Number Formatting

Number Formatting is used by the Flat File parser and composer and Integer, Float and String objects (e.g. parseFloatExpression() Operation, printFloatExpression() Operation, parseIntegerExpression() Operation, printIntegerExpression() Operation).

> Locale characters are by default the system locales. These can be overridden by providing a Locale structure as input parameter to Flat File parser/composer or action script functions.

The format processor uses similar patterns as Oracle and PostgreSQL but is not 100% compatible e.g. there's no support for scientific and roman notation.
All parse and compose functionality is influenced by locales. The Flat File adapter and the parse and print number operations take a parameter of type **Basic Components.Basic Behavior.NumbersLocale**. This parameter can be used to specify the following settings:

Class	Attribute	Type	Description
NumbersLocale	negativeSign	**String**	Characters used to signify negative values. Usually '-'.
	positiveSign	**String**	Characters used to signify positive values. Usually '+'.
	thousandsSeparator	**String**	Characters used to separate units of thousand, e.g. ','.
	decimalPoint	**String**	e.g. '.'
	currencySymbol	**String**	e.g. '$'

To specify a number format, the attribute must have the stereotype **FlatFileAttribute**. Default values are given by the system locales.

Patterns

This table shows the template patterns available for formatting numeric values.

Pattern	Description
`9`	Value with the specified number of digits.
`0`	Value with leading zeros.
`.` (period)	Decimal point.
`,` (comma)	Group (thousand) separator.
`PR`	Negative value in angle brackets.
`S`	Sign anchored to number (uses locale).
`L`	Currency symbol (uses locale).
`D`	Decimal point (uses locale).
`G`	Group separator (uses locale).
`MI`	Minus sign in specified position (if number < 0).
`PL`	Plus sign in specified position (if number > 0).
`SG`	Plus/minus sign in specified position.
`TH` or `th`	Ordinal number suffix.
`V`	Shift specified number of digits (see notes).
`FM`	Fill mode prefix, will discard any leading spaces.

Usage Notes

- A sign formatted using SG, PL, or MI is not anchored to the number; for example, `to_char(-12, 'S9999')` produces ' -12', but `to_char(-12, 'MI9999')` produces '- 12'.
- 9 results in a value with the same number of digits as there are 9s. If a digit is not available it outputs a space.

On this Page:

Related Pages:

- printFloatExpression() Operation

- TH does not convert values less than zero and does not convert fractional numbers.
- V effectively multiplies the input values by 10^n, where n is the number of digits following V. `to_char()` does not support the use of V combined with a decimal point. (E.g., 99.9V99 is not allowed.)
- Locale characters are by default the system locales. These can be overriden by providing a Locale structure as input parameter to flatfile parser/composer or script function.

Examples

The examples below use the following locale:

Locale Setting	Value
Negative sign	-
Positive sign	+
Thousands separator	(blank)
Decimal point	,
Currency symbol	$

Number	Format	Output String	Description
-0.1	'FM9.99'	'-.1'	
485	'FM999MI'	'485'	
148.5	'FM999.990'	'148.500'	
148.5	'FM999.999'	'148.5'	
12	'FM9990999.9'	'0012.'	
0.0	'FM999999999999.000'	'.000'	
0.0	'FM999999999990.000'	'0.000'	
0.0	'FM999999999990.099'	'0.0'	
0.0	'FM999999999990.999'	'0.'	
123.12	'99.99'	' ##.##'	Parsing not possible.
0.1	'0.9'	' 0.1'	
-0.1	'99.99'	' -.10'	no leading zeros (9), minus sign sticks to first number
12	'9990999.9'	' 0012.0'	
485	'999'	' 485'	leading blank comes from the plus sign, that is not displayed
-485	'999'	'-485'	
485	'9 9 9'	' 4 8 5'	
1485	'9,999'	' 1,485'	hard coded group separator
1485	'9G999'	' 1 485'	group separator from locale
148.5	'999.999'	' 148.500'	hard coded decimal point
148.5	'999D999'	' 148,500'	decimal point from locale
3148.5	'9G999D999'	' 3 148,500'	group separator and decimal point from locale
12	'99V999'	' 12000'	value increased by factor 1000, because 3 digit are following the specification of V

12.4	'99V999'	' 12400'	value increased by factor 1000, because 3 digit are following the specification of V
12.45	'99V9'	' 124'	value increased by factor 10, because 1 digit is following the specification of V, last decimal is omitted because no decimals are defined in format
-485	'999S'	'485-'	displaying the sign from locale in rear of the number
-485	'999MI'	'485-'	displaying a minus sign in rear of the number, if number is negative
485	'999MI'	'485 '	no minus sign added if number is positive, but empty space instead
485	'PL999'	'+ 485'	adding a plus sign in front of number if number is positive (PL), no trimming of leading spaces (empty sign in this case)
-485	'PL999'	' -485'	adding no plus sign in front of number because number is not positive (PL), negative system sign, no trimming of leading spaces
485	'999PL'	' 485+'	displaying a plus sign in rear of the number, if number is positive (PL), no trimming of leading spaces (empty sign in this case)
-485	'999PL'	'-485 '	displaying a plus sign in rear of the number, if number is positive (PL), no trimming of trailing spaces (empty sign in this case, because number is not positive)
485	'SG999'	'+485'	
-485	'SG999'	'-485'	
-485	'9SG99'	'4-85'	
-485	'999PR'	'<485>'	
485	'L999'	'$ 485'	displaying the currency symbol in front of the number (L), no trimming of leading spaces (empty sign in this case)
0	'999th'	' 0th'	
1	'999th'	' 1st'	
482	'999th'	' 482nd'	
485	'"Good number:"999'	'Good number: 485'	
485.8	'"Pre:" 999" Post: " .999'	'Pre: 485 Post: .800'	