Product Price Formula

extension for Magento2

User Guide

version 1.0

Contents

1. Introduction

The document is a User Guide for extension **Product Price Formula** created for Magento2 websites. It describes the extension functionality and provides some tips for a quick start.

The purpose of Product Price Formula is to provide more flexibility with the product price calculation. The website administrator can now use any custom mathematical formula to calculate the final product price. The input to formula is custom options the customer chooses on the form or the product attributes defined by the website administrator.

The price formula can be of any complexity, have math functions and constants, have conditional branching.

The extension will be useful for those who need a custom method of price calculation not supported by Magento, like:

- Price calculation based on the object size or dimensions
- Complex tier price calculation based on quantity and the custom options selected
- Single setup fees for a bulk purchase not dependent on the quantity ordered
- Additional fees that depend on multiple custom options or product attributes together

2. Installation

2.1. System Requirements

The extension requires Magento 2.x

2.2. Installation

The extension is provided as a .zip archive with the source code and the installation instructions.

Unpack the source code into **/app/code/Itoris/ProductPriceFormula/** folder on your server. And run the following commands in the SSH console:

```
php bin/magento module:enable --clear-static-content Itoris_ProductPriceFormula
php bin/magento setup:upgrade
```

Then log into Magento backend and flush cache in **System -> Cache Management -> Flush Magento Cache**

If you experience any issues with the extension installation please contact us here - https://www.itoris.com/contact-us.html

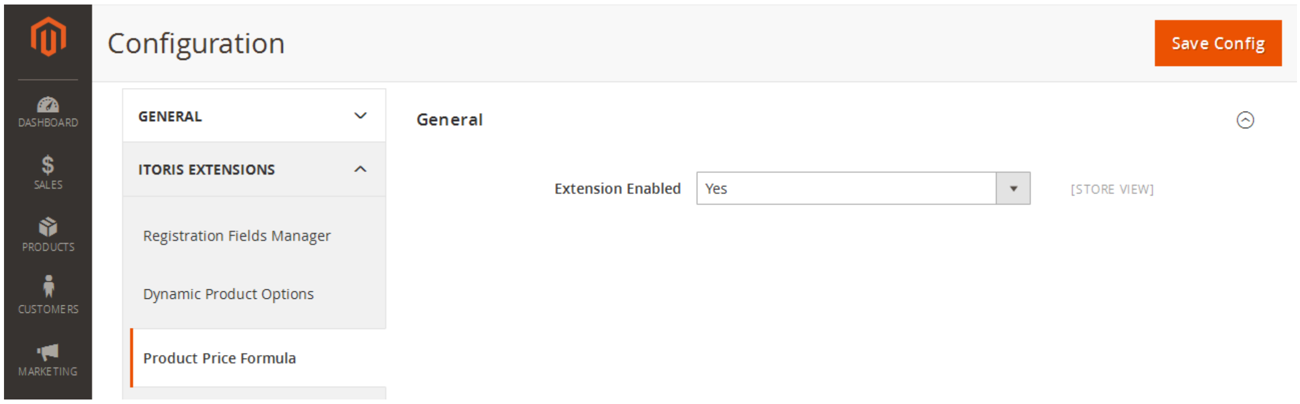
2.3. License

The extension has full open source code. One license/purchase can be used on a single production Magento2 website and its development instances. The extension can be customized for the license owner needs. Redistribution of the extension or its parts is not allowed. Please read more details here - https://www.itoris.com/magento-extensions-license.html

3. How to Use

3.1. Extension Activation

The extension is enabled by default. But if you need to temporarily disable it please do it following:
STORES -> Configuration -> ITORIS EXTENSIONS -> Product Price Formula -> Extension Enabled = Yes/No

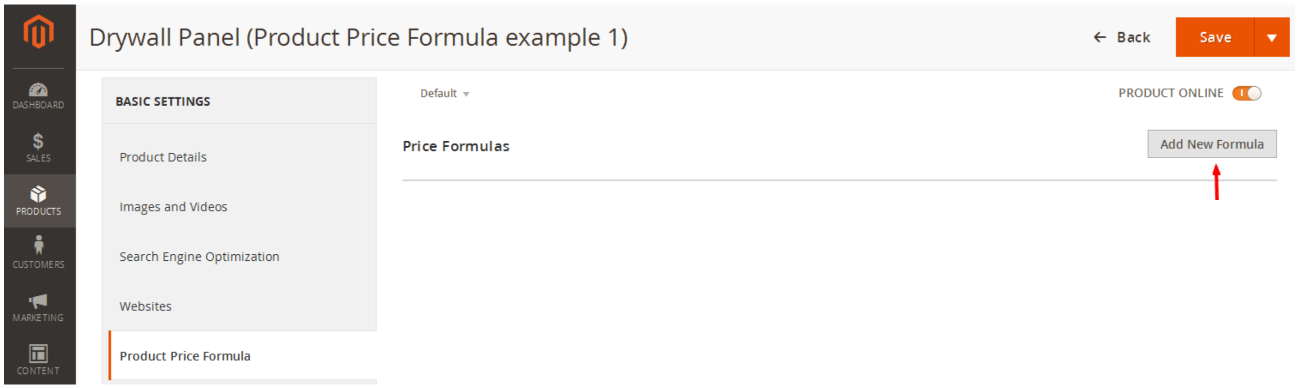


Extension Enabled: Yes [STORE VIEW]

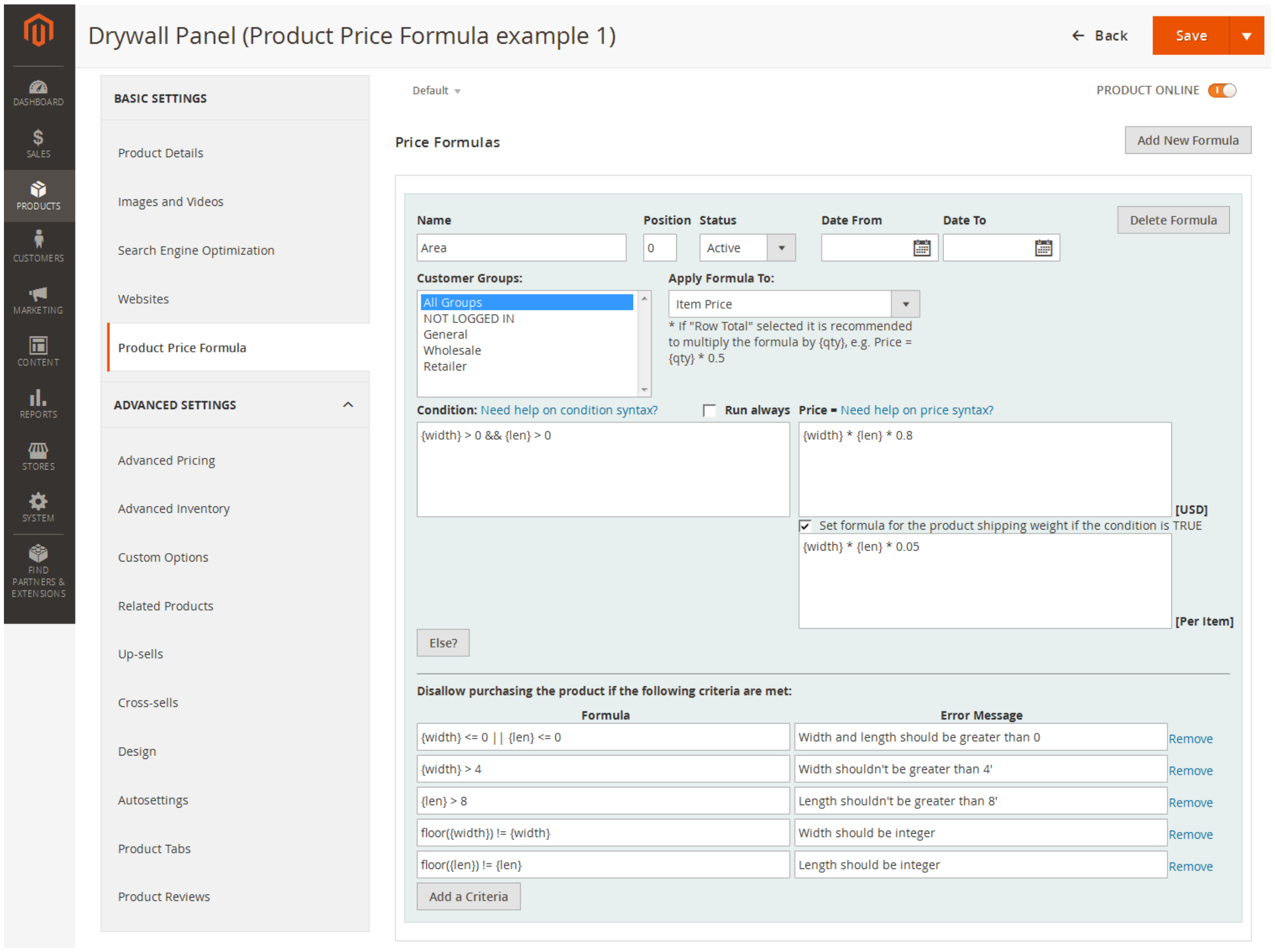
3.2. Create New Formula

Formulas can be created and edited for each individually following:
PRODUCTS -> Catalog -> {product} -> Product Price Formula

Click the "Add New Formula" button to create a new formula:



3.3. Formula Editor Overview



Drywall Panel (Product Price Formula example 1)

PRODUCT ONLINE

Price Formulas

Name	Position	Status	Date From	Date To
Area	0	Active		

Customer Groups: All Groups, NOT LOGGED IN, General, Wholesale, Retailer

Apply Formula To: Item Price

* If "Row Total" selected it is recommended to multiply the formula by {qty}, e.g. Price = {qty} * 0.5

Condition: Need help on condition syntax?

```
{width} > 0 && {len} > 0
```

Run always

Price = Need help on price syntax?

```
{width} * {len} * 0.8
```

[USD]

☑ Set formula for the product shipping weight if the condition is TRUE

```
{width} * {len} * 0.05
```

[Per Item]

Disallow purchasing the product if the following criteria are met:

Formula	Error Message
{width} <= 0 || {len} <= 0	Width and length should be greater than 0
{width} > 4	Width shouldn't be greater than 4'
{len} > 8	Length shouldn't be greater than 8'
floor({width}) != {width}	Width should be integer
floor({len}) != {len}	Length should be integer

The form has the following elements:

Add New Formula - *button*. Product can have multiple accumulative formulas executed by chain. Click this button if you want to add another formula to the product. Make sure to set the correct order in field Position.

Delete Formula - *button*. If you no longer need a formula you can delete it using this button. You should save the product to finish deletion.

Name - *text field*. Enter the formula title here. Not visible on frontend.

Position - *text field*. If you have multiple formulas in the same product you should set the correct order of execution.

Status - *dropdown*. Formula is enabled by default. You may temporarily disable it using this dropdown.

Date From - Date To - *range selector with calendar*. If you plan to have a limited promotion specify the "from-to" dates when the formula should be active. You can set both, either, or none dates.

Customer Group - *multiple list box*. Choose groups the formula is active for. All Groups is set by default.

Apply Formula To - *dropdown*. This dropdown allows setting what the calculated value should be applied to. If "**Item Price**" selected the formula result will be applied to the item price. The row total will be calculated as the item price multiplied by the quantity. If "**Row Total**" selected the formula result will be applied to the row total not depending on the quantity selected. The item price will be calculated as the division of the calculated row total and the quantity.

Show Product Price on Frontend as - *dropdown*. Visible if "Row Total" is selected in the previous dropdown. If "**Default**" selected the customer will see the "price per item" on the product view on frontend. If "**Multiplied by the QTY**" selected the customer will see the row total value. It's useful when you calculate the package price based on multiple conditions, to show the final price to the customer before adding product to cart.

Condition - *text area*. Condition defines when formula should be executed, for example:
`{width} > 0 && {len} > 0` - making sure the width and the length entered is positive
`{print_label} && {txt.length} >= 15 || {print_default}` - if customer has selected to print label and (&&) entered 15 or more characters of text, or (||) the default label selected.

Need help on condition syntax? - *link*. Shows tips on the condition syntax.

Run always - *checkbox*. This checkbox will disable the Condition textarea, meaning the formula will always run without conditions. Also, you will not be able to fork the condition using button "Else".

Price= - *text area*. Enter your formula here. It should result to a float value. For example:
`{width} * {len} * 0.8` - calculate the area of a rectangle and multiply it by the rate
`PI * sqrt({radius}) * 0.8` - calculate the area of a circle and multiply it by the rate

Need help on price syntax? - *link*. Shows tips on the price syntax.

Set formula for the product shipping weight if the condition is TRUE - *checkbox*. Check this checkbox if you want to override the product shipping weight. If checked a text area appears below it where you enter your custom formula for the product weight.

Else? - button. Using this button you can fork your condition and add another formula. New set of fields will appear for Condition, Formula, and Weight. For example, custom tier price for quantities up to 10, 20, and 30 and if length is 20 or greater:

```
if ({qty} < 10 && {len} >= 20) Price = 20;
else if ({qty} < 20 && {len} >= 20) Price = 18;
else if ({qty} < 30 && {len} >= 20) Price = 16;
etc.
```

Disallow purchasing the product if the following criteria are met: Formula and error message - *text boxes*. Your custom validation criteria. Enter the formula and the error message. You can create multiple validation messages. Examples:

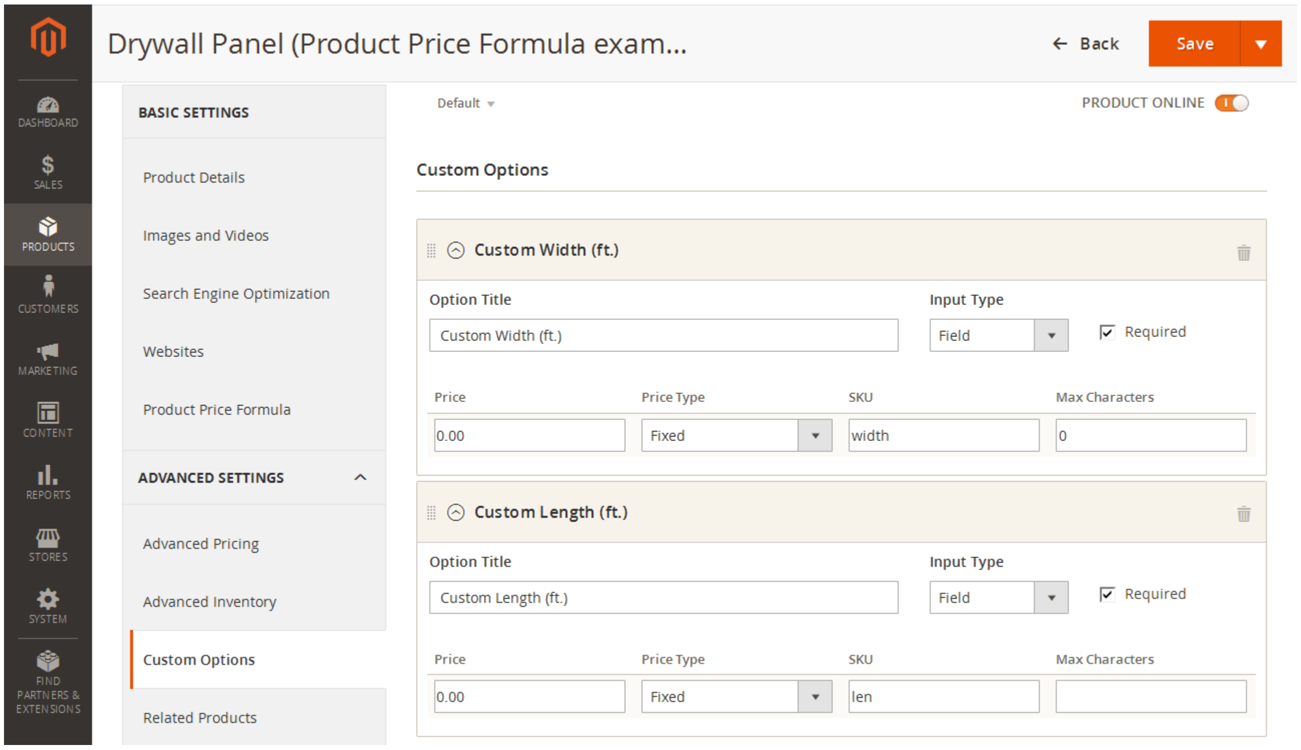
```
if ({width} <= 0 || {len} <= 0) Error = "Width and length should be greater than 0"
if (floor({width}) != {width}) Error = "Width should be integer"
etc.
```

3.4. Input

You can pass data into your formula via dynamic variables. Variables are enclosed into curly braces: {variable}. All variables currently supported by the extension are listed in the Appendix A.

3.4.1. Custom Options

You can pass the data entered by customer via the custom option. To use custom option in formula the option should have the unique SKU. In the screenshot below we have 2 options - Width and Length, SKUs are "width" and "len" accordingly. Our variables will be {width} and {len}:



Drywall Panel (Product Price Formula exam...

Custom Options

Custom Width (ft.)

Option Title	Input Type	
Custom Width (ft.)	Field	☑ Required

Price	Price Type	SKU	Max Characters
0.00	Fixed	width	0

Custom Length (ft.)

Option Title	Input Type	
Custom Length (ft.)	Field	☑ Required

Price	Price Type	SKU	Max Characters
0.00	Fixed	len	

If custom option is a Field or Textarea the dynamic variable returns a string. If string is numeric it is converted into the number automatically.

If custom option is a Dropdown, Checkbox, or Radio, i.e. has sub-options, the variables will return the sub-option title. You can use such variables as Boolean variables, i.e.:

```
if ({red} || {blue}) Price = 10;
else if ({green} && {qty} > 20) Price = 8;
```

If you have a Dropdown with numeric sub-options, you can use the values in the formula as well:

```
if ({size10} || {size20} || {size30}) Price = {size10} * 0.5 + {size20} * 0.4 + {size30} * 0.3;
```

If variable is not set it returns 0/false.

You can get the option price using variable {sku.price}, for example:

```
if ({leather}) Price = {leather.price};
if ({cloth}) Price = {cloth.price};
```

If your price relies on the length of text entered you can use variable {sku.length}, for example:

```
if ({custom_text.length} > 0) Price = {custom_text.length} * 0.02;
```

If you have extension Dynamic Product Options installed that supports quantities for options you can use variable {sku.qty}, for example:

```
if ({ram}) Price = 500 + {ram.qty} * 20;
```

3.4.2. Product Attributes

In addition to custom options you can use product attributes in your formula like {attribute_code}. Find the attribute code in backend following **STORES -> Attributes -> Product**

3.4.3. Configurable Options

If you have a configurable product you can get the ID of selected sub-product via variable {configurable_pid}. For example:

```
if ({configurable_pid} == 961) Price = 299;
else if ({configurable_pid} == 962) Price = 289;
else if ({configurable_pid} == 963) Price = 319;
```

3.4.4. Other variables

Selected quantity: {qty}

Price after product options selected: {configured_price}

Price before options selected: {initial_price}

Price after all calculations applied: {price}

Special price configured in the product: {special_price}

3.5. Accumulative Price

If you have a long formula it is possible to set up a few smaller ones and make the price accumulative. Create multiple formulas by clicking "Add New Formula" button. Set the correct order in field Position. The accumulative price is summed via variable {price}. Each next formula has {price} calculated by the previous formula. Example:

[Formula 1, position 1] - Material price

```
if ({cloth} && {width} > 0 && {len} > 0) Price = {width} * {len} * 5;
else if ({leather} && {width} > 0 && {len} > 0) Price = {width} * {len} * 10;
```

[Formula 2, position 2] - if chair selected in addition

```
if ({chair}) Price = {price} + 50 * {chair.qty}
```

[Formula 3, position 3] - discount for a bulk purchase

```
if ({qty} < 10) Price = {price} * 1;
else if ({qty} < 20) Price = {price} * 0.9;
else if ({qty} < 30) Price = {price} * 0.8;
else Price = {price} * 0.7;
```

3.6. Sub-Conditions

The extension allows having conditions directly in the formula using a special syntax. For example:

```
Price = {price} + ({size10} ? 24.99 : 0) + ({size20} ? 44.99 : 0) + ({size30} ? 64.99 : 0);
```

Here, it adds the custom option price to the final price depending on the dropdown option selected.

3.7. Mathematical functions

You can use math functions like sin, cos, tan, etc. in your formulas or conditions. For example:

```
if ({side1} > 0 && {size2} > 0 && {angle} > 10) Price=0.5 *{size1} *{size2} *sin({angle}) *{rate}
```

Please find the list of all supported math functions in Appendix A.

Appendix A

Use the following condition and math operators:

Operator	Explanation	Example
()	Sub condition	({sku1} + {sku2}) / PI
+	Addition	{sku1} + 10
-	Subtraction	{sku1} - 10
*	Multiplication	2 * PI * {sku_radius}
/	Division	{sku1} / 1.5

Math functions:

Function	Explanation
abs(x)	Returns the absolute value of x
acos(x)	Returns the arccosine of x, in radians
asin(x)	Returns the arcsine of x, in radians
atan(x)	Returns the arctangent of x as a numeric value between -PI/2 and PI/2 radians
atan2(y,x)	Returns the arctangent of the quotient of its arguments
ceil(x)	Returns x, rounded upwards to the nearest integer
cos(x)	Returns the cosine of x (x is in radians)
exp(x)	Returns the value of Ex
floor(x)	Returns x, rounded downwards to the nearest integer
log(x)	Returns the natural logarithm (base E) of x
max(x,y,z,...,n)	Returns the number with the highest value
min(x,y,z,...,n)	Returns the number with the lowest value
pow(x,y)	Returns the value of x to the power of y
random()	Returns a random number between 0 and 1
round(x)	Rounds x to the nearest integer
sin(x)	Returns the sine of x (x is in radians)
sqrt(x)	Returns the square root of x
tan(x)	Returns the tangent of an angle

Constants:

Constant	Explanation
E	Returns Euler's number (approx. 2.718)
LN2	Returns the natural logarithm of 2 (approx. 0.693)
LN10	Returns the natural logarithm of 10 (approx. 2.302)

LOG2E	Returns the base-2 logarithm of E (approx. 1.442)
LOG10E	Returns the base-10 logarithm of E (approx. 0.434)
PI	Returns PI (approx. 3.14)
SQRT1_2	Returns the square root of 1/2 (approx. 0.707)
SQRT2	Returns the square root of 2 (approx. 1.414)

Variables:

Variable	Explanation
{configured_price}	Price after product options selected
{initial_price}	Price before options selected
{price}	Price after all calculations applied
{special_price}	Special price configured in the product
{attribute_code}	Any product attribute name enclosed into {}
{option_sku}	Call any product option by its SKU enclosed into {}
{option_sku.qty}	The quantity of sub-option if Dynamic Product Options installed
{option_sku.price}	Get the price of option by sku
{option_sku.length}	Get the length of entered text
{configurable_pid}	Returns the ID of currently selected product within the configurable product
{qty}	Product quantity selected