Strings formatteren in C#

Tekenreeksen in een bepaald formaat weergeven is een wezenlijk onderdeel van het programmeren, o.a. datums en geldbedragen kunnen moeilijk zijn.

Strings formatteren

Strings kan je formatteren door de padding- en uitlijningsopties te specifiëren:

example	output
String.Format("--{0,10}--", "test");	-- test--
String.Format("--{0,-10}--", "test");	--test --

Getallen formatteren

In het laatste voorbeeld verbruikt de eerste placeholder 10 tekens. De standaard uitlijning is rechts, maar het minteken verandert die naar links. Getallen formatteren is cultuur afhankelijk. Een geldbedrag zal op mijn netbook weergegeven worden als €9,99, op een netbook voor Londen als £9.99, en op een netbook voor de US als $9.99..

specifier	type	format	output (double 1.2345)	output (int -12345)
c	currency	{0:c}	£1.23	-£12,345.00
d	decimal (whole number)	{0:d}	System.FormatException	-12345
e	exponent / scientific	{0:e}	1.234500e+000	-1.234500e+004
f	fixed point	{0:f}	1.23	-12345.00
g	general	{0:g}	1.2345	-12345
n	number	{0:n}	1.23	-12,345.00
r	round trippable	{0:r}	1.23	System.FormatException
x	hexadecimal	{0:x4}	System.FormatException	ffffcfc7

Zelf getallen formatteren

specifier	type	format	output (double 1234.56)
0	zero placeholder	{0:00.000}	1234.560

#	digit placeholder	{0:#.##}	1234.56
.	decimal point placeholder	{0:0.0}	1234.6
,	thousand separator	{0:0,0}	1,235
%	percentage	{0:0%}	12,34%
‰	per mille (je zal dit teken moeten knippen en plakken om het te gebruiken :) Multiplies a number by 1000 and inserts a localized per mille symbol in the result string.	{0:0‰}	12,3456%

Er bestaat ook nog een groep separator. Dat is handig om het formaat aan te passen aan de doorgegeven parameter:

```
String.Format("{0:£#,##0.00;(£#,##0.00);Nothing}", value);
```

Dat geeft "£1,240.00" als de parameter 1243.56 is. Als de parameter een negatief getal is zal het tussen de haakjes staan, en als de parameter 0 is zal de tekst "Nothing" geretourneerd worden.

Voorbeelden

Let erop dat de manier waarop getallen worden geschreven cultuurafhankelijk zijn. Een punt betekent in de VS iets anders dan bij ons!

Helemaal bovenaan geef je aan dat je de namespace `System.Globalization` wilt gebruiken:

```
using System.Globalization;
```

In de klasse `WerkenMetGegevens` voeg je de volgende methode toe:

```
class WerkenMetGegevens
{
    public string FormatNumericSample()
    {
        double value;

        value = 123;
        Console.WriteLine(value.ToString("00000"));
        Console.WriteLine(String.Format("{0:00000}", value));
        // Displays 00123

        value = 1.2;
        Console.WriteLine(value.ToString("0.00", CultureInfo.InvariantCulture));
        Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                    "{0:0.00}", value));
        // Displays 1.20

        Console.WriteLine(value.ToString("00.00", CultureInfo.InvariantCulture));
        Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                    "{0:00.00}", value));
        // Displays 01.20

        CultureInfo daDK = CultureInfo.CreateSpecificCulture("da-DK");
        Console.WriteLine(value.ToString("00.00", daDK));
        Console.WriteLine(String.Format(daDK, "{0:00.00}", value));
```

```
// Displays 01,20

value = .56;
Console.WriteLine(value.ToString("0.0", CultureInfo.InvariantCulture));
Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                            "{0:0.0}", value));
// Displays 0.6

value = 1234567890;
Console.WriteLine(value.ToString("0,0", CultureInfo.InvariantCulture));
Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                            "{0:0,0}", value));
// Displays 1,234,567,890

CultureInfo elGR = CultureInfo.CreateSpecificCulture("el-GR");
Console.WriteLine(value.ToString("0,0", elGR));
Console.WriteLine(String.Format(elGR, "{0:0,0}", value));
// Displays 1.234.567.890

value = 1234567890.123456;
Console.WriteLine(value.ToString("0,0.0", CultureInfo.InvariantCulture));
Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                            "{0:0,0.0}", value));
// Displays 1,234,567,890.1

value = 1234.567890;
Console.WriteLine(value.ToString("0,0.00", CultureInfo.InvariantCulture));
Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                            "{0:0,0.00}", value));
// Displays 1,234.57

value = 1.2;
Console.WriteLine(value.ToString("#.##", CultureInfo.InvariantCulture));
Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                "{0:#.##}", value));
// Displays 1.2

value = 123;
Console.WriteLine(value.ToString("#####"));
Console.WriteLine(String.Format("{0:#####}", value));
// Displays 123

value = 123456;
Console.WriteLine(value.ToString("[##-##-##]"));
Console.WriteLine(String.Format("{0:[##-##-##]}", value));
// Displays [12-34-56]

value = 1234567890;
Console.WriteLine(value.ToString("#"));
Console.WriteLine(String.Format("{0:#}", value));
// Displays 1234567890

Console.WriteLine(value.ToString("(###) ###-####"));
Console.WriteLine(String.Format("{0:(###) ###-####}", value));
// Displays (123) 456-7890

value = 1.2;
Console.WriteLine(value.ToString("0.00", CultureInfo.InvariantCulture));
Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                "{0:0.00}", value));
// Displays 1.20

Console.WriteLine(value.ToString("00.00", CultureInfo.InvariantCulture));
Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                "{0:00.00}", value));
// Displays 01.20

Console.WriteLine(value.ToString("00.00",
                    CultureInfo.CreateSpecificCulture("da-DK")));
Console.WriteLine(String.Format(CultureInfo.CreateSpecificCulture("da-DK"),
                    "{0:00.00}", value));
// Displays 01,20
```

```
        value = .086;
        Console.WriteLine(value.ToString("#0.##%", CultureInfo.InvariantCulture));
        Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                        "{0:#0.##%}", value));
        // Displays 8.6%

        value = 86000;
        Console.WriteLine(value.ToString("0.###E+0", CultureInfo.InvariantCulture));
        Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                            "{0:0.###E+0}", value));
        // Displays 8.6E+4
        value = 1234567890;
        Console.WriteLine(value.ToString("#,,", CultureInfo.InvariantCulture));
        Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                        "{0:#,,}", value));
        // Displays 1235

        Console.WriteLine(value.ToString("#,,,", CultureInfo.InvariantCulture));
        Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                        "{0:#,,,}", value));
        // Displays 1

        Console.WriteLine(value.ToString("#,##0,,", CultureInfo.InvariantCulture));
        Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                        "{0:#,##0,,}", value));
        // Displays 1,235
        value = .086;
        Console.WriteLine(value.ToString("#0.##%", CultureInfo.InvariantCulture));
        Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                        "{0:#0.##%}", value));
        // Displays 8.6%
        value = .00354;
        string perMilleFmt = "#0.## " + '\u2030';
        Console.WriteLine(value.ToString(perMilleFmt, CultureInfo.InvariantCulture));
        Console.WriteLine(String.Format(CultureInfo.InvariantCulture,
                                        "{0:" + perMilleFmt + "}", value));
        // Displays 3.54‰
        return value.ToString();
      }
}
```

Oefening

1. Plaats de code hierboven in een static methode met de naam `FormatNumericSample` in de klasse met de naam `WerkenMetGegevens` in de namespace met de naam `LerenWerkenMetCSharp`.
2. Test deze methode in de `Main` methode in `Program.cs`.

Datums formatteren

Datumformaten zijn afhankelijk van de ingestelde CultureInfo. De voorbeelden tonen de UK cultuur. Ik weet wel dat dit saai is. Maar probeer er toch iets van te onthouden. Die kennis komt zeker van pas als je een eindwerk moet maken. Werken met datums is geen sinecure!

specifier	type	output (June 8, 1970 12:30:59)
d	Short Date	08/06/1970
D	Long Date	08 June 1970
t	Short Time	12:30

T	Long Time	12:30:59
f	Full date and time	08 June 1970 12:30
F	Full date and time (long)	08 June 1970 12:30:59
g	Default date and time	08/06/1970 12:30
G	Default date and time (long)	08/06/1970 12:30:59
M	Day / Month	8 June
r	RFC1123 date string	Mon, 08 Jun 1970 12:30:59 GMT
s	Sortable date/time	1970-06-08T12:30:59
u	Universal time, local timezone	1970-06-08 12:30:59Z
Y	Month / Year	June 1970

Specifiers waarmee je datums à la carte kan formatteren

specifier	**type**	**output (June 8, 1970 12:30:59)**
d	dag van de maand van 1 tot 31	8
dd	dag van de maand van 01 tot 31	08
ddd	Short Day Name	Mon
dddd	Full Day Name	Monday
f	tiende van een seconde in datum- en tijdswaarde	
ff	honderdste van een seconde in datum- en tijdswaarde	
fff	duizendste van een seconde in datum- en tijdswaarde dat gaat zo verder tot fffffff, een tienmiljoenste!	
F	als niet nul, een tiende van een seconde in datum- en tijdswaarde dat gaat weerom tot FFFFFFF	
g gg	de tijdsrekening	A.D.
h	2 cijfers tijd van 1 tot 12	12
hh	2 cijfers tijd (12 uren) van 01 tot 12	12
H	2 cijfers tijd van 0 tot 23	
HH	2 digit hour (24 hour) van 00 tot 23	12

K	Tijdszone	
m	2 digit minute van 0 tot 59	30
mm	2 digit minute van 00 tot 59	30
M	maand van 1 tot 12	
MM	maand van 01 tot 12	06
MMM	Short Month name	Jun
MMMM	Month name	June
s	seconde van 1 tot 59	
ss	seconde van 01 tot 59	59
t	het eerste gedeelte van AM/PM	
tt	AM/PM	PM
y	jaar van 0 tot 99	
yy	jaar van 00 tot 99	70
yyyy	4 digit year	1970
:	seperator, e.g. {0:hh:mm:ss}	12:30:59
/	seperator, e.g. {0:dd/MM/yyyy}	08/06/1970

Voorbeelden

```
public static void FormatDate()
{

    DateTime date1 = new DateTime(2008, 8, 29, 19, 27, 15);
    CultureInfo ci = CultureInfo.InvariantCulture;
    Console.WriteLine("Datum in cultuur van computer {0}.\n", date1.ToString("d, M",
                CultureInfo.InvariantCulture));
    // Displays 29, 8
    Console.WriteLine(date1.ToString("d MMMM",
                CultureInfo.CreateSpecificCulture("en-US")));
    // Displays 29 August
    Console.WriteLine(date1.ToString("d MMMM",
                CultureInfo.CreateSpecificCulture("es-MX")));
    // Displays 29 agosto
    Console.WriteLine(date1.ToString("dd, MM",
                CultureInfo.InvariantCulture));
    // 02, 01
    Console.WriteLine(date1.ToString("ddd d MMM",
                CultureInfo.CreateSpecificCulture("en-US")));
    // Displays Fri 29 Aug
    Console.WriteLine(date1.ToString("ddd d MMM",
                CultureInfo.CreateSpecificCulture("fr-FR")));
    // Displays ven. 29 août

    Console.WriteLine(date1.ToString("dddd dd MMMM",
                CultureInfo.CreateSpecificCulture("en-US")));
    // Displays Friday 29 August
    Console.WriteLine(date1.ToString("dddd dd MMMM",
                CultureInfo.CreateSpecificCulture("it-IT")));
    // Displays venerdì 29 agosto
```

```
    Console.WriteLine(date1.ToString("hh:mm:ss.f", ci));
    // Displays 07:27:15.0
    Console.WriteLine(date1.ToString("hh:mm:ss.F", ci));
    // Displays 07:27:15
    Console.WriteLine(date1.ToString("hh:mm:ss.ff", ci));
    // Displays 07:27:15.01
    Console.WriteLine(date1.ToString("hh:mm:ss.FF", ci));
    // Displays 07:27:15.01
    Console.WriteLine(date1.ToString("hh:mm:ss.fff", ci));
    // Displays 07:27:15.018
    Console.WriteLine(date1.ToString("hh:mm:ss.FFF", ci));
    // Displays 07:27:15.018
    date1 = new DateTime(70, 08, 04);
    Console.WriteLine(date1.ToString("MM/dd/yyyy g",
                      CultureInfo.InvariantCulture));
    // Displays 08/04/0070 A.D.
    Console.WriteLine(date1.ToString("MM/dd/yyyy g",
                      CultureInfo.CreateSpecificCulture("fr-FR")));
    // Displays 08/04/0070 ap. J.-C.

    date1 = new DateTime(2008, 1, 1, 18, 9, 1);
    Console.WriteLine(date1.ToString("h:m:s.F t",
                      CultureInfo.InvariantCulture));
    // Displays 6:9:1 P
    Console.WriteLine(date1.ToString("h:m:s.F t",
                      CultureInfo.CreateSpecificCulture("el-GR")));
    // Displays 6:9:1 μ
    date1 = new DateTime(2008, 1, 1, 18, 9, 1, 500);
    Console.WriteLine(date1.ToString("h:m:s.F t",
                      CultureInfo.InvariantCulture));
    // Displays 6:9:1.5 P
    Console.WriteLine("In het Grieks {0}\n", date1.ToString("h:m:s.F t",
                      CultureInfo.CreateSpecificCulture("el-GR")));
    // Displays 6:9:1.5 μ
    date1 = new DateTime(1, 12, 1);
    DateTime date2 = new DateTime(2010, 1, 1);
    Console.WriteLine(date1.ToString("%y"));
    // Displays 1
    Console.WriteLine(date1.ToString("yy"));
    // Displays 01
    Console.WriteLine(date1.ToString("yyy"));
    // Displays 001
    Console.WriteLine(date1.ToString("yyyy"));
    // Displays 0001
    Console.WriteLine(date1.ToString("yyyyy"));
    // Displays 00001
    Console.WriteLine(date2.ToString("%y"));
    // Displays 10
    Console.WriteLine(date2.ToString("yy"));
    // Displays 10
    Console.WriteLine(date2.ToString("yyy"));
    // Displays 2010
    Console.WriteLine(date2.ToString("yyyy"));
    // Displays 2010
    Console.WriteLine(date2.ToString("yyyyy"));
    // Displays 02010
}
```

Oefening

1. Plaats de code hierboven in een static methode met de naam `FormatDate` in de klasse met de naam `WerkenMetGegevens` in de namespace met de naam `LerenWerkenMetCSharp`.
2. Test deze methode in de `Main` methode in `Program.cs`.

Bronnen

- Standard Numeric Format Strings
- Custom Numeric Format Strings
- Standard Date and Time Format Strings
- Custom Date and Time Format Strings
- Standard TimeSpan Format Strings
- Custom TimeSpan Format Strings
- Enumeration Format Strings
- Composite Formatting
- Performing Formatting Operations

Culture information

String.**Format** beschikt over een methode die een CultureInfo argument aanvaardt als een an **IFormatProvider**. Dat is belangrijk als je internationale software maakt. De manier van datums, boekhouding enz. kan lokaal heel verschillende zijn. Je moet in dat geval altijd de CultureInfo meegeven en niet voortgaan op standaard **String**.**Format**. Als je geen specifieke cultuur nodig hebt kan je de **System**.**Globalization**.**CultureInfo**.**InvariantCulture** meegeven. Dat stelt de standaard formattering in op de Engelse cultuur. **ToString**() Je kan die formattering ook gebruiken in de **ToString**() methode: // Vietnamees geldsymbool

```
sAnswer += kostprijs.ToString("c", new CultureInfo("vi-VN")) + "\n";
```

// UK

```
sAnswer += kostprijs.ToString("c", new CultureInfo("en-GB")) + "\n";
```

Oefening

1. Plaats de code hierboven in een static methode met de naam `CultureInfo` in de klasse met de naam `WerkenMetGegevens` in de namespace met de naam `LerenWerkenMetCSharp`.
2. Test deze methode in de `Main` methode in `Program.cs`.