

System.Int16 Structure

```
[ILAsm]
.class public sequential sealed serializable Int16 extends
System.ValueType implements System.IComparable, System.IFormattable,
System.IComparable`1<int16>, System.IEquatable`1<int16>

[C#]
public struct Int16: IComparable, IFormattable, IComparable<Int16>,
IEquatable<Int16>
```

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.IFormattable**
- **System.IComparable<System.Int16>**
- **System.IEquatable<System.Int16>**

Summary

Represents a 16-bit signed integer.

Inherits From: System.ValueType

Library: BCL

Thread Safety: This type is safe for multithreaded operations.

Description

The `System.Int16` data type represents integer values ranging from negative 32,768 to positive 32,767; that is, hexadecimal: 0x8000 to 0x7FFF.

Int16.MaxValue Field

```
[ILAsm]  
.field public static literal int16 MaxValue = 32767  
  
[C#]  
public const short MaxValue = 32767
```

Summary

Contains the maximum value for the `System.Int16` type.

Description

The value of this constant is 32,767 (hexadecimal 0X7FFF).

Int16.MinValue Field

```
[ILAsm]  
.field public static literal int16 MinValue = -32768  
  
[C#]  
public const short MinValue = -32768
```

Summary

Contains the minimum value for the `System.Int16` type.

Description

The value of this constant is -32,768 (hexadecimal 0X8000).

Int16.CompareTo(System.Int16) Method

```
[ILAsm]  
.method public final hidebysig virtual int32 CompareTo(int16 value)  
  
[C#]  
public int CompareTo(short value)
```

Summary

Returns the sort order of the current instance compared to the specified `System.Int16`.

Parameters

Parameter	Description
<i>value</i>	The <code>System.Int16</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

[*Note:* This method is implemented to support the `System.IComparable<Int16>` interface.]

Int16.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

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

Exceptions

Exception	Condition
-----------	-----------

System.ArgumentException

value is not a `System.Int16` and is not a null reference.

1

2

Int16.Equals(System.Int16) Method

```
[ILAsm]  
.method public hidebysig virtual bool Equals(int16 obj)  
  
[C#]  
public override bool Equals(int16 obj)
```

Summary

Determines whether the current instance and the specified `System.Int16` represent the same value.

Parameters

Parameter	Description
<i>obj</i>	The <code>System.Int16</code> to compare to the current instance.

Return Value

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

Description

[*Note:* This method is implemented to support the `System.IEquatable<Int16>` interface.]

Int16.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.Int16`, returns `false`.

Description

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

Int16.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` containing the hash code for the current instance.

Description

The algorithm used to generate the hash code is unspecified.

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

Int16.Parse(System.String) Method

```
[ILAsm]  
.method public hidebysig static int16 Parse(string s)  
  
[C#]  
public static short Parse(string s)
```

Summary

Returns the specified `System.String` converted to a `System.Int16` value.

Parameters

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

Return Value

The `System.Int16` value obtained from `s`.

Description

This version of `System.Int16.Parse` is equivalent to `System.Int16.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	<code>s</code> is a null reference.
System.FormatException	<code>s</code> is not in the correct style.
System.OverflowException	<code>s</code> represents a number greater than <code>System.Int16.MaxValue</code> or less than <code>System.Int16.MinValue</code> .

1

2 **Example**

3 This example demonstrates parsing a string to a `System.Int16`.

4

5 [C#]

```
6 using System;
7 public class Int16ParseClass {
8     public static void Main() {
9         string str = " 100 ";
10        Console.WriteLine("String: \"{0}\" <Int16> {1}",str,Int16.Parse(str));
11    }
12 }
```

13 The output is

14

15 String: " 100 " <Int16> 100

16

Int16.Parse(System.String, System.Globalization.NumberStyles) Method

```
[ILAsm]  
.method public hidebysig static int16 Parse(string s, valuetype  
System.Globalization.NumberStyles style)  
  
[C#]  
public static short Parse(string s, NumberStyles style)
```

Summary

Returns the specified `System.String` converted to a `System.Int16` value.

Parameters

Parameter	Description
<i>s</i>	A <code>System.String</code> containing the value to convert. The string is interpreted using the style specified by <i>style</i> .
<i>style</i>	Zero or more <code>System.Globalization.NumberStyles</code> 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 <code>System.Globalization.NumberStyles.Integer</code> style.

Return Value

The `System.Int16` value obtained from *s*.

Description

This version of `System.Int16.Parse` is equivalent to `System.Int16.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	s is a null reference.
System.FormatException	s is not in the correct style.
System.OverflowException	s represents a number greater than System.Int16.MaxValue or less than System.Int16.MinValue.

1

2

Int16.Parse(System.String, System.IFormatProvider) Method

```
[ILAsm]  
.method public hidebysig static int16 Parse(string s, class  
System.IFormatProvider provider)  
  
[C#]  
public static short Parse(string s, IFormatProvider provider)
```

Summary

Returns the specified `System.String` converted to a `System.Int16` value.

Parameters

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

Return Value

The `System.Int16` value obtained from *s*.

Description

This version of `System.Int16.Parse` is equivalent to `System.Int16.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

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

System.FormatException	s is not in the correct style.
System.OverflowException	s represents a number greater than System.Int16.MaxValue or less than System.Int16.MinValue.

1

2

Int16.Parse(System.String, System.Globalization.NumberStyles, System.IFormatProvider) Method

```
[ILAsm]
.method public hidebysig static int16 Parse(string s, valuetype
System.Globalization.NumberStyles style, class System.IFormatProvider
provider)

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

Summary

Returns the specified `System.String` converted to a `System.Int16` value.

Parameters

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

Return Value

The `System.Int16` 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.

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 <code>System.Int16.MaxValue</code> or less than <code>System.Int16.MinValue</code> .

1

2

Int16.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 <code>System.IFormatProvider</code> that supplies a <code>System.Globalization.NumberFormatInfo</code> 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 formatting information in the `System.Globalization.NumberFormatInfo` instance supplied by *provider*.

Description

This version of `System.Int16.ToString` is equivalent to `System.Int16.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.

Int16.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 <code>System.String</code> containing a character that specifies the format of the returned string.
<i>provider</i>	A <code>System.IFormatProvider</code> that supplies a <code>System.Globalization.NumberFormatInfo</code> 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 formatting 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.

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

This method is implemented to support the `System.IFormattable` interface.

]

The following table lists the characters that are valid for the `System.Int16` 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 Exceptions

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

3

4

Int16.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 formatted using the general format specifier ("G"). The string takes into account the current system culture.

Description

This version of `System.Int16.ToString` is equivalent to `System.Int16.ToString(null, null)`.

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

Int16.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 <code>System.String</code> that specifies the format of the returned string. [<i>Note:</i> For a list of valid values, see <code>System.Int16.ToString(System.String, System.IFormatProvider).</code>]

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.Int16.ToString` is equivalent to `System.Int16.ToString(format, null)`.

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

Exceptions

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

Example

This example demonstrates converting a `System.Int16` to a string.

```
[C#]
```

```
using System;  
public class Int16ToStringExample {
```

```

1      public static void Main() {
2          Int16 i = 16;
3          Console.WriteLine(i);
4          string[] formats = {"c", "d", "e", "f", "g", "n", "p", "x" };
5          foreach(string str in formats)
6              Console.WriteLine("{0}: {1}", str, i.ToString(str));
7      }
8  }

```

9 The output is

```

10
11 16
12
13
14 c: $16.00
15
16
17 d: 16
18
19
20 e: 1.600000e+001
21
22
23 f: 16.00
24
25
26 g: 16
27
28
29 n: 16.00
30
31
32 p: 1,600.00 %
33
34
35 x: 10
36

```

37