Attribute Import-Export

Extension version: 1.3.0
Magento compatibility: CE 2.3, 2.4

Table of Contents

1. Introduction

Sometimes Magento webshops need to inherit a couple of special attributes (ex. product attributes, attribute sets, which are used on product pages) from another webshop. There are many reasons why such a task should be done: switching from another webshop engine; extending your current site; or just simply copying another similar shop. The migration of attributes can be a long and frustrating job, especially when it is done manually. You have to create each attribute manually, specify the settings, and maybe the most time consuming is to type in the options. Some webshops have thousands of options, which can be very hard to maintain.

The Code007 Attribute Import-Export module will save you precious hours. You will be able to export your chosen attributes from your old Magento site into a CSV file. This will contain the properties of the attributes, the attribute options and option swatches as well. All you have to do is to import this file into your Magento site.

If you are migrating from another webshop engine or just building your Magento store from scratch, you can create your own CSV file based on the provided Sample file (it can be found on the Import screen). The import process will take just a couple of seconds.

1.1 Migrate your Magento 1.x attributes into Magento 2

It is a very big step to migrate everything from Magento 1.x to 2.x. Many of the extension developers consider it tiresome and prefer to stick with Magento 1.x. Now there's an opportunity to ease the switch between the two as the Attribute Import-Export module has also been written for Magento 1.x (can be purchased separately) and the exported CSV is compatible with the Attribute Import-Export module for Magento 2. You will be able to migrate your attributes in just a couple of clicks.

1.2 Feature List

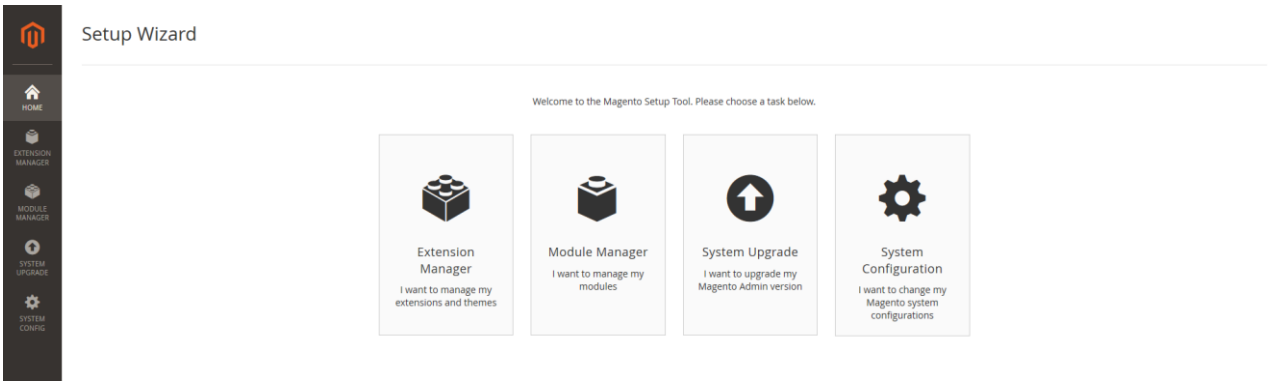
- **Export** attributes: you can choose from the available system and user defined attributes and save them into a CSV file
- While exporting attributes, the related attribute options and swatches are also exported (if available)
- **Import** attributes: import your attributes and attribute options from CSV files
- Import CSV format is compatible with the Code007 AttributeImportExport extension for Magento 1.x, so attributes can be migrated easily

2. Installation

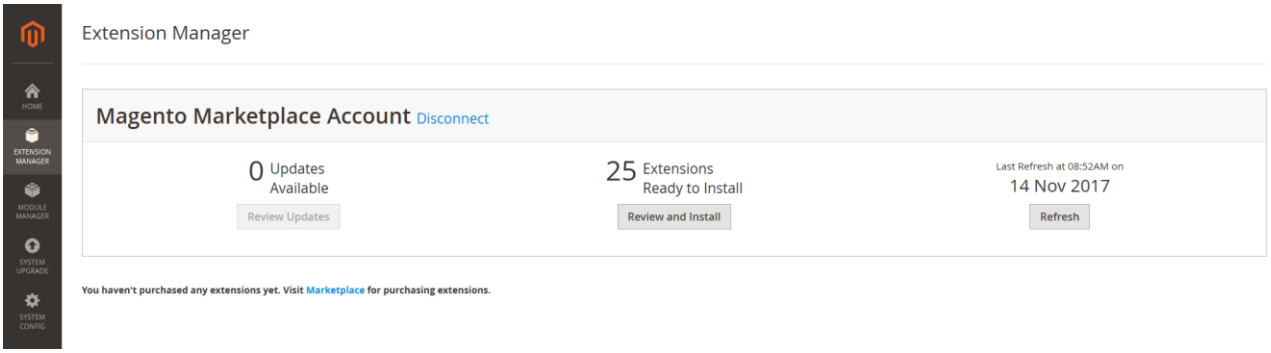
2.1 Web Setup Wizard

The easiest way to install the extension is by using the Web Setup Wizard from the Magento backend. To do this, follow the steps below:

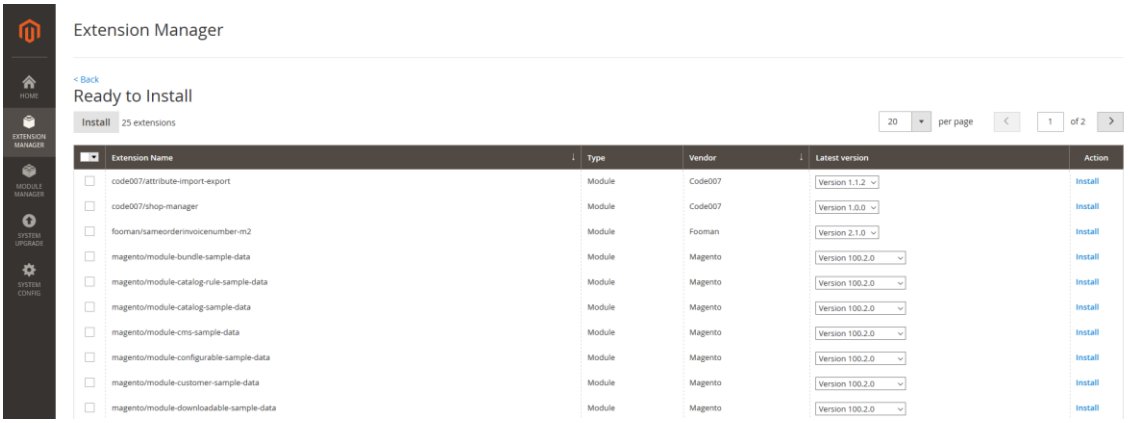
a. Open **System → Tools → Web Setup Wizard**



b. Click on **Extension Manager**
c. You will be asked to log in with your Magento Marketplace access keys. If you don't have any access keys, then you can create one in your Marketplace account in the **My Products → Access Keys** section
d. Press the **Review and Install** button



e. In the next screen you will get the list of available extensions attached to your Marketplace account. Choose **code007/attribute-import-export** and click on **Install**.



f. Follow the instructions step by step. First you will need to start the readiness check, then you can create a backup (it is optional) and the last step is the component installation.

2.2 Manual installation

To install the module manually, you will need its source code. Open Magento Marketplace, sign in and navigate to **My Account → My purchases**. If you have purchased the Attribute Import / Export module, you will be able to download its source code from this page.
Unpack the source code into the following directory:

<Magento root directory>/app/code/Code007/AttributeImportExport

If any of these directories are missing, create them. Please pay attention to the upper- and lowercase characters in the directory names. Open the terminal and navigate to the root directory of your Magento and run the following command:

php bin/magento setup:upgrade

The next step is to check the status of the module 'Code007_AttributeImportExport' with the following command:

php bin/magento module:status

If the module is disabled, run the following command to enable it:

php bin/magento module:enable Code007_AttributeImportExport

3. How to use

3.1 Importing attributes

Log in to the Magento backend and go to **System → Import**, then choose entity type „Attribute”. Choose an import behavior:

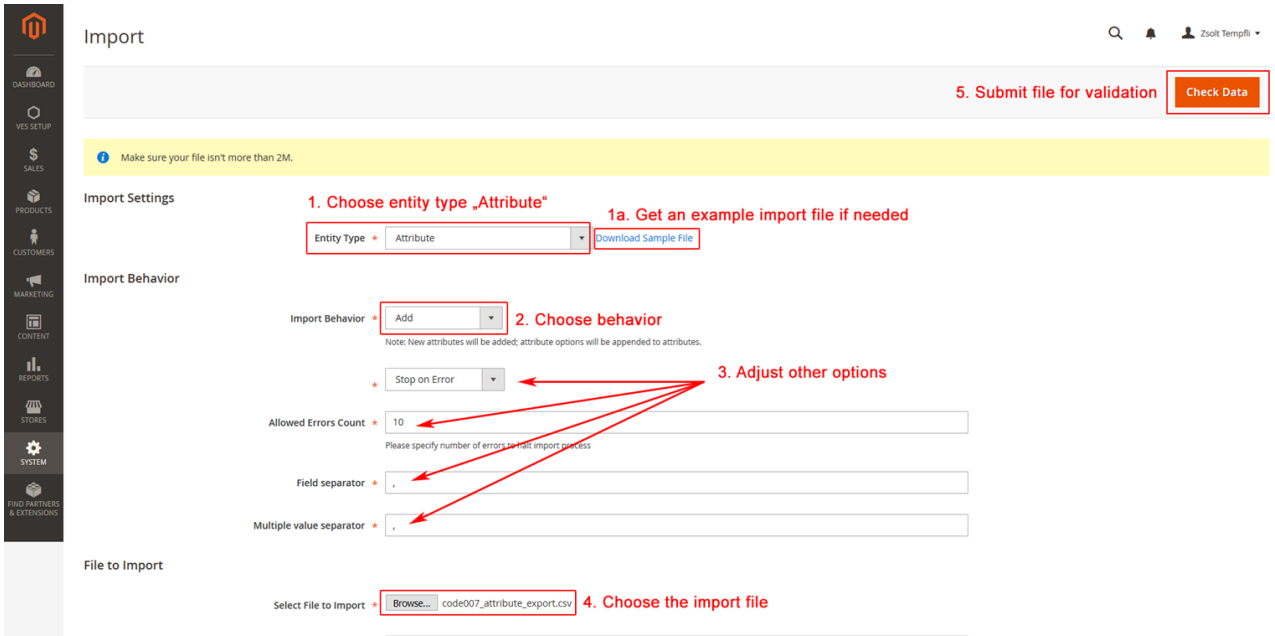
Behavior	Create new attributes	Create new attribute options / swatches	Update existing attributes	Update existing attribute options / swatches	Regenerate / create new ID	Preserve connection to entities	Remove attributes	Remove attribute options / swatches
Add	yes	yes	no	no	yes	yes	no	no
Update	yes	yes	yes	yes	no	yes	no	no
Replace	yes	yes	yes	no	yes	no	no	yes
Delete	no	no	no	no	no	no	yes	yes

- **Add**: when an attribute from the import file is not found, it will be created, otherwise it will be skipped; attribute options are always appended, so no overwriting will occur

- **Update**: when an attribute from the import file is not found, it will be created, otherwise it will be updated; attribute options are added if their option_id does not match with an option_id from the database, if matches found, then the options will be updated; the same applies to swatches

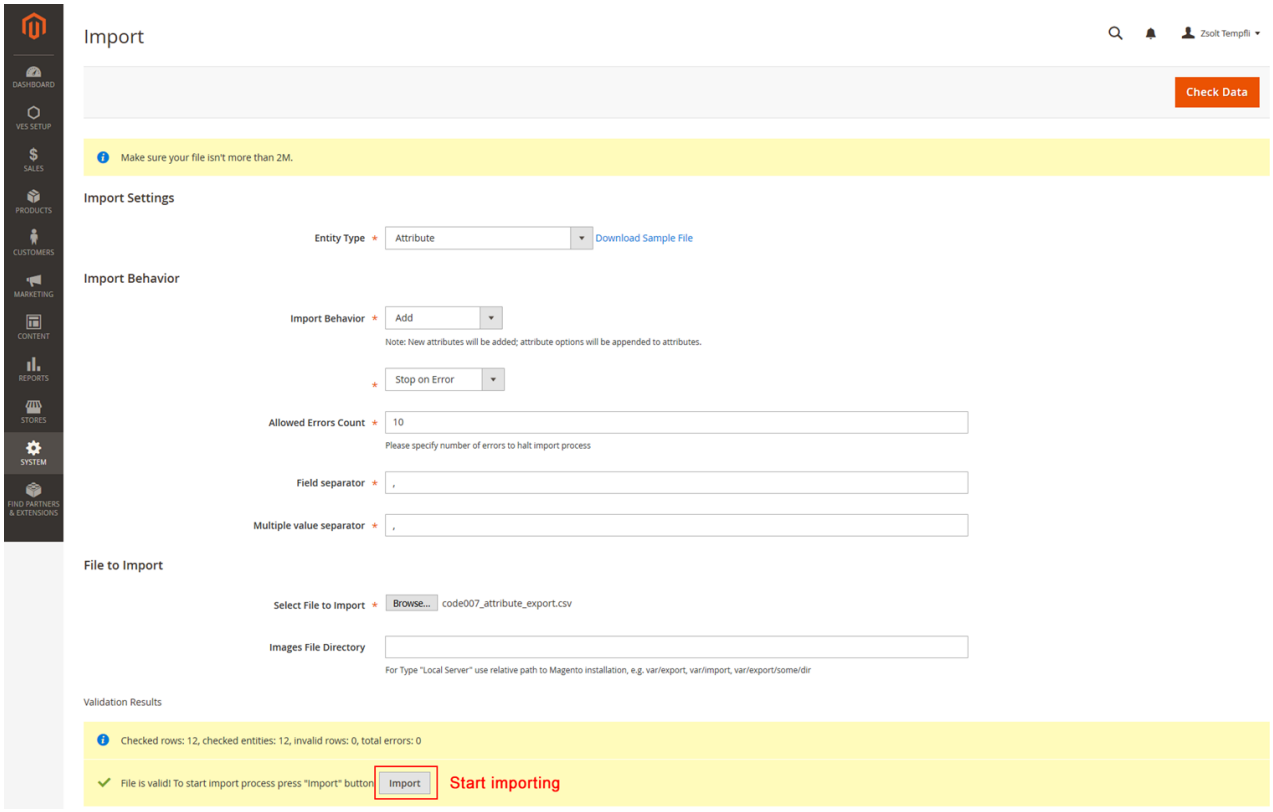
- **Replace**: when an attribute from the import file is not found, it will be created, otherwise it will be updated; existing attribute options will be removed from the database, then the new options will be created; the connections to the entities will be broken

- **Delete**: will delete all attributes which can be found both in the import file and in the webshop.

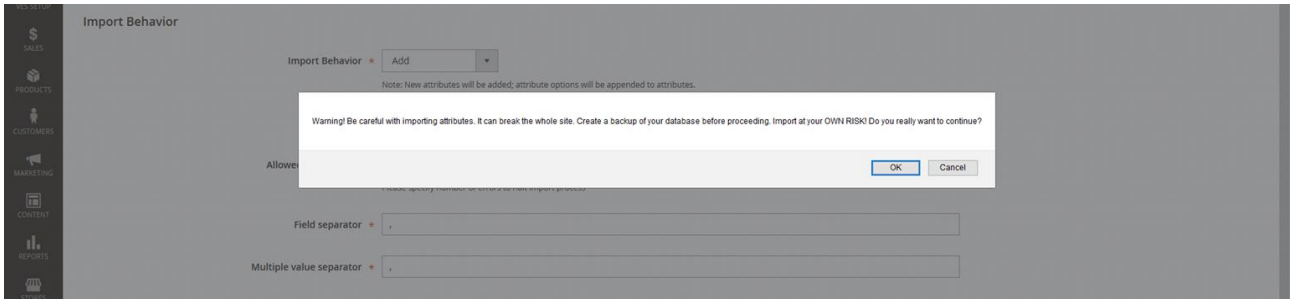


CSV field separator can be also adjusted together with other options, like the number of errors which will halt the import process. Images File Directory option is not used, so it can be left empty. Choose the file to import. If you are not sure how an import file should look like, then you can download a sample import file which contains a full example.

When your file is chosen, press the „Check Data” button. On the bottom of the screen a yellow band will appear with the validation results of the file. If the validation is passed, then you can start the import process by pressing the „Import” button.

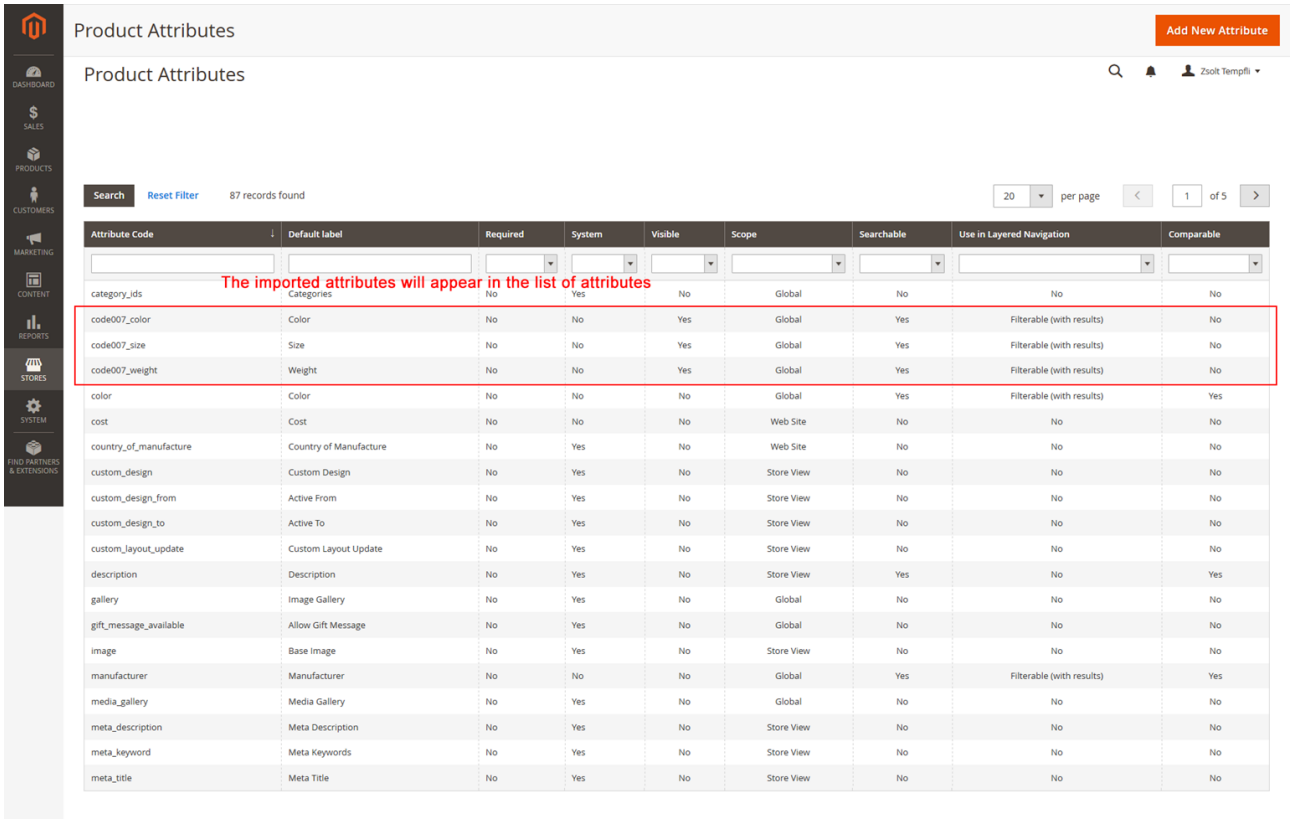


It is recommended that you create a backup of your database before importing, so you will be able to restore your webshop in the worst scenario. A final warning will remind you of this:



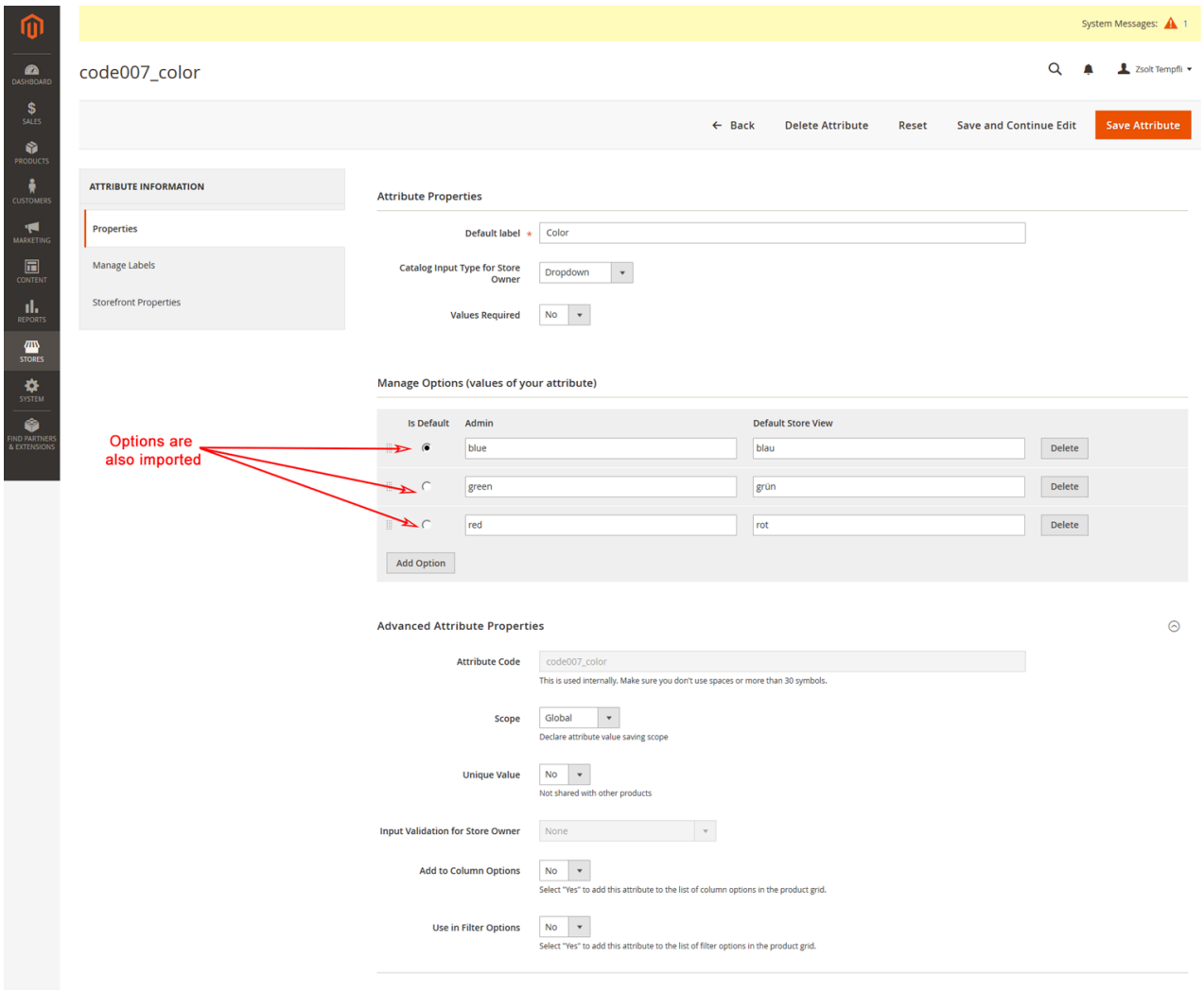
Now, if you proceed, the importing process will start. It will take just a couple of seconds to finish.

When the import process is done, you will get a notification whether it has succeeded or failed. You will be able to check the results yourself, either directly in your database client or in case of a product attribute import by going to **Stores → Attributes / Product**. The newly imported attributes will be found between the others. Remember that other attribute types (ex. Customer, Order, etc.) don't have a visual representation like Product Attributes have.



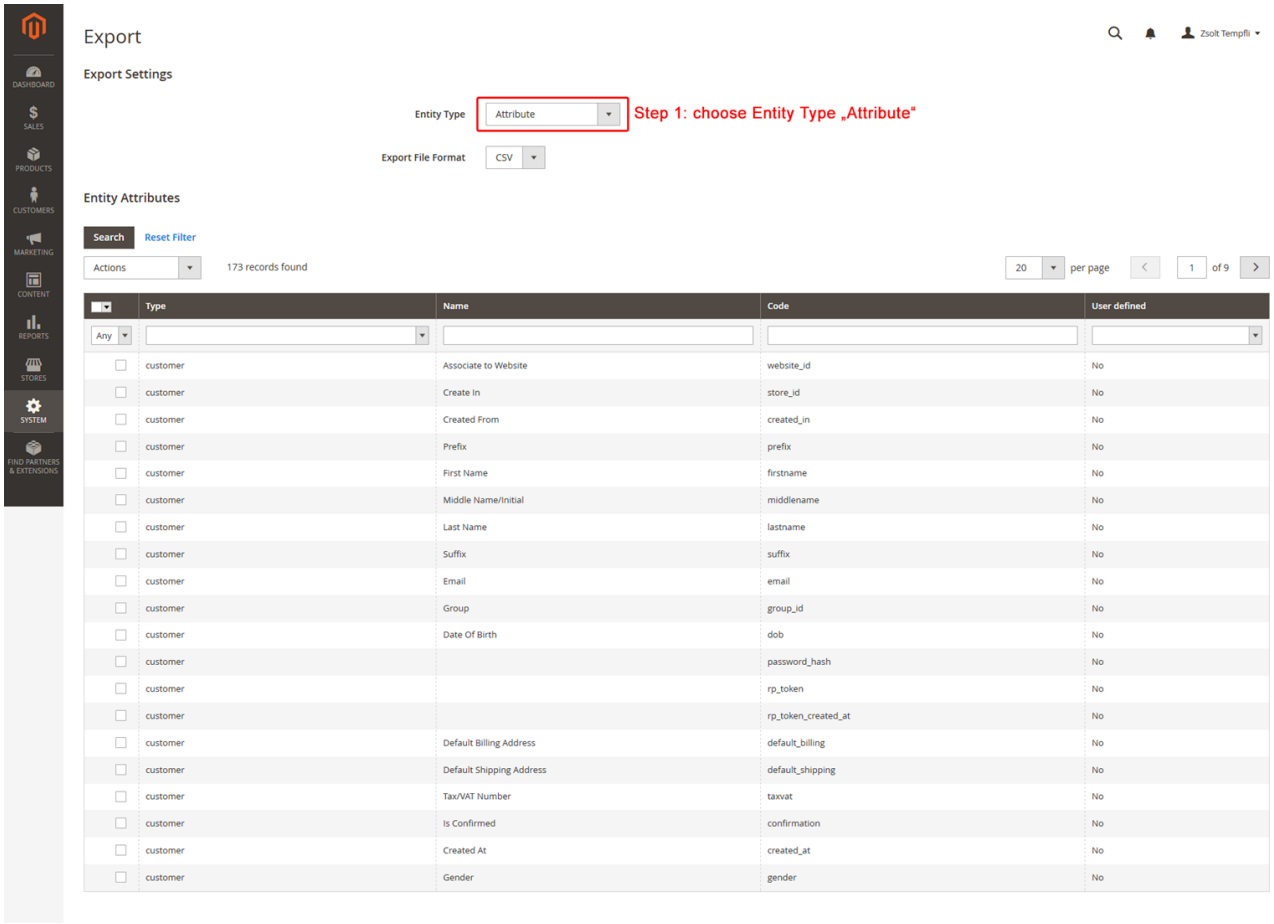
Attribute Code	Default label	Required	System	Visible	Scope	Searchable	Use in Layered Navigation	Comparable
category_ids	Categories	No	Yes	No	Global	No	No	No
code007_color	Color	No	No	Yes	Global	Yes	Filterable (with results)	No
code007_size	Size	No	No	Yes	Global	Yes	Filterable (with results)	No
code007_weight	Weight	No	No	Yes	Global	Yes	Filterable (with results)	No
color	Color	No	No	No	Global	Yes	Filterable (with results)	Yes
cost	Cost	No	No	No	Web Site	No	No	No
country_of_manufacture	Country of Manufacture	No	Yes	No	Web Site	No	No	No
custom_design	Custom Design	No	Yes	No	Store View	No	No	No
custom_design_from	Active From	No	Yes	No	Store View	No	No	No
custom_design_to	Active To	No	Yes	No	Store View	No	No	No
custom_layout_update	Custom Layout Update	No	Yes	No	Store View	No	No	No
description	Description	No	Yes	No	Store View	Yes	No	Yes
gallery	Image Gallery	No	Yes	No	Global	No	No	No
gift_message_available	Allow Gift Message	No	Yes	No	Global	No	No	No
image	Base Image	No	Yes	No	Store View	No	No	No
manufacturer	Manufacturer	No	No	No	Global	Yes	Filterable (with results)	Yes
media_gallery	Media Gallery	No	Yes	No	Global	No	No	No
meta_description	Meta Description	No	Yes	No	Store View	No	No	No
meta_keyword	Meta Keywords	No	Yes	No	Store View	No	No	No
meta_title	Meta Title	No	Yes	No	Store View	No	No	No

After the import you will see that the attribute options (if they are specified) will appear for each imported attributes:



3.2 Exporting attributes

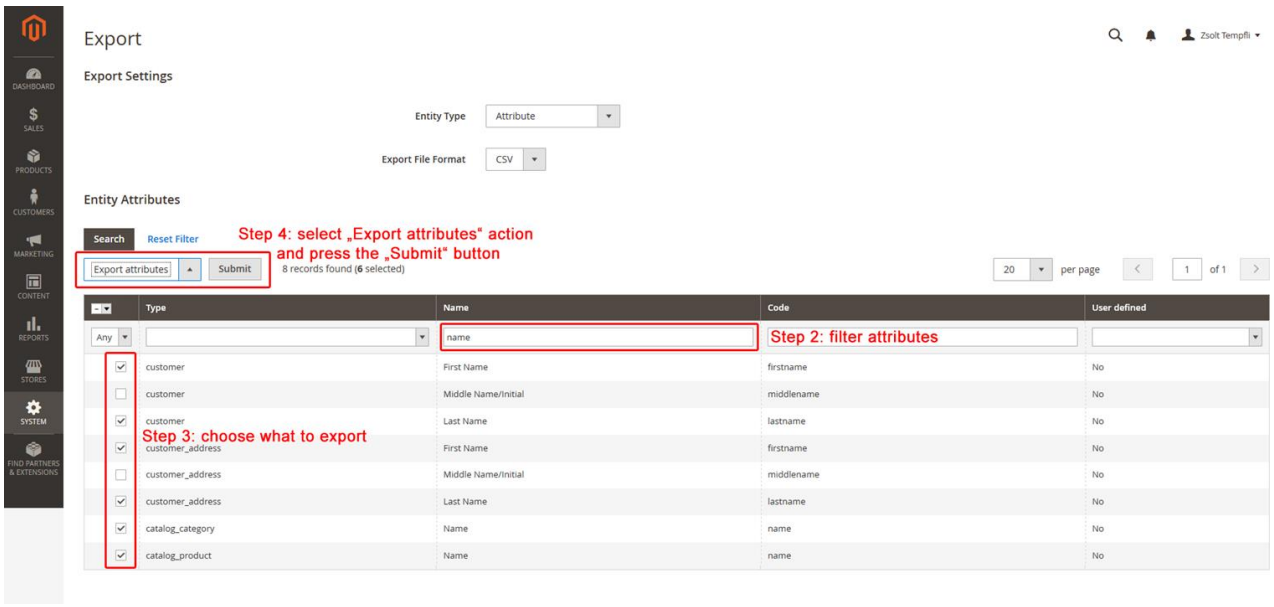
Exporting is a very simple task. Visit **System → Export** in the Magento backend and choose the *Attribute* entity type.



A grid representation of all attributes will appear, where you will be able to browse through all attributes of the webshop. If you don't want to export everything, just a few attributes, ex. those which are named partially „name", then enter the „name" string into the filter field on top of the Name column and press Enter or click on the Search button to filter.

A nice feature of Magento grids is that they are preserving the selections of previous pages, so in case there is more than one page with attributes and you have selected a couple, and then navigated to the second page, the selections from the first page will be remembered on submission.

When you are done with the filtering and the selection, choose the „Export attributes" option from the dropdown menu (below the „Search" button) and press the „Submit" button. The latter button will be shown only when the „Export attributes" are chosen.



Pressing the „Submit" button will put a new job into the export queue. Make sure that the queue processor is started, or start it with the following command:

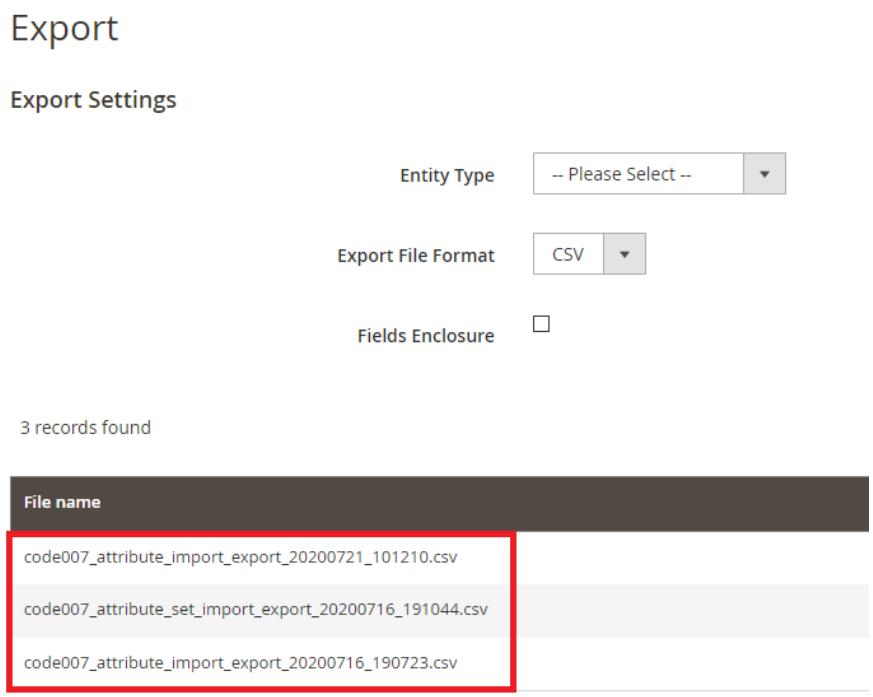
php bin/magento queue:consumer:start exportProcessor

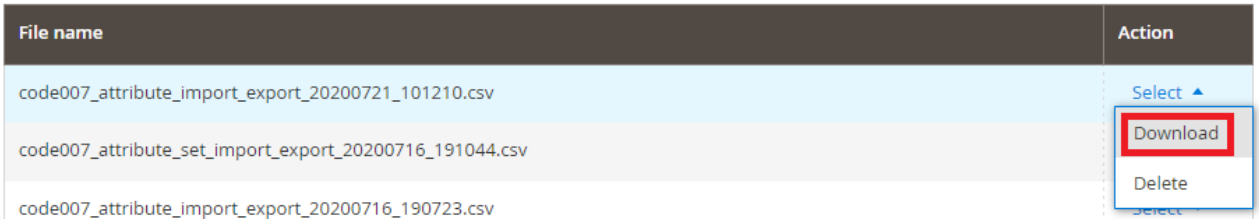
It is important to run the above command with the user which runs also the website, to prevent user permission errors (which will not let you download the generated files).

When the queue has been processed, the exported file will show up in

<path_to_magento_root>/var/export

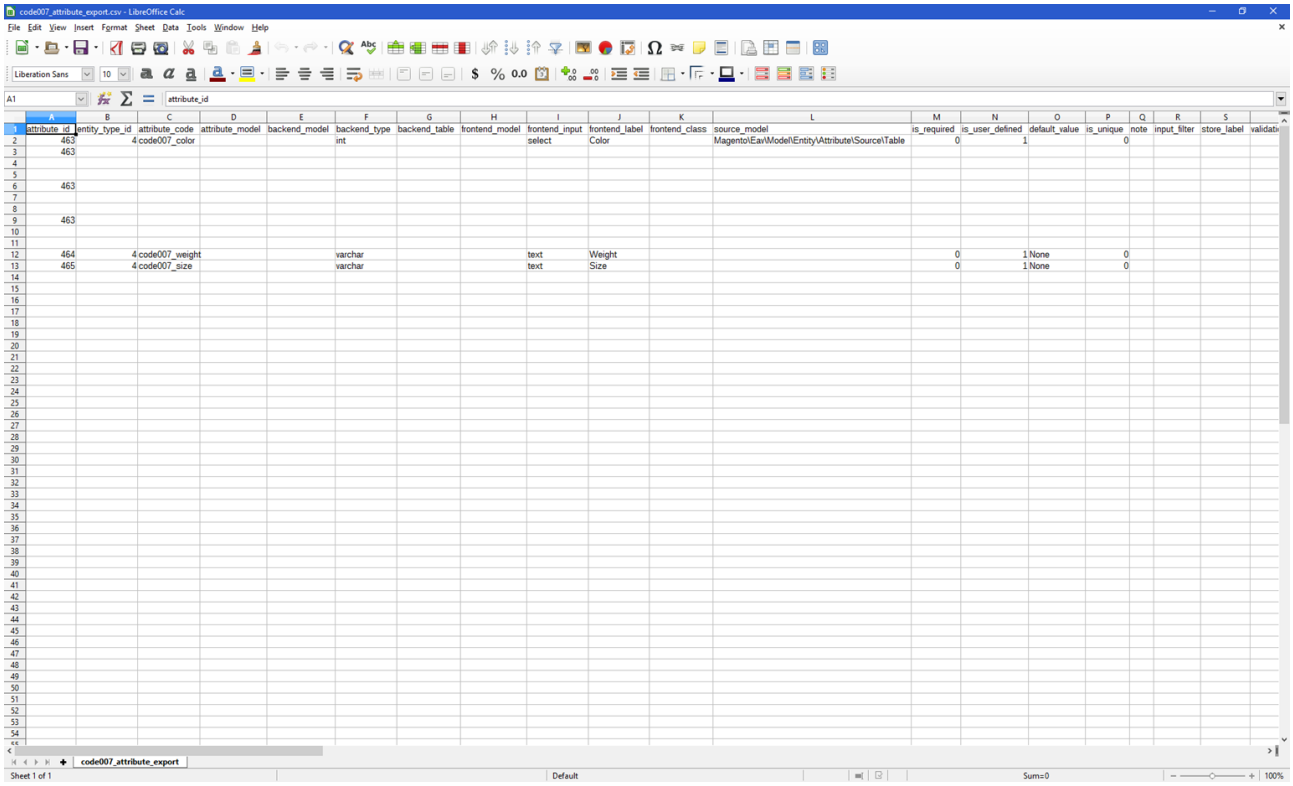
folder, and also in the list of exported files:





If the download button doesn't work, then most probably you have file permission errors (ex. you have run the queue consumer as root).

The CSV file looks something similar:



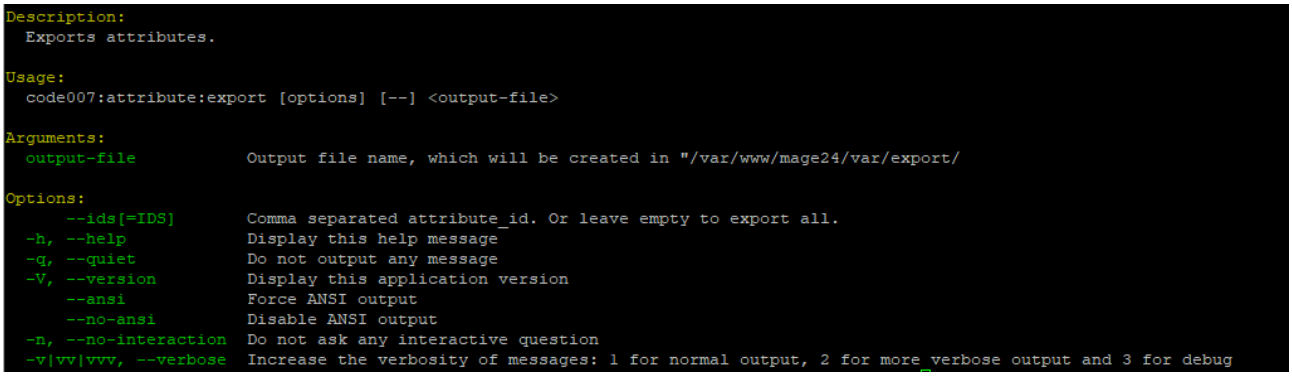
You will be able to import this file into any other Magento webshop which will have the Code007 Attribute Import-Export extension.

3.3 Exporting attributes in console

From version 1.3.0 it is possible to export attributes using command line. To get help, open the terminal and enter the following command in your Magento directory:

bin/magento help code007:attribute:export

This command will show the available command line options and arguments:

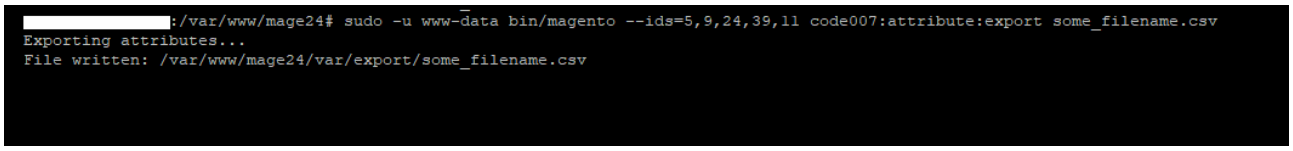


- The **--ids** argument is optional, you can use it to specify which attributes you want to export. This is a comma separated list. When not specified, all the attributes will be exported.
- The **<output-file>** is required, specifying paths is not allowed. Type only the file name and it will be saved to <Magento root>/var/export/<output-file>. Be careful, the specified file will be overwritten.

Example usage:

bin/magento code007:attribute:export --ids=5,9,24,39,11 some_filename.csv

The output will be something similar:



3.4 Importing attribute sets

This task is very similar to importing attributes. Log in to the Magento backend and go to **System → Import**, then choose the *Attribute set* entity type. Choose the *Add* import behavior, select the file to be imported and press the *Check Data* button in the top right corner of the page. To start the importing process press the *Import* button and confirm the warning popup.
Before importing attribute sets make sure that all the referenced attributes (from the CSV file) are already present in Magento.

3.5 Exporting attribute sets

Visit **System → Export** in the Magento backend and choose the *Attribute Set* entity type. Check the attribute sets to be exported and select the *Export attribute sets* and press the *Submit* button.

The attribute set export works very similar to the attribute export, so make sure the queue consumers are running (see here how to run it).

4. File formats

All the imported and exported files are in CSV (comma separated values) format. Most of the spreadsheet editor software (ex. Libreoffice, Open office, Microsoft Excel, etc.) can handle this file format. The most important thing is to pay attention when saving the file and before importing it to Magento, to use the same separator character (usually it is a , or ;)

4.1 Attribute file format

Importing an attribute means that all the data structure which relates to it (options, swatches) will need to be imported as well. There are some basic columns which must be present. These are the following:

attribute_id: the attribute id, not necessarily the real id of an attribute, but for consistency. If an attribute option has an attribute_id, it means that it is related to the attribute row with this attribute_id. Format: integer. Apply to: attribute, attribute option

entity_type_id: one of the following values are valid: 1 = customer, 2 = customer address, 3 = catalog category, 4 = catalog product, 5 = order, 6 = invoice, 7 = creditmemo, 8 = shipment. Format: integer. Apply to: attribute

attribute_code: this is a unique code of an attribute, it represents the attribute of an entity. Format: string. Apply to: attribute

attribute_model: optional, it can point to a model class name. Format: string. Apply to: attribute

backend_model: optional, it can point to a model class name. Format: string. Apply to: attribute

backend_type: usually it is one of the following: *static*, *varchar*, *int*, *text*, *datetime*, *decimal*. Format: string. Apply to: attribute

frontend_model: optional, it can point to a model class name. Format: string. Apply to: attribute

frontend_input: usually it is one of the following: *select*, *text*, *date*, *hidden*, *boolean*, *multiline*, *textarea*, *image*, *price*, *weight*, *media_image*, *gallery*. Format string. Apply to: attribute

frontend_label: a frontend label of the attribute. Format: string. Apply to: attribute

frontend_class: optional, a css class for the frontend input. Format: string. Apply to: attribute

source_model: optional, it can point to a model class name. Format: string. Apply to: attribute

is_required, **is_user_defined**, **is_unique, visible**, **required**, **system**, **user_defined**, **is_used_in_grid**, **is_visible_in_grid**, **is_filterable_in_grid**, **is_searchable_in_grid**: boolean switches (0 = no, 1 = yes). Format: integer. Apply to: attribute

default_value: optional, a default value of the attribute. Format: string. Apply to: attribute

note: a short note for the user, which will be shown somewhere around the input. Format: string. Apply to: attribute

input_filter: optional, ex. *date*. Format: string. Apply to: attribute

store_label: Format: string, Apply to: attribute

validation_rules, **options**: optional, serialized php object. Format: serialized string. Apply to: attribute

multiline_count: optional. Format: integer. Apply to: attribute

sort_order: sort order, the lowest number will be at the top. Format: integer. Apply to: attribute, attribute option

_dataType: the module's internal type column. Use the following strings: *eav_attribute* = attribute; *eav_attribute_option* = attribute option; *eav_attribute_option_value* = attribute option value; *eav_attribute_option_swatch* = attribute option swatch

_dataSerialized: the module's internal column; it specifies which of the columns are php serialized strings. Separate the columns by comma character. Format: string. Apply to: attribute

option_id: the id of an option. Format: integer. Apply to: attribute option, attribute value, attribute option swatch

value_id: the id of an option value. Format: integer. Apply to: attribute value

store_id: Format: integer. Apply to: attribute value

value: the value of an attribute option. Format: string. Apply to: attribute value

The following column names are specific to product attributes only:

frontend_input_renderer: optional, it can point to a model class name. Format: string.

is_global: attribute scope. 0 = Store View, 1 = Global, 2 = Website

is_visible: boolean switches (0 = no, 1 = yes). Format: integer.

is_searchable: boolean switches (0 = no, 1 = yes). Format: integer.

is_filterable: boolean switches (0 = no, 1 = yes). Format: integer.

is_comparable: boolean switches (0 = no, 1 = yes). Format: integer.

is_visible_on_front: boolean switches (0 = no, 1 = yes). Format: integer.

is_html_allowed_on_front: boolean switches (0 = no, 1 = yes). Format: integer.

is_used_for_price_rules: boolean switches (0 = no, 1 = yes). Format: integer.

is_filterable_in_search: boolean switches (0 = no, 1 = yes). Format: integer.

used_in_product_listing: boolean switches (0 = no, 1 = yes). Format: integer.

used_for_sort_by: boolean switches (0 = no, 1 = yes). Format: integer.

apply_to: optional, comma separated values of product types (ex. simple, virtual, etc.). Format: string

is_visible_in_advanced_search: boolean switches (0 = no, 1 = yes). Format: integer.

position: sorting position of product attribute. The lowest goes to the top. Format: integer

is_wysiwyg_enabled: boolean switches (0 = no, 1 = yes). Format: integer.

is_used_for_promo_rules: boolean switches (0 = no, 1 = yes). Format: integer.

is_required_in_admin_store: boolean switches (0 = no, 1 = yes). Format: integer.

search_weight: Format: integer

additional_data: optional. Format: json string

4.2 Attribute set file format

attribute_set_id: the attribute set id, not necessarily the real id of an attribute set, but it is for consistency. Ex. if an attribute group has an attribute_set_id, it means that it is related to the attribute set row with this attribute_set_id. Format: integer. Apply to: attribute set, attribute group, entity attribute

entity_type_id: one of the following values are valid: 1 = customer, 2 = customer address, 3 = catalog category, 4 = catalog product, 5 = order, 6 = invoice, 7 = creditmemo, 8 = shipment. Format: integer. Apply to: attribute

attribute_set_name: a label for the attribute set. Format: string. Apply to: attribute set

sort_order: sort order of the entity. The lowest goes to the top. Format: integer. Apply to: attribute set, attribute group, entity attribute

_dataType: the module's internal type column. Use the following strings: *eav_attribute_set* = attribute set; *eav_attribute_set_group* = attribute set group; *eav_attribute_set_entity_attribute* = entity attribute

attribute_group_id: the attribute set group id. Format: integer. Apply to: attribute set group, entity attribute

attribute_group_name: the label of the group. Format: string. Apply to: attribute set group

default_id: Format: integer. Apply to: attribute set group

attribute_group_code: attribute group code (similar to the attribute_code column of an attribute). Format: string. Apply to: attribute set group

tab_group_code: optional. Format: string. Apply to: attribute set group

_attribute_code: the attribute_code of the attribute which is referenced. Format: string. Apply to: entity attribute

5. Migrating from Magento 1 to Magento 2

The Code007 Attribute Import-Export extension has been written for both Magento 1.x and 2.x. The importing and exporting mechanisms are compatible with each other, which means that you have the possibility to migrate your attributes from Magento 1.x to Magento 2.x or vice versa. The only thing you have to change in your CSV files is the PHP class naming related fields.

Example: the following two classes are both pointing to the Catalog Product attribute model

Magento 1.x: catalog/resource_eav_attribute

Magento 2.x: Magento\Catalog\Model\ResourceModel\Eav\Attribute

So, when importing from 1.x to 2.x, make sure to replace all fields into the new notation, finding the equivalent class of Magento 2.x.

The same rule applies when importing from 2.x to 1.x, just vice versa. You will have to change the new style class names into the old 1.x notation.

If you are not sure, it is better to leave these fields empty.