<http://www.daveoncsharp.com/2009/09/formatting-decimals-in-csharp/>

In this post I am going to show you a few different ways how you can format a decimal number (float, double, or decimal).

SETTING THE MAXIMUM ALLOWED DECIMAL PLACES

To format your numbers to a maximum of two decimal places use the format string {0:0.##} as shown in the below example:

string.Format("{0:0.##}", 256.583); // "256.58"

string.Format("{0:0.##}", 256.586); // "256.59"

string.Format("{0:0.##}", 256.58); // "256.58"

string.Format("{0:0.##}", 256.5); // "256.5"

string.Format("{0:0.##}", 256.0); // "256"

SETTING A FIXED AMOUNT OF DECIMAL PLACES

This is similar to the above example but instead of hashes (**‘#’**) in our format string we are going to use zeroes (**‘0’**) as shown below:

string.Format("{0:0.00}", 256.583); // "256.58"

string.Format("{0:0.00}", 256.586); // "256.59"

string.Format("{0:0.00}", 256.58); // "256.58"

string.Format("{0:0.00}", 256.5); // "256.50"

string.Format("{0:0.00}", 256.0); // "256.00"

THE THOUSAND SEPARATOR

To format your decimal number using the *thousand separator*, use the format string {0:0,0} as shown in the below example:

string.Format("{0:0,0.00}", 1234256.583); // "1,234,256.58"

string.Format("{0:0,0}", 1234256.583); // "1,234,257"

SETTING A FIXED AMOUNT OF DIGITS BEFORE THE DECIMAL POINT

To set a minimum amount of three digits before the decimal point use the format string{0:000.##}.

string.Format("{0:00.000}", 1.2345); // "01.235"

string.Format("{0:000.000}", 12.345); // "012.345"

string.Format("{0:0000.000}", 123.456); // "0123.456"

ALIGNMENT

To specify alignment to the Format method you must write your format string as shown below. Note we are using a comma (**‘,’**) to specify the number of characters used for alignment.

{0,[no. of chars]} and if you want to pad with zeroes {0,[no. of chars]:00.00}

string.Format("{0,7:##.00}", 2.356); // " 2.36"

string.Format("{0,-7:##.00}", 2.356); // "2.36 "

string.Format("{0,7:00.00}", 2.356); // " 02.36"

string.Format("{0,-7:00.00}", 2.356); // "02.36 "

POSITIVE NUMBERS, NEGATIVE NUMBERS, AND ZERO

You can include different formats for positive numbers, negative numbers, and zero by using the semicolon character (**‘;’**).

Format string:  
{0:[positive];[negative];[zero]}

string.Format("{0:000.000;(000.000);zero}", 23.43); // "023.430"

string.Format("{0:000.000;(000.000);zero}", -23.43); // "(023.430)"

string.Format("{0:000.000;(000.000);zero}", 0.0); // "zero"

SOME PRE-DEFINED FORMATS

string.Format("{0:C}", 1532.236); // "£1,532.24"

string.Format("{0:C}", -1532.236); // "-£1,532.24"

string.Format("{0:E}", 1532.236); // "1.532236E+003"

string.Format("{0:E}", -1532.236); // "-1.532236E+003"

string.Format("{0:F}", 1532.24); // "1532.24"

string.Format("{0:F}", -1532.24); // "-1532.24"

string.Format("{0:G}", 1532.236); // "1532.236"

string.Format("{0:G}", -1532.236); // "-1532.236"

string.Format("{0:N}", 1532.236); // "1,532.24"

string.Format("{0:N}", -1532.236); // "-1,532.24"

string.Format("{0:P}", 0.1532); // "15.32 %"

string.Format("{0:P}", -0.1532); // "-15.32 %"

string.Format("{0:R}", 1532.236); // "1532.236"

string.Format("{0:R}", -1532.236); // "-1532.236"