

# System.Globalization.NumberFormatInfo

## Class

```
[ILAsm]
.class public sealed serializable NumberFormatInfo extends System.Object
implements System.ICloneable, System.IFormatProvider

[C#]
public sealed class NumberFormatInfo: ICloneable, IFormatProvider
```

### Assembly Info:

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

### Implements:

- **System.ICloneable**
- **System.IFormatProvider**

### Summary

Supplies culture-specific formatting information for string representations of numeric values.

### Inherits From: System.Object

**Library:** BCL

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

### Description

`System.Globalization.NumberFormatInfo` supplies symbols such as currency symbols and decimal separators.

[*Note:* A `System.Globalization.NumberFormatInfo` instance typically contains the set of symbols for a specific language and culture. Instances of `System.Globalization.NumberFormatInfo` can be created to provide customized formatting information.]

# NumberFormatInfo() Constructor

```
[ILAsm]  
public rtspecialname specialname instance void .ctor()  
  
[C#]  
public NumberFormatInfo()
```

## Summary

Constructs and initializes a new instance of the `System.Globalization.NumberFormatInfo` class.

## Description

The new instance is not read-only, and is otherwise identical to the `System.Globalization.NumberFormatInfo` instance returned by the `System.Globalization.NumberFormatInfo.InvariantInfo` property.

# 1 NumberFormatInfo.Clone() Method

```
2 [ILAsm]  
3 .method public final hidebysig virtual object Clone()  
  
4 [C#]  
5 public object Clone()
```

## 6 Summary

7 Creates a copy of the current instance.

## 8 Return Value

9 A `System.Object` that is a copy of the current instance.

## 10 Description

11 The `System.Globalization.NumberFormatInfo.Clone` method returns a new instance  
12 of `System.Globalization.NumberFormatInfo` with property values that are equal to  
13 the property values of the current instance except for the  
14 `System.Globalization.DateTimeFormatInfo.IsReadOnly` property, which is always  
15 `false`.

16  
17 [*Note:* This method is implemented to support the `System.ICloneable` interface.]  
18  
19  
20

# NumberFormatInfo.GetFormat(System.Type) Method

```
[ILAsm]  
.method public final hidebysig virtual object GetFormat(class System.Type  
formatType)  
  
[C#]  
public object GetFormat(Type formatType)
```

## Summary

Returns an object of the specified type that provides formatting services.

## Parameters

Parameter	Description
<i>formatType</i>	The <code>System.Type</code> of the formatting object to be returned.

## Return Value

The current instance, if *formatType* is of type `System.Globalization.NumberFormatInfo`; otherwise, a null reference.

## Description

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

# NumberFormatInfo.ReadOnly(System.Globalization.NumberFormatInfo) Method

```
[ILAsm]
.method public hidebysig static class
System.Globalization.NumberFormatInfo ReadOnly(class
System.Globalization.NumberFormatInfo nfi)

[C#]
public static NumberFormatInfo ReadOnly(NumberFormatInfo nfi)
```

## Summary

Creates a read-only copy of the specified `System.Globalization.NumberFormatInfo` instance.

## Parameters

Parameter	Description
<i>nfi</i>	A <code>System.Globalization.NumberFormatInfo</code> object to copy.

## Return Value

A `System.Object` that is a copy of the current instance, and cannot be altered.

## Exceptions

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

# NumberFormatInfo.CurrencyDecimalDigits Property

```
[ILAsm]
.property int32 CurrencyDecimalDigits { public hidebysig specialname
instance int32 get_CurrencyDecimalDigits() public hidebysig specialname
instance void set_CurrencyDecimalDigits(int32 value) }

[C#]
public int CurrencyDecimalDigits { get; set; }
```

## Summary

Gets or sets the number of decimal places in currency values.

## Property Value

A `System.Int32` containing the number of decimal places in currency values.

## Description

The culture-invariant value for this property is 2.

## Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	The value specified for a set operation is less than 0 or greater than 99.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.CurrencyDecimalSeparator Property

```
[ILAsm]
.property string CurrencyDecimalSeparator { public hidebysig specialname
instance string get_CurrencyDecimalSeparator() public hidebysig
specialname instance void set_CurrencyDecimalSeparator(string value) }

[C#]
public string CurrencyDecimalSeparator { get; set; }
```

## Summary

Gets or sets the symbol used as the decimal separator in currency values.

## Property Value

A `System.String` containing the decimal separator used in currency values.

## Description

The culture-invariant value for this property is ".".

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The value specified for a set operation is a null reference.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.CurrencyGroupSeparator Property

```
[ILAsm]
.property string CurrencyGroupSeparator { public hidebysig specialname
instance string get_CurrencyGroupSeparator() public hidebysig specialname
instance void set_CurrencyGroupSeparator(string value) }

[C#]
public string CurrencyGroupSeparator { get; set; }
```

## Summary

Gets or sets the symbol used to separate groups of digits to the left of the decimal point in currency values.

## Property Value

A `System.String` containing the group separator used in currency values.

## Description

The culture-invariant value for this property is ",".

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The value specified for a set operation is a null reference.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.



# NumberFormatInfo.CurrencyGroupSizes

## Property

```
[ILAsm]
.property int32[] CurrencyGroupSizes { public hidebysig specialname
instance int32[] get_CurrencyGroupSizes() public hidebysig specialname
instance void set_CurrencyGroupSizes(int32[] value) }

[C#]
public int[] CurrencyGroupSizes { get; set; }
```

### Summary

Gets or sets the number of digits in each group to the left of the decimal point in currency values.

### Property Value

A `System.Int32` array containing elements that define the number of digits in each group in currency values.

### Description

All elements of the array except the last are required to be between 1 and 9, inclusive. The last element can be 0.

The first element of the array defines the number of elements in the first group of digits located immediately to the left of the `System.Globalization.NumberFormatInfo.CurrencyDecimalSeparator`. Each subsequent element refers to the next group of digits located to the left of the previous group. If the last element of the array is not zero, any remaining digits are grouped based on the last element of the array. If the last element is zero, the remaining digits are not grouped.

The culture-invariant value for this property is an array with a single element containing the value 3.

### Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The array specified for a set operation is a null reference.
<b>System.ArgumentException</b>	One of the elements in the array specified for a set operation is not between 0 and 9.  -or-

	The array contains an element, other than the last element, that is set to 0.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

1

## 2 Example

3 The following example demonstrates the effects of different  
4 `System.Globalization.NumberFormatInfo.CurrencyGroupSizes` property values.

5 [C#]  
6

```

7 using System;
8 using System.Globalization;
9 class Test {
10     public static void Main() {
11         NumberFormatInfo nfi = new NumberFormatInfo();
12
13         decimal myMoney = 9999999994444333221.00m;
14         nfi.CurrencyGroupSizes = new int[] {1,2,3,4,0};
15         Console.WriteLine("{0}",myMoney.ToString("C",nfi));
16
17         myMoney = 123456789123456.78m;
18         nfi.CurrencyGroupSizes = new int[] {3};
19         Console.WriteLine("{0}",myMoney.ToString("C",nfi));
20
21         nfi.CurrencyGroupSizes = new int[] {3,0};
22         Console.WriteLine("{0}",myMoney.ToString("C",nfi));
23     }
24 }
25
26
```

27 The output is

28  
29 \$999999999,4444,333,22,1.00

30  
31  
32 \$123,456,789,123,456.78

33  
34  
35 \$123456789123,456.78  
36

37

# NumberFormatInfo.CurrencyNegativePattern Property

```
[ILAsm]  
.property int32 CurrencyNegativePattern { public hidebysig specialname  
instance int32 get_CurrencyNegativePattern() public hidebysig specialname  
instance void set_CurrencyNegativePattern(int32 value) }  
  
[C#]  
public int CurrencyNegativePattern { get; set; }
```

## Summary

Gets or sets the format of negative currency values.

## Property Value

A `System.Int32` between 0 and 15 inclusive, which specifies the format of negative currency values.

## Description

The following table describes the valid values for this property. "\$" is used as the value for `System.Globalization.NumberFormatInfo.CurrencySymbol`, "-" is used as the value for `System.Globalization.NumberFormatInfo.NegativeSign`, and 999 represents any numeric value.

Value	Pattern
0	(\$999)
1	-\$999
2	\$-999
3	\$999-
4	(999\$)
5	-999\$
6	999-\$
7	999\$-
8	-999 \$

9	-\$ 999
10	999 \$-
11	\$ 999-
12	\$ -999
13	999- \$
14	(\$ 999)
15	(999 \$)

1  
2 The culture-invariant value for this property is 0.

### 3 Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	The value specified for a set operation is less than 0 or greater than 15.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

### 4 5 Example

6 The following example demonstrates the effects of different  
7 `System.Globalization.NumberFormatInfo.CurrencyNegativePattern` property  
8 values.

```

9
10 [C#]
11 using System;
12 using System.Globalization;
13 class Test {
14     public static void Main() {
15         NumberFormatInfo nfi = new NumberFormatInfo();
16         decimal myMoney = -99999999999999.00m;
17         for (int i = 0; i<=15; i++) {
18             nfi.CurrencyNegativePattern = i;
19             Console.WriteLine("pattern # {0}: {1}", i, myMoney.ToString("C", nfi));
20         }
21     }
22 }
```

```
1
2 The output is
3
4 pattern # 0: ($9,999,999,999,999.00)
5
6
7 pattern # 1: -$9,999,999,999,999.00
8
9
10 pattern # 2: $-9,999,999,999,999.00
11
12
13 pattern # 3: $9,999,999,999,999.00-
14
15
16 pattern # 4: (9,999,999,999,999.00$)
17
18
19 pattern # 5: -9,999,999,999,999.00$
20
21
22 pattern # 6: 9,999,999,999,999.00-$
23
24
25 pattern # 7: 9,999,999,999,999.00$-
26
27
28 pattern # 8: -9,999,999,999,999.00 $
29
30
31 pattern # 9: -$ 9,999,999,999,999.00
32
33
34 pattern # 10: 9,999,999,999,999.00 $-
35
36
37 pattern # 11: $ 9,999,999,999,999.00-
38
39
40 pattern # 12: $ -9,999,999,999,999.00
41
42
43 pattern # 13: 9,999,999,999,999.00- $
44
45
46 pattern # 14: ($ 9,999,999,999,999.00)
47
48
```

1 pattern # 15: (9,999,999,999,999.00 \$)  
2  
3

# NumberFormatInfo.CurrencyPositivePattern Property

```
[ILAsm]
.property int32 CurrencyPositivePattern { public hidebysig specialname
instance int32 get_CurrencyPositivePattern() public hidebysig specialname
instance void set_CurrencyPositivePattern(int32 value) }

[C#]
public int CurrencyPositivePattern { get; set; }
```

## Summary

Gets or sets the format of positive currency values.

## Property Value

A `System.Int32` between 0 and 3 inclusive, containing the format of positive currency values.

## Description

The following table describes the valid values for this property. "\$" is used as the value for `System.Globalization.NumberFormatInfo.CurrencySymbol`, and 999 represents any numeric value.

Value	Pattern
0	\$999
1	999\$
2	\$ 999
3	999 \$

The culture-invariant value for this property is 0.

## Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	The value specified for a set operation is less than 0 or greater than 3.

## System.InvalidOperationException

The current instance is read-only and a set operation was attempted.

### Example

The following example demonstrates the effects of different `System.Globalization.NumberFormatInfo.CurrencyPositivePattern` property values.

[C#]

```
using System;
using System.Globalization;
class Test {
    public static void Main() {
        NumberFormatInfo nfi = new NumberFormatInfo();
        decimal myMoney = 999999999999.00m;
        for (int i = 0; i<=3; i++) {
            nfi.CurrencyPositivePattern = i;
            Console.WriteLine("pattern # {0}: {1}", i, myMoney.ToString("C", nfi));
        }
    }
}
```

The output is

```
pattern # 0: $9,999,999,999,999.00
pattern # 1: 9,999,999,999,999.00$
pattern # 2: $ 9,999,999,999,999.00
pattern # 3: 9,999,999,999,999.00 $
```



# NumberFormatInfo.CurrencySymbol Property

```
[ILAsm]
.property string CurrencySymbol { public hidebysig specialname instance
string get_CurrencySymbol() public hidebysig specialname instance void
set_CurrencySymbol(string value) }

[C#]
public string CurrencySymbol { get; set; }
```

## Summary

Gets or sets the currency symbol.

## Property Value

A System.String containing the currency symbol.

## Description

The culture-invariant value for this property is the Unicode currency symbol 0x00a4.

## Exceptions

Exception	Condition
System.ArgumentNullException	The value specified for a set operation is a null reference.
System.InvalidOperationException	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.CurrentInfo Property

```
[ILAsm]  
.property class System.Globalization.NumberFormatInfo CurrentInfo { public  
hidebysig static specialname class System.Globalization.NumberFormatInfo  
get_CurrentInfo() }  
  
[C#]  
public static NumberFormatInfo CurrentInfo { get; }
```

## Summary

Gets a System.Globalization.NumberFormatInfo instance containing formatting information for the current system culture.

## Property Value

A read-only System.Globalization.NumberFormatInfo containing the settings for the current system culture.

## Description

This property is read-only.

# NumberFormatInfo.InvariantInfo Property

```
[ILAsm]
.property class System.Globalization.NumberFormatInfo InvariantInfo {
public hidebysig static specialname class
System.Globalization.NumberFormatInfo get_InvariantInfo() }

[C#]
public static NumberFormatInfo InvariantInfo { get; }
```

## Summary

Gets a `System.Globalization.NumberFormatInfo` instance containing formatting information that is culture-independent and does not change.

## Property Value

A read-only `System.Globalization.NumberFormatInfo` with property values which are universally supported. The property values of the returned `System.Globalization.NumberFormatInfo` are not impacted by changes to the current culture.

## Description

This property is read-only.

The following table lists the property values of the `System.Globalization.NumberFormatInfo` returned by this property.

Property	Default	Description
CurrencyDecimalDigits	2	The number of decimal places in currency values.
CurrencyDecimalSeparator	","	The string used as the decimal separator in currency values.
CurrencyGroupSeparator	","	The string used to separate groups of digits to the left of the decimal point in currency values.
CurrencyGroupSizes	3	The number of digits in each group to the left of the decimal point in currency values.
CurrencyNegativePattern	0	The format of negative currency values.
CurrencyPositivePattern	0	The format of positive currency values.
CurrencySymbol	0x00a4	The Unicode currency symbol.

NaNSymbol	"NaN"	The string used to represent undefined floating-point values.
NegativeInfinitySymbol	"-Infinity"	The string used to represent negative infinities.
NegativeSign	"-"	The string used to indicate negative values.
NumberDecimalDigits	2	The default number of decimal places.
NumberDecimalSeparator	"."	The string used as the decimal separator.
NumberGroupSeparator	","	The string used to separate groups of digits to the left of the decimal point.
NumberGroupSizes	3	The number of digits in each group to the left of the decimal point.
NumberNegativePattern	1	The format of negative values.
PercentDecimalDigits	2	The default number of decimal places in percent values.
PercentDecimalSeparator	"."	The string used as the decimal separator in percent values.
PercentGroupSeparator	","	The string used to separate groups of digits to the left of the decimal point in percent values.
PercentGroupSizes	3	The number of digits in each group to the left of the decimal in percent values.
PercentNegativePattern	0	The format of negative percent values.
PercentPositivePattern	0	The format of positive percent values.
PercentSymbol	"%"	The percent symbol.
PerMilleSymbol	"‰"	The per mille symbol.
PositiveInfinitySymbol	"Infinity"	The string used to represent positive infinities.
PositiveSign	"+"	The string used to indicate positive values.

1

2

# 1 NumberFormatInfo.IsReadOnly Property

```
2 [ILAsm]  
3 .property bool IsReadOnly { public hidebysig specialname instance bool  
4 get_IsReadOnly() }  
  
5 [C#]  
6 public bool IsReadOnly { get; }
```

## 7 Summary

8 Gets a value indicating whether the current instance is read-only.

## 9 Property Value

10 true if the current instance is read-only; otherwise false.

## 11 Description

12 This property is read-only.

13  
14 [*Note:* Attempting to perform an assignment to a property of a read-only  
15 System.Globalization.NumberFormatInfo causes a  
16 System.InvalidOperationException.]  
17  
18

# NumberFormatInfo.NaNSymbol Property

```
[ILAsm]
.property string NaNSymbol { public hidebysig specialname instance string
get_NaNSymbol() public hidebysig specialname instance void
set_NaNSymbol(string value) }

[C#]
public string NaNSymbol { get; set; }
```

## Summary

Gets or sets the symbol that represents NaN (Not-a-Number) floating-point values.

## Property Value

A System.String containing the symbol for NaN values.

## Description

The culture-invariant value for this property is "NaN".

[Note: For more information on NaN values, see System.Double or System.Single.]

## Exceptions

Exception	Condition
System.ArgumentNullException	The value specified for a set operation is a null reference.
System.InvalidOperationException	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.NegativeInfinitySymbol Property

```
[ILAsm]
.property string NegativeInfinitySymbol { public hidebysig specialname
instance string get_NegativeInfinitySymbol() public hidebysig specialname
instance void set_NegativeInfinitySymbol(string value) }

[C#]
public string NegativeInfinitySymbol { get; set; }
```

## Summary

Gets or sets the symbol that represents negative infinity.

## Property Value

A `System.String` containing the symbol for negative infinity.

## Description

The culture-invariant value for this property is "-Infinity".

[*Note:* For more information on negative infinity, see `System.Double` or `System.Single`.]

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The value specified for a set operation is a null reference.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.



# NumberFormatInfo.NegativeSign Property

```
[ILAsm]
.property string NegativeSign { public hidebysig specialname instance
string get_NegativeSign() public hidebysig specialname instance void
set_NegativeSign(string value) }

[C#]
public string NegativeSign { get; set; }
```

## Summary

Gets or sets the symbol used to represent negative values.

## Property Value

A System.String containing the symbol that indicates a value is negative.

## Description

The culture-invariant value for this property is "-".

## Exceptions

Exception	Condition
System.ArgumentNullException	The value specified for a set operation is a null reference.
System.InvalidOperationException	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.NumberDecimalDigits

## Property

```
[ILAsm]
.property int32 NumberDecimalDigits { public hidebysig specialname
instance int32 get_NumberDecimalDigits() public hidebysig specialname
instance void set_NumberDecimalDigits(int32 value) }

[C#]
public int NumberDecimalDigits { get; set; }
```

### Summary

Gets or sets the number of decimal places for numeric values.

### Property Value

A System.Int32 containing the number of decimal places for numeric values.

### Description

The culture-invariant value for this property is 2.

### Exceptions

Exception	Condition
System.ArgumentOutOfRangeException	The value specified for a set operation is less than 0 or greater than 99.
System.InvalidOperationException	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.NumberDecimalSeparator Property

```
[ILAsm]
.property string NumberDecimalSeparator { public hidebysig specialname
instance string get_NumberDecimalSeparator() public hidebysig specialname
instance void set_NumberDecimalSeparator(string value) }

[C#]
public string NumberDecimalSeparator { get; set; }
```

## Summary

Gets or sets the symbol used as the decimal separator for numeric values.

## Property Value

A System.String containing the decimal separator.

## Description

The culture-invariant value for this property is ".".

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The value specified for a set operation is a null reference.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.NumberGroupSeparator Property

```
[ILAsm]
.property string NumberGroupSeparator { public hidebysig specialname
instance string get_NumberGroupSeparator() public hidebysig specialname
instance void set_NumberGroupSeparator(string value) }

[C#]
public string NumberGroupSeparator { get; set; }
```

## Summary

Gets or sets the symbol used to separate groups of digits to the left of the decimal point for numeric values.

## Property Value

A `System.String` containing the group separator.

## Description

The culture-invariant value for this property is ",".

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The value specified for a set operation is a null reference.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.NumberGroupSizes

## Property

```
[ILAsm]
.property int32[] NumberGroupSizes { public hidebysig specialname instance
int32[] get_NumberGroupSizes() public hidebysig specialname instance void
set_NumberGroupSizes(int32[] value) }

[C#]
public int[] NumberGroupSizes { get; set; }
```

### Summary

Gets or sets the number of digits in each group to the left of the decimal point for numeric values.

### Property Value

A `System.Int32` array containing elements that define the number of digits in each group in numeric values.

### Description

All elements of the array except the last are required to be between 1 and 9, inclusive. The last element can be 0.

The first element of the array defines the number of elements in the first group of digits located immediately to the left of the `System.Globalization.NumberFormatInfo.NumberDecimalSeparator`. Each subsequent element refers to the next group of digits located to the left of the previous group. If the last element of the array is not zero, any remaining digits are grouped based on the last element of the array. If the last element is zero, the remaining digits are not grouped.

The culture-invariant value for this property is an array with a single element containing the value 3.

### Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The array specified for a set operation is a null reference.
<b>System.ArgumentOutOfRangeException</b>	One of the elements in the array specified for a set operation is not between 0 and 9.

## System.InvalidOperationException

The current instance is read-only and a set operation was attempted.

### Example

The following example demonstrates the effects of different `System.Globalization.NumberFormatInfo.NumberGroupSizes` property values.

[C#]

```
using System;
using System.Globalization;
class Test {
    public static void Main() {
        NumberFormatInfo nfi = new NumberFormatInfo();

        decimal data = 9999999994444333221.00m;
        nfi.NumberGroupSizes = new int[] {1,2,3,4,0};
        Console.WriteLine("{0}",data.ToString("N",nfi));

        data = 123456789123456.78m;
        nfi.NumberGroupSizes = new int[] {3};
        Console.WriteLine("{0}",data.ToString("N",nfi));

        nfi.NumberGroupSizes = new int[] {3,0};
        Console.WriteLine("{0}",data.ToString("N",nfi));
    }
}
```

The output is

```
999999999,4444,333,22,1.00
```

```
123,456,789,123,456.78
```

```
123456789123,456.78
```

# NumberFormatInfo.NumberNegativePattern

## Property

```
[ILAsm]  
.property int32 NumberNegativePattern { public hidebysig specialname  
instance int32 get_NumberNegativePattern() public hidebysig specialname  
instance void set_NumberNegativePattern(int32 value) }  
  
[C#]  
public int NumberNegativePattern { get; set; }
```

### Summary

Gets or sets the format of negative values.

### Property Value

A `System.Int32` between 0 and 4 inclusive that specifies the format of negative values.

### Description

The following table describes the valid values for this property. "-" is used as the value for `System.Globalization.NumberFormatInfo.NegativeSign`, and 999 represents any numeric value.

Value	Pattern
0	(999)
1	-999
2	- 999
3	999-
4	999 -

The culture-invariant value for this property is 1.

### Exceptions

Exception	Condition
<code>System.ArgumentOutOfRangeException</code>	The value specified for a set operation is less

	than 0 or greater than 4.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36

## Example

The following example demonstrates the effects of different `System.Globalization.NumberFormatInfo.NumberNegativePattern` property values.

[C#]

```
using System;
using System.Globalization;
class Test {
    public static void Main() {
        NumberFormatInfo nfi = new NumberFormatInfo();
        Double data = -999999999999.00;
        for (int i = 0; i<=4; i++) {
            nfi.NumberNegativePattern = i;
            Console.WriteLine("pattern # {0}: {1}",i,data.ToString("N",nfi));
        }
    }
}
```

The output is

```
pattern # 0: (9,999,999,999,999.00)
pattern # 1: -9,999,999,999,999.00
pattern # 2: - 9,999,999,999,999.00
pattern # 3: 9,999,999,999,999.00-
pattern # 4: 9,999,999,999,999.00 -
```



# NumberFormatInfo.PercentDecimalDigits

## Property

```
[ILAsm]
.property int32 PercentDecimalDigits { public hidebysig specialname
instance int32 get_PercentDecimalDigits() public hidebysig specialname
instance void set_PercentDecimalDigits(int32 value) }

[C#]
public int PercentDecimalDigits { get; set; }
```

### Summary

Gets or sets the number of decimal places in percent values.

### Property Value

A `System.Int32` containing the number of decimal places in percent values.

### Description

The culture-invariant value for this property is 2.

### Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	The value specified for a set operation is less than 0 or greater than 99.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.PercentDecimalSeparator Property

```
[ILAsm]
.property string PercentDecimalSeparator { public hidebysig specialname
instance string get_PercentDecimalSeparator() public hidebysig specialname
instance void set_PercentDecimalSeparator(string value) }

[C#]
public string PercentDecimalSeparator { get; set; }
```

## Summary

Gets or sets the symbol used as the decimal separator in percent values.

## Property Value

A `System.String` containing the decimal separator used in percent values.

## Description

The culture-invariant value for this property is ".".

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The value specified for a set operation is a null reference.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.PercentGroupSeparator Property

```
[ILAsm]
.property string PercentGroupSeparator { public hidebysig specialname
instance string get_PercentGroupSeparator() public hidebysig specialname
instance void set_PercentGroupSeparator(string value) }

[C#]
public string PercentGroupSeparator { get; set; }
```

## Summary

Gets or sets the symbol used to separate groups of digits to the left of the decimal point in percent values.

## Property Value

A `System.String` containing the group separator symbol used in percent values.

## Description

The culture-invariant value for this property is ",".

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The value specified for a set operation is a null reference.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.PercentGroupSizes

## Property

```
[ILAsm]
.property int32[] PercentGroupSizes { public hidebysig specialname
instance int32[] get_PercentGroupSizes() public hidebysig specialname
instance void set_PercentGroupSizes(int32[] value) }

[C#]
public int[] PercentGroupSizes { get; set; }
```

### Summary

Gets or sets the number of digits in each group to the left of the decimal point in percent values.

### Property Value

A `System.Int32` array containing elements that define the number of digits in each group in percent values.

### Description

All elements of the array except the last are required to be between 1 and 9, inclusive. The last element can be 0.

The first element of the array defines the number of elements in the first group of digits located immediately to the left of the `System.Globalization.NumberFormatInfo.PercentDecimalSeparator`. Each subsequent element refers to the next group of digits located to the left of the previous group. If the last element of the array is not zero, any remaining digits are grouped based on the last element of the array. If the last element is zero, the remaining digits are not grouped.

The culture-invariant value for this property is an array with a single element containing the value 3.

### Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The array specified for a set operation is a null reference.
<b>System.ArgumentException</b>	One of the elements in the array specified for a set operation is not between 0 and 9.  -or-

	The array contains an element, other than the last element, that is set to 0.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

1

## 2 Example

3 The following example demonstrates the effects of different  
4 System.Globalization.NumberFormatInfo.PercentGroupSizes property values.

5  
6 [C#]

```

7 using System;
8 using System.Globalization;
9 class Test {
10     public static void Main() {
11         NumberFormatInfo nfi = new NumberFormatInfo();
12
13         decimal data = 9999999994444333221.00m;
14         nfi.PercentGroupSizes = new int[] {1,2,3,4,0};
15         Console.WriteLine("{0}",data.ToString("P",nfi));
16
17         data = 123456789123456.78m;
18         nfi.PercentGroupSizes = new int[] {3};
19         Console.WriteLine("{0}",data.ToString("P",nfi));
20
21         nfi.PercentGroupSizes = new int[] {3,0};
22         Console.WriteLine("{0}",data.ToString("P",nfi));
23     }
24 }

```

25  
26 The output is

```

27
28 999999999944,4433,322,10,0.00 %
29
30
31 12,345,678,912,345,678.00 %
32
33
34 12345678912345,678.00 %
35

```

36

# NumberFormatInfo.PercentNegativePattern Property

```
[ILAsm]
.property int32 PercentNegativePattern { public hidebysig specialname
instance int32 get_PercentNegativePattern() public hidebysig specialname
instance void set_PercentNegativePattern(int32 value) }

[C#]
public int PercentNegativePattern { get; set; }
```

## Summary

Gets or sets the format of negative percent values.

## Property Value

A `System.Int32` between 0 and 2 inclusive that specifies the format of negative percent values.

## Description

The following table describes the valid values for this property. "%" is used as the value for `System.Globalization.NumberFormatInfo.PercentSymbol`, "-" is used as the value for `System.Globalization.NumberFormatInfo.NegativeSign`, and 999 represents any numeric value.

Value	Pattern
0	-999 %
1	-999%
2	-%999

The culture-invariant value for this property is 0.

## Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	The value specified for a set operation is less than 0 or greater than 2.

## System.InvalidOperationException

The current instance is read-only and a set operation was attempted.

### Example

The following example demonstrates the effects of different `System.Globalization.NumberFormatInfo.PercentNegativePattern` property values.

[C#]

```
using System;
using System.Globalization;
class Test {
    public static void Main() {
        NumberFormatInfo nfi = new NumberFormatInfo();
        decimal data = -.9900m;
        for (int i = 0; i<=2; i++) {
            nfi.PercentNegativePattern = i;
            Console.WriteLine("pattern # {0}: {1}", i, data.ToString("P", nfi));
        }
    }
}
```

The output is

```
pattern # 0: -99.00 %
pattern # 1: -99.00%
pattern # 2: -%99.00
```

# NumberFormatInfo.PercentPositivePattern Property

```
[ILAsm]
.property int32 PercentPositivePattern { public hidebysig specialname
instance int32 get_PercentPositivePattern() public hidebysig specialname
instance void set_PercentPositivePattern(int32 value) }

[C#]
public int PercentPositivePattern { get; set; }
```

## Summary

Gets or sets the format of positive percent values.

## Property Value

A `System.Int32` between 0 and 2 inclusive that specifies the format of positive percent values.

## Description

The following table describes the valid values for this property. "%" is used as the value for `System.Globalization.NumberFormatInfo.PercentSymbol`, and 999 represents a numeric value.

Value	Pattern
0	999 %
1	999%
2	%999

The culture-invariant value for this property is 0.

## Exceptions

Exception	Condition
<b>System.ArgumentOutOfRangeException</b>	The value specified for a set operation is less than 0 or greater than 2.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set



	operation was attempted.
--	--------------------------

## Example

The following example demonstrates the effects of different `System.Globalization.NumberFormatInfo.PercentPositivePattern` property values.

[C#]

```
using System;
using System.Globalization;
class Test {
    public static void Main() {
        NumberFormatInfo nfi = new NumberFormatInfo();
        decimal data = .9900m;
        for (int i = 0; i<=2; i++) {
            nfi.PercentPositivePattern = i;
            Console.WriteLine("pattern # {0}: {1}", i, data.ToString("P", nfi));
        }
    }
}
```

The output is

```
pattern # 0: 99.00 %
pattern # 1: 99.00%
pattern # 2: %99.00
```

# NumberFormatInfo.PercentSymbol Property

```
[ILAsm]
.property string PercentSymbol { public hidebysig specialname instance
string get_PercentSymbol() public hidebysig specialname instance void
set_PercentSymbol(string value) }

[C#]
public string PercentSymbol { get; set; }
```

## Summary

Gets or sets the symbol that represents percentage values.

## Property Value

A `System.String` containing the percent symbol.

## Description

The culture-invariant value for this property is "%".

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The value specified for a set operation is a null reference.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.PerMilleSymbol Property

```
[ILAsm]
.property string PerMilleSymbol { public hidebysig specialname instance
string get_PerMilleSymbol() public hidebysig specialname instance void
set_PerMilleSymbol(string value) }

[C#]
public string PerMilleSymbol { get; set; }
```

## Summary

Gets or sets the per mille symbol.

## Property Value

A System.String containing the per mille symbol.

## Description

The culture-invariant value for this property is "%<sub>oo</sub>".

## Exceptions

Exception	Condition
System.ArgumentNullException	The value specified for a set operation is a null reference.
System.InvalidOperationException	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.PositiveInfinitySymbol Property

```
[ILAsm]
.property string PositiveInfinitySymbol { public hidebysig specialname
instance string get_PositiveInfinitySymbol() public hidebysig specialname
instance void set_PositiveInfinitySymbol(string value) }

[C#]
public string PositiveInfinitySymbol { get; set; }
```

## Summary

Gets or sets the symbol that represents positive infinity.

## Property Value

A `System.String` containing the symbol for positive infinity.

## Description

The culture-invariant value for this property is "Infinity".

[*Note:* For more information on positive infinity, see `System.Double` or `System.Single`.]

## Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	The value specified for a set operation is a null reference.
<b>System.InvalidOperationException</b>	The current instance is read-only and a set operation was attempted.

# NumberFormatInfo.PositiveSign Property

```
[ILAsm]
.property string PositiveSign { public hidebysig specialname instance
string get_PositiveSign() public hidebysig specialname instance void
set_PositiveSign(string value) }

[C#]
public string PositiveSign { get; set; }
```

## Summary

Gets or sets the symbol used to represent positive values.

## Property Value

A System.String containing the symbol that indicates the value is positive.

## Description

The culture-invariant value for this property is "+".

## Exceptions

Exception	Condition
System.ArgumentNullException	The value specified for a set operation is a null reference
System.InvalidOperationException	The current instance is read-only and a set operation was attempted.