

# 1 System.Int64 Structure

```
2 [ILAsm]
3 .class public sequential sealed serializable Int64 extends
4 System.ValueType implements System.IComparable, System.IFormattable,
5 System.IComparable`1<int64>, System.IEquatable`1<int64>
6
7 [C#]
8 public struct Int64: IComparable, IFormattable, IComparable<Int64>,
9 IEquatable<Int64>
```

## 9 Assembly Info:

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

## 15 Implements:

- 16 • **System.IComparable**
- 17 • **System.IFormattable**
- 18 • **System.IComparable<System.Int64>**
- 19 • **System.IEquatable<System.Int64>**

## 20 Summary

21 Represents a 64-bit signed integer.

## 22 Inherits From: System.ValueType

23  
24 **Library:** BCL

25  
26 **Thread Safety:** All public static members of this type are safe for multithreaded operations.  
27 No instance members are guaranteed to be thread safe.

## 28 29 Description

30 The `System.Int64` data type represents integer values ranging from negative  
31 9,223,372,036,854,775,808 to positive 9,223,372,036,854,775,807; that is,  
32 hexadecimal 0X8000000000000000 to 0X7FFFFFFFFFFFFFFF.

33

# 1 Int64.MaxValue Field

```
2 [ILAsm]  
3 .field public static literal int64 MaxValue = 9223372036854775807  
4 [C#]  
5 public const long MaxValue = 9223372036854775807
```

## 6 Summary

7 Contains the maximum value for the `System.Int64` type.

## 8 Description

9 The value of this constant is 9,223,372,036,854,775,807 (hexadecimal  
10 0X7FFFFFFFFFFFFFFF).

11

# 1 Int64.MinValue Field

```
2 [ILAsm]  
3 .field public static literal int64 MinValue = -9223372036854775808  
4 [C#]  
5 public const long MinValue = -9223372036854775808
```

## 6 Summary

7 Contains the minimum value for the `System.Int64` type.

## 8 Description

9 The value of this constant is -9,223,372,036,854,775,808 (hexadecimal  
10 0X8000000000000000).

11

# 1 Int64.CompareTo(System.Int64) Method

```
2 [ILAsm]  
3 .method public final hidebysig virtual int32 CompareTo(int64 value)  
4 [C#]  
5 public int CompareTo(long value)
```

## 6 Summary

7 Returns the sort order of the current instance compared to the specified `System.Int64`.

## 8 Parameters

Parameter	Description
<i>value</i>	The <code>System.Int64</code> to compare to the current instance.

## 9 Return Value

11 The return value is a negative number, zero, or a positive number reflecting the sort  
12 order of the current instance as compared to *value*. For non-zero return values, the  
13 exact value returned by this method is unspecified. The following table defines the  
14 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

17 [Note: This method is implemented to support the `System.IComparable<Int64>`  
18 interface.]  
19  
20

21

# 1 Int64.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 <code>System.Object</code> to compare to the current instance.

## 9 Return Value

11 The return value is a negative number, zero, or a positive number reflecting the sort  
12 order of the current instance as compared to *value*. For non-zero return values, the  
13 exact value returned by this method is unspecified. The following table defines the  
14 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

17 [Note: This method is implemented to support the `System.IComparable` interface.]  
18  
19

## 20 Exceptions

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

**System.ArgumentException**

*value* is not a `System.Int64` and is not a null reference.

1

2

# 1 Int64.Equals(System.Int64) Method

```
2 [ILAsm]  
3 .method public hidebysig virtual bool Equals(int64 obj)  
4 [C#]  
5 public override bool Equals(long obj)
```

## 6 Summary

7 Determines whether the current instance and the specified `System.Int64` represent the  
8 same value.

## 9 Parameters

Parameter	Description
<i>obj</i>	The <code>System.Int64</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 [*Note:* This method is implemented to support the `System.IEquatable<Int64>`  
15 interface.]  
16  
17

18

# 1 Int64.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.Int64`, returns `false`.

## 14 Description

15 [Note: This method overrides `System.Object.Equals`.]  
16  
17

18

# 1 Int64.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` containing the 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

# 1 Int64.Parse(System.String) Method

```
2 [ILAsm]  
3 .method public hidebysig static int64 Parse(string s)  
4 [C#]  
5 public static long Parse(string s)
```

## 6 Summary

7 Returns the specified `System.String` converted to a `System.Int64` value.

## 8 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.

## 9 Return Value

11 The `System.Int64` value obtained from `s`.

## 12 Description

13 This version of `System.Int64.Parse` is equivalent to `System.Int64.Parse (s, System.Globalization.NumberStyles.Integer, null)`.

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

## 22 Exceptions

Exception	Condition
<code>System.ArgumentNullException</code>	<code>s</code> is a null reference.
<code>System.FormatException</code>	<code>s</code> is not in the correct style.
<code>System.OverflowException</code>	<code>s</code> represents a number greater than <code>System.Int64.MaxValue</code> or less than <code>System.Int64.MinValue</code> .

1

## 2 **Example**

3 This example demonstrates parsing a string to a `System.Int64`.

4

5 [C#]

```
6 using System;
7 public class Int64ParseClass {
8     public static void Main() {
9         string str = " 100 ";
10        Console.WriteLine("String: \"{0}\" <Int64> {1}",str,Int64.Parse(str));
11    }
12 }
```

13 The output is

14

15 String: " 100 " <Int64> 100

16

# 1 Int64.Parse(System.String, 2 System.Globalization.NumberStyles) Method

```
3 [ILAsm]  
4 .method public hidebysig static int64 Parse(string s, valuetype  
5 System.Globalization.NumberStyles style)  
  
6 [C#]  
7 public static long Parse(string s, NumberStyles style)
```

## 8 Summary

9 Returns the specified System.String converted to a System.Int64 value.

## 10 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.

## 11 Return Value

12 The System.Int64 value obtained from *s*.

## 13 Description

14 This version of System.Int64.Parse is equivalent to System.Int64.Parse(*s*, *style*, null).

15 The string *s* is parsed using the formatting information in a  
16 System.Globalization.NumberFormatInfo initialized for the current system culture.  
17 [Note: For more information, see  
18 System.Globalization.NumberFormatInfo.CurrentInfo.]  
19  
20  
21  
22  
23

## 24 Exceptions

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

<b>System.ArgumentNullException</b>	s is a null reference.
<b>System.FormatException</b>	s is not in the correct style.
<b>System.OverflowException</b>	s represents a number greater than <code>System.Int64.MaxValue</code> or less than <code>System.Int64.MinValue</code> .

1

2

# 1 Int64.Parse(System.String, 2 System.IFormatProvider) Method

```
3 [ILAsm]  
4 .method public hidebysig static int64 Parse(string s, class  
5 System.IFormatProvider provider)  
  
6 [C#]  
7 public static long Parse(string s, IFormatProvider provider)
```

## 8 Summary

9 Returns the specified System.String converted to a System.Int64 value.

## 10 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> .

## 11 12 Return Value

13 The System.Int64 value obtained from *s*.

## 14 Description

15 This version of System.Int64.Parse is equivalent to System.Int64.Parse (*s*,  
16 System.Globalization.NumberStyles.Integer, *provider*).

17  
18 The string *s* is parsed using the culture-specific formatting information from the  
19 System.Globalization.NumberFormatInfo instance supplied by *provider*. If *provider* is  
20 null or a System.Globalization.NumberFormatInfo cannot be obtained from *provider*,  
21 the formatting information for the current system culture is used.

## 22 Exceptions

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

<b>System.FormatException</b>	s is not in the correct style.
<b>System.OverflowException</b>	s represents a number greater than System.Int64.MaxValue or less than System.Int64.MinValue.

1

2

# 1 Int64.Parse(System.String, 2 System.Globalization.NumberStyles, 3 System.IFormatProvider) Method

```
4 [ILAsm]  
5 .method public hidebysig static int64 Parse(string s, valuetype  
6 System.Globalization.NumberStyles style, class System.IFormatProvider  
7 provider)  
  
8 [C#]  
9 public static long Parse(string s, NumberStyles style, IFormatProvider  
10 provider)
```

## 11 Summary

12 Returns the specified System.String converted to a System.Int64 value.

## 13 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> .

## 14 15 Return Value

16 The System.Int64 value obtained from *s*.

## 17 Description

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

## 22 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	s is a null reference.
<b>System.FormatException</b>	s is not in the correct style.
<b>System.OverflowException</b>	s represents a number greater than <code>System.Int64.MaxValue</code> or less than <code>System.Int64.MinValue</code> .

1

2

# 1 Int64.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 Returns a `System.String` representation of the value of the current instance.

### 10 Parameters

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

11

### 12 Return Value

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

### 16 Description

17 This version of `System.Int64.ToString` is equivalent to `System.Int64.ToString("G",`  
18 `provider)`.

19

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

22

# 1 Int64.ToString(System.String, 2 System.IFormatProvider) Method

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

## 8 Summary

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

## 10 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.

## 11 Return Value

12 A `System.String` representation of the current instance formatted as specified by  
13 *format*. The string takes into account the formatting information in the  
14 `System.Globalization.NumberFormatInfo` instance supplied by *provider*.  
15

## 16 Description

17 If *provider* is null or a `System.Globalization.NumberFormatInfo` cannot be obtained  
18 from *provider*, the formatting information for the current system culture is used.  
19

20 If *format* is a null reference, the general format specifier "G" is used.  
21

22 [Note: For a detailed description of formatting, see the `System.IFormattable` interface.  
23

24 This method is implemented to support the `System.IFormattable` interface.  
25

26 ]  
27

28 The following table lists the characters that are valid for the `System.Int64` 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
<b>System.FormatException</b>	<i>format</i> is invalid.

3

4

# 1 Int64.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 formatted using the general  
10 format specifier ("G"). The string takes into account the current system culture.

## 11 Description

12 This version of `System.Int64.ToString` is equivalent to `System.Int64.ToString(null,`  
13 `null)`.

14 [*Note:* This method overrides `System.Object.ToString`.]  
15  
16  
17

18

# 1 Int64.ToString(System.String) Method

```
2 [ILAsm]  
3 .method public hidebysig instance string ToString(string format)  
4 [C#]  
5 public string ToString(string format)
```

## 6 Summary

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

## 8 Parameters

Parameter	Description
<i>format</i>	A <code>System.String</code> that specifies the format of the returned string. [Note: For a list of valid values, see <code>System.Int64.ToString(System.String, System.IFormatProvider)</code> .]

9

## 10 Return Value

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

## 13 Description

14 This method is equivalent to `System.Int64.ToString (format, null)`.

15

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

## 17 Exceptions

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

18

## 19 Example

20 This example demonstrates converting a `System.Int64` to a string.

21

22 [C#]

```
23 using System;  
24 public class Int64ToStringExample {  
25     public static void Main() {
```

```
1     Int64 i = 64;
2     Console.WriteLine(i);
3     String[] formats = {"c", "d", "e", "f", "g", "n", "p", "x" };
4     foreach(String str in formats)
5         Console.WriteLine("{0}: {1}", str, i.ToString(str));
6     }
7 }
```

8 The output is

```
9
10 64
11
12
13 c: $64.00
14
15
16 d: 64
17
18
19 e: 6.400000e+001
20
21
22 f: 64.00
23
24
25 g: 64
26
27
28 n: 64.00
29
30
31 p: 6,400.00 %
32
33
34 x: 40
35
36
```