**[C# int转string 每三位加一个逗号](http://www.cnblogs.com/testsec/p/6096023.html)**



long aaaa = 14200666;

Console.WriteLine(aaaa.ToString("N0"));

Console.WriteLine(string.Format("{0:N0}", 14200666));

Console.WriteLine("架构师 www.itsvse.com");

Console.ReadLine();

1、格式化货币（跟系统的环境有关，中文系统默认格式化人民币，英文系统格式化美元）  
string.Format("{0:C}",0.2) 结果为：￥0.20 （英文操作系统结果：$0.20）  
默认格式化小数点后面保留两位小数，如果需要保留一位或者更多，可以指定位数  
string.Format("{0:C1}",23.15) 结果为：￥23.2 （截取会自动四舍五入）  
格式化多个Object实例  
string.Format("市场价：{0:C}，优惠价{1:C}",23.15,19.82)  
2、格式化十进制的数字（格式化成固定的位数，位数不能少于未格式化前，只支持整形）  
string.Format("{0:D3}",23) 结果为：023  
string.Format("{0:D2}",1223) 结果为：1223，（精度说明符指示结果字符串中所需的最少数字个数。）  
3、用分号隔开的数字，并指定小数点后的位数  
string.Format("{0:N}", 14200) 结果为：14,200.00 （默认为小数点后面两位）  
string.Format("{0:N3}", 14200.2458) 结果为：14,200.246 （自动四舍五入）  
4、格式化百分比  
string.Format("{0:P}", 0.24583) 结果为：24.58% （默认保留百分的两位小数）  
string.Format("{0:P1}", 0.24583) 结果为：24.6% （自动四舍五入）  
5、零占位符和数字占位符  
string.Format("{0:0000.00}", 12394.039) 结果为：12394.04  
string.Format("{0:0000.00}", 194.039) 结果为：0194.04  
string.Format("{0:###.##}", 12394.039) 结果为：12394.04  
string.Format("{0:####.#}", 194.039) 结果为：194  
下面的这段说明比较难理解，多测试一下实际的应用就可以明白了。  
零占位符：  
如果格式化的值在格式字符串中出现“0”的位置有一个数字，则此数字被复制到结果字符串中。小数点前最左边的“0”的位置和小数点后最右边的“0”的位置确定总在结果字符串中出现的数字范围。  
“00”说明符使得值被舍入到小数点前最近的数字，其中零位总被舍去。  
数字占位符：  
如果格式化的值在格式字符串中出现“#”的位置有一个数字，则此数字被复制到结果字符串中。否则，结果字符串中的此位置不存储任何值。   
请注意，如果“0”不是有效数字，此说明符永不显示“0”字符，即使“0”是字符串中唯一的数字。如果“0”是所显示的数字中的有效数字，则显示“0”字符。   
“##”格式字符串使得值被舍入到小数点前最近的数字，其中零总被舍去。  
PS：空格占位符  
string.Format("{0,-50}", theObj);//格式化成50个字符，原字符左对齐，不足则补空格  
string.Format("{0,50}", theObj);//格式化成50个字符，原字符右对齐，不足则补空格  
6、日期格式化  
string.Format("{0:d}",System.DateTime.Now) 结果为：2009-3-20 （月份位置不是03）  
string.Format("{0:D}",System.DateTime.Now) 结果为：2009年3月20日  
string.Format("{0:f}",System.DateTime.Now) 结果为：2009年3月20日 15:37  
string.Format("{0:F}",System.DateTime.Now) 结果为：2009年3月20日 15:37:52  
string.Format("{0:g}",System.DateTime.Now) 结果为：2009-3-20 15:38  
string.Format("{0:G}",System.DateTime.Now) 结果为：2009-3-20 15:39:27  
string.Format("{0:m}",System.DateTime.Now) 结果为：3月20日  
string.Format("{0:t}",System.DateTime.Now) 结果为：15:41  
string.Format("{0:T}",System.DateTime.Now) 结果为：15:41:50  
  
更详细的说明请下面微软对此的说明或者上msdn上查询。  
微软MSDN对string.format的方法说明：  
名称 说明   
String.Format (String, Object) 将指定的 String 中的格式项替换为指定的 Object 实例的值的文本等效项。   
String.Format (String, Object[]) 将指定 String 中的格式项替换为指定数组中相应 Object 实例的值的文本等效项。   
String.Format (IFormatProvider, String, Object[]) 将指定 String 中的格式项替换为指定数组中相应 Object 实例的值的文本等效项。指定的参数提供区域性特定的格式设置信息。   
String.Format (String, Object, Object) 将指定的 String 中的格式项替换为两个指定的 Object 实例的值的文本等效项。   
String.Format (String, Object, Object, Object) 将指定的 String 中的格式项替换为三个指定的 Object 实例的值的文本等效项。  
标准数字格式字符串  
格式说明符 名称 说明   
C 或 c  
货币  
数字转换为表示货币金额的字符串。转换由当前 NumberFormatInfo 对象的货币格式信息控制。  
精度说明符指示所需的小数位数。如果省略精度说明符，则使用当前 NumberFormatInfo 对象给定的默认货币精度。  
  
D 或 d  
十进制数  
只有整型才支持此格式。数字转换为十进制数字 (0-9) 的字符串，如果数字为负，则前面加负号。  
精度说明符指示结果字符串中所需的最少数字个数。如果需要的话，则用零填充该数字的左侧，以产生精度说明符给定的数字个数。  
  
E 或 e  
科学记数法（指数）  
数字转换为“-d.ddd…E+ddd”或“-d.ddd…e+ddd”形式的字符串，其中每个“d”表示一个数字 (0-9)。如果该数字为负，则该字符串以减号开头。小数点前总有一个数字。  
精度说明符指示小数点后所需的位数。如果省略精度说明符，则使用默认值，即小数点后六位数字。  
格式说明符的大小写指示在指数前加前缀“E”还是“e”。指数总是由正号或负号以及最少三位数字组成。如果需要，用零填充指数以满足最少三位数字的要求。  
  
F 或 f  
定点  
数字转换为“-ddd.ddd…”形式的字符串，其中每个“d”表示一个数字 (0-9)。如果该数字为负，则该字符串以减号开头。  
精度说明符指示所需的小数位数。如果忽略精度说明符，则使用当前 NumberFormatInfo 对象给定的默认数值精度。  
  
G 或 g  
常规  
根据数字类型以及是否存在精度说明符，数字会转换为定点或科学记数法的最紧凑形式。如果精度说明符被省略或为零，则数字的类型决定默认精度，如下表所示。  
Byte 或 SByte：3  
Int16 或 UInt16：5  
Int32 或 UInt32：10  
Int64 或 UInt64：19  
Single：7  
Double：15  
Decimal：29  
如果用科学记数法表示数字时指数大于 -5 而且小于精度说明符，则使用定点表示法；否则使用科学记数法。如果要求有小数点，并且忽略尾部零，则结果包含小数点。如果精度说明符存在，并且结果的有效数字位数超过指定精度，则通过舍入删除多余的尾部数字。  
上述规则有一个例外：如果数字是 Decimal 而且省略精度说明符时。在这种情况下总使用定点表示法并保留尾部零。  
使用科学记数法时，如果格式说明符是“G”，结果的指数带前缀“E”；如果格式说明符是“g”，结果的指数带前缀“e”。   
  
N 或 n  
数字  
数字转换为“-d,ddd,ddd.ddd…”形式的字符串，其中“-”表示负数符号（如果需要），“d”表示数字 (0-9)，“,”表示数字组之间的千位分隔符，“.”表示小数点符号。实际的负数模式、数字组大小、千位分隔符以及十进制分隔符由当前 NumberFormatInfo 对象指定。  
精度说明符指示所需的小数位数。如果忽略精度说明符，则使用当前 NumberFormatInfo 对象给定的默认数值精度。  
  
P 或 p  
百分比  
数字转换为由 NumberFormatInfo.PercentNegativePattern 或 NumberFormatInfo.PercentPositivePattern 属性定义的、表示百分比的字符串，前者用于数字为负的情况，后者用于数字为正的情况。已转换的数字乘以 100 以表示为百分比。  
精度说明符指示所需的小数位数。如果忽略精度说明符，则使用当前 NumberFormatInfo 对象给定的默认数值精度。  
  
R 或 r  
往返过程  
只有 Single 和 Double 类型支持此格式。往返过程说明符保证转换为字符串的数值再次被分析为相同的数值。使用此说明符格式化数值时，首先使用常规格式对其进行测试：Double 使用 15 位精度，Single 使用 7 位精度。如果此值被成功地分析回相同的数值，则使用常规格式说明符对其进行格式化。但是，如果此值未被成功地分析为相同数值，则它这样格式 化：Double 使用 17 位精度，Single 使用 9 位精度。  
虽然此处可以存在精度说明符，但它将被忽略。使用此说明符时，往返过程优先于精度。   
  
X 或 x  
十六进制数  
只有整型才支持此格式。数字转换为十六进制数字的字符串。格式说明符的大小写指示对大于 9 的十六进制数字使用大写字符还是小写字符。例如，使用“X”产生“ABCDEF”，使用“x”产生“abcdef”。  
精度说明符指示结果字符串中所需的最少数字个数。如果需要的话，则用零填充该数字的左侧，以产生精度说明符给定的数字个数。   
  
任何其他单个字符  
（未知说明符）  
（未知说明符将引发运行库格式异常。）  
  
自定义数字格式字符串  
格式说明符 名称 说明   
0  
零占位符  
如果格式化的值在格式字符串中出现“0”的位置有一个数字，则此数字被复制到结果字符串中。小数点前最左边的“0”的位置和小数点后最右边的“0”的位置确定总在结果字符串中出现的数字范围。  
“00”说明符使得值被舍入到小数点前最近的数字，其中零位总被舍去。例如，用“00”格式化 34.5 将得到值 35。  
  
#  
数字占位符  
如果格式化的值在格式字符串中出现“#”的位置有一个数字，则此数字被复制到结果字符串中。否则，结果字符串中的此位置不存储任何值。  
请注意，如果“0”不是有效数字，此说明符永不显示“0”字符，即使“0”是字符串中唯一的数字。如果“0”是所显示的数字中的有效数字，则显示“0”字符。  
“##”格式字符串使得值被舍入到小数点前最近的数字，其中零总被舍去。例如，用“##”格式化 34.5 将得到值 35。  
  
.  
小数点  
格式字符串中的第一个“.”字符确定格式化的值中的小数点分隔符的位置；任何其他“.”字符被忽略。  
用作小数点分隔符的实际字符由控制格式化的 NumberFormatInfo 的 NumberDecimalSeparator 属性确定。  
  
,  
千位分隔符和数字比例换算  
“,”字符可作为千位分隔符说明符和数字比例换算说明符。  
千位分隔符说明符：如果在两个数字占位符（0 或 #）之间指定一个或多个“,”字符用于设置数字整数位的格式，则在输出的整数部分中每个数字组之间插入一个组分隔符字符。  
当前 NumberFormatInfo 对象的 NumberGroupSeparator 和 NumberGroupSizes 属性将确定用作数字组分隔符的字符以及每个数字组的大小。例如，如果使用字符串“#,#”和固定区域性对数字 1000 进行格式化，则输出为“1,000”。  
数字比例换算说明符：如果在紧邻显式或隐式小数点的左侧指定一个或多个“,”字符，则每出现一个数字比例换算说明符便将要格式化的数字除以 1000。例如，如果使用字符串“0,,”对数字 1000000000 进行格式化，则输出为“100”。  
可以在同一格式字符串中使用千位分隔符说明符和数字比例换算说明符。例如，如果使用字符串“#,0,,”和固定区域性对数字 10000000000 进行格式化，则输出为“1,000”。   
  
%  
百分比占位符  
在格式字符串中出现“%”字符将导致数字在格式化之前乘以 100。适当的符号插入到数字本身在格式字符串中出现“%”的位置。使用的百分比字符由当前的 NumberFormatInfo 类确定。  
  
E0  
E+0  
E-0  
e0  
e+0  
e-0  
科学记数法  
如果“E”、“E+”、“E-”、“e”、“e+”或“e-”中的任何一个字符串出现在格式字符串中，而且后面紧跟至少一个“0”字符，则数字用科学记数 法来格式化，在数字和指数之间插入“E”或“e”。跟在科学记数法指示符后面的“0”字符数确定指数输出的最小位数。“E+”和“e+”格式指示符号字符 （正号或负号）应总是置于指数前面。“E”、“E-”、“e”或“e-”格式指示符号字符仅置于负指数前面。  
  
\  
转义符  
在 C# 和 C++ 中，反斜杠字符使格式字符串中的下一个字符被解释为转义序列。它与传统的格式化序列一起使用，如“\n”（换行）。  
在某些语言中，转义符本身用作文本时必须跟在转义符之后。否则，编译器将该字符理解为转义符。使用字符串“\\”显示“\”。  
请注意，Visual Basic 中不支持此转义符，但是 ControlChars 提供相同的功能。  
  
’ABC’  
"ABC"  
字符串  
引在单引号或双引号中的字符被复制到结果字符串中，而且不影响格式化。  
  
;  
部分分隔符  
“;”字符用于分隔格式字符串中的正数、负数和零各部分。  
  
其他  
所有其他字符  
所有其他字符被复制到结果字符串中，而且不影响格式化。  
  
标准 DateTime 格式字符串  
格式说明符 名称 说明   
d  
短日期模式  
表示由当前 ShortDatePattern 属性定义的自定义 DateTime 格式字符串。  
例如，用于固定区域性的自定义格式字符串为“MM/dd/yyyy”。  
  
D  
长日期模式  
表示由当前 LongDatePattern 属性定义的自定义 DateTime 格式字符串。  
例如，用于固定区域性的自定义格式字符串为“dddd, dd MMMM yyyy”。  
  
f  
完整日期/时间模式（短时间）  
表示长日期 (D) 和短时间 (t) 模式的组合，由空格分隔。  
  
F  
完整日期/时间模式（长时间）  
表示由当前 FullDateTimePattern 属性定义的自定义 DateTime 格式字符串。  
例如，用于固定区域性的自定义格式字符串为“dddd, dd MMMM yyyy HH:mm:ss”。  
  
g  
常规日期/时间模式（短时间）  
表示短日期 (d) 和短时间 (t) 模式的组合，由空格分隔。  
  
G  
常规日期/时间模式（长时间）  
表示短日期 (d) 和长时间 (T) 模式的组合，由空格分隔。  
  
M 或 m  
月日模式  
表示由当前 MonthDayPattern 属性定义的自定义 DateTime 格式字符串。  
例如，用于固定区域性的自定义格式字符串为“MMMM dd”。  
  
o  
往返日期/时间模式  
表示使用保留时区信息的模式的自定义 DateTime 格式字符串。该模式专用于往返 DateTime 格式（包括文本形式的 Kind 属性）。随后将 Parse 或 ParseExact 与正确的 Kind 属性值一起使用可以对格式化的字符串进行反向分析。  
自定义格式字符串为“yyyy'-'MM'-'dd'T'HH':'mm':'ss.fffffffK”。  
用于此说明符的模式是定义的标准。因此，无论所使用的区域性或所提供的格式提供程序是什么，它总是相同的。  
  
R 或 r  
RFC1123 模式  
表示由当前 RFC1123Pattern 属性定义的自定义 DateTime 格式字符串。该模式是定义的标准，并且属性是只读的。因此，无论所使用的区域性或所提供的格式提供程序是什么，它总是相同的。  
定义格式字符串为“ddd, dd MMM yyyy HH':'mm':'ss 'GMT'”。  
格式化不会修改正在格式化的 DateTime 对象的值。因此，应用程序在使用此格式说明符之前必须将该值转换为协调世界时 (UTC)。  
  
s  
可排序的日期/时间模式；符合 ISO 8601  
表示由当前 SortableDateTimePattern 属性定义的自定义 DateTime 格式字符串。此模式是定义的标准，并且属性是只读的。因此，无论所使用的区域性或所提供的格式提供程序是什么，它总是相同的。  
自定义格式字符串为“yyyy'-'MM'-'dd'T'HH':'mm':'ss”。  
  
t  
短时间模式  
表示由当前 ShortTimePattern 属性定义的自定义 DateTime 格式字符串。  
例如，用于固定区域性的自定义格式字符串为“HH:mm”。  
  
T  
长时间模式  
表示由当前 LongTimePattern 属性定义的自定义 DateTime 格式字符串。  
例如，用于固定区域性的自定义格式字符串为“HH:mm:ss”。  
  
u  
通用的可排序日期/时间模式  
表示由当前 UniversalSortableDateTimePattern 属性定义的自定义 DateTime 格式字符串。此模式是定义的标准，并且属性是只读的。因此，无论所使用的区域性或所提供的格式提供程序是什么，它总是相同的。  
自定义格式字符串为“yyyy'-'MM'-'dd HH':'mm':'ss'Z'”。  
格式化日期和时间时不进行时区转换。因此，应用程序在使用此格式说明符之前必须将本地日期和时间转换为协调世界时 (UTC)。  
  
U  
通用的可排序日期/时间模式  
表示由当前 FullDateTimePattern 属性定义的自定义 DateTime 格式字符串。  
此模式与完整日期/长时间 (F) 模式相同。但是，格式化将作用于等效于正在格式化的 DateTime 对象的协调世界时 (UTC)。  
  
Y 或 y  
年月模式  
表示由当前 YearMonthPattern 属性定义的自定义 DateTime 格式字符串。  
例如，用于固定区域性的自定义格式字符串为“yyyy MMMM”。  
  
任何其他单个字符  
（未知说明符）  
未知说明符将引发运行时格式异常。  
  
自定义 DateTime 格式字符串  
格式说明符 说明   
d  
将月中日期表示为从 1 至 31 的数字。一位数字的日期设置为不带前导零的格式。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
dd  
将月中日期表示为从 01 至 31 的数字。一位数字的日期设置为带前导零的格式。  
  
ddd  
将一周中某天的缩写名称表示为当前 System.Globalization.DateTimeFormatInfo.AbbreviatedDayNames 属性中定义的名称。  
  
dddd（另加任意数量的“d”说明符）   
将一周中某天的全名表示为当前 System.Globalization.DateTimeFormatInfo.DayNames 属性中定义的名称。  
  
f  
表示秒部分的最高有效位。  
请注意，如果“f”格式说明符单独使用，没有其他格式说明符，则该说明符被看作是“f”标准 DateTime 格式说明符（完整日期/时间模式）。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
将此格式说明符与 ParseExact 或 TryParseExact 方法一起使用时，所用“f”格式说明符的数目指示要分析的秒部分的最高有效位位数。   
  
ff  
表示秒部分的两个最高有效位。  
  
fff  
表示秒部分的三个最高有效位。  
  
ffff  
表示秒部分的四个最高有效位。  
  
fffff  
表示秒部分的五个最高有效位。  
  
ffffff  
表示秒部分的六个最高有效位。  
  
fffffff  
表示秒部分的七个最高有效位。  
  
F  
表示秒部分的最高有效位。如果该位为零，则不显示任何信息。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
将此格式说明符与 ParseExact 或 TryParseExact 方法一起使用时，所用“F”格式说明符的数目指示要分析的秒部分的最高有效位最大位数。  
  
FF  
表示秒部分的两个最高有效位。但不显示尾随零（或两个零位）。  
  
FFF  
表示秒部分的三个最高有效位。但不显示尾随零（或三个零位）。  
  
FFFF  
表示秒部分的四个最高有效位。但不显示尾随零（或四个零位）。  
  
FFFFF  
表示秒部分的五个最高有效位。但不显示尾随零（或五个零位）。  
  
FFFFFF  
表示秒部分的六个最高有效位。但不显示尾随零（或六个零位）。  
  
FFFFFFF  
表示秒部分的七个最高有效位。但不显示尾随零（或七个零位）。  
  
g 或 gg（另加任意数量的“g”说明符）   
表示时期或纪元（例如 A.D.）。如果要设置格式的日期不具有关联的时期或纪元字符串，则忽略该说明符。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
h  
将小时表示为从 1 至 12 的数字，即通过 12 小时制表示小时，自午夜或中午开始对整小时计数。因此，午夜后经过的某特定小时数与中午过后的相同小时数无法加以区分。小时数不进行舍入，一位数字的小时 数设置为不带前导零的格式。例如，给定时间为 5:43，则此格式说明符显示“5”。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
hh, hh（另加任意数量的“h”说明符）   
将小时表示为从 01 至 12 的数字，即通过 12 小时制表示小时，自午夜或中午开始对整小时计数。因此，午夜后经过的某特定小时数与中午过后的相同小时数无法加以区分。小时数不进行舍入，一位数字的小时 数设置为带前导零的格式。例如，给定时间为 5:43，则此格式说明符显示“05”。  
  
H  
将小时表示为从 0 至 23 的数字，即通过从零开始的 24 小时制表示小时，自午夜开始对小时计数。一位数字的小时数设置为不带前导零的格式。   
  
HH, HH（另加任意数量的“H”说明符）   
将小时表示为从 00 至 23 的数字，即通过从零开始的 24 小时制表示小时，自午夜开始对小时计数。一位数字的小时数设置为带前导零的格式。   
  
K  
表示 DateTime.Kind 属性的不同值，即“Local”、“Utc”或“Unspecified”。此说明符以文本形式循环设置 Kind 值并保留时区。如果 Kind 值为“Local”，则此说明符等效于“zzz”说明符，用于显示本地时间偏移量，例如“-07:00”。对于“Utc”类型值，该说明符显示字符“Z” 以表示 UTC 日期。对于“Unspecified”类型值，该说明符等效于“”（无任何内容）。  
  
m  
将分钟表示为从 0 至 59 的数字。分钟表示自前一小时后经过的整分钟数。一位数字的分钟数设置为不带前导零的格式。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
mm, mm（另加任意数量的“m”说明符）   
将分钟表示为从 00 至 59 的数字。分钟表示自前一小时后经过的整分钟数。一位数字的分钟数设置为带前导零的格式。  
  
M  
将月份表示为从 1 至 12 的数字。一位数字的月份设置为不带前导零的格式。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
MM  
将月份表示为从 01 至 12 的数字。一位数字的月份设置为带前导零的格式。  
  
MMM  
将月份的缩写名称表示为当前 System.Globalization.DateTimeFormatInfo.AbbreviatedMonthNames 属性中定义的名称。  
  
MMMM  
将月份的全名表示为当前 System.Globalization.DateTimeFormatInfo.MonthNames 属性中定义的名称。  
  
s  
将秒表示为从 0 至 59 的数字。秒表示自前一分钟后经过的整秒数。一位数字的秒数设置为不带前导零的格式。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
ss, ss（另加任意数量的“s”说明符）   
将秒表示为从 00 至 59 的数字。秒表示自前一分钟后经过的整秒数。一位数字的秒数设置为带前导零的格式。  
  
t  
表示当前 System.Globalization.DateTimeFormatInfo.AMDesignator 或 System.Globalization.DateTimeFormatInfo.PMDesignator 属性中定义的 A.M./P.M. 指示符的第一个字符。如果正在格式化的时间中的小时数小于 12，则使用 A.M. 指示符；否则使用 P.M. 指示符。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
tt, tt（另加任意数量的“t”说明符）   
将 A.M./P.M. 指示符表示为当前 System.Globalization.DateTimeFormatInfo.AMDesignator 或 System.Globalization.DateTimeFormatInfo.PMDesignator 属性中定义的内容。如果正在格式化的时间中的小时数小于 12，则使用 A.M. 指示符；否则使用 P.M. 指示符。  
  
y  
将年份表示为最多两位数字。如果年份多于两位数，则结果中仅显示两位低位数。如果年份少于两位数，则该数字设置为不带前导零的格式。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
yy  
将年份表示为两位数字。如果年份多于两位数，则结果中仅显示两位低位数。如果年份少于两位数，则用前导零填充该数字使之达到两位数。  
  
yyy  
将年份表示为三位数字。如果年份多于三位数，则结果中仅显示三位低位数。如果年份少于三位数，则用前导零填充该数字使之达到三位数。  
请注意，对于年份可以为五位数的泰国佛历，此格式说明符将显示全部五位数。   
  
yyyy  
将年份表示为四位数字。如果年份多于四位数，则结果中仅显示四位低位数。如果年份少于四位数，则用前导零填充该数字使之达到四位数。  
请注意，对于年份可以为五位数的泰国佛历，此格式说明符将呈现全部五位数。   
  
yyyyy（另加任意数量的“y”说明符）  
将年份表示为五位数字。如果年份多于五位数，则结果中仅显示五位低位数。如果年份少于五位数，则用前导零填充该数字使之达到五位数。  
如果存在额外的“y”说明符，则用所需个数的前导零填充该数字使之达到“y”说明符的数目。   
  
z  
表示系统时间距格林威治时间 (GMT) 以小时为单位测量的带符号时区偏移量。例如，位于太平洋标准时区中的计算机的偏移量为“-8”。  
偏移量始终显示为带有前导符号。加号 (+) 指示小时数早于 GMT，减号 (-) 指示小时数迟于 GMT。偏移量范围为 –12 至 +13。一位数字的偏移量设置为不带前导零的格式。偏移量受夏时制影响。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
zz  
表示系统时间距格林威治时间 (GMT) 以小时为单位测量的带符号时区偏移量。例如，位于太平洋标准时区中的计算机的偏移量为“-08”。  
偏移量始终显示为带有前导符号。加号 (+) 指示小时数早于 GMT，减号 (-) 指示小时数迟于 GMT。偏移量范围为 –12 至 +13。一位数字的偏移量设置为带前导零的格式。偏移量受夏时制影响。  
  
zzz, zzz（另加任意数量的“z”说明符）   
表示系统时间距格林威治时间 (GMT) 以小时和分钟为单位测量的带符号时区偏移量。例如，位于太平洋标准时区中的计算机的偏移量为“-08:00”。  
偏移量始终显示为带有前导符号。加号 (+) 指示小时数早于 GMT，减号 (-) 指示小时数迟于 GMT。偏移量范围为 –12 至 +13。一位数字的偏移量设置为带前导零的格式。偏移量受夏时制影响。  
  
:  
当前 System.Globalization.DateTimeFormatInfo.TimeSeparator 属性中定义的时间分隔符，用于区分小时、分钟和秒。  
  
/  
当前 System.Globalization.DateTimeFormatInfo.DateSeparator 属性中定义的日期分隔符，用于区分年、月和日。  
  
"  
带引号的字符串（引号）。显示两个引号 (") 之间的任意字符串的文本值。在每个引号前使用转义符 (\)。   
  
'  
带引号的字符串（撇号）。显示两个撇号 (') 字符之间的任意字符串的文本值。  
  
%c  
当自定义 DateTime 格式字符串只包含自定义格式说明符“c”时，表示与该自定义格式说明符关联的结果。也就是说，若要单独使用自定义格式说明符“d”、“f”、“F”、 “h”、“m”、“s”、“t”、“y”、“z”、“H”或“M”，请指定“%d”、“%f”、“%F”、“%h”、“%m”、“%s”、“%t”、 “%y”、“%z”、“%H”或“%M”。有关使用单个格式说明符的更多信息，请参见使用单个自定义格式说明符。  
  
\c  
转义符。当字符“c”前带有转义符 (\) 时，将该字符显示为文本。若要将反斜杠字符本身插入结果字符串，请使用两个转义符（“\\”）。   
  
任何其他字符   
所有其他字符被复制到结果字符串中，而且不影响格式化。  
  
枚举格式字符串  
格式字符串 结果   
G 或 g  
如有可能，将枚举项显示为字符串值，否则显示当前实例的整数值。如果枚举定义中设置了 Flags 属性，则串联每个有效项的字符串值并将各值用逗号分开。如果未设置 Flags 属性，则将无效值显示为数字项。  
  
F 或 f  
如有可能，将枚举项显示为字符串值。如果值可以完全显示为枚举项的总和（即使未提供 Flags 属性），则串联每个有效项的字符串值并将各值用逗号分开。如果值不能完全由枚举项确定，则将值格式化为整数值。  
  
D 或 d  
以尽可能短的表示形式将枚举项显示为整数值。  
  
X 或 x  
将枚举项显示为十六进制值。按需要将值表示为带有前导零，以确保值的长度最少有八位