Smart Processing with Formulas

Download the PDF of this article.

In this Article

Related Articles

The FormAssembly Formula Engine and Editor

As FormAssembly users, you have the option to dynamically customize items like email templates, thank you messages, and display notifications using our FormAssembly formula engine. Additionally, you can use formulas within many of our connectors to to achieve more advanced customization.

> **Note**: The formula engine deals with post-submission processing only. It does not modify the submitted data, and cannot be used inside the form itself. Also, for Enterprise users, formula usage may be restricted on a role-by-role basis.

For calculated fields within the form, see Form Calculations.

If you'd like to learn more about formulas and calculations, check out the recording from our Smart Forms class here.

Formula Structure

The syntax of our formulas is very similar to Excel formulas, but there are some slight differences. In general, each formula consists of a field alias, a function, and a comparison operator, all of which are outlined below.

To make the process of formula building easier we have created a Formula Editor, which allows you to design your formulas in a more simple, dropdown environment, rather than having to write them from scratch.

Field Aliases

When creating a formula, you can reference any field in your form using the **alias** syntax, which is

an internal field name surrounded by two percent signs (e.g., `%%field_name%%`). The alias will be replaced with the actual value submitted with the form when the formula is evaluated.

Here's an example of a formula you could use in an acknowledgment message:

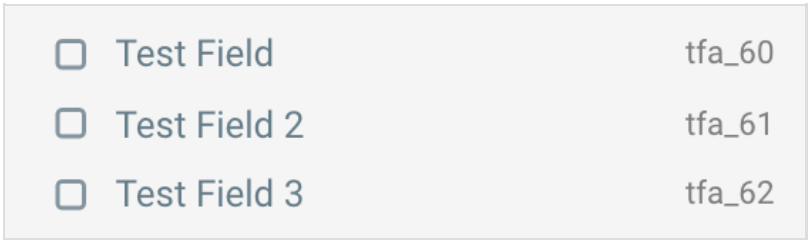
```
Dear %%tfa_salutation%% %%tfa_lastname%%, ...
```

Once evaluated, this formula would be translated to `Dear Mr. Smith`, or `Dear Ms. John` depending on how the user had filled out their form.

Alias List

Field aliases vary from form to form. To see the list of aliases for your form, open the **Notifications** tab of your form settings. At the bottom of this tab, click the **Show the list of available aliases for this form** link. You can also view the available aliases from within the Formula Editor (explained below).

The alias list is also accessible from other places in FormAssembly where formulas and aliases can be used, and you can see the alias for each field on the right side of the Outline view in your Form Builder.



Functions

The engine supports most functions found in MS Excel. The function must be spelled in uppercase and start with the `@` character. Here's a list of the most useful functions:

Logic

@IF

```
@IF(condition,when_true,when_false)
```

Performs a logical test and returns either the second parameter (*if true*) or the third parameter (*if false*).

Example:

```
@IF(%%field_a%%>5,"GOOD","NOT ENOUGH")
```

@AND	@AND(*condition 1*,*condition 2*) Returns TRUE if both conditions are true, FALSE otherwise (logical AND). Example: @IF(@AND(%%field_a%%>5,%%field_b%%>10),"OK","NOT OK")
@OR	@OR(*condition 1*,*condition 2*) Returns TRUE if at least one condition is true, FALSE if all conditions are false (logical OR). Example: @IF(@OR(%%field_a%%>5,%%field_b%%>10),when_true,when_false)
@NOT	@NOT(*condition*) Returns TRUE if at the condition is false, FALSE otherwise (logical NOT). Example: @IF(@NOT(%%field_a%%>5),"OK","NOT OK")

Arithmetic

@COMPUTE	@COMPUTE(%%field_A%%+%%field_B%%) Performs arithmetic calculations on form fields.
@MAX	@MAX(*number1*,*number2*,...) Returns the largest value from the numbers provided.
@MIN	@MIN(*number1*,*number2*,...) Returns the smallest value from the numbers provided.

@ROUND	@ROUND(*number*,*decimal_places*) Returns a number rounded to a specified number of decimal places.

String Operations

@CONCATENATE	@CONCATENATE(*string*,*string*,...) Joins 2 or more strings together. Example: @CONCATENATE(%%tfa_firstname%%," ",%%tfa_lastname%%)
@LEFT	@LEFT(*text*, *number_of_characters*) Extracts a number of characters from a string, starting from the left. Example: @LEFT(%%field_a%%,5)
@RIGHT	@RIGHT(*text*, *number_of_characters*) Extracts a number of characters from a string, starting from the right. Example: @RIGHT(%%field_a%%,5)
@MID	@MID(*text*, *start_position*, *number_of_characters*) Extracts a number of characters starting at any position.

@FIND	@FIND(*text1*, *text2*,*start_position*) Returns the location of a substring in a string (case-sensitive). Returns#VALUE! if string not found. Example: @MID(%%tfa_email%%,@COMPUTE(@FIND("@",%%tfa_email%%)+1), @COMPUTE(@FIND(".",%%tfa_email%%)-@FIND("@",%%tfa_email%%)-1) (Returns the domain part of an email address).
@CONTAINS	@CONTAINS(*text*, *field*) Returns true of text is inside of field. Otherwise, returns false.

Date & Time

@LOCALNOW	@LOCALNOW() Returns the current date *and* time according to your language and time-zone settings.
@LOCALTODAY	@LOCALTODAY() Returns the current date according to your language and time-zone settings.
@NOW	@NOW() Returns the current date as a timestamp. This can be passed to other date functions to extract the day, month or year.
@YEAR	@YEAR(*date_value*) Returns a four-digit year given a valid date. Example: @YEAR(@NOW())

URL Manipulation

@URLENCODE	@URLENCODE(*query_string_param*)
	
	Returns the RFC3986-encoded version of the string passed in.
	
	Example:
	`http://www.tfaforms.com/123?tfa_Name=@URLENCODE("Mike Johnson")`
	
	will output:
	`http://www.tfaforms.com/123?tfa_Name=Mike%20Johnson`

@URLDECODE	@URLDECODE(*query_string_param*)
	
	Decodes RFC3986-encoded version of the string passed in.
	
	Example:
	`http://www.tfaforms.com/123?tfa_Name=@URLDECODE("Mike%20Johnson")`
	
	will output:
	`http://www.tfaforms.com/123?tfa_Name=Mike Johnson`

@SUBSTITUTE	@SUBSTITUTE(*base_string*,*match*,*match_replacement*)
	
	Replaces all occurrences of `match` with `match_replacement` in `base_string`.
	
	Example:
	`@SUBSTITUTE("One Two Three Four","Four","Five")`
	
	will output:
	`One Two Three Five`

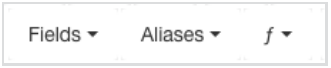
Comparison Operators

A comparison operator can be used to compare two values in a `@IF` statement. The result is a logical value, either `TRUE` or `FALSE`.

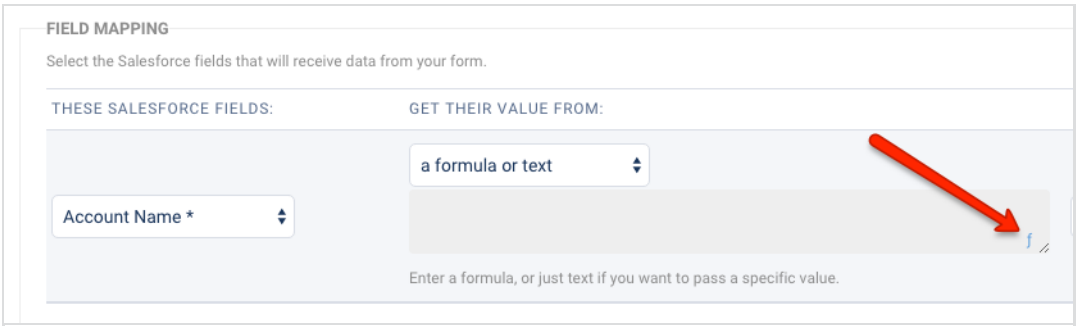
Operator	Name	Example
=	Equal to	`@IF(%%field_a%%="A",when_true,when_false)`
>	Greater than	`@IF(%%field_a%%>5,when_true,when_false)`
<	Less than	`@IF(%%field_a%%<5,when_true,when_false)`
>=	Greater than or equal to	`@IF(%%field_a%%>=5,when_true,when_false)`
<=	Lower than or equal to	`@IF(%%field_a%%<=5,when_true,when_false)`
<>	Not equal	`@IF(%%field_a%%<>5,when_true,when_false)`

The Formula Editor

As mentioned earlier, we have created a Formula Editor to make the process of formula creation easier on your end. This editor can be found on the Display tab, the Notifications tab, and within many of our connectors. To use the Formula Editor, look for fields, aliases, or f-value dropdown menus.

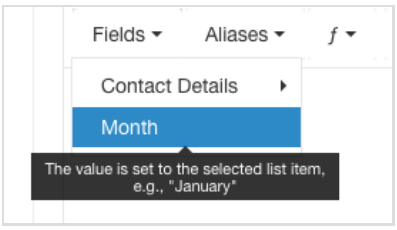


Additionally, the Formula Editor can be found by clicking the ***f* icon** next to the redirect link or from within the connectors (where formulas are applicable).

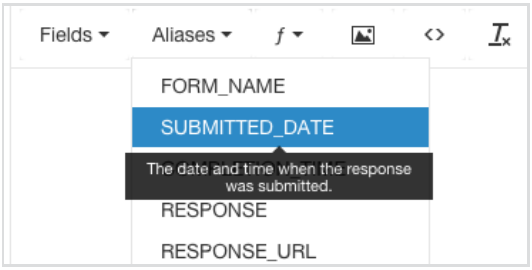


Using the Formula Editor

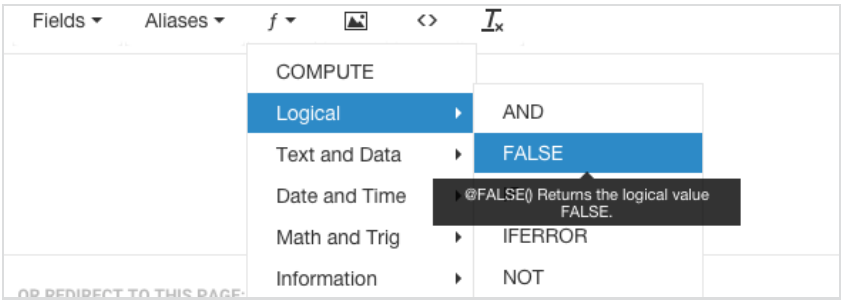
The FormAssembly Formula Editor will automatically be populated with the values and aliases from your form. To find a field alias from your form, simply click on the **fields** dropdown menu, and you will see your available options. If you hold your mouse over an option, a pop-up will appear telling you all of the available values for that particular field.



To insert one of our generic aliases (those that are available on all forms, regardless of what fields you have), click on the **Aliases** dropdown menu. Here again, if you hover over a particular option, a pop-up will appear, telling you more information.



Finally, to find functions to use in your form, you can click on the f-value dropdown menu for all of the formula building logic options. You can build incredibly complex formulas within our formula editor, and to get a better idea of what is possible, you can explore the Functions section below for a complete explanation of each specific function. You can also find specific examples of formulas at the bottom of this article.



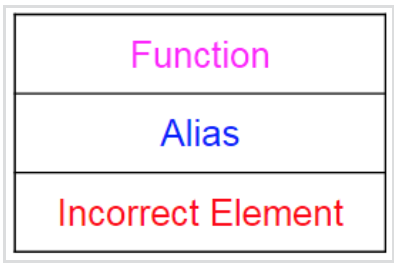
Additional examples can be found on our Formulas for Common Usage page.

Syntax Highlighter and Checker

In the Formula Editor, you will find that certain functions and aliases will be highlighted in different colors. Additionally, if you create a formula with an error in it, a "SyntaxError" will pop up at the bottom of the window. These two tools are designed to help you build formulas more efficiently.

The Syntax Highlighter

To help you visually distinguish the functions from the aliases in your formula, the syntax highlighter will automatically color-code the elements. Depending on the element, you can expect the following colors:

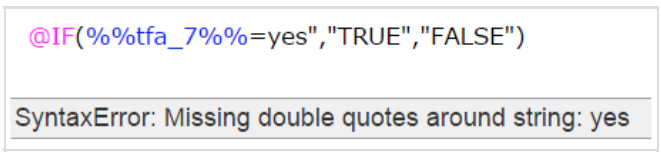


If your element is red, it might be due to an incorrectly capitalized function or a misspelled alias. You should check for SyntaxErrors at the bottom of the Formula Editor to help resolve any issues.

The Syntax Checker

At the bottom of the Formula Editor, you will find that the syntax checker pops up with various errors while you are creating your formulas. The syntax checker is designed to:

1. check your alias names to make sure that they are correctly formatted,
2. check your function names for correct spelling and capitalization,
3. check the number of function arguments, and
4. check for cases where your formula may result in a nonsensical return.



Use the suggestions of the syntax checker and the sample formulas below as a starting point to build your formulas.

Example - Redirect Based On a Field Value

Here's the explanation for each part:

@IF	The logical function used to test the value of a field. Note that the `@IF` statements are nested to perform the equivalent of a if/else/otherwise logical construct.
%%tfa_field%%	The alias of the field you need to test. The actual name will be different for your form. To find the correct alias, click on the link at the bottom of the Notifications tab.
"Page 1"	The expected value. If you are testing a multiple-choice field, you must enter the choice value. To find the choice values for an alias, click on the link at the bottom of the Notifications tab.
"http://your.site.tld/page_1.html"	This is the page where the user will be redirected if `"Page 1"` was selected.

Formulas and Checkbox Values

Checkbox fields are handled a bit differently, as each choice must be checked separately.

☐Blue

☐Red

Assuming the choice aliases are `tfa_blue` and `tfa_red`, you can use this formula:

```
@IF(%%tfa_blue%%="Blue", "You've selected blue", "");
@IF(%%tfa_red%%="Red", "You've selected red", "");
```

Auto-Responder Email Template

Finally, here's an example that could be used in an auto-responder email template:

```
Dear @IF(%%tfa_salutation%%="Mr.","Sir","Madam")
```

`@IF(`*condition*`,` *statement_if_true*`,` *statement_if_false*`)` is a **function**. It will be interpreted by the engine and replaced with the result of the evaluation.

`%%tfa_salutation%%` is the **alias** for the Salutation field in the web form (here, a drop-down menu with just `Mr.` and `Ms.`).

In plain English, this tests the submitted value for the Salutation field. If the value is `Mr.`, the email will include `Dear Sir`. Otherwise, the email will include `Dear Madam`.

1. Go to **My Forms** and click on the form you need to configure. This opens the form properties panel on the right-hand side.
2. Click on the **Notifications** tab.
3. In the **redirect to this page** field, instead of typing in the web address (URL) of the page, enter a formula. The syntax should look like this:

```
@IF(%%tfa_field%%="Page 1", "http://your.site.tld/page_1.html",@IF(%%tfa_field%%="
Page 2", "http://your.site.tld/page_2.html",
@IF(%%tfa_field%%="Page 3", "http://your.site.tld/page_3.html",
"http://your.site.tld/default_page.html"
)))
```