



ModPlus

Guide to “Parametrization” plugin

Table of contents

General information

Strategies

Group 1

1. Self

2. Families

3. Types

Group 2

4. In Host

5. In Group

6. In Room

7. In Space

8. In Mass

9. In Solid

10. Solid Touch

11. Room Boundaries

12. Connection SubElements

13. Nested Families

14. MEP Insulation

15. Rebar In System

17. In MEP System

18. Curtain Wall Components

19. One To Many

20. Parts

16. Materials

Elements filter

Formula Editor

Rules for writing a formula

Writing to a conditionally initial element

Fields

Functions

Mathematical functions

Logical functions

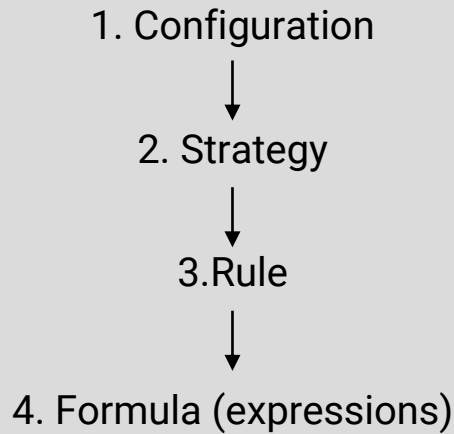
Modifying functions

Get value function

General information

The “Parameterization” plugin is a multifunctional tool that allows you to fill out element parameter values using different strategies, according elements filter and using formulas.

Several levels can be distinguished in the plugin:



PARAMETRIZATION

Configuration: Study

Study

Strategies

- AR
1/1 4. In Host
- KR
2/3 4. In Host
- By Room
2/2 6. In Room
- Group
1/2 17. In MEP System
- Equipment
1/1 3. Types
- Pipes
2/7 1. Self

Strategy

Strategy processes elements that have information about the Hosts. Such elements include: Rebar, Area Reinforcement, Path Reinforcement, Rebar Container, Openings, Stacked Walls, and Families (including elements of Curtain Walls)

AR

Enter strategy description

Data source

File: Sheet:

Strategy filter: Host

Filter elements by c And Filter elements by p

Strategy filter: Target element

Filter elements by c And Filter elements by p

Rules

#	On	Host	Target element	Formula	Formula options
1	<input checked="" type="checkbox"/>	Categories: AND Parameters:	Categories: AND Parameters:	$\$[Mark]=\{\$[Length]/100\}$ $\$[MP_Note]=IF(\$[Len...$ $\$[MP_floor]=Parquet$ $\$[Comments]=IF(\$[Struct...$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Process elements on current View Processing elements from linked files Execute

Actions must be performed in exactly this order after launching the plugin:

1. Create a **configuration**, name it so that it is generally clear what will be processed within this configuration. The configuration with all its strategies can be exported and imported.
2. Select a suitable **strategy** from the list. It is recommended to specify a name and description for it as well. Several strategies can be added within one configuration. Individual strategies with all their rules can be exported and imported.
3. Add a **rule** within the created strategy. Rules are identified by numbers assigned during creation. There can be several rules within one strategy. If necessary, fill in the **elements filter** for the target and/or conditionally source elements.
4. Open the formula editor window for the current rule and write the **expression** into the formula field. You can write several expressions within one rule, each on a new line.
5. You can then add the necessary strategies, rules, and expressions to the current configuration according to the tasks to be performed.
6. In the lower part of the plugin window select the **way of processing elements**: selected elements, on the current view, on selected views or in the whole project.
7. Click the **Execute** button.

The plugin provides element parameters in the following way:

1. The first strategy in the list is taken, followed by the first rule within that strategy, and then the first expression within the rule. Data is sequentially written to the parameter values of the elements according to the first expression, then the elements are processed according to the second expression, and so on.
2. Next, all expressions of the second rule of the first strategy are applied, then the third rule, and so on.
3. After all the rules of the first strategy have been executed, the plugin moves to the next strategy in the list and executes all the rules in it. And so - all strategies within the current configuration.
4. After all strategies have been processed, a result window appears with a report of all operations performed.

PARAMETERIZATION RESULTS
↗ | ✕

Configuration: Study. Total number of strategies: 4. Total number of rules: 9. Total time spent: 00:00:07.5213827
 Searching for relationships between element instances: 00:00:06.6517018
 Total number of processed elements: 305

Expand all

Collapse all

Export to txt

6. In Room - By Room 201 6 0 00:00:00.4723703

Rules executed: 2 ^

- Rule 1 201 6 0 00:00:00.4671992

 - Expression 1. \$[Comments]=@[Number].@[Name] 201 6 0

!

Value setting is skipped, because the parameter "Comments" has the same value 6

✓

Processed 201
 - Rule 2 0 0 0 00:00:00

Rule skipped because no formula is given

17. In MEP System - Group 0 0 0 00:00:00.0791633

Rules executed: 1 v

1. Self - Pipes 0 0 114 00:00:00.0441326

Rules executed: 2 v

Processed elements

Select all

Isolate all

← 1 / 5 →

[525967](#) [526009](#) [526060](#) [526105](#) [719031](#)

[719081](#) [719396](#) [719399](#) [531074](#) [709967](#)

[709968](#) [719033](#) [719360](#) [709951](#) [699904](#)

[701081](#) [719036](#) [719402](#) [736289](#) [736320](#)

[736409](#) [736497](#) [709969](#) [719039](#) [719363](#)

[709952](#) [719042](#) [719045](#) [719048](#) [531062](#)

[531063](#) [531061](#) [719051](#) [719366](#) [735913](#)

[736002](#) [709956](#) [719054](#) [719369](#) [709957](#)

[719057](#) [719372](#) [735718](#) [735813](#) [709958](#)

[719060](#) [719375](#) [735669](#) [709959](#) [719063](#)

In the plugin settings for each strategy, you can set at what point to fix parameter values in the document.

Option 1. When setting a value in a parameter.

Parameter changes in the document will be fixed immediately after writing the value to the parameter of each element. In this case, subsequent formula expressions can use parameter values modified in previous formula expressions. However, this option may require **more processing time**.

Option 2. When processing a rule.

Parameter changes in the document will be fixed after processing all expressions within a single rule. In this case, it is not possible to modify and then reuse the same parameter value within the same rule, as the modified parameter value is not yet fixed in the document.

Option 3. When processing a strategy.

Parameter changes in the document will be fixed after processing all rules within the strategy. In this case, it is not possible to modify and then reuse the same parameter value within the same strategy, as the modified parameter value is not yet fixed in the document.

PARAMETRIZATION

Configuration: Study

Study

Strategies

Add strategy

Import Export

- AR
1/1 4. In Host
- KR
2/3 4. In Host
- By Room
1/2 6. In Room
- Group
1/2 17. In MEP System
- Equipment
1/1 3. Types
- Pipes
2/7 1. Self
- Enter strategy name
3/3 1. Self

Strategy

Strategy processes elements located in Rooms

By Room

Enter strategy description

Data source

File: Sheet:

Strategy filter: Room

Filter elements by parameters:

Strategy filter: Target element

Filter elements by c And Filter elements by p

Rules

	#	On	Room	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Parameters:	Categories: AND Parameters:	\${Comments}=@[Numb...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
⋮	2	<input type="checkbox"/>	Parameters:	Categories: AND Parameters:		Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Process elements on current View Processing elements from linked files Execute

- There is a **checkbox** for each strategy and each rule which allows you to disable individual strategies and rules in the current configuration. If the checkbox is unchecked, the strategy or rule will not be taken for the plugin after clicking the **Execute**.
- When you move the cursor over a strategy in the list, the buttons that allow you to **duplicate** and **remove** a strategy, as well as change the **background color of the strategy** in the list, become available.
- The **Data source** is used in the CV (Cell Value) function.
- Strategies and rules can be moved around in the list, rules can be dragged to other strategies with the left mouse button pressed.
- Rules can be copied to the **clipboard** and pasted both in the current strategy and in other strategies and configurations.
- Rules can be **renumbered** according to their current position in the strategy.
- You can change the **background color for the first cell** of each rule.

- The settings of the **elements filters** per strategy apply to all strategy rules that do not have the corresponding elements filters filled in the rules. If only the category filter is filled in the strategy filter and only the parameter filter is filled in the rule filter, the result will be a combined filter with categories and parameters set. It also works in the opposite way – you can set only categories for the strategy filter, and only parameters for the rule filter. If both the category filter and the parameter filter are filled in the rule, the strategy filter will be ignored.
- If it is possible to use elements filters by categories and parameters, you should use them as much as possible in order to make the formulas more simple and reduce the plugin runtime and the number of possible errors.

Strategies

The strategies differ in the algorithm used to process the elements. They can be divided into several groups:

Group 1. Strategies that process elements without considering interrelationships with other elements

Group 2. Strategies that process elements with respect to their relationship to a conventionally source element (host, room, mass, etc.)

Group 3. A strategy that processes materials.

Before selecting a strategy you should understand, if the value of an element parameter depends on the relationships with other elements.

If yes, it is necessary to determine how the target element is linked to the conditional source element and select a suitable strategy from group 2.

If no, and the element is not a material, one of the group 1 strategies could be applied.

Group 1. Strategies that process elements without considering interrelationships with other elements

1. Self – element instances processing
2. Families – family instances processing
3. Types – types processing

All strategies in the group allow you to write to the target parameter value of the target element:

- arbitrary value
- value of the target parameter as part of the expression
- value of another parameter of the target element
- composite value from the parameters of the target element
- empty value (values of string type parameters will be deleted, numeric type parameters will be set to zero).

1. Self – instance of elements processing

You must be sure to fill in the elements filter by category and/or elements filter by parameters in the **Target elements** column!

Example 1.

	#	On	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Walls AND Parameters:	[\$Comments]=wall [\$Mark]=[\$Mark]_S [\$MP_Level]=FORMAT([\$Base Constraint],d2) level [\$MP_Note]=[\$Family]_[\$Width].1 [\$MP_Designation]=	Tolerance: 0,001 Case sensitive: No Write empty: Yes Dissociate: No

Walls category is selected as the target element in the elements filter by category, the elements filter by parameter is not set, so any walls in the model will be processed.

The target parameter value of the target element will be set as:

- arbitrary value: “wall” will be set as the value of the “Comments” parameter
- value of the target parameter as part of the expression: the previous value of this parameter with the suffix “_S” it will be set as the value of the “Mark” parameter
- value of another parameter of the target element: the value of the parameter “MP_Level” will be filled in as follows: the level number will be extracted from the parameter “Base Constraint” and set as the form of a two-digit number (FORMAT(x,f) function) followed by the word “level”. The result will be “03 level”
- composite value from the parameters of the target element: the value of the parameter “MP_Note” will consist of the value of the parameter “Family” symbol “_” of the value of the parameter “Width” and end with the suffix “.1”. The result will have the form “Base wall_300.1”
- value of the “MP_Designation” parameter will be deleted.

Example 2 – names must be assigned as views based on the value associated with the level of the view.

⋮	2	<input checked="" type="checkbox"/>	Categories: Views AND Parameters: Family and Typ...	$[View\ Name]=AR_IF([Associated\ Level]\sim Level,RS...$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $[View\ Name]=AR_IF([Associated\ Level]\sim Level,RSTR([Associated\ Level],2)\ LSTR([Associated\ Level],5),[Associated\ Level])$					

As the target element in the elements filter by categories “Views” is selected, in the elements filter by parameters it is set that the parameter “Family and Type” must contain “AR”.

The value of the “View Name” parameter will consist of the prefix “AR_”, if the value of the “Associated Level” parameter contains “Level”, 2 final characters of the “Associated Level” parameter value string (RSTR(s,l) function), a space, first 4 characters of the “Associated Level” parameter value string (LSTR(s,l) function) is recorded after the prefix. If the “Associated Level” parameter value does not contain “Level”, the “Associated Level” parameter value is set after the prefix (IF(c, true, false) function).

For example, the “Associated Level” parameter value is “Level 02”. The value “AR_02 level” will be set for the “View Name” parameter.

More information about usage of functions can be found in the [Functions](#) section.

Example 3 – it is necessary to fill in the value of the “Mark” parameter for rebars.


⋮	3	<input checked="" type="checkbox"/>	Categories: Structural Rebar AND Parameters:	$[Mark]=IF([MP_dimension\ in\ linear\ meters]=yes,...$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $[Mark]=IF([MP_dimension\ in\ linear\ meters]=yes,[Bar\ Diameter]-1m,[Bar\ Diameter]-ROUND(0.1*[Bar\ Length],!5))$					

“Structural Rebar” is selected as the target element in the elements filter by category.

If the checkbox is checked in the parameter “MP_Dimension in linear meters”, then in the value of the parameter “Mark” will be set the value of the parameter “Bar diameter” with suffix “-lm”, otherwise the value of the parameter “Bar Diameter”, the symbol “-” and the value of the parameter “Bar Length” converted to cm and rounded off in multiples of 5 will be specified (ROUND(a1,a2) function).

For example, in the parameter “MP_Dimension in linear meters” the checkbox is not checked, the bar diameter is 12 mm, the bar length is 1580 mm. The following expression will be written into the value of the “Mark” parameter: 12-160.

Example 4 – check if the wall is a pylon.


	4	<input checked="" type="checkbox"/>	Categories: Walls AND Parameters:	<code>[\$MP_Note]=IF(OR({[\$Length]/[\$Width]}<4,{\\$[Un...</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:					
<code>[\$MP_Note]=IF(OR({[\$Length]/[\$Width]}<4,{\\$[Unconnected·Height]-250\\$[Length]}>4),pylon,wall)</code>					

“Walls” is selected as the target element in the elements filter by category.

Because of conditions are checked based on the results of arithmetic operations, these operations are enclosed in curly brackets. The parentheses that are not part of functions, but are involved in arithmetic operations are shielded with the “\” symbol.

If at least one of the conditions is met (OR(c1,c2, ...,cn) function), “pylon” will be set as the value of the “MP_Note” parameter. If none of the conditions is met, “wall” will be specified.

Example 5 – it is necessary to write the mark value in meters with three decimal places to the level name.

	5	<input checked="" type="checkbox"/>	Categories: Levels AND Parameters:	<code>[\$Name]=Level FORMAT(\$[Elevation]/1000,f3)</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:					
<code>[\$Name]=Level·FORMAT(\$[Elevation]/1000,f3)</code>					

“Levels” is selected as the target element in the elements filter by category.

The value of the parameter “Name” will consist of the prefix “Level”, followed by the value of the parameter “Elevation” divided by 1000 and written with 3 decimal places (FORMAT(x,f) function)

You can read more about the rules of writing expressions in the [Formula editor](#) section.

2. Families – instance of family processing

The strategy is a special case of the Self for processing loaded families. It is allowed not to use the elements filter in the **Family instance** column.

Example 1

	#	On	Family instance	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Windows AND Parameters: Level<3	$[\text{MP_Mark}] = \text{Window}$ $[\text{Mark}] = \text{IF}([\text{Width}] < 1200, \text{W}-1, \text{W}-2)$ $[\text{Designation}] = [\text{Width}] \times [\text{Height}] \text{ mm}$ $[\text{Comments}] =$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Formula:


```
[$MP_Mark]=Window
[$Mark]=IF([$Width]<1200,W-1,W-2)
[$Designation]=[$Width]x[$Height]·mm
[$Comments]=
```

The strategy allows you to write to the target parameter value of the target element:

- arbitrary value:
 $[\text{MP_Mark}] = \text{OK}$
Result: OK
- value obtained by using another parameter of the target element in the formula condition:
 $[\text{Mark}] = \text{IF}([\text{Width}] < 1200, \text{OK}-1, \text{OK}-2)$
Result: OK-1
- composite value from the parameters of the target element:
 $[\text{Designation}] = [\text{Width}] \times [\text{Height}] \text{ mm}$
Result: 900x2100 mm
- delete the parameter value of the target element:
 $[\text{Comments}] =$


You can read more about the rules of writing expressions in the [Formula editor](#) section.

Example 2 – it is necessary for the Mechanical Equipment to fill in the value of the “Mark” parameter in such a way that each element has a unique value of the parameter.

	2	<input checked="" type="checkbox"/>	Categories: Mechanical Equipment AND Parameters: Type~E04050	$[\text{Mark}] = \text{EINDEX}() - 04050$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $[\text{Mark}] = \text{EINDEX}() - 04050$					

“Mechanical Equipment” is selected as the target element in the elements filter by category, the value of the “Type” parameter must contain “E04050”. The value of the “Mark” parameter will contain the ordinal number of the element in the view or in the model, depending on the way the elements are processed (EINDEX(i) function) with the suffix “E04050”.

Elements are processed in the order in which they are created in the model. If a model contains additional instances, which should be marked, it is possible to set a numeric argument specifying the index offset in the EINDEX(i) function. For example, it is necessary to start numbering from 87-E04050:

	3	<input checked="" type="checkbox"/>	Categories: Mechanical Equipment AND Parameters: Type~E04050	$[\text{Mark}] = \text{IF}([\text{Mark}] = , \text{EINDEX}(86) - 04050,)$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $[\text{Mark}] = \text{IF}([\text{Mark}] = , \text{EINDEX}(86) - 04050,)$					

In this case, if the “Mark” parameter value is not filled in, the number starting from 87-E04050 will be recorded. Elements with the value of this parameter filled in will be skipped.

Example 3 – it is necessary to find the mirrored elements.

⋮	4	<input checked="" type="checkbox"/>	Categories: AND Parameters:	<code>\$(Comments)=IF(TMIRRORED(),\$(ID),)</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: <code>\$(Comments)=IF(TMIRRORED(),\$(ID),)</code>					

No elements filter is filled out for the target element, so all categories of created family instances will be processed.

The “Comments” parameter will be filled with the element ID when the element has been mirrored (TMIRRODED() function). If the element has not been mirrored, nothing will happens because the “Write empty result” option is disabled.

Example 4 – it is necessary to fill in the value of the parameter "MP_Level".

⋮	5	<input checked="" type="checkbox"/>	Categories: AND Parameters:	<code>\$(MP_Level)=FIRSTTRUE(HAS(\$(Level));\$[L...</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: <code>\$(MP_Level)=FIRSTTRUE(HAS(\$(Level));\$[Level];HAS(\$[Base-Constraint]);\$[Base-Constraint];HAS(\$[Reference-Level]);\$[Reference-Level];Check parameters)</code>					

No elements filter is filled out for the target element, so all categories of placed family instances will be processed.

The “MP_Level” will be set to the value of the parameter of the target element (HAS(p) function). If all three parameters are missing, the value of the parameter “MP_Level” will be set for “Check parameters” (FIRSTTRUE(c1:v1;...;cN:vN;v) function).

More information about usage of functions can be found in the [Functions](#) section.

Example 5 – values of several parameters must be set for another, taking into account that some elements may not have some parameters.

⋮	6	<input checked="" type="checkbox"/>	Categories: AND Parameters:	$[\text{Comments}] = [\text{Name short}] - [\text{Material} \dots]$ $[\text{Comments}] = \text{REPLACE}([\text{Comments}], - \dots$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $[\text{Comments}] = [\text{Name short}] - [\text{Material}] - [\text{Nomenclature series}] - [\text{Set}]$ $[\text{Comments}] = \text{REPLACE}([\text{Comments}], \%NO_PARAMETER\%-, -)$					

No elements filter is filled in for the target element, so all categories of placed elements of the families will be processed.

The first expression will set the value of several parameters for the “Comments” parameter. If one of the element parameters is not found, the plugin “%NO_PARAMETER%” instead of its value. With the second expression, in case of absence of any parameter, the value “-%NO_PARAMETER%” will be replaced by “-” (REPLACE(s,s1,s2) function).

3. Types – types processing

This strategy fills in the values of type parameters only.

Unlike the Self strategy, this strategy can be used to process all types in a document, regardless of whether their instances are placed in the model. To do this, you must select the "Process elements in Document" method.

To process the types of selected elements or elements placed in the view, select the "Process picked elements" or "Process elements on current View" options, respectively.

It is obligatory to fill in the elements filter by category and/or elements filter by parameters (only type parameters) in the **Type** column!

Example 1.

For all types whose "Type Name" parameter contains "MP", this string will be set as the value of the "URL" parameter.

#	On	Type	Formula	Formula options
1	<input checked="" type="checkbox"/>	Categories: AND Parameters: Type~MP	<code>\${URL}=MP</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Example 2.

2	<input checked="" type="checkbox"/>	Categories: Windows AND Parameters:	<code>\${MP_Name}=\${Type Name}</code> <code>\${MP_Designation}=\${Rough Height}x\${Ro...</code> <code>\${Glass}=MP_Glass_Transparent Blue</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
---	-------------------------------------	-------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------

Formula:

```


${MP_Name}=${Type Name}
${MP_Designation}=${Rough Height}x${Rough Width}
${Glass}=MP_Glass_Transparent Blue


```

For a filter by category, "Windows" is selected.

The value of the "MP_Name" parameter will be set as the value of the "Type Name" parameter.

The value of the "MP_Designation" parameter will contain an expression of the value of the "Rough Width" parameter, the letter "x" and the value of the "Rough Height" parameter. For example, 1470x1460.

For the “Glass” parameter we will assign the material “MP_Glass_Transparent Blue”.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

Group 2. Strategies that process elements taking into account their relationship to the conditionally source element

4. In Host – processing elements in the Host
5. In Group – processing elements in the Group
6. In Room – processing elements in the Room
7. In Space – processing elements in the Space
8. In Mass – processing elements in the Mass
9. In Solid – processing elements in the Solid
10. Solid Touch – processing elements that touch their solids
11. Room Boundaries – processing room boundaries
12. Connection SubElements – processing sub-elements of Structure Connections
13. Nested Families – processing nested families
14. MEP Insulation – processing Pipe and Duct Insulation
15. Rebar In System – processing of rebars in the system
17. In MEP System – processing elements of MEP system
18. Curtain Wall Components – processing Curtain Wall components
19. One To Many – processing elements in relation to specified source elements

Features of work with strategies of group 2

It is recommended to close Schedules and unnecessary views before starting strategies that work with geometry. It is better to start element processing on the current view at a fine detail level. When starting element processing in a document, the geometry is always taken as for a fine detail level.

If the target parameter (whose value is filled in) is the target element parameter, then the conditionally source elements can be either in the current file or linked files (e.g., in an InMass strategy, the mass can be in a linked file). If the target parameter is the parameter of the conditionally source element, the target elements can be in both the current and linked files (e.g., in the OneToMany strategy, the source element can be in the current file and the target elements in the linked file).

In all group strategies, several ways of interaction between the target element and the conditional source element are possible:

1. Apply **an elements filter to a conditionally source element** to target elements. In doing so, you can write in the value of the target element parameter:

- arbitrary value
- value of the target parameter as part of the expression
- value of another parameter of the target element
- composite value from the parameters of the target element
- empty value

2. Copy of the parameter value of a conditional source element into the parameter value of the target element, including in compound expressions:

$\$[\text{parameter of target element}] = @[\text{parameter of conditionally source element}]$

3. Use **the parameter values of conditionally source elements in the conditions of formulas** for the parameter values of target elements:

$\$[\text{target element parameter}] = \text{IF}(@[\text{conditional source element parameter}] > 10, A, B)$

4. Combine the above methods:

`#[parameter of target element]=IF(#[parameter of conditionally source element1]>10,#[parameter of conditionally source element2],B)`

5. In the value of the conditional source element parameter, write the results of calculating the expression for the target element using the functions of adding and sum.

`JOIN(#[conditional source element parameter],)=#[target element parameter]`

4. In Host – processing elements in the Host

The next categories are included: Rebar, Area Reinforcement, Path Reinforcement, Rebar Container, Openings, Stacked Walls, and Families (including elements of Curtain Walls).

Be sure to fill in the elements filter by category and/or elements filter by parameter in the **Host** column!

The elements filter by category presents only categories of elements that can serve as the host for other elements.

Example 1.

	#	On	Host	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Walls AND Parameters: Type~Exterior	Categories: Structural Re... AND Parameters:	$[\text{MP_product_mark}] = [\text{Mark}]$ $[\text{Comments}] = \text{IF}([\text{REBAR_ELEM_LENGTH}] < 200, t1, t2)$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:						
$[\text{MP_product_mark}] = [\text{Mark}]$						
$[\text{Comments}] = \text{IF}([\text{REBAR_ELEM_LENGTH}] < 200, t1, t2)$						

In the elements filter by category, “Walls” is selected as a host, in the elements filter by parameters it is set that the parameter “Type name” must contain “Exterior”.

Target elements belong to the “Structural Rebar” category.

According to the filter settings, the Structural Rebars placed in the exterior walls will be processed, with only the parameters of the target element used in the formulas.

Example 2.

	#	On	Host	Target element	Formula	Formula options
⋮	2	<input checked="" type="checkbox"/>	Categories: Floors AND Parameters:	Categories: Structural Ar... AND Parameters:	$[\text{Comments}] = [\text{MP_Not...}]$ $[\text{MP_Grouping}] = \text{IF}([\text{Thickness}] < 300, F1, F2)$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:						
$[\text{Comments}] = [\text{MP_Note_rebar}]$						
$[\text{MP_Grouping}] = \text{IF}([\text{Thickness}] < 300, F1, F2)$						

Host – elements of the category “Floors”.

Target element – elements of the category “Structural Area Reinforcement”.

The value of the base parameter “MP_Note” with suffix “_rebar” is copied into the target element parameter “Comments”.

The value of the “MP_Grouping” parameter of the target element is filled depending on the value of the “Thickness” parameter of the floors (host), but its value is not copied.

Example 3 – it is necessary to specify in the value of the door parameter in the wall of which material it is placed.

⋮	3	<input checked="" type="checkbox"/>	<div style="border: 1px solid #ccc; padding: 5px;"><p>Walls ≡</p><p>And ▾</p><p>[Width >= 200 AND Width <= 250] ≡</p></div>	Categories: Doors AND Parameters:	\${MP_Note}=TOLOWER(I...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:						
<code>\${MP_Note}=TOLOWER(IF(STRLEN(@[Type])=24,SSTR(@[Type],12,8),SSTR(@[Type],12,5))) wall</code>						

Host – elements of the “Walls” category with a thickness of 200 to 250 mm.

Target elements belong to the “Doors” category.

The value of the “MP_Note” parameter of the target element is filled depending on the number of characters in the value of the “Type” parameter of the wall (STRLEN(s) function). If there are 24 characters, a substring of 8 characters starting from the 12th character will be extracted from the “Type” parameter value (SSTR(s,I,I) function), otherwise a substring of 5 characters will be extracted. This substring will be converted to lower case (TOLOWER(s) function) and set before the word “wall”.

For example, a door is placed in a wall with the dimension “MP_Interior_Concrete_200”. This door will be sampled by the elements filter by wall parameters. There are 24 characters in the string, so a substring of 8 characters is extracted and converted to lower case. Result: “wall concrete”.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

5. In Group – processing elements in the Group

It is possible not to use the elements filter by parameter in the **Group** column and the elements filters by category and parameter in the **Target element** column.

It must be possible to vary of the target parameter value of the target element must be able to vary by group instances.

Example 1 – it is necessary to specify for the walls which group they are included in.

	#	On	Group	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Parameters:	Categories: Walls AND Parameters:	[\$[Comments]=in group...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: [\$[Comments]=in group · \"@[Type]\"						

The elements filter by parameters for the group is not filled in.

Target elements belong to the “Walls” category.

According to the filter settings, walls included in any groups will be processed. The value of the target element “Comments” parameters will be filled in with the “in group” prefix and quoted value of the parameter “Type” of the conditionally source element (group). Because of quotation marks refer to control characters, they are escaped with \.

Example 2 – Seeking of group related elements.

⋮	2	<input checked="" type="checkbox"/>	Parameters:	Categories: AND Parameters:	[\$[Comments]=in group	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: [\$[Comments]=in group						

Because of the elements filters for the group and target element are not set, all elements within any groups will be processed.

The value of the “Comments” parameter of an element that is part of a group will be filled with “in group” value.

6. In Room – processing elements in the Room

It is possible not to use the elements filter by parameter in the **Room** column and the elements filters by category and parameter in the **Target element** column.

Example 1 – electrical appliances should be marked with the number of the space in which they are located.

	#	On	Room	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Parameters:	Categories: AND Parameters:	<code>#[Comments]=@[Numb...</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: <code>#[Comments]=@[Number].@[Name]</code>						

Because of the elements filter by parameter for room is not set, elements falling into any rooms will be processed.

The elements filter for the target element is also not set, so all room elements will be processed.

The value of the “Comments” parameter of the room element will contain the values of the “Number” and “Name” parameters of the room which element belongs. If the element belongs to several rooms, there will be taken parameter values of only one room, where the centroid of the element is located.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

7. In Space – processing elements in the Space

It is possible not to use the elements filter by parameter in the **Space** column and the elements filters by category and parameter in the **Target element** column.

Example 1 – it is necessary for electrical appliances to indicate the number of the space in which they are located.

	#	On	Space	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Parameters:	Categories: Electrical Fixtures AND Parameters:	$#[\text{Location}] = @[\text{Number}]$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $#[\text{Location}] = @[\text{Number}]$						

Because of the elements filter by parameters for spaces is not set, elements falling into any spaces will be processed.

Target elements belong to the “Electrical Fixtures” category.

The value of the space parameter “Number” will be set as the value of the target element parameter “Location”.

Example 2 – the space must be counted and set as a value of space parameter.

⋮	2	<input checked="" type="checkbox"/>	Parameters: N...	Categories: Lighting Fixtures AND Parameters:	$\text{SUM}(@[\text{MP_Note}]) = 1$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $\text{SUM}(@[\text{MP_Note}]) = 1$						

Elements filter by parameters for spaces: the parameter “Number” must be greater than the value “30”.

Target element – elements of the category “Lighting Fixtures”.

The value of the space parameter “MP_Note” will record the number of lights of each space with a number greater than 30 (SUM(p) function).

8. In Mass – processing elements with Solid geometry and are located in Mass

Masses must have a Solid form. It is recommended to run the strategy on a 3D view.

It is possible not to use the elements filter by parameter in the **Mass** column and the elements filters by category and parameter in the **Target element** column.

Example 1 – the section number for all elements (mass repeat the geometry of sections) must be filled in.

	#	On	Mass	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Parameters:	Categories: AND Parameters:	\${MP_Section Number}...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Formula:

```


$[MP_Section Number]=@[Mark]


```

Because of the filter of elements by parameters for masses is not filled in, the elements falling into all masses will be processed.

The elements filter is also not set for the target element, so all elements that fall into any masses will be processed.

The value of the “MP_Section Number” parameter of the element will contain the value of the “Mark” parameter of the mass in which the element is located.

While the strategy only processes elements that have a Solid form, hollow elements will not be taken. For openings you can additionally use In Host strategy to transfer parameter values from walls (host) to parameters of these elements:

	#	On	Host	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Walls AND Parameters:	Categories: Doors AND Parameters: Famil...	\${MP_Section Number}=@[...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Formula:

```


$[MP_Section Number]=@[MP_Section Number]


```

9. In Solid – processing elements with Solid geometry and are located inside other elements that have Solid geometry

Be sure to populate the elements filter by category in the **Parent element** column!

Example 1 – specify the structural material of the wall structural of these walls for elements located in this wall.

	#	On	Parent element	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Walls AND Parameters:	Categories: AND Parameters:	\$_[Comments]=in the w...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: \$_[Comments]=in the wall -- SSTR(@[Structural Material Name],3)						

The elements of the category “Walls” are selected as the parent element in the elements filter by category.

There is no elements filter filled for the target element, so all elements that have a Solid geometry and are located in walls will be processed.

The value of the “Comments” parameter of the target element will contain “in the wall –” followed by the value of the “Name” parameter of the reference parameter “Structural Material” of the wall, written starting from the 3rd character in the string (SSTR(s,l) function).

In the plugin settings, the “Auxiliary rays” algorithm for finding solid entries must be set so that the plugin can determine whether the target element belongs to only one parent element, in which the centroid of the target element's Solid geometry.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

More information about usage of functions can be found in the [Functions](#) section.

Example 2 – copy the value of the room parameter located in the linked file to the value of the space parameter of the host file.

⋮	2	<input checked="" type="checkbox"/>	Categories: Rooms AND Parameters:	Categories: Spaces AND Parameters:	[\$[Comments]]=@[Depar...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: [\$[Comments]]=@[Department]						

The elements of the category “Rooms” are selected as the parent element in the elements filter by category.

Target elements belong to the “Spaces” category.

The value of the “Department” parameter of room located in the linked file will be set as the value of the “Comments” space parameter.

Example 3 – write as the pipe parameter value marks of the floors crossed by these pipes.

⋮	3	<input checked="" type="checkbox"/>	Categories: Floors AND Parameters:	Categories: Pipes AND Parameters:	[\$[Mark]]=[\$[Mark]]@[Ma...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: [\$[Mark]]=[\$[Mark]]@[Mark]						

The elements of the “Floors” category are selected as the parent element in the elements filter by category.

Target elements belong to the “Pipes” category.

The «Mark» parameter value of pipes will be filled in with values of the same parameter of floors which this pipe intersects. The values of the floor instances are divided with the «space» symbol. The pipe «Mark» parameter must be empty by the start of the rule execution.

In order to the plugin to be able to determine whether the target element belongs to multiple parent elements, the “Boolean operations” algorithm for finding solid entries must be set in the plugin settings.

10. Solid Touch – processing elements with Solid geometry and have flat solid faces that touch* each other

Be sure to fill in the elements filter by category in the **Touchable element** column!

Example 1.

	#	On	Touchable element	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Rooms AND Parameters:	Categories: Floors AND Parameters:	<code>\$(MP_Number of the apartment r...</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: <code>\$(MP_Number of the apartment room)=@[Number]_@[Name]</code>						
⋮	2	<input checked="" type="checkbox"/>	Categories: Floors AND Parameters:	Categories: Rooms AND Parameters:	<code>\$(Floor finish)=@[Floor finish type]</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: <code>\$(Floor finish)=@[Floor finish type]</code>						

Rule 1

Touchable elements belong to the “Rooms” category.

Target elements belong to the “Floors” category.

The parameter “MP_Number of the apartment room” value of the target element (floor) will be recorded to the value composed of the parameters “Number” and “Name” values of the contacted element (room).

Rule 2

Touchable elements belong to the “Floors” category.

Target elements belong to the “Rooms” category.

The “Floor finish” parameter value of the target element (room) will be copied into the “Floor finish type” parameter value of the touching element (floor).

* In the plugin settings you can set the tolerant distance between element faces from 0 to 20 mm.

Example 2 – it is necessary to calculate the length of the joints at the joints between the interior and exterior walls.

⋮	3	<input checked="" type="checkbox"/>	Categories: Walls AND Parameters: Type~...	Categories: Walls AND Parameters: Type~200	$[\text{Comments}] = [\text{Comments}] + 1$ $[\text{MP_joint length}] = \{([\text{Top Constraint Elevation}] - [\text{Base Constraint Elevation}]) * [\text{Comments}] * 2\}$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:						
$[\text{Comments}] = \{[\text{Comments}] + 1\}$ $[\text{MP_joint length}] = \{([\text{Top Constraint Elevation}] - [\text{Base Constraint Elevation}]) * [\text{Comments}] * 2\}$						

Touchable elements belong to the “Walls” category, the parameter “Type” must contain “Outdoor”.

Target elements belong to the “Walls” category, the parameter “Type” must contain “200”.

“Comments” parameter filled with the target element and the number of contact points between a target and contacted elements. The expression is enclosed in curly braces so that the result of an arithmetic operation is added as the parameter value.

The “MP_joint length” parameter consists of wall height multiply the contact points number multiply 2.

The value of the “Comments” parameter should be equal to 0 by the start of running the strategy. Therefore, before this strategy the Self strategy is executed. This helps to set «0» to the «Comments» of the target element:

	#	On	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Walls AND Parameters: Type~200	$[\text{Comments}] = 0$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

You can read more about the rules of writing expressions in the [Formula editor](#) section.

11. Room Boundaries – processing room boundaries elements

Walls, columns, and area boundaries are included.

It is possible not to use the elements filter in the **Room** and **Target element** columns.

Example 1 – set the number of rooms which enclosed by a wall to its parameter.

	#	On	Room	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Parameters:	Categories: Walls AND Parameters:	<code>\$(Mark)=IF(\$(Mark)!=,\$(Mark)\,...</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: <code>\$(Mark)=IF(\$(Mark)!=,\$(Mark)\,@[Number],@[Number])</code>						

Because of there is no elements filter by parameters for room, elements enclosing any rooms will be processed.

Target elements belong to the “Walls” category.

The room numbers, for which the wall serves as a boundary, will be recorded in the wall’s “Mark” parameter, separated by commas. Because the plugin processes elements one by one, the room number of the next room will be appended to the previous value of the “Mark” parameter, separated by a comma. To correctly display the value of the parameter, a condition is used to prevent the addition of an extra comma in the resulting expression.

At the moment of launching this strategy, the “Mark” parameter value must be empty. Therefore, before this strategy the Self strategy is executed, which clears the value of this parameter of the target element.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

12. Connection SubElements – processing SubElements of Structure Connections

Plates, bolts and other structural connection elements are included.

For SubElements, only element categories that can be included in Structural Connections are presented in the elements filter by category. The elements by parameters and the list of parameters in the formula editor are not provided.

Be sure to fill out the elements filter by category in the **SubElement** column!

Example 1 – it is necessary to fill in the name of the plate profile.

	#	On	Connection	SubElement	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Parameters:	Categories: Plates	<code>\$(Profile Name)=- \$[Thi...</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:						
<code>\$(Profile Name)=- \$[Thikness]·ROUND(\$[Width])L=ROUND(\$[Length])/\$(Structural·Material]</code>						

Because of the elements filter by parameters for Structural Connections is not filled in, the elements falling into all connections will be processed.

SubElement – elements of the “Plates” category.

The parameter “Profile Name” value of the target element contains an integral expression with the values of other parameters of the target element, including rounded ones (ROUND(a1) function).

For example : - 10x236L=100/Aluminum.

Example 2 – specify the Connection type for SubElements.

⋮	2	<input checked="" type="checkbox"/>	Parameters: Type~Link	Categories: Anchors, Bolts, Plat...	<code>\$(MP_Node)=@[Type]</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:						
<code>\$(MP_Node)=@[Type]</code>						

Connection – the “Type” parameter must contain “Link”.

SubElement – elements of the “Anchors”, “Plates”, “Bolts”, “Profiles” categories.

The “MP_Node” parameter value of the SubElement will be set as the “Type” parameter value of the Structural Connection, which includes the SubElement.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

13. Nested Families – processing nested families

It is possible not to use elements filters in the **Parent family** and **Nested family** columns.

Example 1 – transfer the system name from the elements of the parent family to the elements of the nested family.

	#	On	Parent family	Nested family	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Mechanic... AND Parameters:	Categories: Mechanical Equ... AND Parameters:	<code>\${System Name}=@[S...</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: <code>\${System Name}=@[System Name]</code>						

Parent families belong to the “Mechanical Equipment” category.

Nested families belong to the “Mechanical Equipment” category.

The “System Name” parameter value of the nested family will be set as the value of the “System Name” parameter of the parent family.

Example 2 – it is necessary to record information about the parent family for all nested families.

⋮	2	<input checked="" type="checkbox"/>	Категории: И Параметры:	Категории: И Параметры:	<code>\${Комментарии}=в...</code>	Допуск: 0,001 Учет регистра: Нет Запись пустых: Нет Отвязать: Нет
Формула: <code>\${Комментарии}=вложенное \, ID ·@[ID] ·@[Семейство ·и ·типоразмер]</code>						

Because of the elements filters for both parent and nested families are not set, all instances of common nested families in the model will be processed.

The nested family “Comments” parameter value will be set with the text “nested, ID” ID of the parent family followed by the family name and the type of parent family.

Example 3 – transfer the system name of the elements of the nested family to the elements of the parent family.

⋮	3	<input checked="" type="checkbox"/>	Категории: Сантехни... И Параметры: Семейст...	Категории: Сантехни... И Параметры: Семейст...	JOIN(@[ADSK_Систе...	Допуск: 0,001 Учет регистра: Нет Запись пустых: Нет Отвязать: Нет
Формула: JOIN(@[ADSK_Система],)=\$[ADSK_Система]						

Parent families belong to the “Plumbing Fixtures” category, the parameter “Family” must contain “Sink”.

Nested families belong to the “Sanitary Fixtures” category, the parameter “Family” must contain “Siphon”.

The “MP_System” parameter value of the parent family will be set as the “MP_System” parameter value of the nested family (JOIN(p,s) function).

You can read more about the rules of writing expressions in the [Formula editor](#) section.

14. MEP Insulation – processing Pipe Insulation elements and/or Duct Insulation elements

It is obligatory to fill in the elements filter by category and/or elements filter by parameter in the column **Pipe or Duct**!

Example 1 – the Duct Area value must be set as the Insulation parameter value.

	#	On	Pipe or Duct	Insulation	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Ducts AND Parameters:	Categories: Duct Insu... AND Parameters:	\${MP_Quantity}=@[Area]	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Pipes or Ducts belong to the “Ducts” category.

Insulations belong to the “Duct Insulation” category.

The “MP_Quantity” parameter value of the Insulation will be set as the “Area” parameter value of the Duct.

Example 2 – Pipe Insulation with a 6 mm thickness must be added.

1. Create insulation by executing the Self strategy:

	#	On	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Pipes AND Parameters:	\${Insulation Type}=MP_insulation tubes	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

2. Set the required Insulation Thickness using the 4. MEP Insulation strategy:

	#	On	Categories: Pipes AND Parameters:	Categories: AND Parameters: Type=MP_insulati...	Formula	Formula options
⋮	2	<input checked="" type="checkbox"/>			\${Insulation Thickness}=6	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Pipes or Ducts belong to the “Pipes” category.

Insulation – category filter not filled in, Type name is “MP_insulation tubes”.

The “Insulation Thickness” value will be set as the value of “6”.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

15. Rebar In System – processing rebars as part of reinforcement system

The next categories are included: Structural Area Reinforcement and Structural Path Reinforcement.

Be sure to fill in the elements filter by category and/or elements filter by parameter in the **System** column!

Example 1 – copy the value of the parameter from the Structural Area Reinforcement to the Structural Rebars included in this system.

#	On	System	Rebar	Formula	Formula options
1	<input checked="" type="checkbox"/>	Categories: Structural Area Reinfo... AND Parameters:	Parameters:	\${[MP_Note]}=@[Mark]	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Systems belong to the “Structural Area Reinforcement” category.

The Rebar column is not filled with the elements filter by parameters, so all rebars belonging to the specified systems will be processed.

The “MP_Note” parameter value of the Structural Rebars will be set as the “Mark” parameter value of the Structural Area Reinforcement.

Example 2 – it is required to fill in the value of the Rebar parameter depending on the length of the Rebar included in the Structural Area Reinforcement.

2	<input checked="" type="checkbox"/>	Categories: Structural Area Reinfo... AND Parameters:	Parameters:	\${[MP_Mark]}=FIRSTTRUE...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $\text{\$[MP_Mark]}=\text{FIRSTTRUE}(\text{INRANGE}(\text{\$[Bar_Length]},0..2000):t1;\text{INRANGE}(\text{\$[Bar_Length]},2001..4000):t2;\text{INRANGE}(\text{\$[Bar_Length]},4001..6000):t3;t4)$					

Systems belong to the “Structural Area Reinforcement” category.

The Rebar column is not filled with the elements filter by parameters, so all rebars belonging to the specified systems will be processed.

The “MP_Mark” parameter value of the Structural Rebars will be set as the value depending on which range the “Bar Length” parameter value of the Structural Rebar falls into (INRANGE(x,min..max) function).

For example, the value “t2” will be recorded for a 2550 mm long Rebar and “t4” for a 6900 mm long Rebar.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

More information about usage of functions can be found in the [Functions](#) section.

17. In MEP System – processing elements that are part of a MEP system

The next categories are included: electrical system, piping systems, and duct systems.

Be sure to fill in the elements filter by category and/or elements filter by parameter in the **MEP system** column!

Example 1 – set the item number for elements of piping systems.

	#	On	MEP system	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Piping System... AND Parameters: Type~Polye...	Categories: AND Parameters:	$\$[MP_Position]=3$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $\$[MP_Position]=3$						

MEP Systems belong to the “Piping Systems” category.

The elements filter is not filled for the target element, so all elements included in Piping Systems containing “Polyethylene” in the “Type” parameter value will be processed.

The “MP_Position” parameter value of the target elements that satisfy the filter conditions will be set to “3”.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

Example 2 – the cross-sectional area of the ducts must be calculated.

⋮	2	<input checked="" type="checkbox"/>	Categories: Duct Systems AND Parameters: Type~Supply	Categories: Ducts AND Parameters:	$\$[Area]=IF(\$[Size]\sim x, FIRSTTRUE...$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $\$[Area]=IF(\$[Size]\sim x, FIRSTTRUE(STRINDEX(\$[Size],x)=3:{{SSTR(\$[Size],0,3)}*{SSTR(\$[Size],4)}/1000000}};STRINDEX(\$[Size],x)=4:{{SSTR(\$[Size],0,4)}*{SSTR(\$[Size],5)}/1000000}}:0),\{ \$[Size]*\$[Size]\}/1000000*PI()/4)$						

MEP Systems belong to the “Duct Systems” category whose “Type” parameter contains “Supply”.

Target elements belong to the “Ducts” category.

The parameter “Area” value of the target elements satisfying the filter conditions will contain the result of the duct area calculation. If the duct parameter value “Size” contains “x”, then depending on the position of “x” in the parameter value (STRINDEX(s1,s2) function), the substrings of this parameter value (SSTR(s,l,l) function) will be multiplied and divided by 1000000. If there is no “x” in the value of the “Size” parameter, then the area will be calculated as the value of the “Size” parameter squared, multiplied by the number π , and divided by 1000000 and by 4.

For example, the “Size” parameter value: “300x500”. The value contains “x”, its index in the expression is 3 (the first character corresponds to index 0), so the substring “300” (the first 3 characters) will be extracted from the string and multiplied by “500” – the substring from the 5th character (index 4, starting from 0), then this product is divided by 1000000. Result: 0,15.

More information about usage of functions can be found in the [Functions](#) section.

Example 3 – writing the names of Plumbing Fixtures into a system parameter without duplicates in alphabetical order.

⋮	3	<input checked="" type="checkbox"/>	Categories: Piping Systems AND Parameters:	Categories: Plumbing Fixt... AND Parameters:	JOIN(@[Comments],, ,ds)...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: JOIN(@[Comments],, ,ds)=[MP_Name]						

MEP systems belong to the “Piping Systems” category.

Target element belong to the "Plumbing Fixtures" category.

The values of the “MP_Name” parameter of all plumbing fixtures connected to this system will be recorded in the "Comments" parameter of the MEP system, separated by commas (using the JOIN(p,s) function). The use of the third argument “ds” in the function allows duplicate names to be removed and arranged in alphabetical order.

18. Curtain Wall Components – processing components of the Curtain Wall

The next categories are included: imposts, panels, doors and windows. Conventional source element is the walls.

It is possible not to use the elements filter by parameter in the **Curtain Wall** column and the elements filters by category and parameter in the **Target element** column.

Example 1 – transfer the "Mark" parameter value from the Curtain Wall to the panels and imposts of this Curtain Wall.

	#	On	Curtain Wall	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Parameters:	Categories: AND Parameters:	<code>\$(Mark)=@[Mark]</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Because of there is no elements filter by parameters for Curtain Walls, all Curtain Walls will be processed.

The elements filter is also not set for the target element, so all elements falling into any Curtain Walls will be processed.


In the "Mark" parameter value of the target elements will be set the "Mark" parameter value of the Curtain Wall in which they are included.

You can read more about the rules of writing expressions in the [Formula editor](#) section.


19. One To Many – processing elements with the possible to use possibility of usage of any specified element of the model as a source element

If more than one element is found by the filters of the source element, the last found element will be used.

Be sure to fill in the elements filters by category in the **Source element** and **Target element** columns!

The source element can be selected in the model using the  button, in this case the filters will be filled in automatically.

Example 1 – copy the values of several parameters from one duct to all other ducts revealed on the current view.

#	On	Source element	Target element	Formula	Formula options
1	<input checked="" type="checkbox"/>	Categories: Ducts AND Parameters: ID=743... 	Categories: Ducts AND Parameters:	$[\text{MP_Kit}] = @[\text{MP_Kit}]$ $[\text{MP_Position}] = @[\text{M...}]$ $[\text{MP_Note}] = @[\text{MP_N...}]$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $[\text{MP_Kit}] = @[\text{MP_Kit}]$ $[\text{MP_Position}] = @[\text{MP_Position}]$ $[\text{MP_Note}] = @[\text{MP_Note}]$					

The source element was selected on the view using the button, the elements filter by category and parameters was filled automatically.

The target elements belong to the “Ducts” category.

The “MP_Kit”, “MP_Position” and “MP_Note” parameters value of the target elements will be copied to the values of these parameters of the source element.

Example 2 – calculate and record the number of pipe fixtures to the parameter of the selected cube element.

#	On	Source element	Target element	Formula	Formula options
2	<input checked="" type="checkbox"/>	Categories: Generic Models AND Parameters: ID=719315	Categories: Pipes AND Parameters: Type Name~P...	$\text{SUM}(@[\text{Comments}]) = \text{IF}(\$[\text{Length}] > 2000, \{\text{ROUND}(\$[\text{Length}] / 2000) + 1\}, 2)$	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula: $\text{SUM}(@[\text{Comments}]) = \text{IF}(\$[\text{Length}] > 2000, \{\text{ROUND}(\$[\text{Length}] / 2000) + 1\}, 2)$					

The source element was selected on the view using the button, the elements filter by category and parameters was filled automatically.

The target elements belong to the “Pipes” category whose Type Name contains the value “PN-10”.

For pipes longer than 2000 mm the number of fixings is calculated as the pipe length divided by 2000 added 1 and rounded to integers (ROUND(a1) function). For pipes shorter than 2000 mm the number of fixings is 2 (IF(c, true, false) function).

Example 3 – it is necessary to calculate the number of bathrooms and balconies in each apartment and enter this number as the room parameter value with tag = 1.

⋮	3	<input checked="" type="checkbox"/>	Categories: Rooms AND Parameters: AR_Apartment...	Categories: Rooms AND Parameters: Name=Bathro...	SUM(@[AR_ Bathrooms n...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:						
SUM(@[AR_ Bathrooms · number])=IF(\$[AR_Apartment · ID]=@[AR_Apartment · ID],1,0)						

The source elements belong to the "Rooms" category, the parameter "AR_Apartment ID" must be equal to 1.

The target elements belong to the "Premises" category, the parameter "Name" should be "Bathroom" or "Balcony".

The source element's parameter "AR_ Bathrooms number" value will contain the number of rooms named "Bathroom" and "Balcony" (SUM(p) function), which have the same "AR_Apartment ID" parameter as for the source element (IF(c, true, false) function). It means they are in the same apartment.

Example 4 – calculate the number of ducts of each length and enter the corresponding value in the parameter of each duct.

⋮	4	<input checked="" type="checkbox"/>	Categories: Generic Models AND Parameters: ID=698108	Categories: Ducts AND Parameters:	JOIN(@[Comments],,)=\$[L... \$[MP_Count]=COUNT(@[C...	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
Formula:						
JOIN(@[Comments],,)=\$[Length] \$[MP_Count]=COUNT(@[Comments], \$[Length])						

The source element was selected on the view using the button, the elements filter by category and parameters was filled in automatically. It is an auxiliary element for calculations.

The target elements belong to the “Ducts” category.

Using the first expression, the lengths of all ducts will be set as the “Comments” parameter value of the source element (JOIN(p,s) function).

With the second expression, the number of ducts with a length equal to the length of this duct will be set as the value of the “MP_Count” parameter of each duct (COUNT(s1,s2) function).

Because of both expressions are written to one rule, the “When setting value to parameter” option must be selected in the plugin settings for correct processing! For the “When processing rule” option, each expression must be written to a separate rule with the same filters.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

More information about usage of functions can be found in the [Functions](#) section.

20. Part – processing of parts considering the source elements from which the parts were created

If a part is created from several source elements, the first element will be considered the source element.

It is possible not to use the elements filter by parameter in the **Part** column and the elements filters by category and parameter in the **Source element** column.

Example 1 – transfer the "Mark" parameter value from walls to the parts created from them.

	#	On	Source element	Part	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Categories: Walls AND Parameters:	Parameters:	<code>\$(Comments)=@[Mark]</code>	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Formula:

```
$(Comments)=@[Mark]
```

The source elements belong to the "Walls" category.

Since no element filter by parameters is set for the parts, all parts will be processed.

The value of the "Mark" parameter from the original wall will be recorded in the "Comments" parameter of the part created from it.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

Group 3. A strategy that processes materials

16. Materials – processing all materials of the current document regardless of the elements selection option

Example 1 – write the substring “concrete” to a parameter of materials which contain this substring in their names.

	#	On	Material	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	Parameters: Name~Concrete	\${Comments}=Concrete	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

For the filter of elements by parameters it is set that materials with the parameter “Name” containing “Concrete” will be processed.

The material parameter “Comments” will contain the substring “Concrete”.

You can read more about the rules of writing expressions in the [Formula editor](#) section.

Elements filter

The elements filter consists of two filters – elements filter by category and elements filter by parameter, with a logical operator AND/OR between them.




Both filters or only one of them may be present in different strategies.

The filter by category is a list of valid element categories.

A filter by parameter is a list of conditions separated by a logical operator AND or OR.


Filter by parameters allows you to filter elements:

- by parameter value
- by presence or absence of parameters
- by the presence of elements in the group or assembly
- by ID and TypeID value
- by evaluation from a surface (Z coordinate).

	#	On	Target element	Formula	Formula options
⋮	1	<input checked="" type="checkbox"/>	<input type="text" value="Walls, Floors"/>  And  <input type="text" value="Structural=1 AND !Group"/> 		Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Example – the plugin will process walls and floors that are not in groups, where the "Structural" parameter value is checked.

Formula Editor





 FORMULA EDITOR ✕

[Instruction](#)

Tolerance: 0.001 Case sensitive Write empty result Dissociate

```
1 $[target parameter of the target element]=IF(@  
[parameter of the conditionally initial element]>100,  
$[parameter of the target element],0)  
2 SUM(@[target parameter of the conditionally initial  
element])=$[parameter of the target element]
```

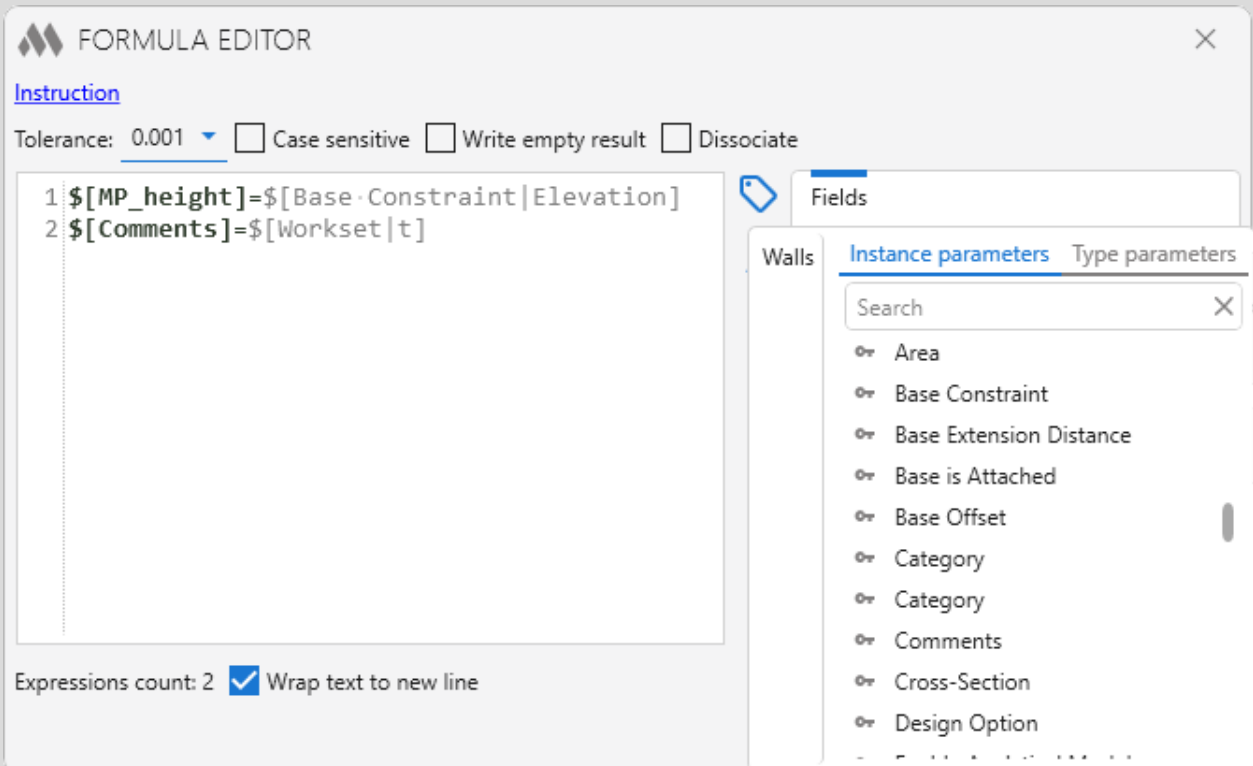
Fields


   

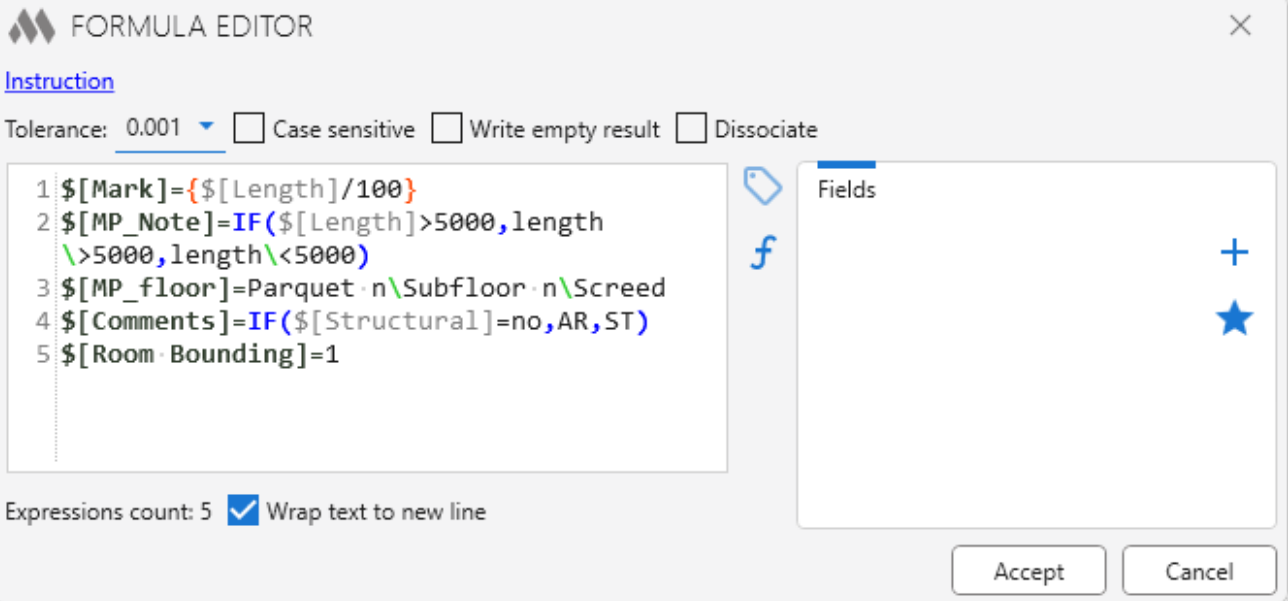
Expressions count: 2 Wrap text to new line

Rules for writing a formula:

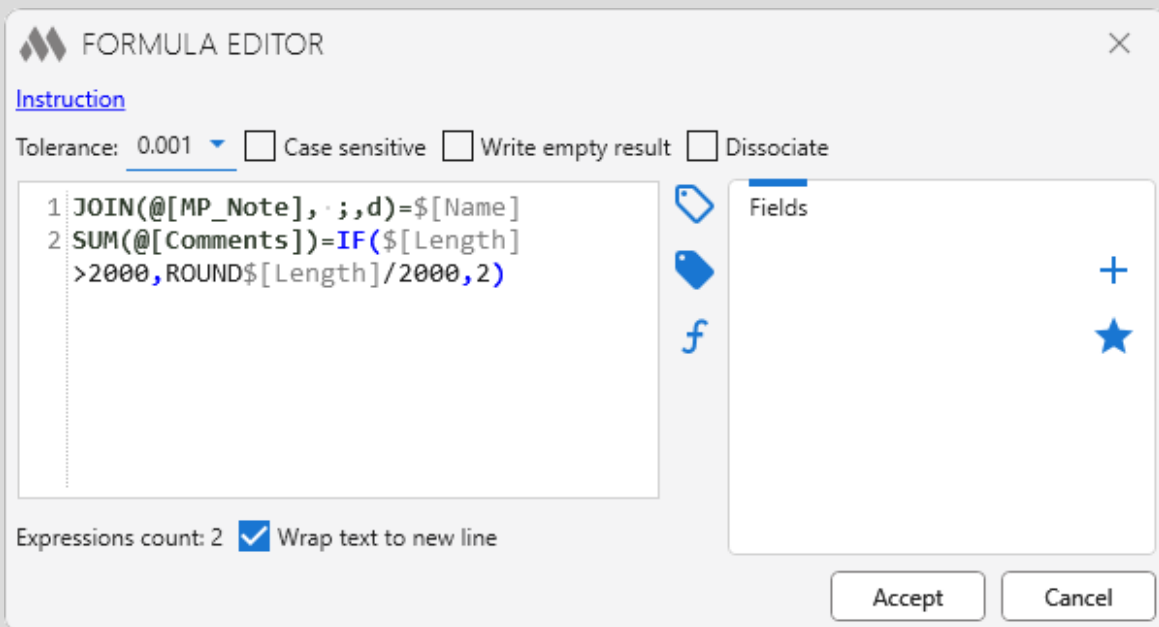
1. A formula consists of the expressions.
2. The expressions can use the parameters of the target element and the parameters of the conditionally source element. Conditional source elements are room for the In Room strategy, group for the In Group strategy, etc.
3. The target element parameters must be enclosed in square brackets started by "\$" character appended. The square brackets contain the parameter identifier – parameter name, GUID of a shared parameters or system name of a built-in parameters.
4. The parameters of the conditionally source element are denoted in the same way as the parameters of a target element, but using the initial "@" character.
5. Each expression can begin:
 - with the target parameter of the target element, for which the result of the expression calculation will be written, and the "=" sign
 - with the add or sum function for the conditionally source element and the "=" sign.



6. You can record the parameter name manually or select from a parameter list . The parameter lists become available after the current rule's elements filter by category is set. The instance parameter list appears when at least one category element is placed in the model. The parameter list is not available for conditionally sourced elements from linked files
7. If you click on the "key" (parameter key) in front of a parameter in the list, a GUID of the shared parameter or the system name of the built-in parameter will set as of the name.
8. If a parameter is a reference to another element of a model (e.g., the "Base Constraint" parameter references the "Level" element), you can get the value of the reference element parameter. To do this, specify two parameter identifiers, separated by a vertical line, where the second parameter refers to the reference element (Example in the strategy).
9. If a specified parameter contains both the instance and the type of element, the instance parameter will be processed. To forcefully specify the type parameter it is necessary to add "|t" at the end of the parameter name.



10. The default expression is a string. If an expression should be defined as a value of an arithmetic operation in an expression, it must be enclosed in curly braces "{}".
11. If the parameter is an argument of an arithmetic function (for example, MIN, ABS, etc.), an argument of an arithmetic condition (>, <, >=, <=), or the parameter is enclosed in curly braces "{}", a numeric value will be taken from the parameter value, regardless of the parameter's data type. For example, the number 10 will be taken from the value "10 m³", and the number 1 will be taken from the value "Level 1".
12. To use control characters in an expression as normal string characters, you must escape them. Using the "\" character before the character. Control characters that can be escaped are () { } < > , : ; and " .
13. To wrap text "Multiline Text" parameters, insert "\n" before the part of the expression that you want to move to the next line.
14. For Yes/No parameters, the value of the parameter when checked equals "1", "true" or "yes" and when unchecked equals "0", "false" or "no". For example, to uncheck a parameter you should write the expression: \$[Parameter]=0.



15. If the strategy has a conditionally source element, the results of the expression calculation for the target element can be set for the conditionally source element using either add or sum function.
16. JOIN(p,s) is a join function. The results of the expression calculation will be joined into a single string using the specified <s> separator and, after processing all matching target elements, will be written to the specified string parameter of the conditional source element <p>. The <s> separator does not require special characters to be escaped. Function can have two or three arguments. As the third, optional, argument you can specify the symbol "d", indicating that duplicate values should be removed from the result list, or the symbol "s", indicating that the result list should be sorted alphabetically. If you specify "ds" as the third argument, the actions for "d" and "s" will be performed simultaneously

```

JOIN(@[Comments],)= $[Mark]
JOIN(@[Comments],, )= $[Mark]
JOIN(@[Comments],,d)= $[Mark]
JOIN(@[Comments],, ,d)= $[Mark]
JOIN(@[Comments],,s)= $[Mark]
JOIN(@[Comments],, ,ds)= $[Mark]

```

[Example in the strategy 1](#)

[Example in the strategy 2](#)

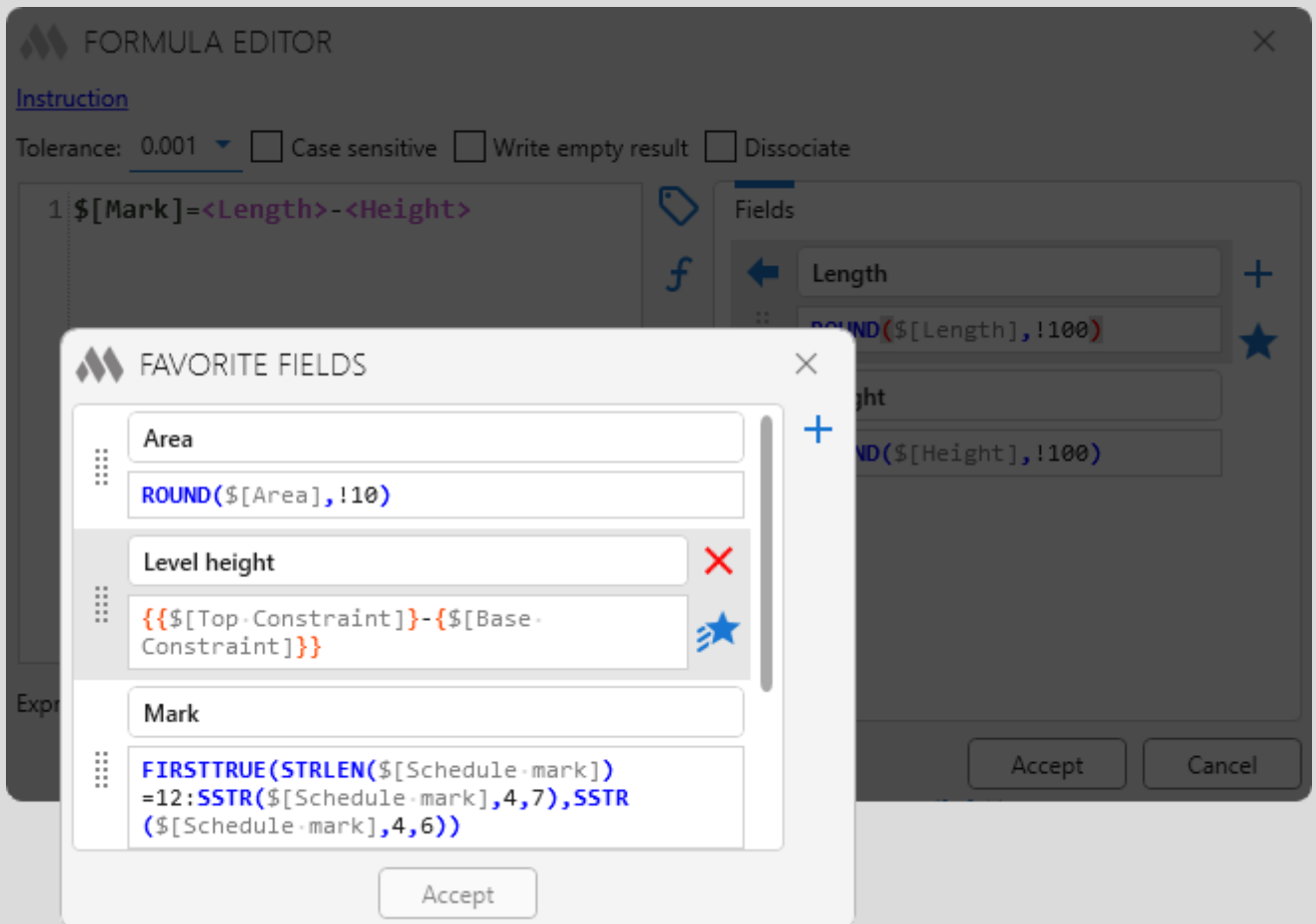
[Example in the strategy 3](#)

17. SUM(p) is a summation function. After all matching target elements have been processed, the numeric result of the expression calculation will be summed and set as the specified numeric or string parameter of the conditional source element <p>. If at least one of the results of the expression calculation is not a number or the parameter value in the target element is not filled, the results will not be set.

SUM(@[Comments])=\$[Length]

[Example in the strategy 1](#)

[Example in the strategy 2](#)



18. Fields allow you to take out parts of a formula and replace them with keys, making it easier to write long formulas or use repeating parts of formulas.
19. Fields consist of two properties – field key and field value. Fields are inserted into a formula as a field key enclosed in angle brackets (<>). The list of formula fields must not contain fields with the same keys.
20. Fields can be inserted into each other.
21. Frequently used fields can be saved to Favorite fields. Fields added to favorites can be edited and their value updated in all rules where they have been added.
22. The list of favorite fields can be opened from the plugin settings menu.

Functions

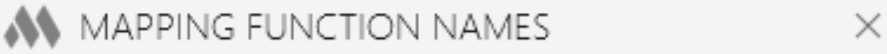
You can use different functions in expressions.

The following types of functions are available in the plugin:

- mathematical functions
- logical functions
- modifying functions
- get value functions.

Functions can be recorded manually or selected from a list by pressing the button **f**.

In the plugin settings, using the “Mapping function names” command, you can set custom names for functions used in the formula editor. Except for the functions JOIN and SUM.



MAPPING FUNCTION NAMES

In this window, you can define your own names for the functions used in the formula editor. Function names must not contain spaces and control characters (brackets, commas, dots, etc.). Case is not important

EINDEX	<input type="text"/>	^
EXP	<input type="text"/>	×
FIRSTTRUE	FT	×
FORMAT	<input type="text"/>	×
HAS	<input type="text"/>	×
IF	<input type="text"/>	×
INRANGE	<input type="text"/>	×
LN	<input type="text"/>	×

Accept

Mathematical functions

Arguments of mathematical functions are treated as default mathematical formulas and do not require curly braces.

MIN(x,y,...,n). Returns the smaller of numbers.

`MIN($[Volume],@[Volume],100)`

MAX(x,y, ...,n). Returns the larger of numbers.

`MAX($[Volume],@[Volume],100)`

ABS(x). Returns the absolute value of the number.

`ABS(-10/$[Length])`

SQRT(x). Returns the square root of a specified number.

`SQRT($[Length])`

POW(x,y). Returns a specified number raised to the specified power.

`POW($[Length],3)`

SIN(a). Returns the sine of a specified angle. Angle is given in degrees.

`SIN(60)`

COS(a). Returns the cosine of a specified angle. Angle is given in degrees.

`COS(60)`

TAN(a). Returns the tangent of a specified angle. Angle is given in degrees.

`TAN(60)`

ASIN(x). Returns the angle (in degrees) whose sine is the specified number.

`ASIN($[Length])`

ACOS(x). Returns the angle (in degrees) whose cosine is the specified number.

`ACOS($[Length])`

ATAN(x). Returns the angle (in degrees) whose tangent is the specified number.

ATAN(\$[Length])

LOG(x,y). Returns the logarithm of a specified number with a specified base.

LOG(\$[Length], 5)

LN(x). Returns the natural logarithm (with base e) of the specified number.

LN(\$[Length])

EXP(x). Returns e multiplied by the specified power.

EXP(\$[Length])

PI(). Returns the ratio of the circumference of a circle to its diameter.

10 * PI()

Mathematical functions. Rounding

ROUND(a1) or **ROUND(a1, a2)** or **ROUND(a1, a2, a3)**. Rounds a value to the nearest integer or to the specified number of fractional digits.

When rounding, you can use one of two rounding conventions:

- ToEven: when a number is halfway between two others, it is rounded toward the nearest even number. The argument is specified as an "e".
- AwayFromZero: when a number is halfway between two others, it is rounded toward the nearest number that is away from zero. The argument is specified as a "z".

Function can have from 1 to 3 arguments, defining the rules of operation:

·1 argument (number) – rounds a double-precision floating-point value to the nearest integer value. Values in the middle are rounded to the nearest even number (the rounding convention is ToEven).

$\text{ROUND}(\text{\$[Length]}) \longrightarrow \text{ROUND}(55.1438)=55$

Example in strategy

·2 arguments (number, number) – rounds a double-precision floating-point value to the specified number of decimal places. The values in the middle are rounded to the nearest even number (the rounding convention is ToEven). The second argument can be an integer (1, 2, 3, etc.), a fractional number (0.1, 0.5, 0.01, etc.), or an integer preceded by an exclamation mark (!10, !50, !100, etc.). A whole number indicates the number of decimal places, and a fractional number or an integer preceded by an exclamation mark indicates the rounding format.

$\text{ROUND}(\text{\$[Length]},3) \longrightarrow \text{ROUND}(55.1438,3)=35,144$

$\text{ROUND}(\text{\$[Length]},!30) \longrightarrow \text{ROUND}(55.1438,!30)=60$

$\text{ROUND}(\text{\$[Length]},0.005) \longrightarrow \text{ROUND}(55.1438,0.005)=55,145$

Example in strategy

·2 arguments (number, letter) – rounds a double-precision floating-point value to an integer using the specified rounding convention. The second argument can be a letter "z" to indicate the AwayFromZero convention or an "e" to indicate the ToEven convention.

$\text{ROUND}(\text{\$[Length]},z) \longrightarrow \text{ROUND}(14.5,z)=15$

$\text{ROUND}(\text{\$[Length]},e) \longrightarrow \text{ROUND}(14.5,e)=14$

·3 arguments (number, number, letter) – rounds a double-precision floating-point value to the specified number of fractional digits, using the specified rounding convention. The second argument can be an integer (1, 2, 3, etc.), a fractional number (0.1, 0.5, 0.01, etc.), or an integer preceded by an exclamation mark (!10, !50, !100, etc.). A whole number indicates the number of decimal places, and a fractional number or an integer preceded by an exclamation mark indicates the rounding format.

$\text{ROUND}(\text{\$[Length]},3,z)$

$\text{ROUND}(\text{\$[Length]},!30,z)$

$\text{ROUND}(\text{\$[Length]},0.005,z)$

ROUNDUP(x). Returns the smallest integral value greater than or equal to the specified number

$\text{ROUNDUP}(\text{\$[Length]}) \longrightarrow \text{ROUNDUP}(55.1438)=56$

ROUNDDOWN(x). Returns the largest integral value less than or equal to the specified number.

$\text{ROUNDDOWN}(\text{\$[Length]}) \longrightarrow \text{ROUNDDOWN}(55.1438)=55$

Logical functions

Logical functions calculate a condition and then return different responses depending on the value of the condition.

In addition to expressions with operators, other [logical functions](#) that return a Boolean value can be used as conditions of logical functions (TRUE/FALSE).

Allowed operators in logical functions:

- = – equals. For numeric or string values
- <> or != – is not equal. For numeric or string values
- > – greater. Only for numeric values
- >= – greater than or equal to. Only for numeric values
- < – less. Only for numeric values
- <= – less than or equal to. Only for numeric values
- ~ – contains. Only for string values
- !~ – not contains. Only for string values.

It is possible to use the conditions "Starts with", "Ends with", "Not starts with", and "Not ends with".

To use the "Starts with" condition, specify the "Contains" condition and add an "*" at the end of the second argument.

`$(Level) ~ Lev*`

To use the "Ends with" condition, specify the "Contains" condition and add an "*" to the beginning of the second argument

`$(Level) ~ * Lev`

To use the "Not starts with" condition, specify the "Not contains" condition and add an "*" at the end of the second argument

`$(Level) !~ Lev *`

To use the "Not ends with" condition, specify the "Not contains" condition and add an "*" to the beginning of the second argument

`$(Level) !~ * Lev`

Logical functions

Functions that return the specified value according to the fulfillment/non-fulfillment of the condition:

IF(c, true, false). Returns the first value <true> if the condition <c> is true. Returns the second value <false> if the condition <c> is not true. Any other function that returns a Boolean value (TRUE/FALSE) can be used as a condition <c>.

IF(\$[Length]>100, Long,Short)

Example in strategy

FIRSTTRUE(c1:v1;c2:v2;...;cN:vN;v). Returns the first value of <v> whose <c> condition is true. If all conditions <c> are not true, returns last <v> value. Any other functions that return a Boolean value (TRUE/FALSE) can be used as <c> conditions.

FIRSTTRUE(\$[Length]>100:A;AND(\$[Length]>50,\$[Height]>50):B;C)

Example in strategy

Functions that can be used in the context of other logical functions:

AND(c1,c2, ...,cn). Returns TRUE if all conditions <c> are true. Otherwise returns FALSE.

IF(AND(\$[Length]>100,\$[Height]>100), Big,Small)

OR(c1,c2, ...,cn). Returns TRUE if at least one condition <c> is true. Otherwise returns FALSE.

IF(OR(\$[Length]>100,\$[Height]>100), Big,Small)

Example in strategy

NOT(c). Returns TRUE if condition <c> is false. Returns FALSE if condition <c> is true. The function can be used as a condition of other conditional functions. Any other function returning a Boolean value (TRUE/FALSE) can be used as a condition <c>

NOT(TMIRRORED())

INRANGE(x,min..max). Returns TRUE if the number <x> is greater than or equal to <min> and less than or equal to <max>. Otherwise returns FALSE.

AND(INRANGE(\$[Length],1000..5000),INRANGE(\$[Height], 1000..5000))

Example in strategy

HAS(p). Returns TRUE if parameter <p> exists. Otherwise returns FALSE. If \$[] is specified, then the parameter will be checked in the target element. If @[] is specified, then the parameter will be checked in the conditionally source element.

IF(HAS(\$[Length]), yes,no)

Example in strategy

TMIRRORED(). Returns TRUE if the target element is a family that was mirrored. Otherwise it returns FALSE.

IF(TMIRRORED(), Mirrored,ok)

Example in strategy

SMIRRORED(). Returns TRUE if the conditional source element is a family that was mirrored. Otherwise it returns FALSE.

IF(SMIRRORED(), In the mirrored,ok)

Modifying functions

TOUPPER(s). Returns the value converted to uppercase

TOUPPER(\$[Level]) → TOUPPER(Level 06)=LEVEL 06

TOLOWER(s). Returns the value converted to lowercase

TOLOWER(\$[Level]) → TOUPPER(Level 06)=level 06

Example in strategy

LSTR(s, I). Returns the specified number of characters from the beginning of the string

LSTR(\$[Level],4) → LSTR(Level 06,5)=Level

Example in strategy

RSTR(s, I). Returns the specified number of characters from the end of the string

RSTR (\$[Level],3) → RSTR (Level 06,3)= 06

Example in strategy

SSTR(s, I, I). Returns a substring.

The function can have 2 or 3 arguments:

·2 arguments (string, number) – extracts a substring from a string. The substring begins at the specified character position and continues to the end of the string. The first character in the string is index 0

SSTR (\$[Level],2) → SSTR (Level 06,2)=vel 06

Example in strategy

·3 arguments (string, number, number) – extracts a substring from the string. The substring starts at the specified character position and has the specified length. The first character in the string is index 0

SSTR (\$[Level],2,3) → SSTR (Level 06,2,2)=vel

Example in strategy

REPLACE(s,s1,s2). Returns the value of <s>, with all found occurrences of <s1> replaced by the value of <s2>. If the <s1> value is not found in <s>, it returns the <s> value unchanged. All values – <s>, <s1>, <s2> – are treated as strings. The function is case sensitive.

REPLACE(Test,s,x) → Text

REPLACE(\$[Comments],1,5) → (var. 1,1,5)=var. 5

Example in strategy

FORMAT(x,f). Converts a number <x> to a string using the specified <f> format. The following table shows specifiers of valid formats:

Standard:

- D or d. The number is converted to a string of decimal digits (0-9); if the number is negative, it is preceded by a negative sign. This format is only available for integer types. If the source number is fractional, it will be converted to integer by discarding the fractional part. The minimum number of characters in the output string is specified by the precision specifier. Missing characters in the string are replaced by zeros. If the precision specifier is not specified, the minimum value that allows representing an integer without zeros at the beginning is used by default.

FORMAT(123,d) → 123

FORMAT(123,d5) → 00123

FORMAT(123.3,d) → 123

Example in strategy

- F or f. The number is converted into a string of the form “-ddd.ddd...”, where each character “d” denotes a digit (0-9). If the number is negative, a negative sign is placed at the beginning of the string. The required number of digits of the fractional part is specified by the precision specifier. If there is no accuracy specifier, the numerical accuracy determined by the current operating system settings is used. Separator of integer and fractional part is defined by plugin settings.

FORMAT(15,f2) → 15.00
FORMAT(15.5,f3) → 15.500

Example in strategy

Customizable:

- 0. Replaces zero with the corresponding digit, if any. Otherwise, the resulting string will contain zero. The format causes the value to be rounded to the nearest digit preceding the decimal point-separator. Separator of integer and fractional parts is determined by plugin settings.

FORMAT(15,0.00) → 15.00
FORMAT(15.455,0.00) → 15.46
FORMAT(15.455,000.0) → 015.5

DISTINCT(s1,s2). Returns differing values from the <s1> list, considering <s2> as the separator of values in the list. The function ignores initial and final spaces when comparing list values. For example: when processing the list "a, a, b" with "," separator, the function will return "a, b" because the values "a" and " a" will be considered the same. The function is case sensitive

DISTINCT(\$[Comments],\,)

Get value function

STRLEN(s). Returns the string length.

STRLEN(\$[Comments]) → STRLEN(Этаж 06)=7

Example in strategy

STRINDEX(s1,s2). Returns the index starting from zero of the first occurrence of substring <s2> in string <s1>. Returns -1 if the <s2> substring is not found in the <s1> string. Case insensitive function

STRINDEX(\$[Comments],_) → STRINDEX(Тип_A_15,_) = 4

Example in strategy

LSTRINDEX(s1,s2). Returns the index starting from zero of the last occurrence of substring <s2> in string <s1>. Returns -1 if the <s2> substring is not found in the <s1> string. Case insensitive function

LSTRINDEX(\$[Comments],_) → LSTRINDEX(Тип_A_15,_) = 6

COUNT(s1,s2). Returns the number of times the <s2> substring occurs in the <s1> string. If the <s2> substring is not found in the <s1> string, returns 0. The function is case insensitive

COUNT(\$[Comments],_) → COUNT(Тип_A_15,_) = 2

Example in strategy

VALUESTR(p). Returns the value of the parameter as a string with units. The function must contain one argument that is a pointer to the target element parameter (\$[]) or to the conditional source element parameter (@[]). The function is relevant only for those parameters whose units are displayed in the standard properties palette

\$[Mark]=VALUESTR(\$[Diameter]) → \$[Mark]=150 mm

EINDEX(i). Returns the index of the successful use of the current formula expression. Each element that matches the filtering conditions of the rule is processed by the formula expression of that rule. If the element was processed successfully – i.e. a parameter value is set - the counter of the successful use of the expression is incremented by 1. This function allows you to insert the counter value into the expression. You can specify an optional numeric argument in the function that sets the index offset. For example, if you set the argument to 5, the first value obtained by the function is 6.

$\$[\text{Mark}]=\text{C-EINDEX}() \longrightarrow \$[\text{Mark}]=\text{C-1}$

$\$[\text{Mark}]=\text{C-EINDEX}(5) \longrightarrow \$[\text{Mark}]=\text{C-6}$

Example in strategy

CV(cell). Returns the value from the specified cell of the strategy data source (Excel file). The cell address in the format "A1" should be specified as the <cell> argument. Important: the retrieved cell value is not validated! If the value contains control characters, operators and the like, it may cause errors in further calculation of the expression.

CV(A1)

Example of use:

Data source

File: Product.xlsx Sheet: 1.

Strategy filter: Family instance

Filter elements by category: And Filter elements by parameters:

Rules

#	On	Family instance	Formula	Formula options
1	<input checked="" type="checkbox"/>	Categories: Walls AND Parameters:	$\$[\text{MP_Manufacturing plant}]=\text{CV}(\text{B3})$ $\$[\text{MP_Unit of measurement}]=\text{m}$ $\$[\text{MP_Product code}]=\text{CV}(\text{C3})$ $\$[\text{MP_Technical characteristic}]=\text{CV}(\text{E5})$	Tolerance: 0,001 Case sensitive: No Write empty: Yes Dissociate: No

Formula:

$\$[\text{MP_Manufacturing plant}]=\text{CV}(\text{B3})$
 $\$[\text{MP_Unit of measurement}]=\text{m}$
 $\$[\text{MP_Product code}]=\text{CV}(\text{C3})$
 $\$[\text{MP_Technical characteristic}]=\text{CV}(\text{E5})$

The values are taken from the cells specified in the expressions from the sheet "1." of the Excel document "Product".

CVIF(c1,c2,...,cN,r). Looks for the first row in the strategy data source (Excel file) where all <cN> conditions are met, and returns the value from the specified <r> column of that row. All <cN> conditions on the left side must contain a column number or letter. The <r> value must also contain the column number or letter. All <cN> conditions are checked using the same rules and can contain the same operators as conditional functions. Important: the retrieved cell value is not validated! If the value contains control characters, operators and the like, it may cause errors in further calculation of the expression

CVIF(A=5,B=10,C
CVIF(1~text,2!=15,5)

Example of use:

Data source

File: Product.xlsx Sheet: 2

Strategy filter: Target element

Conduits And Filter elements by parameters:

Rules

#	On	Target element	Formula	Formula options
1	<input checked="" type="checkbox"/>	Categories: AND Parameters: Type~smooth	\${MP_Name}=CVIF(c=SM,f>1000,a)	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No
2	<input checked="" type="checkbox"/>	Categories: AND Parameters: Type~corrugated	\${MP_Name}=CVIF(c=CO,f>1000,a)	Tolerance: 0,001 Case sensitive: No Write empty: No Dissociate: No

Formula:
\${MP_Name}=CVIF(c=CO,f>1000,a)

The value is taken from the cell in column "a" of the first row that meets the conditions for the cells in columns "c" and "f" on sheet "2." of the Excel document "Product".