PVB CONTACT FORM 7 CALCULATOR PRO 1.4.0 – DOCUMENTATION

CONTENTS

1. INSTALLATION

Contact Form 7 version 5.0 or later is required.

1. Download the *pvb-contact-form-7-calculator.zip* file to your computer.
2. Unzip the file.
3. Upload the *pvb-contact-form-7-calculator* directory to your */wp-content/plugins/* directory.

2. FORM TAGS

The plugin adds support for the following special tags that you can add to any CF7 form:

2.1. CALCULATED VALUE

A calculated value is a read-only input element where a value is calculated according to a predefined formula.

In the free version, you can have one calculated value per form. In the Pro version, you can have as many as you need.

2.1.1. EXAMPLE

```
[calculation total min:0 max:1000 precision:2 "item_price * quantity"]
```

2.1.2. FORMULA

The calculation formula is a mathematical expression that may contain the following elements:

- **Number literals**;
- **Values taken from other fields** – simply use the field name in the equation. Non-numerical values are evaluated as 0;
- **Parentheses** – for grouping operations, e.g. (a+b)*c
- **Operators:**
  - addition (a + b)
  - subtraction (a - b)
  - multiplication (a * b)
  - division (a / b)
  - exponentiation (a ^ b)
  - modulo (a % b)
- **Functions:**
  - Square root: **sqrt**(x)
  - Logarithm: **log**(number, base)
  - Extract day number from date field: **fn_day**(x)
  - Extract month number from date field: **fn_month**(x)
  - Extract year from date field: **fn_year**(x)
  - Extract day of the year from date field: **fn_day_of_year**(x)
  - Extract weekday number from date field: **fn_weekday**(x) – Monday is 1, Sunday is 7
  - Extract number of business days between two date fields: **fn_business_days**(x, y)
  - Convert between two currencies *(Pro version only)*: **fn_currency_convert**(USD_EUR, x)
  - Custom PHP code *(Pro version only)*: **fn_php**(`return tan(deg2rad($_POST{'degrees'}));`) – see below for details
  - Spreadsheet cell reference *(Pro version only)*: **fn_spreadsheet**(http://www.example.com/pricelist.xls, 1, B, 10) – see below for details
  - Road distance *(Pro version only)*: **fn_distance**(address1_formatted, address2_formatted, km) – see below for details
  - Travel time *(Pro version only)*: **fn_travel_time**(address1_formatted, address2_formatted) – see below for details

2.1.2.1. RUNNING PHP CODE INSIDE THE FORMULA *(PRO VERSION ONLY)*

To run PHP code inside the formula, enclose it within a **fn_php**(…) call.

NOTES ON RUNNING PHP CODE:

- Running PHP code is **disabled by default** since it is a significant security risk. You can enable it from the plugin options page, but only if you are 100% sure you know what you are doing!
- Do *not* add any PHP opening/closing tags (*<?php … ?>*).
- Use a **return** statement to set the value that will be used in the calculation formula.
- You can escape double-quotes in your PHP code using the **&quot;** entity.
- You can use the **$_POST** global to access form data.
- Since Contact Form 7 uses square brackets - "[" and "]" - to delimit form tags, square brackets in your PHP code will not be handled properly. If you need to access array elements, **please use curly braces**. In PHP, both square brackets and curly braces can be used interchangeably for accessing array elements (e.g. $array[42] and $array{42} will both do the same thing).

2.1.2.2. REFERENCING AN EXTERNAL SPREADSHEET INSIDE THE FORMULA *(PRO VERSION ONLY)*

You can use the **fn_spreadsheet**(…) function inside your formula to reference a cell in a spreadsheet.

The **fn_spreadsheet** function is **disabled by default**. You can enable it from the plugin options page.

The function takes four parameters:

1. URL or full path to a local file where the spreadsheet document is located;
2. Number of the worksheet in the document (1, 2, 3…)
3. Column of the cell (A, B, C…)
4. Row of the cell (1, 2, 3…)

Supported formats are:

- Google Sheets
- Microsoft Excel 2007+ (XLSX)
- Microsoft Excel 97-2003 (XLS)
- Comma-separated values (CSV)

2.1.2.3. CALCULATING TRAVEL DISTANCE AND TIME *(PRO VERSION ONLY)*

If you have set a Google services API in the plugin options, you can use this function to calculate distance using the Google Maps API.

The **fn_distance**(…) function will take three parameters:

1. Starting point address. This is anything that can be parsed by Google Maps. If you use the input from an address field, you can use the formatted address by appending **_formatted** to the field name (e.g. if the address field name is called "office", use "office_formatted" in this parameter)
2. End point address. Same as above.
3. Distance units. Use "**miles**" for miles, "**km**" for kilometers, or "**m**" for meters (default is meters).

The **fn_travel_time**(…) function works similarly, but returns driving time. Again, it takes three parameters:

1. Starting point address.
2. End point address.
3. Time units. Use "**h**" for miles, or "**m**" for minutes (default is minutes).

2.1.3. OPTIONS

- **Minimum** – by specifying the "*min:X*" attribute, you can set a minimum value for the result of the calculation. For example, if you specify "*min:0*", when the result is negative, it will always be set to zero instead.
- **Maximum** – by specifying the "*max:Y*" attribute, you can set a maximum value for the result of the calculation. For example, if you specify "*max:100*", when the result exceeds 100, it will always be set to 100 instead.
- **Precision** – optionally, you can specify the "precision" attribute, which specifies the maximum number of digits after the decimal point. For example, with "precision:2", 0.786 will be rounded to 0.79. You can specify *"precision:0"* to get only integer results.

2.2. VARIABLE *(PRO VERSION ONLY)*

The variable field is similar to the calculation field, but it maps a list of text values (e.g. from a drop-down menu) to a list of number values.

2.2.1. EXAMPLE

```
[variable price "text:" "One 1" "Two 2" "Three 3" "Four 4" "default 0"]
```

2.2.2. VALUES

The first value indicates the source text field and should end with a colon.

Each text value is then mapped to a number, and defined in the CF7 tag as "(text) (space) (number)".

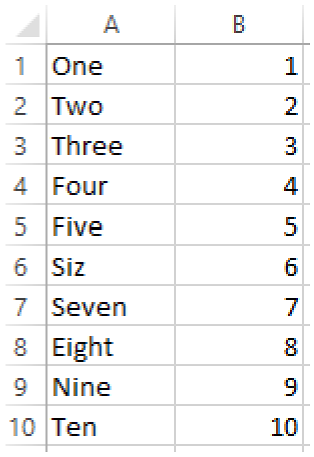
You can also specify a default value if none of the text values match. If not specified, it defaults to 0.

2.3. SPREADSHEET REFERENCE *(PRO VERSION ONLY)*

This field works similarly to the variable field above, but instead of a hard-coded list, it takes the values from an external spreadsheet.

2.3.1. EXAMPLE

```
[spreadsheet_reference price sheet:1 cells:A1-B10
"http://www.example.com/list.xls"]
```



	A	B
1	One	1
2	Two	2
3	Three	3
4	Four	4
5	Five	5
6	Siz	6
7	Seven	7
8	Eight	8
9	Nine	9
10	Ten	10

2.3.2. VALUES

- The “sheet” option indicates the number of the sheet in the spreadsheet document (1, 2, 3…)
- The “cells” option points to a two-column set of cells that contain the text-to-number mapping (text on the left, numbers on the right – see example above).
- The field value contains the URL or local path to a spreadsheet file (supported formats are: Google Sheets, Microsoft Excel, or comma-separated values).

2.4. SPREADSHEET DROPDOWN *(PRO VERSION ONLY)*

The spreadsheet dropdown will display a simple drop-down menu with a list of options taken from an external spreadsheet.

2.4.1. EXAMPLE

```
[spreadsheet_dropdown product sheet:1 cells:A1-A10
"http://www.example.com/list.xls"]
```

2.4.2. VALUES

- The “sheet” option indicates the number of the sheet in the spreadsheet document (1, 2, 3…)
- The “cells” option points to a column set of cells that contain the list of options.
- The field value contains the URL or local path to a spreadsheet file (supported formats are: Google Sheets, Microsoft Excel, or comma-separated values).

2.5. ADDRESS INPUT *(PRO VERSION ONLY)*

If you have entered a valid Google Maps API key on the plugin settings page, you can use this to generate an address input field with autocomplete features.

This tag will automatically generate three hidden fields alongside the main input field. You can use them in your calculations and email templates.

For example, If your main field name is “**xyz**”, the available additional form fields would be as follows:

- **xyz_lat** – location latitude (in decimal degrees)
- **xyz_lng** – location longitude (in decimal degrees)
- **xyz_formatted** – the full formatted address (including street, city, state, zip, etc.)

2.5.1. EXAMPLE

```
[address home]
[address work]
```

2.6. CALCULATE BUTTON

The calculate button will trigger an AJAX request that will fill in all the calculated values in the form.

2.6.1. EXAMPLE

```
[calculate_button "Click here to calculate"]
```

3. HIDING FIELDS

You can add the “cf7-hide” option to a [calculation] or [variable] field to make it hidden. For example:

```
[calculation total cf7-hide "item_price * quantity"]

[variable price cf7-hide "text:" "One 1" "Two 2" "Three 3" "Four 4"]
```

4. STYLING FIELDS

Each of the three calculation elements supported by this plugin have their own class names, respectively: ***wpcf7-calculation***, ***wpcf7-variable***, and ***wpcf7-calculate_button***. You can apply CSS styles to those classes, for example:

```
.wpcf7-calculation { border: red }
```

5. SCRIPTING

You can trigger calculation programmatically from your own Javascript code by calling the **`cf7Calculate`** function. It takes a single argument a jQuery object constructed from the respective `<form>` element. For example:

```
<script>
jQuery(document).ready(function($){
    var recalc = function() {
        // Find any forms on the current page and activate the calculations
        cf7Calculate(jQuery('form'));
    };

    // Trigger the recalc function when any text input changes
    jQuery('input[type=text],input[type=number],select').on('change',
recalc);

    // Trigger the recalc function when any checkboxes or radio buttons are
clicked
    jQuery('input[type=checkbox],input[type=radio]').on('click', recalc);
});
</script>
```

After the calculation AJAX call is made, the plugin will trigger the ***wpcf7calculate*** jQuery event. If the AJAX call fails, ***wpcf7calculatefail*** will be triggered. You can code your own handlers for these events, for example:

```
<script>
jQuery('form#calculator').on('wpcf7calculate', function() {
  alert('Your calculation is ready!');
}
</script>
```