

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]
- *Version*: 1.0.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** may be created to provide customized formatting information.]

1 NumberFormatInfo() Constructor

```
2 [ILASM]  
3 public rtspecialname specialname instance void .ctor()  
4 [C#]  
5 public NumberFormatInfo()
```

6 Summary

7 Constructs and initializes a new instance of the
8 **System.Globalization.NumberFormatInfo** class.

9 Description

10 The new instance is not read-only, and is otherwise identical to the
11 **System.Globalization.NumberFormatInfo** instance returned by the
12 **System.Globalization.NumberFormatInfo.InvariantInfo**
13 property.

14

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

10 A **System.Object** that is a copy of the current instance.

11 Description

12 The **System.Globalization.NumberFormatInfo.Clone** method
13 returns a new instance of
14 **System.Globalization.NumberFormatInfo** with property values
15 that are equal to the property values of the current instance. If the
16 current instance is read-only, the clone is also read-only.

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

20

1 NumberFormatInfo.GetFormat(System.Type) Method

2

```
3 [ILASM]  
4 .method public final hidebysig virtual object  
5 GetFormat(class System.Type formatType)  
6  
7 [C#]  
8 public object GetFormat(Type formatType)
```

8 Summary

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

11 Parameters

12
13

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

14
15
16

15 Return Value

17 A formatting object of the specified **System.Type**. If no formatting
18 object of the specified type is available, or *formatType* is a null
19 reference, returns the formatting object for the current culture.

20 Description

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

23

1 NumberFormatInfo.ReadOnly(System.Global 2 ization.NumberFormatInfo) Method

```
3 [ILASM]  
4 .method public hidebysig static class  
5 System.Globalization.NumberFormatInfo ReadOnly(class  
6 System.Globalization.NumberFormatInfo nfi)  
  
7 [C#]  
8 public static NumberFormatInfo ReadOnly(NumberFormatInfo  
9 nfi)
```

10 Summary

11 Creates a read-only copy of the specified
12 **System.Globalization.NumberFormatInfo** instance.

13 Parameters

14
15

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

16
17
18

Return Value

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

21 Exceptions

22
23

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

24
25
26

1 NumberFormatInfo.CurrencyDecimalDigits Property

```
3 [ILASM]
4 .property int32 CurrencyDecimalDigits { public hidebysig
5 specialname instance int32 get_CurrencyDecimalDigits()
6 public hidebysig specialname instance void
7 set_CurrencyDecimalDigits(int32 value) }

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

10 Summary

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

12 Property Value

13

14 A **System.Int32** containing the number of decimal places in currency
15 values.

16 Description

17 The culture-invariant value for this property is 2.

18 Exceptions

19

20

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.

21

22

23

1 NumberFormatInfo.CurrencyDecimalSepar 2 ator Property

```
3 [ILASM]  
4 .property string CurrencyDecimalSeparator { public  
5 hidebysig specialname instance string  
6 get_CurrencyDecimalSeparator() public hidebysig specialname  
7 instance void set_CurrencyDecimalSeparator(string value) }  
  
8 [C#]  
9 public string CurrencyDecimalSeparator { get; set; }
```

10 Summary

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

13 Property Value

14

15 A **System.String** containing the decimal separator used in currency
16 values.

17 Description

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

19 Exceptions

20

21

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.

22

23

24

1 NumberFormatInfo.CurrencyGroupSepara 2 tor Property

```
3 [ILASM]  
4 .property string CurrencyGroupSeparator { public hideby sig  
5 specialname instance string get_CurrencyGroupSeparator()  
6 public hideby sig specialname instance void  
7 set_CurrencyGroupSeparator(string value) }  
  
8 [C#]  
9 public string CurrencyGroupSeparator { get; set; }
```

10 Summary

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

13 Property Value

14

15 A **System.String** containing the group separator used in currency
16 values.

17 Description

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

19 Exceptions

20

21

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.

22

23

24

1 NumberFormatInfo.CurrencyGroupSizes

2 Property

```
3 [ILASM]
4 .property class System.Int32[] CurrencyGroupSizes { public
5 hidebysig specialname instance class System.Int32[]
6 get_CurrencyGroupSizes() public hidebysig specialname
7 instance void set_CurrencyGroupSizes(class System.Int32[]
8 value) }
9
10 [C#]
11 public int[] CurrencyGroupSizes { get; set; }
```

11 Summary

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

14 Property Value

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

18 Description

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

21
22 The first element of the array defines the number of elements in the
23 first group of digits located immediately to the left of the
24 **System.Globalization.NumberFormatInfo.CurrencyDecimalSepa**
25 **rator**. Each subsequent element refers to the next group of digits
26 located to the left of the previous group. If the last element of the
27 array is not zero, any remaining digits are grouped based on the last
28 element of the array. If the last element is zero, the remaining digits
29 are not grouped.

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

33 Exceptions

34
35

Exception	Condition
System.ArgumentNullException	The array specified for a set operation is a null reference.
System.ArgumentOutOfRangeException	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.

1
2
3

Example

4
5
6
7
8

The following example demonstrates the effects of different **System.Globalization.NumberFormatInfo.CurrencyGroupSizes** property values.

9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

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

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

        myMoney = 123456789123456.78m;
        nfi.CurrencyGroupSizes = new int[] {3};
        Console.WriteLine("{0}",myMoney.ToString("C",nfi));

        nfi.CurrencyGroupSizes = new int[] {3,0};
        Console.WriteLine("{0}",myMoney.ToString("C",nfi));
    }
}
```

29
30
31
32
33
34
35
36

The output is
\$999999999,4444,333,22,1.00

\$123,456,789,123,456.78

1 \$123456789123,456.78
2
3

1 NumberFormatInfo.CurrencyNegativePattern Property

```
3 [ILASM]  
4 .property int32 CurrencyNegativePattern { public hidebysig  
5 specialname instance int32 get_CurrencyNegativePattern()  
6 public hidebysig specialname instance void  
7 set_CurrencyNegativePattern(int32 value) }  
  
8 [C#]  
9 public int CurrencyNegativePattern { get; set; }
```

10 Summary

11 Gets or sets the format of negative currency values.

12 Property Value

13

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

16 Description

17 The following table describes the valid values for this property. "\$" is
18 used as the value for
19 **System.Globalization.NumberFormatInfo.CurrencySymbol**, "-" is
20 used as the value for
21 **System.Globalization.NumberFormatInfo.NegativeSign**, and 999
22 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**

4
5

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

6
7 **Example**

8

9 The following example demonstrates the effects of different
10 **System.Globalization.NumberFormatInfo.CurrencyNegativePattern**
11 **ern** property values.

12
13

[C#]

```

14 using System;
15 using System.Globalization;
16 class Test {
17     public static void Main() {
18         NumberFormatInfo nfi = new NumberFormatInfo();
19         decimal myMoney = -999999999999.00m;
20         for (int i = 0; i<=15; i++) {
21             nfi.CurrencyNegativePattern = i;
22             Console.WriteLine("pattern # {0}:
23 {1}", i, myMoney.ToString("C", nfi));
24         }
25     }
26 }
27

```

28 The output is

29
30 pattern # 0: (\$9,999,999,999,999.00)
31
32

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

45

1 NumberFormatInfo.CurrencyPositivePattern Property

```
3 [ILASM]
4 .property int32 CurrencyPositivePattern { public hidebysig
5 specialname instance int32 get_CurrencyPositivePattern()
6 public hidebysig specialname instance void
7 set_CurrencyPositivePattern(int32 value) }

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

10 Summary

11 Gets or sets the format of positive currency values.

12 Property Value

13

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

16 Description

17 The following table describes the valid values for this property. "\$" is
18 used as the value for
19 **System.Globalization.NumberFormatInfo.CurrencySymbol**, and
20 999 represents any numeric value.

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

21
22 The culture-invariant value for this property is 0.

23 Exceptions

24
25

Exception	Condition
System.ArgumentOutOfRangeException	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.

1
2
3

Example

4 The following example demonstrates the effects of different
5 **System.Globalization.NumberFormatInfo.CurrencyPositivePatte**
6 **rn** property values.

```
7 [C#]  
8  
9 using System;  
10 using System.Globalization;  
11 class Test {  
12     public static void Main() {  
13         NumberFormatInfo nfi = new NumberFormatInfo();  
14         decimal myMoney = 9999999999999.00m;  
15         for (int i = 0; i<=3; i++) {  
16             nfi.CurrencyPositivePattern = i;  
17             Console.WriteLine("pattern # {0}:  
18 {1}", i, myMoney.ToString("C", nfi));  
19         }  
20     }  
21 }  
22
```

23 The output is

```
24  
25 pattern # 0: $9,999,999,999,999.00  
26  
27  
28 pattern # 1: 9,999,999,999,999.00$  
29  
30  
31 pattern # 2: $ 9,999,999,999,999.00  
32  
33  
34 pattern # 3: 9,999,999,999,999.00 $  
35
```

36

1 NumberFormatInfo.CurrencySymbol

2 Property

```
3 [ILASM]
4 .property string CurrencySymbol { public hidebysig
5 specialname instance string get_CurrencySymbol() public
6 hidebysig specialname instance void
7 set_CurrencySymbol(string value) }

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

10 Summary

11 Gets or sets the currency symbol.

12 Property Value

13

14 A **System.String** containing the currency symbol.

15 Description

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

18 Exceptions

19

20

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.

21

22

23

1 NumberFormatInfo.CurrentInfo Property

```
2 [ILASM]  
3 .property class System.Globalization.NumberFormatInfo  
4 CurrentInfo { public hidebysig static specialname class  
5 System.Globalization.NumberFormatInfo get_CurrentInfo() }  
6 [C#]  
7 public static NumberFormatInfo CurrentInfo { get; }
```

8 Summary

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

11 Property Value

12

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

15 Description

16 This property is read-only.

17

1 NumberFormatInfo.InvariantInfo

2 Property

```
3 [ILASM]
4 .property class System.Globalization.NumberFormatInfo
5 InvariantInfo { public hidebysig static specialname class
6 System.Globalization.NumberFormatInfo get_InvariantInfo() }
7
8 [C#]
9 public static NumberFormatInfo InvariantInfo { get; }
```

9 Summary

10 Gets a **System.Globalization.NumberFormatInfo** instance
11 containing formatting information that is culture-independent and does
12 not change.

13 Property Value

14
15 A read-only **System.Globalization.NumberFormatInfo** with
16 property values which are universally supported. The property values
17 of the returned **System.Globalization.NumberFormatInfo** are not
18 impacted by changes to the current culture.

19 Description

20 This property is read-only.

21
22 The following table lists the property values of the
23 **System.Globalization.NumberFormatInfo** returned by this
24 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
4 instance bool get_IsReadOnly() }
5
6 [C#]
7 public bool IsReadOnly { get; }
```

7 Summary

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

9 Property Value

10

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

12 Description

13 This property is read-only.

14

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

18

1 NumberFormatInfo.NaNSymbol Property

```
2 [ILASM]
3 .property string NaNSymbol { public hidebysig specialname
4 instance string get_NaNSymbol() public hidebysig
5 specialname instance void set_NaNSymbol(string value) }
6
7 [C#]
8 public string NaNSymbol { get; set; }
```

8 Summary

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

11 Property Value

12

13 A **System.String** containing the symbol for NaN values.

14 Description

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

16

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

19 Exceptions

20

21

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.

22

23

24

1 NumberFormatInfo.NegativeInfinitySymbol 2 of Property

```
3 [ILASM]  
4 .property string NegativeInfinitySymbol { public hidebysig  
5 specialname instance string get_NegativeInfinitySymbol()  
6 public hidebysig specialname instance void  
7 set_NegativeInfinitySymbol(string value) }  
  
8 [C#]  
9 public string NegativeInfinitySymbol { get; set; }
```

10 Summary

11 Gets or sets the symbol that represents negative infinity.

12 Property Value

13

14 A **System.String** containing the symbol for negative infinity.

15 Description

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

17

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

20 Exceptions

21

22

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.

23

24

25

1 NumberFormatInfo.NegativeSign Property

```
2 [ILASM]
3 .property string NegativeSign { public hidebysig
4 specialname instance string get_NegativeSign() public
5 hidebysig specialname instance void set_NegativeSign(string
6 value) }
7
8 [C#]
9 public string NegativeSign { get; set; }
```

9 Summary

10 Gets or sets the symbol used to represent negative values.

11 Property Value

12

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

15 Description

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

17 Exceptions

18

19

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.

20

21

22

1 NumberFormatInfo.NumberDecimalDigits

2 Property

```
3 [ILASM]
4 .property int32 NumberDecimalDigits { public hidebyref
5 specialname instance int32 get_NumberDecimalDigits() public
6 hidebyref specialname instance void
7 set_NumberDecimalDigits(int32 value) }

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

10 Summary

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

12 Property Value

13

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

16 Description

17 The culture-invariant value for this property is 2.

18 Exceptions

19

20

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.

21

22

23

1 NumberFormatInfo.NumberDecimalSeparator Property

```
3 [ILASM]
4 .property string NumberDecimalSeparator { public hidebysig
5 specialname instance string get_NumberDecimalSeparator()
6 public hidebysig specialname instance void
7 set_NumberDecimalSeparator(string value) }

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

10 Summary

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

13 Property Value

14

15 A **System.String** containing the decimal separator.

16 Description

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

18 Exceptions

19

20

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.

21

22

23

1 NumberFormatInfo.NumberGroupSeparat 2 or Property

```
3 [ILASM]  
4 .property string NumberGroupSeparator { public hidebysig  
5 specialname instance string get_NumberGroupSeparator()  
6 public hidebysig specialname instance void  
7 set_NumberGroupSeparator(string value) }  
8  
9 [C#]  
10 public string NumberGroupSeparator { get; set; }
```

10 Summary

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

13 Property Value

14

15 A **System.String** containing the group separator.

16 Description

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

18 Exceptions

19

20

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.

21

22

23

1 NumberFormatInfo.NumberGroupSizes

2 Property

```
3 [ILASM]
4 .property class System.Int32[] NumberGroupSizes { public
5 hidebysig specialname instance class System.Int32[]
6 get_NumberGroupSizes() public hidebysig specialname
7 instance void set_NumberGroupSizes(class System.Int32[]
8 value) }
9
10 [C#]
11 public int[] NumberGroupSizes { get; set; }
```

11 Summary

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

14 Property Value

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

18 Description

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

21
22 The first element of the array defines the number of elements in the
23 first group of digits located immediately to the left of the
24 **System.Globalization.NumberFormatInfo.NumberDecimalSepar-**
25 **ator**. Each subsequent element refers to the next group of digits
26 located to the left of the previous group. If the last element of the
27 array is not zero, any remaining digits are grouped based on the last
28 element of the array. If the last element is zero, the remaining digits
29 are not grouped.

30
31 The culture-invariant value for this property is an array with a single
32 element containing the value 3.

33 Exceptions

34
35

Exception	Condition
System.ArgumentNullException	The array specified for a set operation is a null reference.
System.ArgumentOutOfRangeException	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.

1
2
3

Example

4
5
6
7
8

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

9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

```
[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));
    }
}
```

28
29
30
31
32
33
34
35

The output is

999999999,4444,333,22,1.00

123,456,789,123,456.78

1 123456789123,456.78
2
3

1 NumberFormatInfo.NumberNegativePattern Property

```
3 [ILASM]
4 .property int32 NumberNegativePattern { public hidebysig
5 specialname instance int32 get_NumberNegativePattern()
6 public hidebysig specialname instance void
7 set_NumberNegativePattern(int32 value) }

8 [C#]
9 public int NumberNegativePattern { get; set; }
```

10 Summary

11 Gets or sets the format of negative values.

12 Property Value

13

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

16 Description

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

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

21
22 The culture-invariant value for this property is 1.

23 Exceptions

24

25

Exception	Condition
System.ArgumentOutOfRangeException	The value specified for a set operation is less than 0 or greater than 4.
System.InvalidOperationException	The current instance is read-only and a set operation was attempted.

1
2
3

Example

4 The following example demonstrates the effects of different
5 **System.Globalization.NumberFormatInfo.NumberNegativePatte**
6 **rn** property values.

```
7 [C#]  
8  
9 using System;  
10 using System.Globalization;  
11 class Test {  
12     public static void Main() {  
13         NumberFormatInfo nfi = new NumberFormatInfo();  
14         Double data = -9999999999999.00;  
15         for (int i = 0; i<=4; i++) {  
16             nfi.NumberNegativePattern = i;  
17             Console.WriteLine("pattern # {0}:  
18 {1}", i, data.ToString("N", nfi));  
19         }  
20     }  
21 }  
22
```

23 The output is
24
25 pattern # 0: (9,999,999,999,999.00)
26
27
28 pattern # 1: -9,999,999,999,999.00
29
30
31 pattern # 2: - 9,999,999,999,999.00
32
33
34 pattern # 3: 9,999,999,999,999.00-
35
36

1 pattern # 4: 9,999,999,999,999.00 -
2
3

1 NumberFormatInfo.PercentDecimalDigits 2 Property

```
3 [ILASM]  
4 .property int32 PercentDecimalDigits { public hidebysig  
5 specialname instance int32 get_PercentDecimalDigits()  
6 public hidebysig specialname instance void  
7 set_PercentDecimalDigits(int32 value) }  
  
8 [C#]  
9 public int PercentDecimalDigits { get; set; }
```

10 Summary

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

12 Property Value

13

14 A **System.Int32** containing the number of decimal places in percent
15 values.

16 Description

17 The culture-invariant value for this property is 2.

18 Exceptions

19

20

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.

21

22

23

1 NumberFormatInfo.PercentDecimalSepara 2 tor Property

```
3 [ILASM]  
4 .property string PercentDecimalSeparator { public hideby sig  
5 specialname instance string get_PercentDecimalSeparator()  
6 public hideby sig specialname instance void  
7 set_PercentDecimalSeparator(string value) }  
  
8 [C#]  
9 public string PercentDecimalSeparator { get; set; }
```

10 Summary

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

13 Property Value

14

15 A **System.String** containing the decimal separator used in percent
16 values.

17 Description

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

19 Exceptions

20

21

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.

22

23

24

1 NumberFormatInfo.PercentGroupSeparator 2 Property

```
3 [ILASM]  
4 .property string PercentGroupSeparator { public hidebysig  
5 specialname instance string get_PercentGroupSeparator()  
6 public hidebysig specialname instance void  
7 set_PercentGroupSeparator(string value) }  
  
8 [C#]  
9 public string PercentGroupSeparator { get; set; }
```

10 Summary

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

13 Property Value

14
15 A **System.String** containing the group separator symbol used in
16 percent values.

17 Description

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

19 Exceptions

20
21

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.

22
23
24

1 NumberFormatInfo.PercentGroupSizes 2 Property

```
3 [ILASM]  
4 .property class System.Int32[] PercentGroupSizes { public  
5 hidebysig specialname instance class System.Int32[]  
6 get_PercentGroupSizes() public hidebysig specialname  
7 instance void set_PercentGroupSizes(class System.Int32[]  
8 value) }
```

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

11 Summary

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

14 Property Value

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

18 Description

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

21
22 The first element of the array defines the number of elements in the
23 first group of digits located immediately to the left of the
24 **System.Globalization.NumberFormatInfo.PercentDecimalSepar-**
25 **ator**. Each subsequent element refers to the next group of digits
26 located to the left of the previous group. If the last element of the
27 array is not zero, any remaining digits are grouped based on the last
28 element of the array. If the last element is zero, the remaining digits
29 are not grouped.

30
31 The culture-invariant value for this property is an array with a single
32 element containing the value 3.

33 Exceptions

34
35

Exception	Condition
System.ArgumentNullException	The array specified for a set operation is a null reference.
System.ArgumentOutOfRangeException	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.

1

2

Example

3

4

The following example demonstrates the effects of different

5

System.Globalization.NumberFormatInfo.PercentGroupSizes

6

property values.

7

8

[C#]

9

```
using System;
```

10

```
using System.Globalization;
```

11

```
class Test {
```

12

```
    public static void Main() {
```

13

```
        NumberFormatInfo nfi = new NumberFormatInfo();
```

14

```
        decimal data = 9999999994444333221.00m;
```

15

```
        nfi.PercentGroupSizes = new int[] {1,2,3,4,0};
```

16

```
        Console.WriteLine("{0}",data.ToString("P",nfi));
```

17

18

```
        data = 123456789123456.78m;
```

19

```
        nfi.PercentGroupSizes = new int[] {3};
```

20

```
        Console.WriteLine("{0}",data.ToString("P",nfi));
```

21

```
        nfi.PercentGroupSizes = new int[] {3,0};
```

22

```
        Console.WriteLine("{0}",data.ToString("P",nfi));
```

23

```
    }
```

24

25

```
}
```

26

27

28

The output is

29

```
99999999944,4433,322,10,0.00 %
```

30

31

32

```
12,345,678,912,345,678.00 %
```

33

34

35

1 12345678912345,678.00 %
2
3

1 NumberFormatInfo.PercentNegativePattern 2 n Property

```
3 [ILASM]  
4 .property int32 PercentNegativePattern { public hidebysig  
5 specialname instance int32 get_PercentNegativePattern()  
6 public hidebysig specialname instance void  
7 set_PercentNegativePattern(int32 value) }  
  
8 [C#]  
9 public int PercentNegativePattern { get; set; }
```

10 Summary

11 Gets or sets the format of negative percent values.

12 Property Value

13

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

16 Description

17 The following table describes the valid values for this property. "%" is
18 used as the value for
19 **System.Globalization.NumberFormatInfo.PercentSymbol**, "-" is
20 used as the value for
21 **System.Globalization.NumberFormatInfo.NegativeSign**, and 999
22 represents any numeric value.

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

23

24 The culture-invariant value for this property is 0.

25 Exceptions

26

27

Exception	Condition
System.ArgumentOutOfRangeException	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.

1
2
3

Example

4 The following example demonstrates the effects of different
5 **System.Globalization.NumberFormatInfo.PercentNegativePatte**
6 **rn** property values.

```
7 [C#]  
8  
9 using System;  
10 using System.Globalization;  
11 class Test {  
12     public static void Main() {  
13         NumberFormatInfo nfi = new NumberFormatInfo();  
14         decimal data = -.9900m;  
15         for (int i = 0; i<=2; i++) {  
16             nfi.PercentNegativePattern = i;  
17             Console.WriteLine("pattern # {0}:  
18 {1}", i, data.ToString("P", nfi));  
19         }  
20     }  
21 }  
22
```

23 The output is

```
24  
25 pattern # 0: -99.00 %  
26  
27  
28 pattern # 1: -99.00%  
29  
30  
31 pattern # 2: -%99.00  
32
```

33

1 NumberFormatInfo.PercentPositivePattern 2 n Property

```
3 [ILASM  
4 .property int32 PercentPositivePattern { public hidebysig  
5 specialname instance int32 get_PercentPositivePattern()  
6 public hidebysig specialname instance void  
7 set_PercentPositivePattern(int32 value) }  
  
8 [C#]  
9 public int PercentPositivePattern { get; set; }
```

10 Summary

11 Gets or sets the format of positive percent values.

12 Property Value

13

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

16 Description

17 The following table describes the valid values for this property. "%" is
18 used as the value for
19 **System.Globalization.NumberFormatInfo.PercentSymbol**, and
20 999 represents a numeric value.

Value	Pattern
0	999 %
1	999%
2	%999

21

22 The culture-invariant value for this property is 0.

23 Exceptions

24

25

Exception	Condition
System.ArgumentOutOfRangeException	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.

1
2
3

Example

4 The following example demonstrates the effects of different
5 **System.Globalization.NumberFormatInfo.PercentPositivePatter**
6 **n** property values.

```
7 [C#]  
8  
9 using System;  
10 using System.Globalization;  
11 class Test {  
12     public static void Main() {  
13         NumberFormatInfo nfi = new NumberFormatInfo();  
14         decimal data = .9900m;  
15         for (int i = 0; i<=2; i++) {  
16             nfi.PercentPositivePattern = i;  
17             Console.WriteLine("pattern # {0}:  
18 {1}", i, data.ToString("P", nfi));  
19         }  
20     }  
21 }  
22
```

23 The output is

```
24  
25 pattern # 0: 99.00 %  
26  
27  
28 pattern # 1: 99.00%  
29  
30  
31 pattern # 2: %99.00  
32
```

33

1 NumberFormatInfo.PercentSymbol 2 Property

```
3 [ILASM]  
4 .property string PercentSymbol { public hidebysig  
5 specialname instance string get_PercentSymbol() public  
6 hidebysig specialname instance void  
7 set_PercentSymbol(string value) }  
  
8 [C#]  
9 public string PercentSymbol { get; set; }
```

10 Summary

11 Gets or sets the symbol that represents percentage values.

12 Property Value

13

14 A **System.String** containing the percent symbol.

15 Description

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

17 Exceptions

18

19

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.

20

21

22

1 NumberFormatInfo.PerMilleSymbol

2 Property

```
3 [ILASM]
4 .property string PerMilleSymbol { public hidebysig
5 specialname instance string get_PerMilleSymbol() public
6 hidebysig specialname instance void
7 set_PerMilleSymbol(string value) }

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

10 Summary

11 Gets or sets the per mille symbol.

12 Property Value

13

14 A **System.String** containing the per mille symbol.

15 Description

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

17 Exceptions

18

19

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.

20

21

22

1 NumberFormatInfo.PositiveInfinitySymbol 2 Property

```
3 [ILASM]  
4 .property string PositiveInfinitySymbol { public hidebysig  
5 specialname instance string get_PositiveInfinitySymbol()  
6 public hidebysig specialname instance void  
7 set_PositiveInfinitySymbol(string value) }  
  
8 [C#]  
9 public string PositiveInfinitySymbol { get; set; }
```

10 Summary

11 Gets or sets the symbol that represents positive infinity.

12 Property Value

13

14 A **System.String** containing the symbol for positive infinity.

15 Description

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

17

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

20 Exceptions

21

22

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.

23

24

25

1 NumberFormatInfo.PositiveSign Property

```
2 [ILASM]
3 .property string PositiveSign { public hidebysig
4 specialname instance string get_PositiveSign() public
5 hidebysig specialname instance void set_PositiveSign(string
6 value) }
7
8 [C#]
9 public string PositiveSign { get; set; }
```

9 Summary

10 Gets or sets the symbol used to represent positive values.

11 Property Value

12

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

15 Description

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

17 Exceptions

18

19

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.

20

21