

System.Byte Structure

```
[ILASM]
.class public sequential sealed serializable Byte extends
System.ValueType implements System.IComparable,
System.IFormattable

[C#]
public struct Byte: IComparable, IFormattable
```

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.IComparable**
- **System.IFormattable**

Summary

Represents an 8-bit unsigned integer.

Inherits From: System.ValueType

Library: BCL

Thread Safety: This type is safe for multithreaded operations.

Description

The **System.Byte** data type represents integer values ranging from 0 to positive 255 (hexadecimal 0xFF).

1 Byte.MaxValue Field

```
2 [ILASM]  
3 .field public static literal unsigned int8 MaxValue = 255  
4 [C#]  
5 public const byte MaxValue = 255
```

6 Summary

7 Contains the maximum value for the **System.Byte** type.

8 Description

9 The value of this constant is 255 (hexadecimal 0xFF).

10

1 Byte.MinValue Field

```
2 [ILASM]  
3 .field public static literal unsigned int8 MinValue = 0  
4 [C#]  
5 public const byte MinValue = 0
```

6 Summary

7 Contains the minimum value for the **System.Byte** type.

8 Description

9 The value of this constant is 0.

10

Byte.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 object.

Parameters

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

Return Value

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

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

Description

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

Exceptions

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

1
2
3

Byte.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 System.Object 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.Byte**, returns **false**.

Description

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

1 Byte.GetHashCode() Method

```
2 [ILASM]  
3 .method public hidebysig virtual int32 GetHashCode()  
4  
5 [C#]  
6 public override int GetHashCode()
```

6 Summary

7 Generates a hash code for the current instance.

8 Return Value

9

10 A **System.Int32** containing the hash code for the current instance.

11 Description

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

13

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

15

Byte.Parse(System.String) Method

```
[ILASM]
.method public hidebysig static unsigned int8 Parse(string
s)

[C#]
public static byte Parse(string s)
```

Summary

Returns the specified **System.String** converted to a **System.Byte** value.

Parameters

Parameter	Description
s	A System.String containing the value to convert. The string is interpreted using the System.Globalization.NumberStyles.Integer style.

Return Value

The **System.Byte** value obtained from s.

Description

This version of **System.Byte.Parse** is equivalent to **System.Byte.Parse (s, System.Globalization.NumberStyles.Integer, null)**.

The string s is parsed using the formatting information in a **System.Globalization.NumberFormatInfo** initialized for the current system culture. [Note: For more information, see **System.Globalization.NumberFormatInfo.CurrentInfo**.]

Exceptions

Exception	Condition
System.ArgumentNullException	s is a null reference.
System.FormatException	s is not in the correct style.
System.OverflowException	s represents a number greater than System.Byte.MaxValue or less than System.Byte.MinValue .

1

2 **Example**

3

4 The following example demonstrates the **System.Byte.Parse** method.

5

6 [C#]

7 using System;

8 public class ByteParseClass {

9 public static void Main() {

10 string str = " 100 ";

11 Console.WriteLine("String: \"{0}\" <Byte>

12 {1}",str,Byte.Parse(str));

13 }

14 }

15

16 The output is

17

18 String: " 100 " <Byte> 100

19

Byte.Parse(System.String, System.Globalization.NumberStyles) Method

```
[ILASM]
.method public hidebysig static unsigned int8 Parse(string
s, valuetype System.Globalization.NumberStyles style)

[C#]
public static byte Parse(string s, NumberStyles style)
```

Summary

Returns the specified **System.String** converted to a **System.Byte** value.

Parameters

Parameter	Description
<i>s</i>	A System.String containing the value to convert. The string is interpreted using the style specified by <i>style</i> .
<i>style</i>	Zero or more System.Globalization.NumberStyles values that specify the style of <i>s</i> . Specify multiple values for <i>style</i> using the bitwise OR operator. If <i>style</i> is a null reference, the string is interpreted using the System.Globalization.NumberStyles.Integer style.

Return Value

The **System.Byte** value obtained from *s*.

Description

This version of **System.Byte.Parse** is equivalent to **System.Byte.Parse** (*s*, *style*, null).

The string *s* is parsed using the formatting information in a **System.Globalization.NumberFormatInfo** initialized for the current system culture. [Note: For more information, see **System.Globalization.NumberFormatInfo.CurrentInfo**.]

Exceptions

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

1
2
3

System.FormatException	s is not in the correct style.
System.OverflowException	s represents a number greater than System.Byte.MaxValue or less than System.Byte.MinValue .

Byte.Parse(System.String, System.IFormatProvider) Method

```
[ILASM]
.method public hidebysig static unsigned int8 Parse(string
s, class System.IFormatProvider provider)

[C#]
public static byte Parse(string s, IFormatProvider
provider)
```

Summary

Returns the specified **System.String** converted to a **System.Byte** value.

Parameters

Parameter	Description
<i>s</i>	A System.String containing the value to convert. The string is interpreted using the System.Globalization.NumberStyles.Integer style.
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.NumberFormatInfo containing culture-specific formatting information about <i>s</i> .

Return Value

The **System.Byte** value obtained from *s*.

Description

This version of **System.Byte.Parse** is equivalent to **System.Byte.Parse** (*s*, **System.Globalization.NumberStyles.Integer**, *provider*).

The string *s* is parsed using the culture-specific formatting information from the **System.Globalization.NumberFormatInfo** instance supplied by *provider*. If *provider* is **null** or a **System.Globalization.NumberFormatInfo** cannot be obtained from *provider*, the formatting information for the current system culture is used.

Exceptions

1
2
3

Exception	Condition
System.ArgumentNullException	s is a null reference.
System.OverflowException	s represents a number greater than System.Byte.MaxValue or less than System.Byte.MinValue .
System.FormatException	s is not in the correct style.

Byte.Parse(System.String, System.Globalization.NumberStyles, System.IFormatProvider) Method

```
[ILASM]  
.method public hidebysig static unsigned int8 Parse(string  
s, valuetype System.Globalization.NumberStyles style, class  
System.IFormatProvider provider)
```

```
[C#]  
public static byte Parse(string s, NumberStyles style,  
IFormatProvider provider)
```

Summary

Returns the specified **System.String** converted to a **System.Byte** value.

Parameters

Parameter	Description
<i>s</i>	A System.String containing the value to convert. The string is interpreted using the style specified by <i>style</i> .
<i>style</i>	Zero or more System.Globalization.NumberStyles values that specify the style of <i>s</i> . Specify multiple values for <i>style</i> using the bitwise OR operator. If <i>style</i> is a null reference, the string is interpreted using the System.Globalization.NumberStyles.Integer style.
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.NumberFormatInfo containing culture-specific formatting information about <i>s</i> .

Return Value

The **System.Byte** value obtained from *s*.

Description

The string *s* is parsed using the culture-specific formatting information from the **System.Globalization.NumberFormatInfo** instance supplied by *provider*. If *provider* is **null** or a **System.Globalization.NumberFormatInfo** cannot be obtained from *provider*, the formatting information for the current system culture is used.

1 **Exceptions**
2
3

Exception	Condition
System.ArgumentNullException	s is a null reference.
System.FormatException	s is not in the correct style.
System.OverflowException	s represents a number greater than System.Byte.MaxValue or less than System.Byte.MinValue .

4
5
6

Byte.ToString(System.IFormatProvider)

Method

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

[C#]
public string ToString(IFormatProvider provider)
```

Summary

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

Parameters

Parameter	Description
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.NumberFormatInfo containing culture-specific formatting information.

Return Value

A **System.String** representation of the current instance formatted using the general format specifier, ("G"). The string takes into account the information in the **System.Globalization.NumberFormatInfo** instance supplied by *provider*.

Description

This version of **System.Byte.ToString** is equivalent to **System.Byte.ToString("G", provider)**.

If *provider* is **null** or a **System.Globalization.NumberFormatInfo** cannot be obtained from *provider*, the formatting information for the current system culture is used.

Byte.ToString(System.String, System.IFormatProvider) Method

```
[ILASM]
.method public final hidebysig virtual string
ToString(string format, class System.IFormatProvider
provider)

[C#]
public string ToString(string format, IFormatProvider
provider)
```

Summary

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

Parameters

Parameter	Description
<i>format</i>	A System.String containing a character that specifies the format of the returned string.
<i>provider</i>	A System.IFormatProvider that supplies a System.Globalization.NumberFormatInfo instance containing culture-specific formatting information.

Return Value

A **System.String** representation of the current instance formatted as specified by *format*. The string takes into account the information in the **System.Globalization.NumberFormatInfo** instance supplied by *provider*.

Description

If *provider* is **null** or a **System.Globalization.NumberFormatInfo** cannot be obtained from *provider*, the formatting information for the current system culture is used.

If *format* is a **null** reference, the general format specifier "G" is used.

The following table lists the characters that are valid for the **System.Byte** type:

Format Characters	Description
-------------------	-------------

"C", "c"	Currency format.
"D", "d"	Decimal format.
"E", "e"	Exponential notation format.
"F", "f"	Fixed-point format.
"G", "g"	General format.
"N", "n"	Number format.
"P", "p"	Percent format.
"X", "x"	Hexadecimal format.

1
2
3
4
5
6

[*Note:* For a detailed description of formatting, see the **System.IFormattable** interface.]

This method is implemented to support the **System.IFormattable** interface.]

7 **Exceptions**
8
9

Exception	Condition
System.FormatException	<i>format</i> is invalid.

10
11
12

1 Byte.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
8 instance.

9 Return Value

10

11 A **System.String** representation of the current instance formatted
12 using the general format specifier ("G"). The string takes into account
13 the current system culture.

14 Description

15 This version of **System.Byte.ToString** is equivalent to
16 **System.Byte.ToString (null, null)**.

17

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

19

Byte.ToString(System.String) Method

```
[ILASM]
.method public hidebysig instance string ToString(string
format)

[C#]
public string ToString(string format)
```

Summary

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

Parameters

Parameter	Description
<i>format</i>	A System.String that specifies the format of the returned string. [Note: For a list of valid values, see System.Byte.ToString(System.String, System.IFormatProvider) .]

Return Value

A **System.String** representation of the current instance formatted as specified by *format*. The string takes into account the current system culture.

Description

This version of **System.Byte.ToString** is equivalent to **System.Byte.ToString** (*format*, **null**).

If *format* is **null**, the general format specifier "G" is used.

Exceptions

Exception	Condition
System.FormatException	<i>format</i> is invalid.

Example

The following example demonstrates the **System.Byte.ToString** method.

```
[C#]
```

```
1      using System;
2      public class ByteToStringExample {
3          public static void Main() {
4              Byte b = 8;
5              Console.WriteLine(b);
6              String[] formats = {"c", "d", "e", "f", "g", "n",
7                  "p", "x" };
8              foreach(String str in formats)
9                  Console.WriteLine("{0}: {1}", str,
10                     b.ToString(str));
11          }
12      }
13
```

14 The output is

15 8

16
17
18
19 c: \$8.00

20
21
22 d: 8

23
24
25 e: 8.000000e+000

26
27
28 f: 8.00

29
30
31 g: 8

32

1	
2	n: 8.00
3	
4	
5	p: 800.00 %
6	
7	
8	x: 8
9	
10	