Creatio Academy

Formulas

Process formulas

Version 8.0

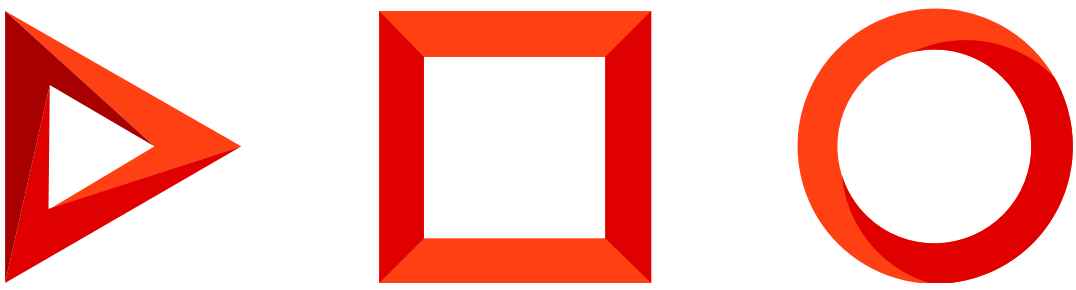

Table of Contents

Process formulas

PRODUCTS: ALL CREATIO PRODUCTS

The formula window in the business process element settings enables you to solve many tasks without developer involvement. The functions of the formula range from automatically generating email message texts, determining the conditions for transitions between flows. Using formulas requires knowledge of basic syntax rules, which are covered in this article.

Basic syntax rules

The formula syntax will be familiar to anyone who worked with C#. When entering formulas, it is important to follow typing. If possible, use values of one type, for example, text values with text values, numeric values with numeric values. Otherwise, you must convert values to the proper type.

In addition, we recommend you to familiarize yourself with the basic operators that will help to implement complex conditions in your formulas.

""	Text strings must be enclosed in quotes.
+	Used to connect values.
==	Defines the equality of two values.
!=	Defines the inequality of two values.
<, >	Compares the two values (less than, greater than).
>=, <=	Compares the two values (greater or equal, less or equal).
&&	Boolean “And.”
||	Boolean “Or.”
true, false	Boolean values “True” and “False.”
\n, 	Text line wrap.

Build a complex text

The [*Formula*] field is usually used to generate variable text.

Example. Use a formula to generate a header for the activity to compile product bundle. The header text must contain the product and the customer for whom the bundle is compiled.

To do this, add the [*Read data*] element parameters and constant text values to the formula:

```
"Compile bundle" + [#Read products.First element of the resulting collection.Name#] + "for custo
```

Note. Add spaces between static data and quotation marks "" to separate the resulting text.

Example. Copy a complex text to the [*Recommendation for filling in the page*] business process element

You can combine two and more strings using the formula dialog. Use the + character for concatenation. To introduce a new line, use the \n control character, for example:

```
"1. Make an appointment with the manager." + "\n" + "2. Discuss the deal tactics with the manage
```

String literals must always be wrapped in straight double quotes (" "). To ensure that control characters work as intended, check the “Is multiline” box. Otherwise, all newline characters will be filtered.

Note. To wrap the text lines you can also use the “
” HTML tag.

Use different types of data

When working with the [*Formula*] element, use a single type of data. There is no need to memorize the typification — on the formula edit page, the data type of each parameter is marked with:

- Id — unique identifier;
- 123 — numeric;
- 0.5 — fractional;
- T — text;
- — lookup value;
- — time and date value.

Example. Use a formula to generate a header for the activity to compile product bundle. Specify the product name, customer and the date of product compilation.

In this case, the compilation date parameter cannot be added in the usual way. To do this, you need to convert the date value to the text value:

```
"Compile bundle" + [#Read products.First element of the resulting collection.Name#] + " for cust
```

To convert the [*#Read order.The first element of the resulting collection.Planned date of completion#*] parameter, enclose it in parentheses and add the .ToString() property. In this case, the business process will work correctly.

Use date and time parameters

To execute business processes with the use of operations with date and time, you can use the C# DateTime structure. The main characteristics and methods are as follows:

.Date	Returns the date value of the selected parameter.
.Hour	Returns the hours value of the selected date parameter.
DateTime.MinValue	The minimum value of date and time, 00:00, UTC, January 1, 0001.
.TotalMinutes	Returns the full date and time value in minutes.
.TotalHours	Returns the full date and time value in hours.
.TotalDays	Returns the full date and time value in days.
.AddMinutes(), .AddHours(), .AddDays()	Increase the selected value of the date and time for a certain number of minutes, hours or days.

Example. In the lead qualification process, before transitioning between conditional flows, Creatio must check whether the decision date field is populated.

To check whether a date field is populated, use the != operator and the DateTime.MinValue parameter:

```
[#Read Lead data after Qualification.First element of the resulting collection.Decision date#]!=
```

Example. To transition between conditional flows, Creatio must compare closing dates of two opportunities.

To compare two date values, use the == operator and the .Date property:

```
[#Read opportunity data 1.First element of the resulting collection.Closing date#].Date
== [#Read opportunity data 2.First element of the resulting collection.Closing date#].Date
```

Example. During the execution of the business process, it is necessary to calculate the time it took to close the opportunity.

If you need to calculate the difference between two date values, use the following construct:

```
(decimal)RoundOff(([#Read opportunity data.First element of the resulting collection.Closing
 date#]-[#Read opportunity data.First element of the resulting collection.Creation date#]).Total
```

In the formula window, select the [*RoundOff*] function and fill it in with the necessary process element parameters, in our case, the difference of values, and then add the .TotalMinutes property. As a result, you will get the elapsed time in minutes. Use the .TotalHours or .TotalDays properties to obtain same result in hours or days.

Example. To transition between conditional flows in a business process, you must determine whether the lead was generated more than 12 hours ago.

To do this, add the.TotalHours property to the element parameter, enclose the parameter value in parentheses, and then compare it with a numeric value:

```
(decimal)RoundOff(([#System variable.Current date and time#]-[#Read lead data.First element of t
```

When working with date and time parameters, you can also use .AddMinutes(), .AddHours() and .AddDays() functions to increment the time and date to a certain value. For example, you can bring the date and time value to user's time zone when using the [*Read data*] element.

Note. In Creatio, the "Date/Time" type data is stored in UTC. The [*Read data*] element does not adjust this data to the current user's time zone.

Example. When performing lead registration business process, you need to set the call time to 3 hours after the lead creation.

To do this, add the number of hours in the .AddHours() value:

```
([#Read lead data.First element of the resulting collection.Creation date#].AddHours(3)).Hour
```

Work with lookup parameters and conditional flows

The functionality of the [*Formula*] element in a conditional flow is same as in other process elements. That is, the basic rules and operators are relevant. Conditional flows are used to transition to the next process element, if transition conditions are met.

Often, values of the lookup parameters must be compared with a specific lookup value.

Example. Check whether an opportunity is on the “Commercial offer” stage of the corporate business process.

To do this, compare the current opportunity stage in the conditional flow with the value in the lookup:

```
[#Read opportunity data.First element of the resulting collection.Stage#]==[#Lookup.Opportunity
```

Example. A conditional flow must be activated if the lead contact lookup field is populated.

To check whether the lookup field is populated, use the following condition:

```
[#Read lead data.First element of the resulting collection.Contact#]!= Guid.Empty
```

To check whether the lookup field is not populated, use the following construct:

```
[#Read lead data.First element of the resulting collection.Contact#]==Guid.Empty
```

Example. A transition in the invoice approval must check whether the approval has been acquired.

To do this, use the following condition:

```
[#Invoice approval.Result#] == "Approved"
```

In this case, the approval result is checked.

Note that there may be several solutions of this case. For example, you can use a more complex structure:

```
[#Invoice approval.Result#] == "Rejected" || [#Invoice approval.Result#] == "Approval pending"
```

In this case, Creatio checks whether the approval status is not “Rejected” or “Approval pending.”

Example. A conditional flow in a sale process must be activated if the presentation has been conducted.

To check the status of your presentation, use the following structure:

```
[#Give presentation.Result#] == true
```

If the presentation has not been conducted, the conditional flow will not be activated.