

DeltaMaster clicks!

02/2008

Greetings, fellow data analysts!

Pi, the mathematical constant representing the ratio of any circle's circumference to its diameter, has fascinated mathematicians for centuries. In their attempts to approximately identify its exact value, they have produced a wide range of algorithms and traced the number to over a trillion decimal places (not to forget the "3" before the decimal point). "It's absolutely useless – but an incredible feeling!" explains Professor Albrecht Beutelspacher from the Institute for Mathematics at the University of Giessen.

This type of sentimentalism, however, has no place in management reporting. Digits alone don't lend insight even in theory. Instead, they get in the way of understanding important relationships. In this edition of *clicks!*, we will explain how you can tame the numbers in your reports and format them accordingly.

Best regards,

Your Bissantz and Company team

Straight talk for stock data

We are certain that you have read a lot of about stock market "traumas" and share "crises" in the past few months. In Germany, some newspapers even use these provocative words next to a vertically upright arrow to symbolize a small(!) improvement of the DAX over the previous day. Sparklines, on the other hand, just tell it like it is.

<http://www.bissantz.com/sparklines/deltalife.asp>



Stock prices from 7 December 2007 through 6 February 2008 for DAX companies: 39 trading days, 1,170 values.

"Industry Reporting" power seminar

18 February 2008, Nuremberg

The battle against anorexic reporting and useless BI shelfware has begun!

www.bissantz.com/ir

OLAP seminar

18 - 21 February 2008, Nuremberg

The first of six seminars in 2008 will kick off this month. This three-day training covers data preparation (SQL), OLAP modeling and MDX programming.

www.bissantz.com/olap-seminar

"Service goes live"

Blaupunkt, Hildesheim

20 - 21 February 2008

We'll show how management control and service complement each other in this event sponsored by the German Customer Service Organization.

DeltaMaster@Work

28 February 2008, Nuremberg

Create reports that make a statement!

www.bissantz.com/dm@w

Archive

<http://www.bissantz.com/clicks/en>

Tip of the month *Formatting values based on views*

How easily others can read and understand your reports often depends on how you format your numbers. When we say “formatting” in this sense, we are not referring to the font type, color, style or other design aspects of the print layout. Instead, we mean how many digits and characters should we use and which ones should we choose in order to avoid pseudo-accuracy? Who profits from knowing the exact cent value of million-dollar figures... and in which decision-making situations is this precision important?

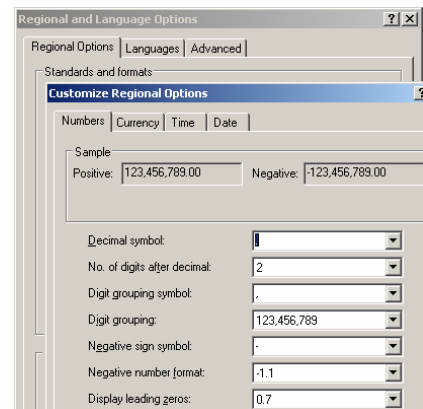
In this edition of *clicks!* we will explore the options *DeltaMaster* offers for formatting numbers as well as how you can influence them.

Windows language options and regions

In general, the properties that apply for *DeltaMaster* are based on those of the entire computer on which it runs. In your Windows control panel, you can customize these “regional and language options”, for example:



- If you use a comma (e.g. in German-speaking countries) in place of a decimal point
- How you display negative values
- Which currency symbols should be placed before or after the value
- In which order the day, month and year should appear for dates

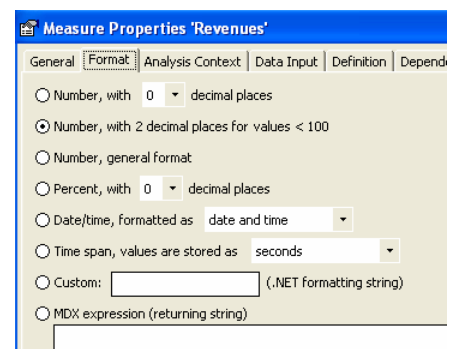


These and other properties also determine the appearance of *DeltaMaster*. This is a great advantage when you want to distribute reports in different countries, because the recipients can view the analyses in the form that is familiar to them – and not the report author.

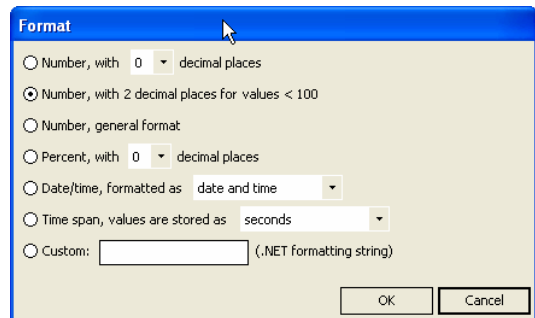
Although you can make individual customizations for measures and calculated elements, *DeltaMaster* will generally utilize the Windows properties.

Standard formats in *DeltaMaster*

You can determine how each measure should be displayed on the *Format* tab under *Measure properties*.



For calculated members such as a budget-actual variance, simply click on the *Format* link. Here you can access almost the exact same options as in the *Calculated measures editor*.



Normally, you do not need to bother with formatting issues. *DeltaMaster* automatically assigns measures (and members) a format as a number or a percentage. If the value is a number smaller than 100, *DeltaMaster* will use two decimal spaces and none if the value is higher. If the value is a percentage, *DeltaMaster* will automatically select this option when you add a quotient measure or define a relative variance in the *Calculated members editor*. These two possibilities have proven very useful in business applications.

Depending on the value, a number with a *General format* will be expressed either as an integer or in scientific notation (1.08E+07).

You can also customize the number of decimal spaces used. As mentioned previously, your personal Windows settings will determine how *DeltaMaster* displays positive or negative numbers or if a period or comma is used to mark decimals or groups of thousands.

When you work with dates, you can also see the direct relationship with the control panel. The *Date and time*, *Long/short date* and *Long/short time* options are identical to the properties there.

Time span is suited for any type of time measures including contract periods, processing times, length of telephone conversations and delivery deadlines. You can also express a large volume of seconds, for example, in days, hours and minutes.

Format strings

If you require even more customization than the examples above, you can also apply .NET format strings by simply entering them into the desired input field. When working with measures (not calculated members!), you can dynamically set the number format, for example, based on the current view. This is done using an MDX statement which returns a format string as the output as we will explain later in the document.

The format strings consist of one or (usually) more placeholders which can be replaced by numbers, digits and characters depending on the value to be displayed. If you have ever created a user-defined numerical format such as “#.##0” in Excel, you will understand the general concept.

The pool of characters and the possible combinations thereof that influence the output format is immense. Since a complete documentation would go beyond the usual scope of *clicks!*, we will simply concentrate on a few typical usage scenarios.

Symbol	Name	Explanation/Example
0	0 placeholder	Stands for a digit. Leading zeroes before the decimal point and trailing zeroes after the decimal point will be displayed so that all numbers will be generated with a specific length. For example, the number "3.14" with the format "000.000" would be displayed as "003.140". This format limits the number of digits to the right of the decimal point but not to the left. The digits to the left of the decimal point, however, will be "filled" if there are more digits available than variables are given.
#	Digit placeholder	Stands for a digit like in the example above. In this case, however, the leftmost zeros preceding the decimal and the rightmost zeros following the decimal will not be displayed. The value '0', therefore, would not be displayed at all. The number "3.14" in the format "###.###" would have the output "3.14".
.	Decimal point	The first point in the character string determines where the decimal point should be located. This is also necessary when a certain number of digits should be placed after the decimal point. How the decimal point is displayed (e.g. as a "normal" period in English speaking countries or as a comma in German-speaking countries) depends on the control panel settings. A period should always be used to represent a decimal point in a format string.
,	Thousand separator and number scaling	A comma can have one of two meanings. If it stands between two digits or 0 placeholders, the output will be a thousand separator (as specified in the control panel settings). The number "3140" with the format "#,#" has an output of "3,140". If one or more commas stand at the end of a format string or directly before a period (i.e. as a placeholder for the decimal point), then the value for each comma will be divided by 1000. As a result, a value of 3,500,000 with the format "#,," will be divided by 1000 twice and then simply be rounded to "4".
%	Percentage placeholder	A value with this format would be multiplied by 100 and displayed in the output with a percentage sign.
E0 E+0 E-0	Scientific notation	In scientific notations numbers will be displayed as a multiple to the 10 th power.
\	Escape character	The 'backslash' sign annuls the placeholder effect of the following character so only the character that follows the backslash will be shown, i.e. to display a percentage sign without previously multiplying the value by 100, you would write "%".
"abc" 'abc'	Literal string	Everything that is located within double or single quotation marks will be output as such. If you format the value 3.14 as "#.## 'mil'", the output will be "3.14 mil". Characters that have no meaning as a placeholder also will be transferred to the output as is.

For more information, please refer to the online documentation for. Microsoft NET Framework:

[http://msdn2.microsoft.com/en-us/library/fbxf59x\(VS.80\).aspx](http://msdn2.microsoft.com/en-us/library/fbxf59x(VS.80).aspx)

John Sheehan's "cheat sheet" also provides many useful tips:

<http://john-sheehan.com/blog/wp-content/uploads/msnet-formatting-strings.pdf>

Displaying values in thousands or millions

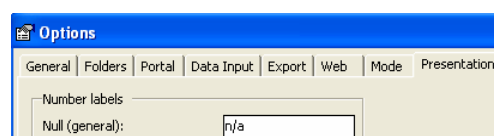
In order to summarize the small print for the common scenario that you want to display values in thousands or millions, simply use “#,#,” for thousands and “#,#,,” for millions. Both strings will suppress decimal spaces and group the numbers by thousands.

Varying formats for positive and negative numbers

You can also use conditional formatting to distinguish positive numbers from negative ones. The trick is to combine two or three valid format strings with a semicolon. If you enter two strings, the first one will be valid for positive values and the second for negative ones. If you enter three, the first one will be valid for positive values, the second for negative ones and the third for zero values.

Please do not confuse the number 0 with a missing value.

You can change how *DeltaMaster* presents these in the *Extras* menu under *Options, Presentation*.



If you want to display negative values in parentheses, enter “#,#;(-#,#)”.

Creating view-dependant formatting with MDX

As mentioned above, you can use MDX to calculate the format string so that it dynamically changes based on the current view.

For example: While a global enterprise thinks in terms of billions on a group level, its country-level operations think in millions, their sales regions also calculate in thousands and account managers even count their own customers’ revenues down to the last cent.

You can account for these different points of view in *DeltaMaster*. When formatting the measure, you simply use an MDX expression similar to this one:

```
iif([BusinessUnit].CurrentMember.Level.Ordinal=0, "#,###, bil",
    iif([BusinessUnit].CurrentMember.Level.Ordinal=1, "#,##, mil",
        iif([BusinessUnit].CurrentMember.Level.Ordinal=2, "#,##, tsd", "0,0.00")
    )
)
```

You can enter this expression as is in *DeltaMaster*, however, the indentations and multiple rows have only been used to emphasize the structure. Please note that you should always write the name of the dimension in brackets and ensure that you enter the exact “.CurrentMember” name using upper-case and lower-case letters.

In this expression, the “iif” operator appears three times in a nested form. It generally has this syntax

iif(<Condition>, <Return if condition has been met>, <Return if condition has not been met >)

and works similarly to the “IF” function in Excel. If the condition has been met, the first value (i.e. the first string) will be returned. Otherwise, the second string will be returned – and this, in turn, can contain another “iif” expression. The conditions check whether the currently selected member in the dimension ([BusinessUnit].CurrentMember) is located on a certain level. If yes, we have found the appropriate string; if not, the next “iif” expression will run a query on the next level below. As a result, we receive different format strings depending on if our interests lie on an enterprise, country or regional-office level.

“#,#,,,, bil” is the format output for the top member with the ordinal “0”. The three commas at the end of the number create a division through a billion. In order to make this clear, the literal string “bil”, which was displayed in the report, was used. For currency specifications or physical units (e.g. kg, m²), you should use the *Units* field under *Measure properties* (*General* tab). In this case, the unit will only be entered once behind the name of the measure and not repeated in every cell.

Pseudo-inaccuracy

As you can see in the example above, formatting is related to scaling, division and rounding up or down – and that is exactly what makes the report so refreshing. Preparing reports for display is always the final step in the editing process for *DeltaMaster*. It has no influence on the calculation logic. *DeltaMaster* calculates all reports and analyses with the original precision of the data, regardless of how it is later displayed on the screen.

Questions? Comments?

Just contact your Bissantz team for more information!