

AMAZZING FILTER

Installation and configuration guide

Installation

Module is installed in a regular way. Simply upload your archive and click install. Don't forget to uninstall other layered navigation modules in order to avoid possible interference.

Indexation

Once module is installed, you should run indexation once. Indexation data is used by module to filter products quickly basing on selected parameters like features, attributes, tags, etc...

You can always check the total number of indexed products on **Indexation tab**. Keep in mind that only active and visible products are indexed.

Indexation data is updated automatically when products are saved/updated in a native way, for example when you update product sheet and click "Save" button. So, you don't have to re-index products manually every day.

If products are not updated natively (for example if rows are inserted directly in database), you might need to re-index them manually on **Indexation tab**. There you will find 2 buttons:

1. **INDEX MISSING PRODUCTS** - When this button is clicked, only products that haven't been indexed before will be indexed. Other products will not be processed.
2. **REINDEX ALL PRODUCTS** - When this button is clicked all products will be re-indexed, no matter if they have been indexed before or not.

Except manual re-indexation, you can set up a **Cron task**, that will update indexation data automatically at specified time intervals. If you use multistore, indexation URLs are different for each shop.

Dev Store ✓	shop2 ✓	test shop 1 ✓
Total indexed: 905	Total indexed: 905	Total indexed: 905
Missing in index: 0	Missing in index: 0	Missing in index: 0
Erase index	Erase index	Erase index
Cron indexation	Cron indexation	Cron indexation

Index missing products URL
https://cronurl&id_shop=1&action=index-missing

Index all products URL
https://cronurl&id_shop=1&action=index-all

You can use the following cron commands (Pay attention to quotes around indexation_url):

```
curl -L "indexation_url" or wget -O /dev/null -q "indexation_url"
```

For example:

```
curl -L "https://domain.com/module/amazingfilter/cron?token=xxx&id_shop=1&action=index-all"
```

NOTE: indexation is a resource consuming task, so it is not recommended to call cron tasks very often, like "every 5 minutes", or so. 1-2 times per day should be enough. If you need to call cron tasks more often, you can [contact us](#), and together we will try to find an optimal way of keeping indexation data up to date.

NOTE 2: When you add new shops in multistore system, or when you add bulk catalog price rules, it is recommended to reindex all data

Filter templates

Once module is installed, you will have a general filter template for all existing categories. If you don't select any categories in template settings, it will be applied to all available categories, including new created.

If you want to display a specific filter template on selected categories, you can create a new one for those categories only. Templates with selected categories have higher priority compared to templates without selected categories.

You can add/remove/update filter criteria and arrange them in required order by drag-n-dropping

The screenshot shows the 'Template for all category pages' configuration page. At the top, there's a title bar with a green checkmark and a 'Scroll Up' button. Below this is a 'Selected categories' dropdown menu set to 'All available (including new created)'. The main section is divided into two tabs: 'FILTERS' and 'ADDITIONAL SETTINGS', with the latter being highlighted by a red box and a '1' badge. Under the 'FILTERS' tab, there are three filter criteria: 'Subcategories of current page' (Structure: Folded, Type: Checkbox), 'Attribute Color' (Sort by: Name, Type: Radio button), and 'Standard parameter Price' (Type: Slider). Each criterion has a 'MINIMIZED' checkbox and icons for settings and deletion. A red arrow points to the 'Attribute Color' criterion. Below these is a dashed box containing an 'Attribute Size' criterion (Sort by: Name, Type: Text box) and a '+ ADD NEW' button, which is also highlighted by a red box. Another red arrow points to the '+ ADD NEW' button.

Next to FILTERS you will notice ADDITIONAL SETTINGS tab. On that tab you can define custom settings that will override values specified in General settings. For example you can set custom sorting only for current template, or you can set different number of products per page, etc...

Except category pages, you can also display filter on the following pages: *New products, Specials, Bestsellers, Search results, Products by manufacturer, Products by supplier and Main page.*

All of these pages have adjustable templates, that are activated automatically on module installation. If you don't want to display filters on some of those pages, you can simply deactivate the template by clicking the green check mark next to **Edit** button.

Module position and hook settings

Once module is installed, it is hooked to left column and is positioned at the top of it. You can easily change current hook and ordering on **Hook settings** tab.

If your theme doesn't use column hooks, or if you want to place your filter in some specific place that doesn't have predefined hooks, you can use the special hook: `displayAmazingFilter`. In order to display this hook, you can insert the following code anywhere in your tpl: `{hook h='displayAmazingFilter'}`

Adjusting module view

Initial module layout is seamlessly integrated with "classic" theme (Or "default-bootstrap" in PS 1.6).

The easiest way to add custom styles/scripts is to add "custom.css/js" to module override subfolder in your theme:

`/themes/your_theme/css/modules/amazingfilter/views/css/custom.css` (orange subfolder is for PS 1.6)

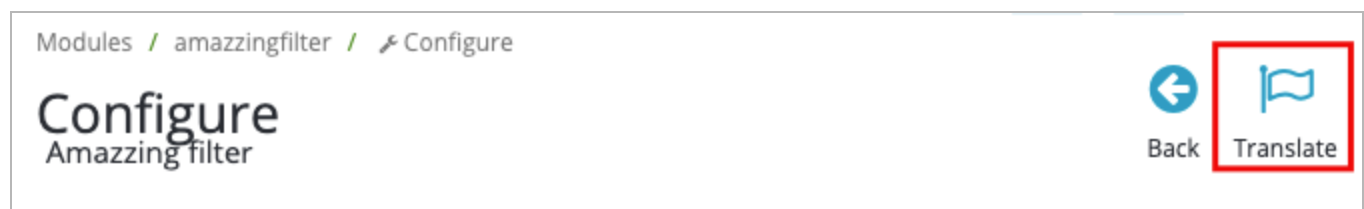
`/themes/your_theme/js/modules/amazingfilter/views/js/custom.js` (orange subfolder is for PS 1.6)

Other css/js/tpl files can be overridden in a regular way, same as you do with other modules. For example, if you want to customize filter block layout, you should add modified file here:

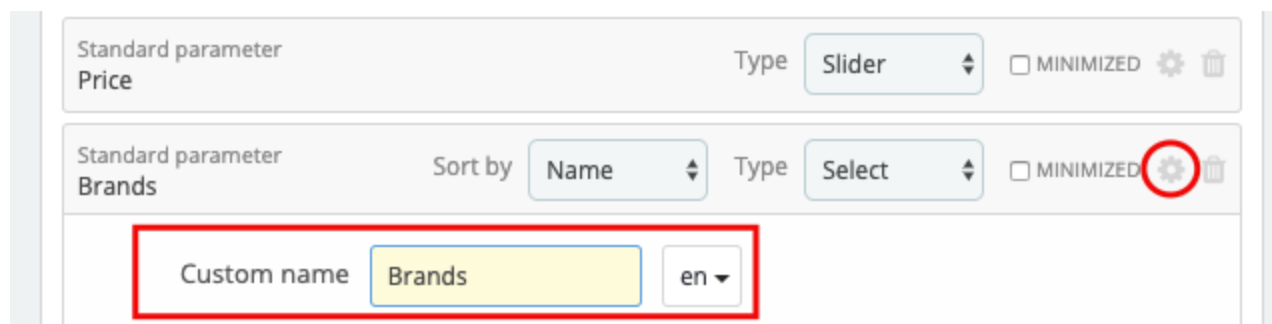
`/themes/your_theme/modules/amazingfilter/views/templates/hook/amazingfilter.tpl`

Translations

Module translations can be edited by clicking on standard **Translate** button in the top right corner on module configuration page:



Also, you can add custom multilingual subtitles for each filter criterion:



General settings

Most General Settings are self explanatory. Here is the description of some specific options:

Out of stock behaviour: You can exclude out-of-stock products from the filtered results, or move them to the end of the list, preserving current ordering. Or you can just ignore the out-of-stock status.

Count stock for combinations: If this option is enabled, stock is calculated based on selected attributes. For example, you have a product that is available in “red”, “blue” and “green”, but currently only “green” is in stock. If customer selects “red” or “blue”, this product will be treated like out-of-stock. So, if you chose to exclude out-of-stock products in previous option, this product will not be included in filtered result. Or, if you use “sorting by stock”, this product will be positioned after products that are in stock.

Note: this option requires additional processing time. If you have a powerful server, you won't see a significant time difference. However, it is recommended to use it only if you really need it.

Check combinations existence: This option is useful if you want to exclude non-existing combinations from filtering results. For example, you have a product that has only 2 combinations: “color-red; size-M” and “color-green;size-L”. You don't have a combination for “red-L”. So, if customer selects “red” and “L” in filtering criteria, this product will be excluded from filtered result.

Note: same as the option above, this one requires additional processing time.

Use merged values for attributes/features: This option is useful if you need to merge multiple options, having different name, but same meaning. For example shoe sizes *US-10*, *UK-9.5* and *EUR-43* can be merged in one value *EUR-43*. When this option is activated, a new tab appears in side panel, where you can configure merged values.

Layout classes/ids

Here you can specify your theme selectors, that are required for seamless integration with the module. In most cases these will work out of the box without extra modifications.

In some cases you might need to change classes for icons used in module. For example, some themes use “fa fa-xx” instead of “icon-xx”. You can feel free to play with selectors. Don't worry about saving the initial classes/ids, you can easily **RESET** them in one click.

ICON CLASSES USED IN FILTER

RESET

Filter icon

.

icon-filter

Remove one filter icon

.

icon-times

Remove all filters icon

.

icon-eraser

If your theme doesn't use any icon-classes, you can activate option **Load icon font**

Caching settings

Caching is used to minimize repetitive requests to database and complex calculations. You can activate caching for selected resources like **category options**, **attribute options**, **feature options** and **combinations availability**.

Here is how it works in general way: when any user opens a page, some data is fetched from database and processed in a regular way. After this it is saved in a cache-file. So, when the same data is requested next time by another user, there are no repeating requests to database, and no processing involved, data is fetched from cache-file “as is” and it is ready to use.

In current module version caching is updated every hour. So once every hour data is processed in real time and saved in cache. During the next hour data is fetched directly from the cache, without requests to database or complex calculations. When caching hour is over, data is processed in real time again, caching is updated and used during the next hour, and so on...

This approach can decrease page loading time and filtration time on stores with many attributes, combinations and features. But you should be careful if data in your store is updated very often, because obviously caching doesn't use real-time data, it uses data that was saved within the last hour.

Besides resetting every hour, caching data it is automatically erased on the following native actions:

Category options - automatically erased, when a category is added/updated/deleted



Attribute options - automatically erased, when an attribute is added/updated/deleted


Feature options - automatically erased, when a predefined feature value is added/updated/deleted

Combinations availability - automatically erased, when new combinations are added/deleted, or product quantities are updated, or an order is placed.

If you need to reset cached data for any reason, you can use the **Clear cache** button:

ACTIVATE CACHING FOR THE FOLLOWING RESOURCES:

 Caching is used to optimize page loading time. [More info](#) 



Category options	<input checked="" type="checkbox"/> YES <input type="checkbox"/> NO	Cached data size: 5.52kb last update: 2019-06-14 14:40:14
Attribute options	<input checked="" type="checkbox"/> YES <input type="checkbox"/> NO	Cached data size: 5.48kb last update: 2019-06-14 14:18:14

Customer filters

These are specific filtering options that customers can configure individually for themselves.

For example, customer wants to get only products that are size “L”. He goes to “customer filters” on his account page, selects “size-L” and clicks “save” button. After that, when he goes to any category, “L” filter is applied automatically if “size” filter is available on that page, so customer doesn't have to select the same criteria multiple times.

This functionality is not activated initially. If you want to activate it, select at least one criterion, that can be used as a customer filter and click “save”. It can be color, size, manufacturer, or any other attribute/feature.

TECHNICAL TIPS

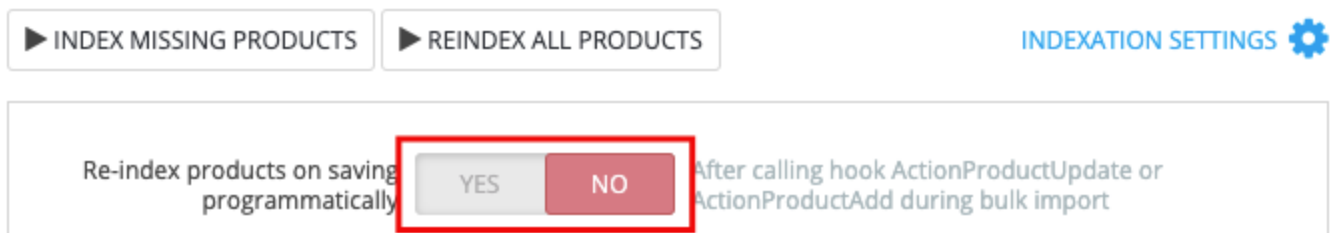
AUTOMATIC RE-INDEXATION ON MASS PRODUCT IMPORT/UPDATE:

If products are added/updated using mass importing tools, the easiest way to re-index them automatically is to make sure they are saved natively using one of the following methods: `$product_obj->save()` or `$product_obj->add()` or `$product_obj->update()`. When these methods are called, each product is re-indexed, so you don’t need extra cron tasks to keep indexation data up to date.

Many bulk importing tools already use those methods, so you don’t need to modify their code. If they don’t use such methods, you should add them. It will be useful not only for automatic re-indexation, but also possibly for other modules that perform specific actions on updating products, like *Menu*, *Search block*, *Tags block*, etc...

Automatic re-indexation triggered by native methods like `$product_obj->update()` will work smoothly if you use a regular shared hosting and import 50-100 products in one request. But indexation is a resource consuming task, so if you try to import/update too many of products in one request, each of them will be re-indexed one by one and the process can become slow. In such cases automatic re-indexation can be optimized in the following way:

- 1) First of all you should block auto-indexation on saving each product individually:



If you don’t want to change settings shown above, you can block auto-indexation by adding URL parameter `no_indexation=1`. If you call importing script from URL, you can add this parameter directly in URL. Or, if script is called another way, you can define it directly in code `$_GET['no_indexation'] = 1;`. This parameter can be used any time, if you need to block automatic re-indexation for any reason.

- 2) During the import process you collect array of `$product_ids`, that should be re-indexed. Note: in some cases products don’t require re-indexation after updating, for example, if only quantity was updated, or only description was updated, or product was processed, but nothing was changed about it. So you don’t have to spend server resources on indexation in such cases. There can be other cases when product doesn’t require re-indexation, so if you are not sure whether it should be re-indexed or not, you can [contact us](#) for support.

- 3) After import is complete you re-index `$product_ids` by calling hook `'actionIndexProduct'`.
This hook can be called anywhere, anytime to re-index selected products.

Here is a simplified code example:

```
$_GET['no_indexation'] = 1; // block automatic indexation on $obj->update()
$updated_ids = array(); // prepare an array of IDs that should be indexed
foreach ($products as $product) { // this is standard loop for most importing tools
    /**
     * bulk importing actions are processed here...
     */
    if (product_should_be_reindexed) {
        $updated_ids[] = $product['id_product'];
    }
}
Hook::exec('actionIndexProduct', array('product' => $updated_ids)); // re-index $updated_ids
```