Unicode character is a separator character.

8 Parameters

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

9

10 Return Value

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

16

1 Char.IsSurrogate(System.String, 2 System.Int32) Method

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

7 Summary

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

10 Parameters

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

11 12 Return Value

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

16 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> .

17
18

1 Char.IsSurrogate(System.Char) Method

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

6 Summary

7 Determines whether the specified Unicode character is a surrogate character.

8 Parameters

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

9

10 Return Value

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

14

1 Char.IsSymbol(System.Char) Method

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

6 Summary

7 Determines whether the specified Unicode character is a symbol character.

8 Parameters

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

9

10 Return Value

11 true if *c* is a member of one of the following categories in
12 System.Globalization.UnicodeCategory:
13 System.Globalization.UnicodeCategory.MathSymbol,
14 System.Globalization.UnicodeCategory.CurrencySymbol,
15 System.Globalization.UnicodeCategory.ModifierSymbol, or
16 System.Globalization.UnicodeCategory.OtherSymbol; otherwise, false.

17

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

2 Method

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

7 Summary

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

10 Parameters

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

11 Return Value

13 true if the character at position *index* in *s* is a member of one of the following
14 categories in System.Globalization.UnicodeCategory:
15 System.Globalization.UnicodeCategory.MathSymbol,
16 System.Globalization.UnicodeCategory.CurrencySymbol,
17 System.Globalization.UnicodeCategory.ModifierSymbol, OR
18 System.Globalization.UnicodeCategory.OtherSymbol; otherwise, false.

19 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> .

20
21

1 Char.IsUpper(System.String, System.Int32)

2 Method

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

7 Summary

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

10 Parameters

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

11

12 Return Value

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

16 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> .

17

18

1 Char.IsUpper(System.Char) Method

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

6 Summary

7 Determines whether the specified Unicode character is an uppercase letter.

8 Parameters

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

9

10 Return Value

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

14

1 Char.IsWhiteSpace(System.String, 2 System.Int32) Method

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

7 Summary

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

10 Parameters

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

11 12 Return Value

13 true if the character at position *index* in *s* either has a code point of 0x0009, 0x000a,
14 0x000b, 0x000c, 0x000d, 0x0085, 0x2028, or 0x2029; or is a member of the following
15 category in System.Globalization.UnicodeCategory:
16 System.Globalization.UnicodeCategory.SpaceSeparator; otherwise, false.

17 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> .

18
19

1 Char.IsWhiteSpace(System.Char) Method

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

6 Summary

7 Determines whether the specified Unicode character is a whitespace character.

8 Parameters

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

9

10 Return Value

11 true if *c* either has a code point of 0x0009, 0x000a, 0x000b, 0x000c, 0x000d, 0x0085,
12 0x2028, or 0x2029; or is a member of the following category in
13 System.Globalization.UnicodeCategory:
14 System.Globalization.UnicodeCategory.SpaceSeparator; otherwise, false.

15

1 Char.Parse(System.String) Method

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

6 Summary

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

8 Parameters

Parameter	Description
s	A System.String containing a single Unicode character.

9

10 Return Value

11 The System.Char value obtained from s.

12 Exceptions

Exception	Condition
System.ArgumentNullException	s is a null reference.
System.FormatException	s does not contain exactly one character.

13

14

1 Char.ToLower(System.Char) Method

```
2 [ILAsm]  
3 .method public hidebysig static valuetype System.Char ToLower(valuetype  
4 System.Char c)  
5 [C#]  
6 public static char ToLower(char c)
```

7 Summary

8 Converts a System.Char to its lowercase equivalent.

9 Parameters

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

10

11 Return Value

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

14 Example

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

```
16 [C#]  
17  
18 using System;  
19 public class CharToLower {  
20     public static void Main() {  
21         Char[] cAry = {'A', 'c', '*'};  
22         foreach (Char c in cAry) {  
23             Console.WriteLine("Char '{0}' ToLower is ", c);  
24             Console.WriteLine("{0}", Char.ToLower(c));  
25         }  
26     }  
27 }
```

28 The output is

29

30 Char 'A' ToLower is a

31

32

33 Char 'c' ToLower is c

34

35

1 Char '*' ToLower is *

2

3

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

10 Description

11 [*Note:* This method overrides `System.Object.ToString`.]
12
13
14

1 Char.ToString(System.IFormatProvider)

2 Method

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

8 Summary

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

11 Parameters

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

12 13 Return Value

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

15 Description

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

17

1 Char.ToUpper(System.Char) Method

```
2 [ILAsm]  
3 .method public hidebysig static valuetype System.Char ToUpper(valuetype  
4 System.Char c)  
  
5 [C#]  
6 public static char ToUpper(char c)
```

7 Summary

8 Converts a System.Char to its uppercase equivalent.

9 Parameters

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

10

11 Return Value

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

14 Example

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

```
16 [C#]  
17  
18 using System;  
19 public class CharToUpper {  
20     public static void Main() {  
21         Char[] cAry = {'A', 'c', '*'};  
22         foreach (Char c in cAry) {  
23             Console.WriteLine("Char '{0}' ToUpper is {1}",  
24                 c, Char.ToUpper(c));  
25             Console.WriteLine();  
26         }  
27     }  
28 }
```

29 The output is

30

31 Char 'A' ToUpper is A

32

33

34 Char 'c' ToUpper is C

35

36

1 Char '*' ToUpper is *
2
3