



Manual for Improved Import & Export extension for Magento 2

Improved Import and Export extension allows importing csv, xml, json, xlsx, ods files with products data and product images to Magento 2. The files can be imported from a remote FTP / SFTP server, Dropbox, direct URL of a source CSV file, and starting from 2.1.1 version Google Sheets. In addition, the extension offers a dedicated category import from CSV files, import of attribute values and new attributes on the fly, import job scheduling with cron jobs , and other.

Besides, you get advanced export capabilities, including order export and export jobs, with attribute and table field mapping.

In this manual you will find all the necessary information on how to use and operate Improved Import and Export extension. However, including a thorough instruction on how to import each and every entity to Magento 2 would be an overkill.

In the guide list you will find step-by-step instructions on how to import every possible entity - such as configurable products or CMS pages - to your Magento 2 store. For every entity we have composed a dedicated blog post where you can find sample files, attribute reference and other useful information.

Now to the extension. Complete feature list of Improved Import/Export Magento 2 Extension includes:

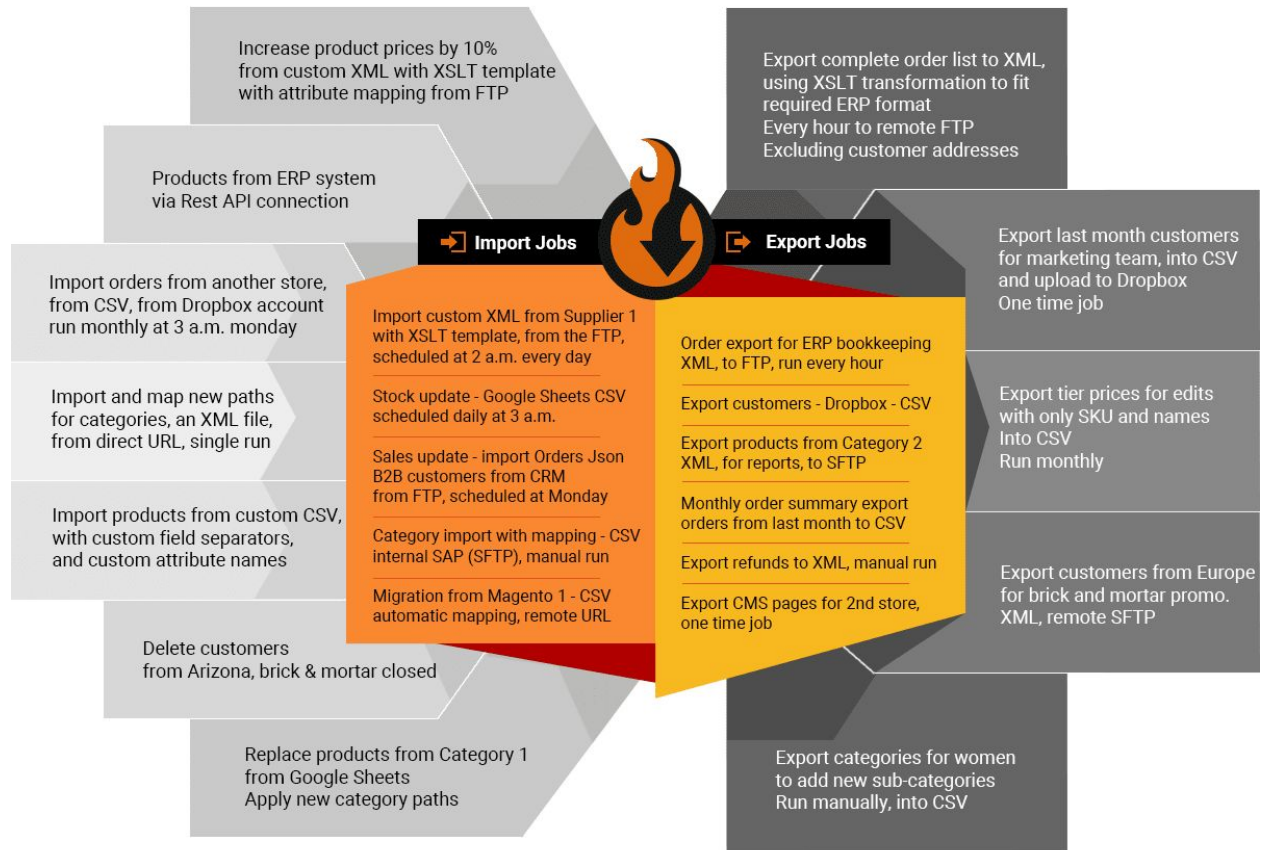
1. Import CSV/XML/Json/XLSX/ODS files from a remote FTP/SFTP server, Dropbox, direct URL or Google Sheets, Rest and Soap APIs;
2. Import product images from a remote FTP/SFTP server, Dropbox or via a direct URL;
3. Dedicated category import from a different file types for selected categories (default M2 store allows importing categories only together with assigned products);
4. Creation of new product attributes on the fly during import process;
5. Magento 2 product attribute import and export, along with attribute sets

6. Creation of new attribute values on the fly: if attribute values don't exist in Magento 2 database, they are created during the import process;
7. Magento 2 product color swatch import with image and color values;
8. **Full support of Google Sheets** - paste links and import directly from the sheets;
9. **Tier price import with products** - imports both products and tier prices from a single file;
10. **Configurable products can be created on the fly** - specify SKU of parent product for a simple product and configurable will be created automatically;
11. Asynchronous import and export process for data heavy tables - now you can easily import and export data heavy tables from your browser;
12. Built-in cron jobs for any kind of import (products , categories, customers, stock, attributes, etc.) – flexible and extremely powerful feature of Magento 2 Enterprise Edition is now available for Community Edition!
13. Full support for configurable products: files can be placed locally or imported from a remote URL source;
14. Fields mapping (in Magento 1.x style!) – you can map Magento 2 product attributes to any custom column in your file! Fully flexible and powerful way to import custom data structures to Magento!
15. **Support for XML, Json, XLSX, ODS** file formats;
16. XSLT templates for transforming XML files into appropriate format;
17. Export jobs with attribute mapping functionality;
18. Run import / export jobs from admin panel or using CLI;
19. Extended debugging functionality with job logging;
20. Order export and import with invoices, credit memos, and other docs;
21. Magento 1 and Shopify presets for import jobs;
22. Hard-coded / default values during mapping;
23. File validation;
24. Inline Edit Mode for cron in a grid;
25. Attributes based export filtering;
26. Sample files in the extension pack and manual;
27. Direct CSV upload;
28. Unzip / untar archives automatically;
29. History with logs.

Improved Import Export Magento 2 Roadmap - vote for the most exciting features and we will implement them first. Leave your suggestions in the comments and we will add them to the poll.

Before we begin, check a brief illustration of the extension features:

Improved Import & Export for Magento 2 - Flow Example



Installation

via Composer

To install the extension using composer follow these steps:

1. Log into your FireBear account and proceed to **My Extensions Download** section.
2. Scroll down to the **Composer credentials** block. Here you will find api keys and installation commands.
From this moment, you can follow the instructions in the **Composer credentials** block.
3. Make sure your composer username and password are generated. Otherwise click 'Refresh api keys' button.

Composer credentials

Username: sampleusername copy

Password: samplepassword copy

4. Navigate to your store root folder in the SSH console of your server:
cd path_to_the_store_root_folder
5. Add FireBear composer repository to your Magento 2 by copying and running relevant command from your root directory. The commands include api keys, and require to be copied from the account dashboard.

1. Add composer repository.

Community edition:

```
composer config repositories.firebear composer https://firebearstudio.com,
```

copy

or

Enterprise edition:

```
composer config repositories.firebear composer https://firebearstudio.com,
```

copy

If you are using Magento 2 Cloud or Commerce - copy and run Enterprise edition command with your api credentials:

```
composer config repositories.firebear composer
https://firebearstudio.com/composer/download/package/type/ee/user/youruser/password/
yourpassword/
```

If you are using Magento 2 Open Source - copy Community and run Community edition command with your api credentials:

```
composer config repositories.firebear composer
https://firebearstudio.com/composer/download/package/type/ce/user/youruser/password/
yourpassword/
```

6. Install the extension by running command:

```
composer require firebear/importexport
```

2. Setup extension.

```
composer require firebear/importexport
```

[copy](#)[Refresh api keys](#)

7. Enable the extension by running:

```
php bin/magento module:enable Firebear_ImportExport
```

8. Deploy content and flush store cache, log out from the backend and log in again. Run:

```
php bin/magento setup:static-content:deploy -f
```

and:

```
php -f bin/magento cache:clean
```

If you require 2.2.1 version of the extension for some reason, you can run:

```
composer require firebear/importexport:2.2.1
```

Upgrading the extension via composer

To upgrade the extension, navigate to your store root folder just as per instructions above and run the following commands:

```
composer update firebear/importexport
```

after:

```
php bin/magento setup:upgrade
```

then:

```
php bin/magento setup:static-content:deploy -f
```

and:

```
php bin/magento cache:clean
```

The extension has been updated.

Manual installation for both Commerce (Enterprise), Cloud and Open Source (Community)

1. Backup your web directory and Magento 2 store database;
2. Download Improved Import installation package;
3. Copy the contents of the package to `/app/code/Firebear/ImportExport` folder. Make sure to create folders if required.
4. Navigate to your store root folder in the SSH console of your server:

```
cd path_to_the_store_root_folder
```

first run:

```
php -f bin/magento module:enable Firebear_ImportExport
```

proceed with:

```
php -f bin/magento setup:upgrade
```

and:

```
php -f bin/magento setup:static-content:deploy
```

5. Flush store cache, log out from the backend and log in again.

```
php -f bin/magento cache:clean
```

Installing B2B Add-On

B2B Add-on for Improved Import and Export extension is purchased separately and installed separately.

To learn more about B2B Add-On, supported entities, and how to use it - [read dedicated B2B Add-On manual](#).

Installing MSI Add-On

MSI Add-on for Improved Import and Export extension is FREE, you can find it in your account if you purchased Improved Import and Export extension. MSI Add-On is installed separately.

To learn more about MSI Add-On, supported entities, and how to use it - [read dedicated MSI Add-On manual](#).

Getting Dropbox API

1. Create [Dropbox account](#).
2. Navigate to [My apps](#) screen and click 'Create app' button.
3. Choose 'Dropbox API', select the most convenient 'Type of access', name your app, and click 'Create app' button.

1. Choose an API

<input checked="" type="radio"/> Dropbox API For apps that need to access files in Dropbox. Learn more	<input type="radio"/> Dropbox Business API For apps that need access to Dropbox Business team info. Learn more
--	--

2. Choose the type of access you need

[Learn more about access types](#)

<input checked="" type="radio"/> App folder – Access to a single folder created specifically for your app.
<input type="radio"/> Full Dropbox – Access to all files and folders in a user's Dropbox.

3. Name your app

Test Firebear App

4. At the app screen find App key, and App secret. You will need to submit these into the appropriate fields when creating an Import Job.

Test Firebear App

Settings	Branding	Analytics
----------	----------	-----------

Status	Development	Apply for production
--------	-------------	----------------------

Development users	Only you	Enable additional users
-------------------	----------	-------------------------

Permission type	App folder ⓘ
-----------------	--------------

App folder name	Test Firebear App	Change
-----------------	-------------------	--------

You need to get these two.

App key	zdkaxq1yp4duyw3
App secret	Show

5. That's it, now you have all necessary credentials to import files from your Dropbox account.

Installing Json library

To import Json files into Magento 2 you first need to add a Json library.

Navigate to your store root folder in the SSH console of your server:

cd path_to_the_store_root_folder

run:

```
composer require salsify/json-streaming-parser 6.0
```

and:

```
composer update
```

That's it. You can now import Json files into your Magento 2.

Installing REST API library

To use Rest API for import purposes, you first need to install [a library](#).

Navigate to your store root folder in the SSH console of your server:

```
cd path_to_the_store_root_folder
```

run:

```
composer require tcdent/php-restclient
```

and:

```
composer update
```

That's it. You can now use Rest API to import entities to your Magento 2.

Standard Magento 2 import

Before going any further with Improved Import, make sure you have read manuals and understand what to do with the default Magento 2 import procedure!

- [The Complete Guide to Magento 2 Product Import / Export](#)

Extension Configuration

Once installed the extension introduces its configuration section to **System > Configuration > Firebear Studio > Import/Export**.

The configuration section is represented with a single setting **Create attribute values on the fly**. The setting controls if the product attributes can be created directly from the import table.

To create new attributes one should use custom column structure that will be covered later in this guide in the **Product attribute values on the fly** import section.


Import and Export Job grids

Most of your time with the extension you will spend at **System > Improved Import/Export > Import Jobs** and **Export Jobs**. There you can find Import and Export Job grids, create new job and edit the existing ones.

Import and Export Job grids have the same look and functionality and for convenience purposes we will cover both grids at the same time.

Import Jobs

🔍 🔔 👤 andrey ▾

 Firebear Improved Import / Export v.2.0.0 **last version**

[Import Jobs](#) | [Export Jobs](#) | [Extension manual](#) | [Sample files](#) | [FAQ](#) | [Support](#) | [Extension details](#) | [Download](#)

[Add New Job](#)

Filters

👁 Default View ▾

⚙ Columns ▾

Actions ▾

6 records found

20 ▾ per page

< 1 of 1 >

<input type="checkbox"/>	ID ↑	Title	Status	Cron	Frequency	Entity Type	Import Source	Action
<input type="checkbox"/>	6	Import products advanced pricing - XML SAP	ENABLED	* * * * *	Custom	Advanced Pricing	File	Select ▾
<input type="checkbox"/>	5	Import products CSV from Magento 1.9 - external shop	ENABLED	0 3 * * 1	Week	Products	File	Select ▾
<input type="checkbox"/>	4	Import customers from CRM - XML	ENABLED	* */1 * * *	Hour	Customers Main File	File	Select ▾
<input type="checkbox"/>	3	Update products stock CSV - warehouse 6	DISABLED	0 3 1 * *	Month	Products	File	Select ▾
<input type="checkbox"/>	2	Import Products XML - Supplier 1	ENABLED		None	Products	File	Select ▾

The grid displays the following information about each job:

- **ID** – unique ID of the job;
- **Title** – title of the job for internal identification purposes;
- **Status** – current status of the job, whether it is enabled or disabled;
- **Cron** – cron schedule, how often the job is launched automatically;
- **Frequency** – one of the predefined frequency values, if selected, how often the job is launched automatically;
- **Entity Type** – the entity the job imports to the store (products, advanced pricing, etc.);
- **Import Source** – source file with the table from which the values should be imported, can be: file, ftp, sftp, url, Google sheet.

All these columns can be adjusted using the '**Columns**' button at the top of the screen. To save the changes you have made to the job grid - use '**Default View**' button. If you are looking for a

particular job - use **'Filters'** button. Last is the **'Action'** column, here you can edit and delete jobs as well as change their statuses. Alternatively, you can apply these actions to multiple jobs simultaneously.

Filters

Default View

Columns

Actions

6 records found

20 per page

<

1 of 1

>

<div><div></div></div>	ID ↑	Title	Status	Cron	Frequency	Entity Type	Import Source	Action
<input type="checkbox"/>	6	Import products advanced pricing - XML SAP	ENABLED	* * * * *	Custom	Advanced Pricing	File	Select
<input type="checkbox"/>	5	Import products CSV from Magento 1.9 - external shop	ENABLED	0 3 * * 1	Week	Products	File	Select
<input type="checkbox"/>	4	Import customers from CRM - XML	ENABLED	* */1 * * *	Hour	Customers Main File	File	Select
<input type="checkbox"/>	3	Update products stock CSV - warehouse 6	DISABLED	0 3 1 * *	Month	Products	File	Select
<input type="checkbox"/>	2	Import Products XML - Supplier 1	ENABLED		None	Products	File	Select
<input type="checkbox"/>	1	Import Job 1	ENABLED	0 3 1 * *	Month	Products	File	Select

Note, that it is also possible to edit the job right from the grid. You only need to click on the job.

Filters

Default View

Columns

Actions

6 records found (1 selected)

20 per page

<

1 of 1

>

<div><div></div></div>	ID ↑	Title	Status	Cron	Frequency	Entity Type	Import Source	Action
<input checked="" type="checkbox"/>	6	Import products advanced pricing - XML SAP	ENABLED	* * * * *	Custom	Advanced Pricing	File	Select
<input type="checkbox"/>	5	Import products CSV from Magento 1.9 - external shop	ENABLED	0 3 * * 1	Week	Products	File	Select
<input type="checkbox"/>	4	Import customers from CRM - XML	ENABLED	* */1 * * *	Hour	Customers Main File	File	Select
<input type="checkbox"/>	3	Update products stock CSV - warehouse 6	DISABLED	0 3 1 * *	Month	Products	File	Select
<input type="checkbox"/>	2	Import Products XML - Supplier 1	ENABLED		None	Products	File	Select
<input type="checkbox"/>	1	Import Job 1	ENABLED	0 3 1 * *	Month	Products	File	Select

This is it for the job grids. Now, let's add a new import job to see how it's done.

Add new Import Job

To add a new import Job click 'Add New Job' button at the top of the job grid. You will be redirected to the New Job screen which is split into four configuration sections:

- General Settings;
- Import Settings;

- Import Behavior;
- Import Source.

Later, as we will learn how to select the import entity, the new configuration sections, specific to the imported entity, will appear. We will cover these sections later, in the chapters dedicated to specific entity import.

General settings

The first set of settings defines the name of the job, schedule and general job behavior.

General Settings

Enable Job

☒ Yes

Job Title *

New Import Job

Frequency *

Custom

▼

Cron Schedule

*

2

7

*

*

Minutes

Hours

Days

Months

Days of Week

↑ Use this field if you have good cron knowledge.

Locale *

English (United States) / English (United States)

▼

Generate Unique Uri if Duplicate

☒ Yes

Re-Index after Import

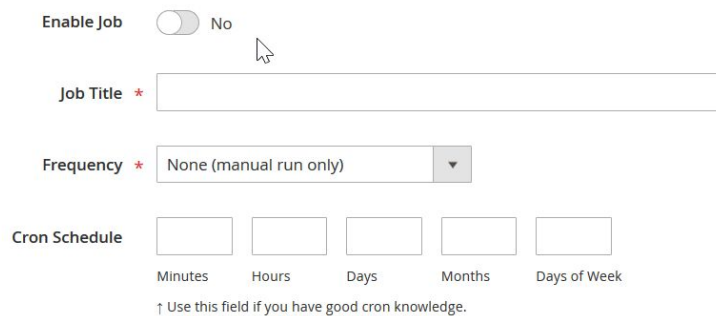
☒ Yes

- Start by switching **Enable Job**. If enabled, the job will automatically run according to the set schedule. If you are going to only run the job manually you can leave the job disabled.
- Then, think about the expressive **Job Title** that describes the job best. It will help you find the job in the grid or understand which jobs are enabled/disabled.
- Next, it will be necessary to configure the **Frequency** of the import job, - how often the job should be automatically executed. You can select one of the predefined values or set up 'Custom' Frequency.
- **Cron Schedule** is where you configure 'Custom' frequency. It allows you to take full advantage of the cron daemon scheduling pattern. If you are not sure how to use it we will teach you how to a bit later.
- After configuring schedule you need to select the **Locale** of the store you are creating the job for.

- If you think that some of the imported products or categories may use the same URLs, switch **Generate Unique Url if Duplicate** on.
- At last, consider enabling **Re-index after Import**. Some of the stores, may be indexed for performance optimization, in such cases re-indexing will be required.

The first section of the import job configuration page is now complete. You know how to change job statuses and create cron schedules. The following GIF illustrates the aforementioned processes:

General Settings

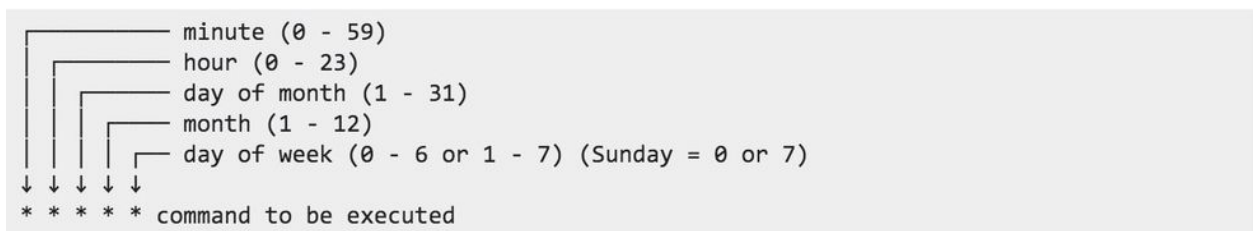


The screenshot shows a form titled "General Settings" with the following fields:

- Enable Job**: A toggle switch currently set to "No".
- Job Title ***: A text input field.
- Frequency ***: A dropdown menu currently set to "None (manual run only)".
- Cron Schedule**: Five input fields for "Minutes", "Hours", "Days", "Months", and "Days of Week". Below these fields is a note: "↑ Use this field if you have good cron knowledge."

How to use cron

Cron is a daemon scheduler for UNIX systems, it is used by Magento 2 and Improved Import and Export extension for scheduling execution of different scripts. It works on the following pattern:



Below you can find a handful of useful links to understand the concepts of the cron scheduler and master it. We advise you to try configuring cron yourself. It will be a great help of the import and export job automation.

- <https://en.wikipedia.org/wiki/Cron>
- <http://crontab-generator.org/>
- <http://www.cronmaker.com/>
- <http://stackoverflow.com/questions/18919151/crontab-day-of-the-week-syntax>

- <http://devdocs.magento.com/guides/v2.0/config-guide/cli/config-cli-subcommands-cron.html>
- <https://firebearstudio.com/blog/magento-2-cron-configuration.html>

Make sure you have set proper schedule before enabling the job or create some test jobs to see how the cron works.

Import settings

Now, it is time to proceed to the second section of the New Job screen – **Import Settings**. Here, you can select the entity that will be imported. Currently, the extension allows you to import the following entities:

- Products
- Categories
- Customers and Addresses (single file)
- Customers Main File
- Customer Addresses
- Advanced Pricing
- Attributes
- Cart Price Rules
- Orders
- CMS pages

In addition to the imported entity you decide whether you want to use Rest API as your Import Source and whether you want to clear product attribute values upon import.

Import Settings



Use API ☐ No

Entity *

Clear Attribute Values ☐ No

↑ Empty/deselect values for ALL optional attributes you are importing. Use carefully.

Remove Product Association ☐ No

↑ Remove Product Association (supported Grouped)

Clear Attribute Values setting allows you to clear values for ALL optional product attributes listed in the imported file. For example. One of your products has optional attributes 'color' and 'size'. If the 'color' attribute is in the table your are importing its value for all imported products

will be reset. However the value of 'size' attribute which was not in the table will remain untouched.

Remove Product Association settings works like the 'replace' behavior for Grouped Products only. If enabled, only the simple products from the currently imported table will be assigned to grouped products. All other simple products will be removed from the grouped products. *For example: you have grouped product A, with simple products B, C, D. If you import grouped product A, with only simple B assigned, with **Remove Product Association** enabled, you will have a grouped A, with only simple B assigned. C and D simples will be removed.*

Depending on the selected import entity you get different configuration sections. Every entity, however, has two sections in common: **Import Behavior** and **Import Source**.


Import behavior

In this section, first you are to specify the Import Behavior. There are four general behavior types:

- **Add/Update** - the new entities from the import table will be added to your store and existing products updated according to the values in table;
- **Only update** - the new entities from the import table will NOT be added to your store, instead, only existing products will be updated according to the values in the table;
- **Replace** - existing entities will be replaced with the products of the same SKU from the import table;
- **Delete** - the entities specified in the import table will be deleted from the store.

Import Behavior 



Import Behavior * 

Validation Strategy * 

Allowed Errors Count *
Please specify number of errors to halt import process

Field Separator *

Multiple value separator *

Category Levels separated by *
The format of categories column in your CSV is :Root Category/Level1/Level2

Categories separated by *
The format of categories column in your CSV is : Root Category/Level1A, Root Category/Level1B

Each value does what it reads. Choose the one that suits your needs. Some entities, however, may come with the behavior type a little different. Such unique behavior types are described individually.

Next, configure **Validation Strategy**. You can 'Stop on Error' or 'Skip error entries' during import process. And specify allowed error quantity, maybe a couple will do.

Then, you are suggested to choose various **separators** in case the file you are importing differs in formatting from the native Magento 2 .CSV file. The separator values are the following:

- **Field Separator;**
- **Multiple value separator;**
- **Category Levels separated by** - when you are importing products, you can decide which separator you use for the categories column to separate category levels. By default / is used to separate multiple category levels. However, if you have /s in category names you may want to use different separator for category levels.
- **Categories separated by** - when you are importing products, you can decide which separator you use for the categories column. By default comma is used to separate multiple category paths. However, if you have commas in category names you may want to use different separator for category paths.


Default values of the Import Job separator fields correspond to the Magento 2 CSV file requirements, so, if you are using such CSV you shouldn't change anything.

If you are using new import table of which separators you are not sure of, you can open it with any editor (like MS Excel) and find what separators are used.

Import source

File types

Import File Type section illustrates lots of useful improvements in comparison with the default import procedure. You can select one of the three file types (the default import allows CSV only): **CSV, XML, XLSX, ODS and Json**. Note that you can use XML files of the following format: [Magento 2 Import Export Sample XML Files](#). Alternatively, you can upload any other format, but it is necessary to use mapping. You can always find up-to-date sample import/export files in all formats at Firebear Github [Import Export Sample Files for Magento 2](#).

Import Source 

Import File Type

Import Source *

Use Image Import Source

- Please Select --
- File
- Dropbox
- Ftp
- Sftp
- Url
- Google Sheet

Json file support

The latest release of the extension 2.2.0 added support of the Json files. A first step to Rest API support and integration of the extension with all possible software.

Right now, you can import Json files. Products, categories, CMS pages, all content that can be imported with CSV and XML can be imported to Magento 2 with Json as well.

Read more about how to import Json files to Magento 2 in our dedicated blog post:

[How to import Json files to Magento 2](#)

We have also composed some sample files to help you get started with Json import. All Magento 2 sample files and Json samples can be found at the FireBear Studio GitHub:

[Get Magento 2 sample Json files](#)

As for customization, Improved Import and Export extension comes with the customization endpoints with can be easily accessed with [Customization module](#). You can start building your integration with POS, CRM or ERP systems right away with Json support for Magento 2.

Excel and OpenOffice files support

Excel XLSX files and OpenOffice ODS files can now be used to export and import entities to Magento 2. XLSX and ODS files have certain limitations when it comes to the number of rows and columns. Here is the comparison table to give you the rough idea. Please note, the numbers for ODS format are valid for OpenOffice.org Calc 3.

File type	Rows	Columns
ODS Calc 3	1048576	16384
XLSX	1048576	16384

Read more about how to import XLSX and ODS files to Magento 2 in our dedicated blog post:

[How to import Excel and OpenOffice files to Magento 2](#)

We have also composed some sample files to help you get started with XLSX and ODS import. All Magento 2 sample files and XLSX and ODS samples can be found at the FireBear Studio GitHub:

[Get Magento 2 sample Excel and OpenOffice files](#)

Please note, ODS files downloaded from Google Sheets cannot be validated by the extension. This is an issue of the Google Sheets converter.

Choosing import source

Next, choose your import source. It may be:

- File
- FTP
- SFTP
- URL
- Dropbox
- Google Sheets
- Rest API
- Soap API

Yes, the Google Sheets, at last! 2.1.1 version of the extension introduced [Full Support of Google Sheets](#). You can now work with your colleagues on the same document, paste it and run the import jobs. Just like this.

By the way, we have composed a [Master Import Table in Google Sheets](#) to provide you with a comprehensive reference to the Magento 2 import. Make sure to take full advantage of it.

Note that for each upload type, the Improved Import / Export Magento 2 extension provides file validation (a big grey button below to the left).

Import File Type

Import Source *

Url *

The file must match import file format.

Use Image Import Source ☐ No

Validate file

Starting from version 2.1.2 the extension also comes with the **Use Image Import Source** switch. The switch decides how the extension will handle image paths if your source is FTP or SFTP.

- If the **Use Image Import Source switch is disabled**, the extension will look for the images inside your Magento 2 root catalog, according to the path specified in any image attribute columns of the import table.
- If the **Use Image Import Source switch is enabled**, the extension will look for the images INSIDE of the FTP/SFTP according to the path specified in any image attribute columns of the import table.

For example:

If the value for the product image in the base_image column is URL:

- with disabled **Use Image Import Source switch is disabled** option the extension will proceed to download the image by the URL specified;
- with enabled **Use Image Import Source switch is enabled** option the extension will try to find the specified URL path INSIDE of the the FTP/SFTP you are importing the table from.

File

In case of 'File', you can use direct file upload or specify its path.

Import File Type

CSV ▼

Import Source *

File ▲

File Upload

Upload

File Path *

The file must match the format. Use relative path to Magento installation, e.g. var/import/products.csv (Make sure folder have correct write permissions - 775 , for more details please see Magento 2 Permissions guide)

Images File Directory

Use relative path to Magento installation, e.g. var/import/images/

Validate file

Images File Directory allows you to specify the path, relative to your Magento 2 folder, where the extension should look for images. If the **Images File Directory** is not specified the extension will use a default folder path relative to your Magento 2 installation: **pub/media/import**.

NOTE:

The actual path to the image is composed of the value you enter in the **Images File Directory** field + value of the *image* attribute.

For example:

Case 1:

- You have uploaded product images to the **pub/media/import/**. And set **Images File Directory** value to **pub/media/import/**.
- In this case, in the relevant image attribute you will only need to specify the name of the image file. Like: **product_image_1.jpg**
- So, the full path combined will be **pub/media/import/product_image_1.jpg**.


Case 2:

- You have uploaded product images to the **pub/media/import/**, and set **Images File Directory** value to **pub/media/import**.
- However, every product image is located in the dedicated folder with the product name. For example: **/product_1/product_image_1.jpg**
- In this case, in the relevant image attribute you will need to specify the path relative to the value you specified in the **Images File Directory** field:
/product_1/product_image_1.jpg

- So the full path combined will be **pub/media/import/product_1/product_image_1.jpg**.

Dropbox

Dropbox upload requires the following parameters to be specified: File Path, App Key, App Secret, and Access Token.

Import Source 

Import File Type

CSV ▼

Import Source *

Dropbox ▼

File Path *

/test.csv

The file must match the format.

App Key *

mhwraq7qvo4k9wyj

App Secret *


Access Token *

Validate file

Full Dropbox configuration and instructions on how to get all required parameters can be found in the beginning of this manual.

FTP and SFTP

For FTP and SFTP, specify File Path, Host, Port, Username, and Password.

Import Source 

Import File Type

CSV ▼

Import Source *

Ftp ▼

File Path *

/test.csv

The file must match the format.

Host *

Port *

Username *

Password *

Validate file

If you are not sure about the credentials - ask your FTP or SFTP provider for the correct ones.

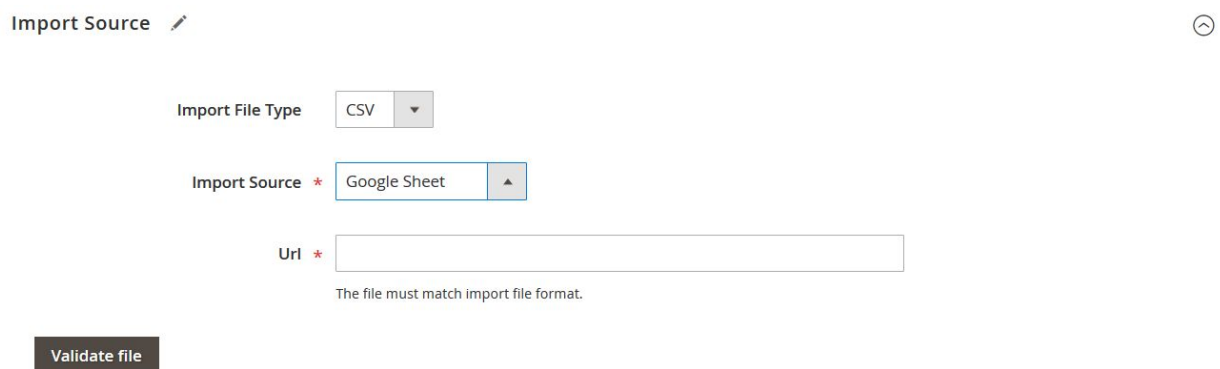
Google Sheets and direct URL

As for Google Sheets and URL upload, it is only necessary to paste the file URL in the appropriate field.

When composing a Google Sheet table you can use as many tabs as you want, the extension supports multiple tabs. To copy the Google Sheet tab URL:

- Click on the tab you want to import;
- Google Sheets will change the URL of the table to include the tab;
- Copy the URL of the tab while you are at its page;
- Paste the URL to the import job.

Remember, to import particular Google Sheet tab you only need to select it and copy the URL.



The screenshot shows a web form titled 'Import Source' with a pencil icon for editing. It contains three main fields: 'Import File Type' with a dropdown menu set to 'CSV', 'Import Source' with a dropdown menu set to 'Google Sheet' and a red asterisk, and 'Url' with an empty text box and a red asterisk. Below the 'Url' field is a small error message: 'The file must match import file format.' At the bottom left is a dark button labeled 'Validate file'. A circular arrow icon is in the top right corner.

For Google Sheets you first need to select the the actual sheet you want to import, in case you have several sheets in the import table, and only then copy and paste the URL.

This was as for the basic import job configuration. Now, you can click 'Save & Run' button to test the newly created job. If the uploaded file has a structure that differs from the default import file, then you will need to apply mapping. If it satisfies the Magento 2 requirements, you can start the import process right now.

Rest and Soap APIs

Rest and Soap APIs are enabled from the **Import Settings** section. Currently it is only possible to import products using API requests.

Use API ☒ Yes

Entity *

Products 

Other Rest and Soap API settings will be available in the **Import Source** section.

Import File Type

XML 

Import Source *

SOAP API 

SOAP Version *

SOAP_1_1 

Web Service Url *

Web Service Url

Call Function *

End point of Request

SOAP Options

SOAP Options

Here you decide which API version you want to use. Specify the Call Function and what do you want to get with SOAP request. If you are not sure how to use the APIs to integrate something, FireBear Studio offers its services. Contact our support team.

XSLT transformation for XML files


The latest release of the extension introduced XSLT transformation for XML files. What it means for the end user? It means that now you can import XML files of ANY format with proper XSLT code template.

Let's start from the beginning. Improved Import and Export extension allows importing XML files, however, of a particular format:

- **XML 1.0**
- **encoding:** UTF-8
- **field separator:** space

- **first row:** column names
- **top-level root name:** Items
- **XML record name:** item

In other words, if you want to import XML file of a different format you first need to transform it. Now, with introduction of the XSLT block you can do it directly from the Import Job screen.

XSLT Configuration 



On ☒ Yes

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <rss xmlns:g="http://base.google.com/ns/1.0" version="2.0">
3   <channel>
4     <title>Your Website</title>
5     <link>http://www.yoursite.com/us</link>
6     <description>Your Website</description>
7     <item>
8       <g:id>myproductid</g:id>
9       <g:title>Working Boots Size 7.5</g:title>
10      <g:description>Excellent for daily use</g:description>
11      <g:google_product_category>Women's > Shoes > Working Boots</g:google_product_category>
12      <g:link>http://www.example.com/product/working-boots</g:link>

```

XSLT is a special programming language that dictates how XML files should be read. For example, your supplier provides you with the custom XML file with the product updates. In this case you can ask your developer to compose an XSLT template that will convert this custom XML into a format readable by the extension.

Don't have a developer or don't know how to use XSLT? No problem!

We will compose an XSLT template for you! Drop us a line.

We will make sure you can import any XML files to your Magento 2.

If you want to know more about what is XSLT and how can one use it - read a dedicated blog article we have composed for you:

[Magento 2 import XSLT transformation - how to import any XML to Magento 2](#)

Import prerequisites

Import is not a straightforward procedure in Magento 2. Improved Import and Export extension's aim is to make it much more user friendly. However, there are some things that cannot be simplified, such as Magento 2 formatted CSV tables with the import entities, such as products.

So, to make sure you are prepared for the import procedure let's revise what do you need to have before attempting product or other entity import.

IMPORTANT!

We strongly advise to read through the [Complete Product Import Guide](#) before referring to import procedure. This guide will help you understand the concepts behind the import procedure and how the Magento 2 addresses different import entities.

Import Table, requirements and where to get

To import anything to Magento 2 using Improved Import and Export extension you need to have a properly formatted CSV or XML table. Native Magento 2 import process only supports CSV file formats, however with the help of FireBear extension you can also take advantage of the XML file format.

The import table should be properly formatted before attempting to import anything. There are several ways of getting properly formatted table:

1. Use [Google Sheet Master Table](#) FireBear team has composed for you. The table comes with the sheets covering every entity import, attribute description and expected values. [Read more about Master Table](#).
2. Download sample import files from [FireBear Studio GitHub](#). There you can find up-to-date import files for every entity if you prefer raw CSV file format.
3. Navigate to your store backend **System > Import**, select entity you want to import and click 'Download Sample File' link.

In case you want to compose the table manually you need to create the following CSV file:

Character set	Unicode (UTF-8)
Field separator	Comma, Tab
Text delimiter	"

Now that you have a properly formatted CSV table it is time to fill it with the import entities and learn how different entities are imported.

How to import products

Importing products and product updates is probably the most used feature of the Improved Import and Export extension. In this chapter we will cover everything you need to know about product import step by step.

Important notes

Product import can be conventionally divided into: product import and product update. The main difference is in REQUIRED product attributes that should be in the table you are importing. You need to remember:

Product import - understands that you will be creating new products from the import file. In this case the required columns or required attributes will be:

- **sku** - a unique product identifier.
- **attribute_set_code** - product must be assigned to an existing attribute set. The attribute set should also contain all the optional product attributes such as: color, size, fabric; and their values.
- **product_type** - Magento 2 needs to understand what type of product it needs to create.
- **name** - a name of the product that will be displayed to the customers.
- **price** - price of the product.

Remember, these attributes are mandatory. In order to create new products you need these attributes in your CSV table. All other product attributes, such as description and image, are optional and you can add them only if you need them.

For example in [this Google Sheet](#) you can see that we can create a configurable product with only 12 columns representing 12 product attributes.

Product update - understands that you will be updating attributes of the product(s) that already exist in your Magento 2 catalog. In this case the required columns will be:

- **sku** - a unique product identifier.
- **attribute_set_code** - product must be assigned to an existing attribute set. The attribute set should also contain all the optional product attributes such as: color, size, fabric; and their values.

It means, that if you already have products in your Magento 2 catalog, and only want to update for example stock and price, you can compose the table with four columns: *sku*, *attribute_set_code*, *price*, *qty*. For example like in the [Google Sheet here](#).

Step 1: select entity, behavior and upload import table

You should have already created a new import job as per **Add new import job** chapter.

First, to import products you need to:

- Set **Entity='Products'** in the **Import Settings** section of the job;
- In the **Import Behavior** specify what the extension has to do with the imported entities: add/update, only update, replace, delete.
- And upload the import table you have composed or downloaded in the Import Source section.

Once everything is done - click Validate button to proceed to the other job settings.

Step 2: map attributes

If the import file has a custom structure that doesn't fit the Magento 2 requirements, you can apply mapping. If not sure, you can check the sample import files at FireBear GitHub: [Import Export Sample Files for Magento 2](#).

The **Map Attributes** section provides the ability to apply presets for the import tables to automatically map attributes from the table with the ones in your Magento 2. There is a 'Select A Platform' field, where you can select a platform from which you are importing the CSV file. For example if you are migrating your Magento 1 store, you can take advantage of Magento 1 preset.

- Select the Magento 1 preset from 'Select a Platform' field. The extension will provide you with the links to the sample files, so you can view the structure of each one.
- Now, click the 'Load Map Attributes' button to automatically map attributes. The extension will compare product attributes from your store (System Attribute) with attributes from the import file (Import Attribute), applying a mapping preset.

Map Attributes



Select A Platform

 ▲

Validate

System Attribute *	Import Attribute *	Default Value
<div>Add New</div>		

Once the preset is applied, the extension will automatically perform mapping for fields that don't satisfy system requirements:

System Attribute *	Import Attribute	Default Value
<input type="text" value="_store"/> ▼	<input type="text" value="store_view_code"/> ▲	<input type="text"/>
<input type="text" value="_product_websites"/> ▼	<input type="text" value="product_websites"/> ▼	<input type="text"/>
<input type="text" value="_attribute_set"/> ▼	<input type="text" value="attribute_set_code"/> ▼	<input type="text"/>
<div>Add New</div>		

At the same time, you can still do everything manually: select a default Magento 2 attribute in the 'System Attribute' and specify an appropriate import attribute. Besides, you can type a default (hard coded) value for each attribute. It will be added for each imported item in the appropriate attribute column.

Validate

System Attribute *	Import Attribute *	Default Value
<input type="text" value="_store"/>	<input type="text" value="store"/>	<input type="text"/>
<input type="text" value="_product_websites"/>	<input type="text" value="websites"/>	<input type="text"/>
<input type="text" value="_attribute_set"/>	<input type="text" value="attribute_set"/>	<input type="text"/>

Add New

To delete a mapping item, click on the black bin icon. When the mapping configuration is prepared hit the 'Validate' button to validate mapping. That covers Map Attributes section.

Creating attributes with mapping

Using Map Attributes section you can create the attributes that are missing from the import file and apply default values for such attributes.

To do this click 'Add New Custom' button in the attributes grid.

System Attribute *	Import Attribute	Default Value
<input type="text" value="qty"/>	<input type="text" value="quantity"/>	<input type="text" value="100"/>

Add New
Add New Custom
Reset mapping

The new line will be added to the attribute grid. Here you will need to specify the Magento 2 attribute that already exists, type in the name of attribute that you want to create and specify the Default Value.

For example:

You are import products that are missing quantity attribute in the imported table. Using Map Attribute section you can:

1. Click 'Add New Custom' button;
2. Select 'qty' Magento 2 attribute which is responsible for storing product quantity;
3. Type in the attribute you want to create. In your example we use 'quantity' however you can name the new attribute whatever you like.
4. In Default Value field specify the value of the new quantity attribute. We specify '100'.

Result:

When the import is complete, every imported product will have 'qty' attribute value set to '100'. Because you specified that you want to use 'qty' attribute that already exists, and which corresponds with 'quantity' attribute you have just created. In this way the value of 'quantity' has been applied to 'qty'.

Applying attribute default value

Another feature available for attribute mapping is setting Default Value for the imported attribute. And there are two ways to do it which are controlled with **Set Default value** setting.

Set Default value ☒ Yes
This will replace all row value with default

Validate

System Attribute *	Import Attribute	Default Value
description ▼	description ▼	<p>Sample description</p> 🗑️


- **When Set Default Value setting is enabled** - the extension will REPLACE attribute value in ALL ROWS. E.g. you are importing 100 products, for every product you have a unique value for the 'description' attribute. If you enable Set Default Value and map the description attribute and set a default value for it. The extension will replace ALL unique values with the Default Value you have specified.
- **When Set Default Value is disabled** - the extension will ADD the Default Value only to the EMPTY ROWS. E.g. you are importing 100 products, for 50 products you have a unique value for the 'description' attribute, other 50 products have an empty value. If Set Default Value setting disabled, the extension will only add a Default Value to the empty rows.

Attribute value mapping

Using Map Attribute section you can also map attribute values you import.

Map Attribute Values 



System Attribute Value *	Import Attribute Value *
<input type="text" value="100"/>	<input type="text" value="56"/> 
<input type="button" value="Add New"/>	

In the Import Attribute Value you specify the value of the imported attributes. For example if you want to adjust qty value set to '100' you need to specify '100' as the value you want to adjust.

In the System Attribute Value you specify the value you want to get. For example if you want to adjust qty value from '100' to '56' you need to specify '56' as the value you want to get.

NOTE: the extension will replace ALL values that match value you typed in System Attribute Value. So play close attention if you are not sure some other attributes can have the same value.

Step 3: map categories

If you need to adjust the categories of the products from the import table you can make use of the **Map Categories** section.

- To start click 'Load Categories From Import File' button. The extension will load all values from the 'categories' column of the import table.
- Next, click 'Add New' button, to map the first category. At the 'Import category path' field select the category from the import table you want to map.
- At the 'New category path' select the category from your Magento 2 store you want the product to belong to. Now, the old category path is mapped with the category path from your store.

Map Categories



Load Categories From Import File

Import category path

New category path

Add New

New Category

NOTE:

'Default Category' in the category path - is a sample Magento 2 root category. If you want to import a root category - make sure to specify its name first in the category path. Remember - 'Default Category' is not a prerequisite for category path, rather an example.

If required, you can add new categories to your store right from the import job. To do this click 'New Category' button.

New Category



Create Category

Category Name *

Parent Category *

Setting root category

It is also possible to select a Root Category that will be used as a reference during category path import.

Map Categories 



Root Category

Shop



Load Categories From Import File

The Root Category you select defines the reference category for starting all category paths you import. For example:

- You are importing category path */Men/Shoes/*.
- You select Root Category value */Shop/*.
- When the import process is complete the result category path will be */Shop/Men/Shoes/*.

This setting can be really useful when you are importing products to different store views or want to rearrange category paths.

Step 4: price rules

At the **Price Rules** section you can adjust product prices specified in the import table. This is particularly useful if you want to add a set value for a certain manufacturer, or if your supplier increases/decreases the prices on a set percent value.

- Click 'Add Rule' button to start price adjustments;
- Next, in the 'Apply' column set whether you want to add a price or flat value to the prices of the product specified in the import table;
- Then, in the 'Value' column specify the value of the price adjustment;
- After, identify the products that will be a subject to the price adjustment using the 'Conditions'.
- If required you can add other lines of price adjustments by clicking 'Add Rule' button once again.

Apply	Value	Conditions
<div>Add Rule</div>		



Remember that the conditions are only for the products from the table you are importing, not for the products currently in your Magento 2 catalog.

Rounding prices to .49 or .99

In addition to adjusting the prices of the imported products you can also let the extension automatically round the price and special price of the products to the nearest value of .49 or 0.99.

Round Price to 0.49 or 0.99 ☒ Yes

Round Special Price to 0.49 or 0.99 ☐ No

Apply	Value	Conditions
<div>Add Rule</div>		

Round Price to and **Round Special Price to** settings if enabled tell the extension to scan the prices for decimal values, and if found, round such values to the closes .49 and .99 values.

This can help you follow the marketing recommendations on the price display and earn an extra buck without any manual price adjustments.


Step 5: Custom logic for creation of configurable products

You will need to use this section if your import table contains configurable products and you want to create them for your product catalog. You can find detailed example of how configurable products can be imported in the [dedicated chapter](#) of this manual. You can also refer to the [dedicated blog post about configurable products](#) where we have tried to create a complete guide to this complex product type.

Improved Import and Export extension version 2.1.1 has added several fast and convenient ways of importing Configurable Products to your Magento 2 store.

To import a configurable product with a bunch of assigned simple products you only need a single attribute. Let's break down how this section works.


In this section, we will only describe what each setting is responsible for and give general impression. If you are looking for in-depth guide on how to import configurable products using Improved Import and Export extension refer to the chapter 'How to import configurable products' of this manual.

Custom logic for creation of configurable products 


On ☒ Yes

Create Configurable Products ☐ No

Configurable Product Condition


Create config product by same attribute of simple products 


Attribute / column name on file

Select A Column 

Product attributes for variations

System Attribute

size (Size) 

color (Color) 

Add New

The new section for configurable product import comes with the following settings:

- **On** - defines whether the custom logic for creating configurable attributes is enabled;

- **Create Configurables Products** - defines whether NEW configurable product should be created at your Magento 2 store if simple products have a reference to it, however the import file lacks this very product and it cannot be found at your store.

For example:

We are importing nine simple products that are referenced to configurable product with SKU:123. However, in the table there is no configurable product with SKU:123, and there is no such product in your Magento catalog. In this case a configurable product with SKU:123 will be created automatically, providing **Create Configurable Products** is enabled. Note: newly created configurable product will have the name same as SKU, and no price applied.

- **Configurable Product Condition** - defines on what conditions the configurable product should be created. There are five condition types:

- **Create config product by the same attribute of simple products** – allows assigning simple products to a configurable product using dedicated column where the SKU (or any other recurring value) of the configurable product is specified.

How is it done.

In the import table, where you list all simple products that should be assigned to the configurable, you need to create a new column with a unique name. In the Master Table we use column 'group'.

In this column, next to the simple products, you need to specify the SKU of the

	A	B	C	D	E	F	G	H	I
1	sku	group	attribute_set_code	product_type	categories	name	description	short_description	price
2	NEW-S-Gray	NEW	Default	simple	Default Category/V	Test Configurable	<p>This is a test simple product that l	<p>That has been imported using Fire <p>This is a test simple proc	68
3	NEW-S-Green	NEW	Default	simple	Default Category/V	Test Configurable	<p>This is a test simple product that l	<p>That has been imported using Fire <p>This is a test simple proc	68
4	NEW-S-Purple	NEW	Default	simple	Default Category/V	Test Configurable	<p>This is a test simple product that l	<p>That has been imported using Fire <p>This is a test simple proc	68
5	NEW-M-Gray	NEW	Default	simple	Default Category/V	Test Configurable	<p>This is a test simple product that l	<p>That has been imported using Fire <p>This is a test simple proc	68
6	NEW-M-Green	NEW	Default	simple	Default Category/V	Test Configurable	<p>This is a test simple product that l	<p>That has been imported using Fire <p>This is a test simple proc	68
7	NEW-M-Purple	NEW	Default	simple	Default Category/V	Test Configurable	<p>This is a test simple product that l	<p>That has been imported using Fire <p>This is a test simple proc	68
8	NEW-L-Gray	NEW	Default	simple	Default Category/V	Test Configurable	<p>This is a test simple product that l	<p>That has been imported using Fire <p>This is a test simple proc	68
9	NEW-L-Green	NEW	Default	simple	Default Category/V	Test Configurable	<p>This is a test simple product that l	<p>That has been imported using Fire <p>This is a test simple proc	68
10	NEW-L-Purple	NEW	Default	simple	Default Category/V	Test Configurable	<p>This is a test simple product that l	<p>That has been imported using Fire <p>This is a test simple proc	68
11	NEW		Default	configurable	Default Category/V	Test Configurable	<p>This is a test configurable product	<p>That has been imported using Fire <p>This is a test configurabl	68

configurable product you want this simple assigned to.

- **Create configurable products by custom rules (part of a line before delimiter)** –allows assigning simple products using SKU (or any dedicated column), where the extension reads part of the value before delimiter (which is specified separately).
- **Create configurable products by custom rules (part of a line after delimiter)** – allows assigning simple products using SKU (or any dedicated column), where

the extension reads part of the value after delimiter (which is specified separately).

- **Create configurable products by custom rules (the number of characters from the beginning of the line)** – allows assigning simple products using SKU (or any dedicated column), where the extension reads first N number of characters from the beginning of the line).
- **Create configurable products by custom rules (the number of characters from the ending of the line)** – allows assigning simple products using SKU (or any dedicated column), where the extension reads first N number of characters from the beginning of the line).
- **Attribute / column name on file** – allows setting up the column, which defines Configurable Product Condition.
- **Delimiter** – allows specifying a delimiter value, which defines Configurable Product Condition.
- **Count Symbols** – allows specifying a number of symbols the extension should read considering Configurable Product Condition.

Next comes the **Product attributes for variations** table, where the store owner selects the product attributes, available in the import table, that the extension should also consider when assigning simple products to a configurable one.

For example:

If several simple products have same attributes: *size* and *color*; and same Configurable Product Condition values, then a configurable product will be created. If some of the attributes differ, then some of the simple products with different attributes, will not be assigned to the configurable one.

Copying simple product attributes to configurables

When creating configurable products using custom logic you don't have to have a separate row with the configurable products. Instead you can only import simple products and configurable product will be created automatically according to the set settings.

However, such configurable products will only have basic (required) attributes filled in, such as: attribute set, name, sku and url. In other words, such products will work however won't have any description, images or other media. To avoid such blank and empty products you can use **Product attributes to copy value** grid of the custom logic.

Product attributes to copy value(Carefully choose the system attributes otherwise config product will not be created correctly)

System Attribute

description (Description)

▼





image (Base)

▼



Add New

In this grid you specify the attributes of the simple products which should be copied to the automatically created configurable product. Please note, that only the attributes of the first simple product in the file will be copied to the automatically created configurable.

For example if you are importing a table that looks like this:

	A	B	C	D	E	F	G	H
1	sku	group	attribute_set_code	product_type	categories	name	description	short_description
2	NEW-S-Gray	NEW	Default	simple	Default Category//	Test Configurable	<p>This is a test simple product that I <p>That has been imported using Fire	<p>This is a test simple proc
3	NEW-S-Green	NEW	Default	simple	Default Category//	Test Configurable	<p>This is a test simple product that I <p>That has been imported using Fire	<p>This is a test simple proc
4	NEW-S-Purple	NEW	Default	simple	Default Category//	Test Configurable	<p>This is a test simple product that I <p>That has been imported using Fire	<p>This is a test simple proc

The attributes of the uppermost simple product that you specify in the grid will be copied to the configurable product. This is really useful if you want to copy meta data or product description and other.

Importing different product types

When importing products to Magento 2 you should consider that different product types, such as configurable, bundle, grouped, require special attributes in the import table.

To get you acquainted with different product types and provide you with the complete set of instructions on how to compose the import table we have created a set of posts, each covering a particular product type:

- [The Complete Guide to Magento 2 Product Import / Export](#)
- [The Complete Guide to Magento 2 Configurable Products, and how to import them](#)
- [The Complete Guide to Magento 2 Grouped Products, and how to import them](#)
- [The Complete Guide to Magento 2 Bundle Products, and how to import them](#)
- [The Complete Guide to Magento 2 Downloadable Products, and how to import them](#)
- [The Complete Guide to Magento 2 Virtual Products, and how to import them](#)

Read these posts to master every product type and make sure you can handle complex import procedures.

This is the end of chapter 'How to import products', please follow the instructions carefully and read through additional materials provided above. Product import is a complex task, however, master it and you save a great deal of time on product management.

How to import categories

With the help of Improved Import and Export extension you can now import product categories in a separate file. To import categories to your Magento 2 store you will need:

- A CSV, XML, XLSX, ODS or Json table formatted to fit Magento 2 requirements. You can find a table with sample data and attribute description it in the [Master Google Sheet](#).
- Follow the instructions in the dedicated post [How to Export and Import Magento 2 Categories](#).

The Magento 2 category import guide is pretty extensive that's why it deserves a separate blog post.

How to import product position in categories

Improved Import and Export extension for Magento 2 allows importing product position for categories. This includes Magento 2 Commerce Visual Merchandiser functionality. To import product position in categories you will need:

- A CSV, XML, XLSX, ODS or JSON table with products complemented with custom attribute ***categories_position***. The use of this custom attribute is explained in the articles below.
- Follow general product Firebear import procedure on product import. The instructions can be found in the dedicated [How to Export and Import Magento 2 Categories](#) post, or in [How to Import & Export Magento 2 Visual Merchandiser Data](#) post.

Remember to check Firebear Blog for updates regularly. We are enhancing the import and export procedures often.

How to import customers

The extension can also import and export customers. Actually, customers import is conveniently split into three entities:

- Customers and addresses ([get sample import table](#));
- Customers main file ([get sample import table](#));
- Customer addresses ([get sample import table](#)).

To import customers to your Magento 2 you will need:

- A CSV, XML, XLSX, ODS or Json table formatted to fit Magento 2 requirements. You can find the sample files by the links 'get sample import table' in the item above.
- Follow the instructions in the dedicated post [How to Import Customers & Customer Addresses to Magento 2](#).

The Magento 2 customer import guide is pretty extensive that's why it deserves a separate blog post.

How to import advanced pricing

Advanced Pricing is the tier prices that can be applied to the products. The tier prices allow you to apply discounts depending on how many products, or product variations a customer belonging to a particular group purchases.

To import advanced pricing you will need:

- A sample file with a complete table of Advanced Pricing. You can get one in the [Google Sheet Master Table](#).
- Follow the instructions in the dedicated post [How to import Magento 2 Advanced Pricing](#).

NOTE:

With Improved Import and Export extension you can import advanced pricing along with the products. To do this - add advanced pricing attributes to the import table you have composed for your products and run Import Job with **Entity = Products**. More information can be found in the dedicated post by the link above.

How to import CMS pages

2.1.2 version of Improved Import and Export extension brings import of the CMS pages. To import CMS pages to your Magento 2 you will need:

- A CSV, XML, XLSX, ODS or Json table formatted to fit Magento 2 requirements. You can find the sample files with all attribute description and sample values at the [Google Sheet Master Table](#).
- Follow the instructions in the dedicated post [How to Import CMS Pages to Magento 2](#).

The Magento 2 CMS Pages import guide is pretty extensive that's why it deserves a separate blog post. Note, that CMS pages composed with [Magento 2 Page Builder](#) can also be imported and exported using the extension.

How to import CMS blocks/Static blocks

CMS or static blocks can be also imported to and exported from Magento 2 store. To import CMS blocks to your Magento 2 you will need:

- A CSV, XML, XLSX, ODS or Json table formatted to fit Magento 2 requirements. Sample files and tables can be found either in the [Google Sheet Master Table](#) or at [Firebear Studio GitHub](#).
- Follow the instructions provided in the dedicated post [How to import and export CMS/Static Blocks to Magento 2](#)

Instructions on how to import and export static blocks would extend the Improved Import and Export manual even further, that's why we have put them into a dedicated article. Note, that CMS blocks composed with [Magento 2 Page Builder](#) can also be imported and exported using the extension.

How to import cart price rules

Starting with version 2.1.2 the extension introduces basic Magento 2 Cart Price Rules import. Now you can import coupon codes to Magento 2 store, specify discount amount, customer groups and other Cart Price Rule settings.

To import coupons to Magento 2 you will need:

- An import table in a CSV, XML, XLSX, ODS or Json format with properly placed cart price rule values. You can find a sample table here [Google Sheet Master Table](#).
- Read through the instructions in the dedicated post on [how to import coupons to Magento 2](#).

The Magento 2 Cart Price Rule import guide is pretty extensive that's why it deserves a separate blog post.

How to import orders

2.2.0 version of Improved Import and Export extension brings import of the orders.

To import orders to your Magento 2 you will need:

- A CSV, XML, XLSX, ODS or Json table formatted to fit Magento 2 requirements. You can find the sample files with all attribute description and sample values at the [FireBear Studio GitHub](#).
- Follow the instructions in the dedicated post [How to Import orders to Magento 2](#).
- Find sample [order import table with ONLY REQUIRED attributes](#) at the Google Sheet, and [FULL attributes order sample table](#).

Putting all information about Magento 2 order import in this guide would increase its already solid size. So take full advantage of the dedicated post whenever you need to import orders to your Magento 2 store.

How to import product and custom attributes

You can now import product and custom attributes to your Magento 2.

To import attributes you will need:

- A CSV, XML, XLSX, ODS or Json table with properly added attributes. You can find sample table with every attribute described at the [Google Sheet Master Table](#).
- Follow instructions in the dedicated post [How to import Magento 2 product and custom attributes](#).

The manual is pretty huge to put instructions in it.

How to import Magento 2.3 MSI entities: stock sources and product qty per source

Magento 2.3 has been released with free MSI extension that introduces multi-source inventory functionality to your Magento 2 store. Firebear Improved Import and Export extension already supports this functionality and even more.

To import MSI entities you will need:

- A sample file/table with the stock sources/warehouses or stock source qty. You can get sample tables from the [Google Sheet Master Table](#) or from [Firebear GitHub](#).
- Follow the instructions in the dedicated blog article [How to import and export Magento 2 MSI and how it works. Magento 2 multi source inventory \(MSI\) import guide and manual](#).

MSI functionality is pretty extensive and deserves a separate manual.

How to import Magento 2 related products, up-sells, and cross-sells

All related product types are imported along with products. However, for every related product type you need to use a dedicated column (product attribute).

To import related products you will need:

- A sample file/table with the products that contains related, up-sells, and cross-sells. You can get sample tables from the [Google Sheet Master Table](#) or from [Firebear GitHub](#).
- Follow the instructions in the dedicated blog article [The Complete Guide to Magento 2 Related Products, Up-sells, and Cross-sells, and how to import them](#).

Importing and exporting products may seem confusing if you are only starting to work with Magento, that's why we have dedicated a separate guide to cover any inquiries.

How to import tracking codes and shipping for orders

All tracking codes and shipping information is imported along with orders. To import tracking codes and shipping information you will need:

- A sample file/table with the tracking codes and shipping details. You can get sample tables from the [Google Sheet Master Table](#).
- Follow the instructions in the dedicated blog article [How to Import Shipping Tracking Number to Magento 2](#).

Importing and exporting products may seem confusing if you are only starting to work with Magento, that's why we have dedicated a separate guide to cover any inquiries.

How to import Magento search terms and synonyms

Search terms and synonyms help you to improve customer experience when it comes to searching for products at your Magento 2. To import search terms and search synonyms you will need:

- A sample file/table with the [search synonym groups](#) and [search terms](#) from Google Sheet Master Table. Or get samples from [Firebear GitHub](#).
- Follow the instructions in the dedicated blog article [Magento 2 search terms and synonyms, and how to import and export them](#).

Importing and exporting products may seem confusing if you are only starting to work with Magento, that's why we have dedicated a separate guide to cover any inquiries.

How to import product position inside of a category

Visual Merchandize or product position inside of a category helps you promote products or build nice category layout. To import product position you will need:

- A sample file/table with the [product position inside category attribute](#) from Google Sheet Master Table. Or get samples from [Firebear GitHub](#).
- Follow the instructions in the dedicated blog article [How to import and export position of the product in the category. Magento 2 visual merchandise](#).

Importing and exporting products may seem confusing if you are only starting to work with Magento, that's why we have dedicated a separate guide to cover any inquiries.

How to import Magento 2 page hierarchy

Page hierarchy lets you compose the navigation menu for your CMS/Static Pages. To import page hierarchy you will need:

- A sample file/table with the [page hierarchy](#) from Google Sheet Master Table. Or get samples from [Firebear GitHub](#).
- Follow the instructions in the dedicated blog article [Magento 2 Page Hierarchy Exploration: Features, Configuration, Import & Export](#).

Importing and exporting page hierarchy may seem confusing if you are only starting to work with Magento, that's why we have dedicated a separate guide to cover any inquiries.

How to import Magento 2 product reviews

Product reviews let you enhance the trustworthiness of the products by allowing customers to leave share their opinion.. To import product reviews you will need:

- A sample file/table with the [product reviews](#) from Google Sheet Master Table. Or get samples from [Firebear GitHub](#).
- Follow the instructions in the dedicated blog article [How to Import & Export Product Reviews in Magento 2](#).

Importing and exporting page hierarchy may seem confusing if you are only starting to work with Magento, that's why we have dedicated a separate guide to cover any inquiries.

How to import Magento 2 URL rewrites

URL rewrites let you healthy store redirects. To import URL rewrites you will need:

- A sample file/table with the [URL rewrites](#) from Google Sheet Master Table. Or get samples from [Firebear GitHub](#).
- Follow the instructions in the dedicated blog article [How to Import and Export URL Rewrites in Magento 2](#).

Importing and exporting Magento 2 product reviews seem confusing if you are only starting to work with Magento, that's why we have dedicated a separate guide to cover any inquiries.

How to import Magento 2 Commerce gift cards

Gift Cards, additional product type in Magento 2 Commerce, let your customer purchase gift cards for them or their friends. To import Gift Cards you will need:

- A sample file/table with the [Gift Cards](#) from Google Sheet Master Table. Or get samples from [Firebear GitHub](#).
- Follow the instructions in the dedicated blog article [How to Import & Export Magento 2 Commerce Gift Cards](#).

Importing and exporting Magento 2 gift cards seem confusing if you are only starting to work with Magento, that's why we have dedicated a separate guide to cover any inquiries. **NOTE:** when importing or exporting gift cards - use 'Products' entity, as gift cards count as a separate product type. [More information in the article](#).

Running the Job

Now that we have covered all the Import Job settings it is high time to run the job.

To run the job from the Magento 2 admin, previously, this was possible only via CLI, hit 'Save & Run' button.

DASHBOARD

SALES

CATALOG

CUSTOMERS

MARKETING

CONTENT

REPORTS

STORES

SYSTEM

FIND PARTNERS & EXTENSIONS

New Job

← Back

View History

Save and Continue Edit

Save & Run

Save Job

General Settings

Enable Job

☒ Yes

Job Title *

Frequency *

None (manual run only) ▼

Cron Schedule

Minutes

Hours

Days

Months

Days of Week

↑ Use this field if you have good cron knowledge.

Locale *

English (United States) / English (United St) ▼

When the import is complete, the system will inform you whether the process went successfully or not. The detailed history log is also available on the appropriate page. In case of unsuccessful import, you will also see a detailed log with errors that are to be fixed:

Run

×

Run

ERROR

Console:

```

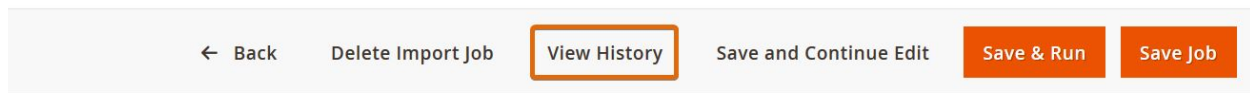
Checked column 73
Finish checking columns
Errors count: 0
Start saving bunches
Finish saving bunches
Data validation failed. Please fix the following errors and upload the file again.
Invalid value in Website column (website does not exist?) in row(s): 0
Checked rows: 1, checked entities: 1, invalid rows: 1, total errors: 1
Import job #2 failed.
Invalid value in Website column (website does not exist?) in rows: 0
Oct 18, 2017, 3:05:58 AM
Data validation is failed. Please fix errors and re-upload the file..
1. Invalid value in Website column (website does not exist?) in rows: 0
2. Data validation is failed. Please fix errors and re-upload the file.. in rows: 1
Job #2 was generated successfully in 2 seconds

```

Starting from version 2.1.2 the log is also saved to `var/log/firebear` folder at your store FTP.

Job history

Another important aspect of the Improved Import and Export Magento 2 extension is a job history which is available for both import and export jobs. You can find it on each job screen. Open the job and click the 'View History' link.



A new screen will be displayed. A grid with all runs is available there. You can download a log of each run, as well as view its type and start/finish time.

History ×

Filters

Default View

Columns

1 records found

20 per page

1 of 1

<input type="checkbox"/>	ID	Type	Start	Finish	Log
<input type="checkbox"/>	446	admin	Dec 29, 2017 8:02:29 AM	Dec 29, 2017 8:02:30 AM	Download

Add new Export Job

To create new Export Job hit the big orange 'Add New Job' button – you will be transferred to a new screen, which is divided into 4 sections: General Settings, Export Settings, Export Behavior, and Export Source. Let's describe each one.

General settings

Here, you can enable/disable the job. Depending on the selected option, the job status is changed from 'Enabled' to 'Disabled' or vice versa.

Next, type a job title. We recommend you to use an expressive title that describes your new job. This will help you find the job in the grid or understand what jobs are enabled/disabled.

Now, it is necessary to configure the frequency of your export job. Although, you can always create a custom schedule (select the 'Custom' option in the 'Frequency' field), it is

recommended to use the 'Cron Schedule' field (the field where you create a custom schedule) only if you understand cron well and have some previous experience in configuring this setting.

Alternatively, it is possible to select one of the predefined schedules:

- None (manual run only)
- Every minute
- Every hour
- Every day at 3:00am
- Every Monday at 3:00am
- Every 1st day of month at 3:00am
- Custom

In the end, you are suggested to specify whether the `additional_attributes` column, where Magento 2 likes to put custom attributes such as color and size, should be split by attribute using **Divide Additional Attributes** setting.

General Settings

Enable Job ☐ No

Job Title *

Frequency * None (manual run only) ▼

Cron Schedule

<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
Minutes	Hours	Days	Months	Days of Week

↑ Use this field if you have good cron knowledge.

Locale * English (United States) / English (United States) ▼

Divide Additional Attributes ☐ No

- If **Divide Additional Attributes** option is disabled - all custom attributes you have added to the product will be put in a single column of the exported file. This can be useful if you are not intended to edit these attributes.
- If **Divide Additional Attributes** option is enabled - all custom attributes will be placed in the dedicated columns making the attribute management much easier, however, taking some of the space of the table.

The first section of the export job configuration page is now complete. You know how to change job statuses and create cron schedules.

Running export jobs on Magento 2 system events

In addition to the cron schedule the export jobs can be automatically executed on certain Magento 2 system events.

Cron Schedule

Minutes Hours Days Months Days of Week

↑ Use this field if you have good cron knowledge.

Events

- catalog_product_save_after
- sales_order_save_after
- checkout_submit_all_after

Currently the events that can trigger the export job execution include:

- **catalog_product_save_after** - the job will trigger after any product in the catalog is saved;
- **sales_order_save_after** - the job will trigger after you add a comment to the order, send shipment or proceed with any action with the order;
- **checkout_submit_after_all** - the job will trigger after the order is created from either store or administrator panel.

You can select multiple events to let the jobs execute whenever you feel necessary.


Export settings

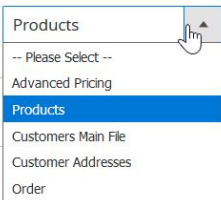
Now, it is necessary to proceed to the second section of the configuration page – Export Settings. Here, you can select an entity that will be exported. Currently, the extension allows you to export the following entities:


- Advanced Pricing
- Products
- Customers Main File
- Customer Addresses
- Orders
- Categories


As you can see, the first 4 entity types are available in the default export procedure. The Improved Import / Export Magento 2 extension adds two new one – Orders and Categories. Thus, you can easily export all orders and categories automatically according to the predefined

schedule. Let's describe how to configure other export job sections on the example of the 'Order' entity type. Select 'Order' in 'Entity'.

Export Settings 

Entity * 


Export Behavior 


Export Source 

Export behavior (CSV, XML Export)

This section allows selecting a file format. You can export orders and related data via the standard CSV format or choose XML, XLSX or ODS. Note that you can use XML files of the following format: [Magento 2 Import Export Sample XML Files](#). You can always find up-to-date sample import/export files in all formats at Firebear Github [Import Export Sample Files for Magento 2](#).

Next, specify field separator, multiple value separator, and field enclosure. Characters you specify there depend on system requirements of the platform you are going to export your order data to.

Export Behavior 

File Format * 

Next, choose entities that will be included into a CSV file you are going to export. As we have selected 'Order' we have the following entities:

- Order;
- Items of Order;
- Addresses;
- Payments;
- Shipments;
- Item of Shipment;
- Invoices;
- Item of Invoice;
- Credit Memos;
- Items of Credit Memo.

- Note that some items may not be compatible.

Export source

While selecting an export source, the extension offers 3 variants: File, FTP, and SFTP.

Choose the first one and specify a file path – your export source configuration is complete.

Date Format - allows you to specify the date format in PHP which will be automatically added to the files you export. For example, if you submit *Y-m-d-hi* the date added to the file will look like 2018-31-12-1031. [Check available PHP date values](#).

Export Source 



Export Source *


File Path *

Use relative path to Magento installation, e.g. var/import/ (Make sure folder have correct write permissions - 775 , for more details please see Magento 2 Permissions guide)

Date Format

Date format to attach at end of the file. Enter php date format default is 'Y-m-d-hi'.

In case of FTP or SFTP, it is also necessary to specify Host, Port, Username, and Password. You can always check whether the provided parameters are correct by clicking the 'Check Connection' button.

Export Source * 

File Path *

The file must match the format.

Host *

Port *

Username *

Password *

[Check Connection](#)

This was a basic export job configuration. Now, you can click 'Save & Run' to test the new job. If exported file has a structure that differs from the default export file, then apply mapping. If it satisfies the requirements of your external system, start export right now. Below, we describe this process in a more detailed manner.

Map attributes

In this section, you can export fields specified in mapping only or all entity-related attributes. Turn the appropriate feature on to get a file with selected options only.

Map Attributes 

Only fields from Mapping ☐ No

Mapping feature allows you to set the conformity between custom data structure and Magento 2 fields. Entity - source of system attribute; System Attribute - select system attributes that should be mapped to custom attributes; Export Attributes - specify custom attributes; DefaultValue - add hard-coded values if necessary.



Entity	System Attribute	Export Attribute	Default Value
Add New			

To do mapping, follow this step by step guide:

- 1) Hit the 'Add New' button;
- 2) Select an entity enabled in the 'Export Behavior' section ('Entity' column);

- 3) Select a system attribute that should have a custom name in the export file ('System Attribute' column);
- 4) Specify a custom name for the attribute selected in the previous step ('Export Attribute');
- 5) If necessary, specify a hardcoded or default value for the attribute ('Default Value' column), which will be provided to all items in the attribute column;
- 6) Repeat the previous steps to map more attributes.


Note that you can delete each attribute mapping by clicking the black bin icon and move rows by clicking on the left dotted area and applying drag&drop to the row.

Map Attributes



Only fields from Mapping
☒ Yes

Mapping feature allows you to set the conformity between custom data structure and Magento 2 fields. Entity - source of system attribute; System Attribute - select system attributes that should be mapped to custom attributes; Export Attributes - specify custom attributes; DefaultValued - add hardcoded values if necessary.

Entity	System Attribute	Export Attribute	Default Value
<div>Add New</div>			

Filters


Filters

Now when you know how to apply mapping and use hardcoded (default) values, we'd like to describe how filters work. Follow these steps to master the feature:

- 1) Click the 'Add Filter' button;
- 2) Select an entity enabled in the 'Export Behavior' section ('Entity' column);
- 3) Select a system attribute that will be used as a basis of a new filter ('Field' column);
- 4) Specify parameters that will be applied for filtering ('Filter' column).
- 5) Repeat the previous steps to create more filters.

Note that parameters applied for filtering vary, depending on the attribute chosen (field).

Filters allow you to import / export only specific entities which match configured filter conditions.

Entity	Field	Filter
<div>Add Filter</div>		

It is possible to delete each filter by clicking the black bin icon. To move rows, click on the left dotted area and drag&drop the row.

Now, you can save and run the new job as well as only save it. The appropriate orange buttons are available on the top of the screen.

The process of running Export Jobs is similar to that of Import Jobs described previously.

Run Import/Export Jobs Via CLI

Import cron jobs can be enabled, disabled, or started from the Magento 2 CLI interface. In the Magento 2 root folder, run the following command:

```
php -f bin/magento import:job:COMMAND JOB_ID
```

import:job:disable *Disable Firebear Import Jobs*

import:job:enable *Enable Firebear Import Jobs*

import:job:run *Generate Firebear Import Jobs*

NOTE:

If you run '**import job:run**' command without specifying the Job ID - **ALL enabled jobs will be executed**.

You will see the following information in the terminal:

```
vitaliy@vitaliy:/var/www/vitaliy/beta % php bin/magento import:job:run 1
Entity customer
Begin data validation
The validation is complete.
Checked rows: 1, checked entities: 1, invalid rows: 0, total errors: 0
Import data validation is complete.
email: jondoe@example.com .... 0.00271s
Checked rows: 0, checked entities: 1, invalid rows: 0, total errors: 0
The import was successful.
```

This is how everything works:

```
ted
  setup:install                Installs the Magento application
  setup:performance:generate-fixtures  Generates fixtures
  setup:rollback              Rolls back Magento Application codebase, media and database
  setup:static-content:deploy  Deploys static view files
  setup:store-config:set      Installs the store configuration. Deprecated since 2.2.0. Use config:set i
instead
  setup:uninstall             Uninstalls the Magento application
  setup:upgrade               Upgrades the Magento application, DB data, and schema
store
  store:list                  Displays the list of stores
  store:website:list          Displays the list of websites
theme
  theme:uninstall             Uninstalls theme
varnish
  varnish:vcl:generate        Generates Varnish VCL and echos it to the command line
vitaliy@vitaliy:/var/www/vitaliy/beta % php bin/magento import:job:run 1
Entity customer
Begin data validation
The validation is complete.
Checked rows: 1, checked entities: 1, invalid rows: 0, total errors: 0
Import data validation is complete.
email: jondoe@example.com .... 0.00271s
Checked rows: 0, checked entities: 1, invalid rows: 0, total errors: 0
The import was successful.
vitaliy@vitaliy:/var/www/vitaliy/beta % php bin/magento import:job:run 1
Entity customer
Begin data validation
The validation is complete.
Checked rows: 1, checked entities: 1, invalid rows: 0, total errors: 0
Import data validation is complete.
email: jondoe@example.com .... 0.00237s
Checked rows: 0, checked entities: 1, invalid rows: 0, total errors: 0
The import was successful.
vitaliy@vitaliy:/var/www/vitaliy/beta %
```

To control export jobs via CLI, use the following commands:

```
php -f bin/magento export:job:COMMAND JOB_ID
```

```
export:job:disable
```

Disable Firebear Export Jobs

`export:job:enable` *Enable Firebear Export Jobs*

`export:job:run` *Start Firebear Export Jobs*

You will see the following information in the terminal:

```
vitaliy@vitaliy:/var/www/vitaliy/beta % php bin/magento export:job:run 1
Entity catalog_product
Exported 2246 rows.
The export is finished.
Job #1 was generated successfully in 3 seconds
```

Both import and export commands:

export	
<code>export:job:disable</code>	Disable Firebear Export Jobs
<code>export:job:enable</code>	Enable Firebear Export Jobs
<code>export:job:run</code>	Generate Firebear Export Jobs
i18n	
<code>i18n:collect-phrases</code>	Discovers phrases in the codebase
<code>i18n:pack</code>	Saves language package
<code>i18n:uninstall</code>	Uninstalls language packages
import	
<code>import:job:disable</code>	Disable Firebear Import Jobs
<code>import:job:enable</code>	Enable Firebear Import Jobs
<code>import:job:run</code>	Generate Firebear Import Jobs

Extension customization endpoints

Improved Import and Export extension has been developed with customization possibilities in mind. That's why we have left customization endpoints for development agencies and in-house developers. If you are not sure how to customize the extension on yourself - you may want to provide this information to the developer you are working with.

[The customization module can be found at FireBear Studio GitHub](#)

With customization module you can create plugins for Improved Import and Export functionality:

- `customChangeData($data)`: changing data whiling saving

- customBunchesData(\$data): changing data whiling saving bunches

The customization module is constantly updating and gains new features. Make sure to bookmark its page and check back regularly.

Use cases and specific features

Magento 2 Google Sheet Import

Improved Import and Export extension support Google Sheets, meaning you can now work in cooperation with your colleagues on a single Google Sheet, then paste its link to the Import Job and hit Run button. Pretty simple, huh?

Moreover, with the new release Firebear Studio introduce Master Import Table, where we have gathered all product types and attributes available, to get you into import process in a quick and easy way. You can read more about the [Master Table here](#).

Categories can be imported with different behavior - Add / Edit , Delete, and Replace which allows flexible category manipulations.

The screenshot displays the 'New Job' page in the Magento 2 Admin interface. The page is titled 'New Job' and features a sidebar with navigation links: SALES, CATALOG, CUSTOMERS, MARKETING, CONTENT, REPORTS, STORES, SYSTEM, and FIND PARTNERS & EXTENSIONS. The main content area has a header with 'New Job' and buttons for 'Back', 'View History', 'Save and Continue Edit', 'Save & Run', and 'Save Job'. Below the header, there is a 'separated by' field with a dropdown arrow and a note: 'The format of categories column in your CSV is : Root Category/Level1A, Root Category/Level1B'. The 'Import Source' section contains an 'Import File Type' dropdown set to 'CSV' and an 'Import Source' dropdown set to '-- Please Select --'. The footer shows 'Copyright © 2017 Magento Commerce Inc. All rights reserved.' and 'Magento ver. 2.2.0' with links for 'Account Activity' and 'Report an Issue'.

To import data from the Google Sheet File hit 'Save and Run' button.

Tier Price import

With 2.1.1 release Tier Prices can be imported along with the products in the same file. This is now possible with **tier_prices** column. Note, how 'prices' are plural, in comparison with native Magento 2 tier_price column. In other words, if you are using native Magento 2 import, you need two separate files to import tier prices and products.

	A	B	C	D	E	F	G
1	sku	group	tier_prices	This is a special attribute introduced in 2.1.1 version of Improved Import and Export extension for Magento 2.			
2	TST-Conf-Simp-S-Gray	TST-Conf	General,360,0,3,All R	It allows you to import tier prices along with the product import. This should be done separately if you are using native Magento 2 import.			
3	TST-Conf-Simp-S-Green	TST-Conf	General,360,0,3,All R	For percent discount the format of the data is the following: Customer Group, Qty, Price, Percent, Website Another tier.			
4	TST-Conf-Simp-S-Purple	TST-Conf	General,360,0,3,All R	For example, the value: General,360,0,3,All Retailer,540,0,5,All			
5	TST-Conf-Simp-M-Gray	TST-Conf	General,100,50,0,All R	Will get you: 3% discount for group General when purchasing 360 items at any Website. And 5% discount for group Retailer when purchasing 540 items at any Website.			
6	TST-Conf-Simp-M-Green	TST-Conf	General,100,50,0,All R	As you can see, you set the Price here to 0.			
7	TST-Conf-Simp-M-Purple	TST-Conf	General,100,50,0,All R	For fixed discount the format of the data is the following: Customer Group, Qty, Price,Percent,Website Another tier.			
8	TST-Conf-Simp-L-Gray	TST-Conf	General,360,0,3,All R	For example, the value: General,100,10,0,All Retailer,200,9,0,All			
9	TST-Conf-Simp-L-Green	TST-Conf	General,360,0,3,All R	Will get you \$10 price for products purchased in quantity of 100, by group General at any Website. And \$9 price for products purchased in quantity of 200, by group Retailer at any Website.			
10	TST-Conf-Simp-L-Purple	TST-Conf	General,360,0,3,All R	As you can see, you set the Percent here to 0.			
11	TST-Conf						
12	TST-GrpBnd-Simple-1						
13	TST-GrpBnd-Simple-2			default	Default	simple	

The tier_prices column is product specific and allows specifying multiple tier prices for each product. As you know, tier prices can be:

1. A percent discount of the original product price;
2. A flat price value.

Percent discount

For percent discount one should use the following data format in the column:

Customer Group,Qty,Price,Percent,Website|next_tier

For example, the value:

General,360,0,3,All|Retailer,540,0,5,All

Will get you:

3% discount for group General when purchasing 360 items at any Website. And 5% discount for group Retailer when purchasing 540 items at any Website.

As you can see, you set the Price here to 0.

Fixed discount

For fixed discount one should use the following data format in the column:

Customer Group,Qty,Price,Percent,Website|next_tier

For example, the value:

General,100,10,0,All|Retailer,200,9,0,All

Will get you

\$10 price for products purchased in quantity of 100, by group General at any Website. And \$9 price for products purchased in quantity of 200, by group Retailer at any Website.

As you can see, you set the Percent here to 0.

Categories import

Categories can be imported with different behavior, which allows for flexible category manipulations:

- Add/Edit
- Delete
- Replace

The process behind the category import is simple and has no peculiarities, you can find the process described in the 'Add new Import Job' chapter of this manual.

To import categories to Magento 2 you need a file with the following structure: [Download sample CSV files for categories import to Magento 2.](#)

Category import offers different opportunities, so you can set a category position and location:

- Import by name and full path to category (Default Category/Women/Dresses);
- Import by category name & path to category;
- Import by parent category id (parent_id).

CSV file columns:

- *name* – Category name;
- *parent_id* – existing parent category id;
- *url_key* – URL keys of category;
- *description* – category description;
- *is_active* – category Enabled / Disabled;
- *include_in_menu* – include category to main menu on frontend;
- *is_anchor* – Is Anchor – required for displaying layered navigation on category;
- *custom_layout_update* – custom XML layout update for category.

Color swatch values import

With Improved Import and Export extension you can also import values for product color swatches, such as color or product images. To import Magento 2 swatch colors you will need to use custom values for the swatch attribute. Let's try importing colors for the attribute 'color'.

	A	B	C	D	E
1	sku	group	attribute_set_code	product_type	color
2	NEW-S-Gray	NEW	Default	simple	Gray type=1 value=#ffff
3	NEW-S-Green	NEW	Default	simple	Green type=1 value=#afff
4	NEW-S-Purple	NEW	Default	simple	Purple type=2 value=https://firebearstudio.com/blog/wp-content/uploads/2014/10/Firebea
5	NEW-M-Gray	NEW	Default	simple	Gray type=1 value=#ffff
6	NEW-M-Green	NEW	Default	simple	Green type=1 value=#afff
7	NEW-M-Purple	NEW	Default	simple	Purple type=2 value=https://firebearstudio.com/blog/wp-content/uploads/2014/10/Firebea
8	NEW-L-Gray	NEW	Default	simple	Gray type=1 value=#ffff
9	NEW-L-Green	NEW	Default	simple	Green type=1 value=#afff
10	NEW-L-Purple	NEW	Default	simple	Purple type=2 value=https://firebearstudio.com/blog/wp-content/uploads/2014/10/Firebea

At the screenshot above you can see that we have a column with attribute 'color'. If you are familiar with Magento 2 import procedure you may find the values for this column strange. Generally, you just type in the names of the existing attribute values, that for the 'color' attribute would be Gray, Green and Purple.

However, here we have an Improved Import and Export extension dedicated values, that allow you to import colors and images for swatch attribute on the fly. The format is the following:

attribute option name|type=1/2|value=hex color code/direct image link

Where:

attribute option name - is the name of the color swatch attribute value, like Gray, Green or Purple

type=1 - defines if the value for the swatch attribute is the hex color code

or

type=2 - defines if the value for the swatch attribute is the direct image link

value= - the value of the swatch attribute, can be either hex color code, or direct image link

For example, if we have a swatch attribute 'color' with swatches Gray, Green and Purple, the values in the import table will look like:

Gray|type=1|value=#808080

Green|type=1|value=#00ff00

Purple|type=1|value=#551a8b

If you want to set images instead of hex colors, the import table will look like:

Gray|type=2|value=https://firebearstudio.com/blog/wp-content/uploads/2014/10/Firebear_logo_294x84_quer.png

Green|type=2|value=https://firebearstudio.com/blog/wp-content/uploads/2014/10/Firebear_logo_294x84_quer.png

Purple|type=2|value=https://firebearstudio.com/blog/wp-content/uploads/2014/10/Firebear_logo_294x84_quer.png

You can find the sample color swatch import table either at [Google Sheets](#), or at [FireBear Studio GitHub](#).

Product attributes import – create new attributes on the fly during the product import

The idea behind the import of product attributes to Magento 2 on the fly during the product import is pretty simple – instead of having only attribute name on a column with attributes, we place there all the data required to create an attribute.

For instance, attribute set, frontend labels, scopes, etc. You can find a structure sample and a sample CSV file below. Please follow the example carefully and import a CSV file with products first. If it works correctly, you can create a custom attribute column and add new attributes. This will reduce possible issues, debug time, and maintenance time.

The general format of an attribute creation column can be described as follows:

attribute|attribute_property_name:attribute_property_value|...

To separate values use pipe “|” symbol. Note, that the new attribute column should ALWAYS start with ‘attribute’, it allows the extension to understand that this is a new attribute.

Composing new attribute

Let’s say, we want to create an attribute for the product size. Let’s list the values of the attribute, and put a required import values after the hyphen in italics next to them:

1. attribute
2. Attribute code will be ‘size’ - *attribute_code:size*
3. Catalog Input Type for Store Owner will be ‘dropdown’ - *frontend_input:select*
4. Values Required will be set to ‘No’ - *is_required:0*
5. Scope will be set to ‘Global’ - *is_global:1*
6. Default Value of the attribute will be size ‘XS’ - *default_value_text:XS text*
7. Unique Value will be set to ‘No’ - *is_unique:0*
8. Add to Column Options will be set to ‘Yes’ - *is_used_in_grid:1*
9. Use in Filter Options will be set to ‘Yes’ - *is_filterable_in_grid:1*
10. Use in Search will be set to ‘Yes’ - *is_searchable:1*
11. Search Weight will be set to ‘3’ - *search_weight:3*
12. Visible in Advanced Search will be set to ‘No’ - *is_visible_in_advanced_search:0*
13. Comparable on Storefront will be set to ‘No’ - *is_comparable:0*
14. Use in Layered Navigation will be set to ‘Yes’ - *is_filterable:1*
15. Use in Search Results Layered Navigation will be set to ‘Yes’ - *is_filterable_in_search:1*
16. Attribute position in layered navigation will be set to “2” - *position:2*
17. Use for Promo Rule Conditions will be set to ‘No’ - *is_used_for_promo_rules:0*
18. Allow HTML Tags on Storefront will be set to ‘Yes’ - *is_html_allowed_on_front:1*
19. Visible on Catalog Pages on Storefront will be set to ‘Yes’ - *is_visible_on_front:1*
20. Used in Product Listing will be set to ‘No’ - *used_in_product_listing:0*
21. Used for Sorting in Product Listing will be set to ‘Yes’ - *used_for_sort_by:1*
22. Default label will be ‘Size’ - *frontend_label_0:Size*
23. Default label for store ID:1 will be ‘Size’ - *frontend_label_1:Size*
24. Will belong to ‘Default’ attribute set - *attribute_set:Default*

Now we have specified all necessary values for the new attribute. We only need to copy and paste attribute values after the hyphen, and separate them with a pipe delimiter. So the column with the new attribute will be named like this:

attribute|attribute_code:size|frontend_input:select|is_required:0|is_global:1|default_value_text:XS
text|is_unique:0|is_used_in_grid:1|is_filterable_in_grid:1|is_searchable:1|search_weight:3|is_vis
ible_in_advanced_search:0|is_comparable:0|is_filterable:1|is_filterable_in_search:1|position:2|is

_used_for_promo_rules:0|is_html_allowed_on_front:1|is_visible_on_front:1|used_in_product_listing:0|used_for_sort_by:1|frontend_label_0:Size|frontend_label_1:Size|attribute_set:Default

Attribute value reference table

In the table below, you will find each property required for attribute creation in Magento 2 as well as appropriate possible values. As you can see, only attribute set and frontend label are required for creating a new attribute in Magento 2 during the import process. Another quite important attribute property is attribute code which actually defines the type of your attribute (text field, dropdown etc). If it is not specified, the attribute will be created with type XXX

[Download CSV file sample for import product attributes on the fly](#)

Code	Description	Default value	Required	type	Values
attribute_set_code	Attribute Set name of the product. Product can only be assigned to a single attribute set.		Yes	text	For example: Default
frontend_label_0	Default label		Yes	text	
frontend_label_[n]	Default label for specified store Where [n] is the store id			text	

frontend_input	Catalog Input Type for Store Owner		select	text: Text Field textarea: Text Area date: Date boolean: Yes/No multiselect: Multiple Select select: Dropdown price: Price media_image: Media Image swatch_visual: Visual Swatch swatch_text: Text Swatch weee: Fixed Product Tax
attribute_code	Attribute Code This is used internally. Make sure you don't use spaces or more than 30 symbols.		text	
is_required	Values Required	0	yes/no	1, 0
is_global	Scope Declare attribute value saving scope	0	select	0: Store View 1: Global 2: Website
default_value_text	Default Value Used for text attributes		text	
default_value_textarea	Default Value Used for textarea attributes		text	

default_value_date	Default Value Used for date attributes		text	Format: mm/dd/YYYY (04/19/2016)
default_value_yesno	Default Value Used for yes/no attributes		yes/no	1,0
is_unique	Unique Value Not shared with other products	0	yes/no	1,0
frontend_class	Input Validation for Store Owner		select	validate-number validate-digits validate-email validate-url validate-alpha validate-alphanu m
is_used_in_grid	Add to Column Options	1	yes/no	1,0
is_filterable_in_grid	Use in Filter Options	1	yes/no	1,0
is_searchable	Use in Search	0	yes/no	1,0
search_weight	Search Weight		integer	
is_visible_in_advanced_s earch	Visible in Advanced Search	0	yes/no	1,0
is_comparable	Comparable on Storefront	0	yes/no	1,0
is_filterable	Use in Layered Navigation	0	yes/no	1,0
is_filterable_in_search	Use in Search Results Layered Navigation	0	yes/no	1,0

position	Attribute position in layered navigation		integer	
is_used_for_promo_rules	Use for Promo Rule Conditions	0	yes/no	1,0
is_html_allowed_on_front	Allow HTML Tags on Storefront	1	yes/no	1,0
is_visible_on_front	Visible on Catalog Pages on Storefront	0	yes/no	1,0
used_in_product_listing	Used in Product Listing	0	yes/no	1,0
used_for_sort_by	Used for Sorting in Product Listing	0	yes/no	1,0

If you need to update the attribute of the product that is already in the catalog you need to include additional attribute to the table *update_attribute_set*.

- If *update_attribute_set=1* - the attribute set of the existing product will be updated.
- If *update_attribute_set=0* or no attribute at all - the attribute set of the existing product will NOT be updated.

Map and create insufficient attributes

Attribute mapping feature of the Improved Import and Export extension is a multifunctional tool. At the moment there are two general uses:

1. You can use attribute mapping to map the import table with different attribute names. Like, when instead of an SKU attribute you would have # attribute.
2. You can use attribute mapping to add attributes that are not available in the table. For example, if you are importing products, and lack 'product_type' attribute in the table, the product won't be created. Instead of editing the table, you can add this attribute from the Map Attributes section.

If the first use is pretty clear, we will cover how can you create new attributes and assign default values to them.

Let's say, you need to import the product table that looks like this:

	A	B	C	D	E	F	G	H	I
1	sku	attribute_set_code	categories	product_websites	name	description	price	qty	
2	training_bag1	Default	Default Category/	base	A bag	<p>A quite nice l	50	100	
3	training_grips1	Default	Default Category/	base	Pushup Grips	<p>Simple pushu	15	100	
4	training_band1	Default	Default Category/	base	Tone Band	<p>A tone band.	10	100	
5	training1	Default	Default Category/	base	Training Kit	<p>A starter training kit.</p>		0	
6									

You do not have these products at you store yet. And if you look closely, this table is missing a required attribute '*product_type*'. Without '*product_type*' attribute Magento 2 won't create any products, because it will not know which product types to create. So what can be done about it?

Using Improved Import and Export feature of Map Attributes section, you can add this missing attribute '*product_type*' and appoint a default value to it. The only thing you need to do, is to fill the import job the way you always do, and 'Add New Custom' attribute in the Map Attribute section, so it would look something like this:

Map Attributes 



Select A Platform

Select



Validate

System Attribute *	Import Attribute	Default Value
product_type	product_type	downloadable
<div> Add New Add New Custom Reset mapping </div>		

So the process is as follows:

1. Click 'Add New Custom' button to create a new line in the mapping table. 'Add New Custom' allows you to specify the missing attribute value manually.
2. In the 'System Attribute' field select the Magento 2 attribute you are missing. In our case it is '*product_type*'
3. In the 'Import Attribute' field type in the name of the attribute that should be added to the table. Actually, you can think of any name, the extension will map it to system '*product_type*' attribute anyways.
4. In the 'Default Value' field type in the default value of this attribute. Now, '*product_type*' attribute supports: *simple*, *downloadable*, *virtual*, *configurable*, *bundle* and *grouped*

values. We have decided to stick with *downloadable* to create downloadable products. NOTE, that the default value will be applied to all products.

5. Run the job.

The extension will "add" a new column which you are missing to the import table. This way, you will have all necessary attributes to create a product. In our case, we have got four downloadable products.

Actions

▼

4 records found

100

▼





per page

<

1

of 1

>

▼	ID	Thumbnail	Name	Type	Attribute Set	SKU	Price	Quantity	Visibility	Status	Websites
<input type="checkbox"/>	20315		A bag	Downloadable Product	Default	training_bag1	\$50.00	100.0000	Catalog, Search	Disabled	Main Website
<input type="checkbox"/>	20316		Pushup Grips	Downloadable Product	Default	training_grips1	\$15.00	100.0000	Catalog, Search	Disabled	Main Website
<input type="checkbox"/>	20317		Tone Band	Downloadable Product	Default	training_band1	\$10.00	100.0000	Catalog, Search	Disabled	Main Website
<input type="checkbox"/>	20318		Training Kit	Downloadable Product	Default	training1		0.0000	Catalog, Search	Disabled	Main Website

That's it. You have just learned how to create attributes missing in the imported table using 'Map Attributes' section of Improved Import and Export extension.

Configurable Product Import

All the examples in this chapter are written in reference to the [Master Import Table](#) we have composed for your convenience. Before proceeding to creating configurable products during the import process, please, read carefully about the Master Table and its use.

Improved Import and Export extension offers five scenarios for Configurable Product import. These scenarios can be selected in the **Configurable Product Condition** field of the Import Job settings. For more information on the scenarios read **Custom logic for creation of configurable products** chapter of this manual.

Create config product by the same attribute of simple products

This is the first scenario. It allows creating configurable products using the dedicated column for simple products, where the required configurable product SKU is specified.

	A	B	C	D	E	F
1	sku	group	attribute_set_code	product_type	categories	name
2	NEW-S-Gray	NEW	Default	simple	Default Category/V	Test Configurable
3	NEW-S-Green	NEW	Default	simple	Default Category/V	Test Configurable
4	NEW-S-Purple	NEW	Default	simple	Default Category/V	Test Configurable
5	NEW-M-Gray	NEW	Default	simple	Default Category/V	Test Configurable
6	NEW-M-Green	NEW	Default	simple	Default Category/V	Test Configurable
7	NEW-M-Purple	NEW	Default	simple	Default Category/V	Test Configurable
8	NEW-L-Gray	NEW	Default	simple	Default Category/V	Test Configurable
9	NEW-L-Green	NEW	Default	simple	Default Category/V	Test Configurable
10	NEW-L-Purple	NEW	Default	simple	Default Category/V	Test Configurable

As per the GIF image above you can see, that we have nine simple products rows 2-10. The service 'group' column for these products contains NEW - which is an SKU of the configurable product, these simples belong to.

Let's configure an Import job, to import these simple products, and assign them to the newly created configurable product. We will omit the general and other settings and focus solely on **Custom logic for creation of configurable products**.



On ☒ Yes

Create Configurable Products ☒ Yes

Configurable Product Condition

Attribute / column name on file

Product attributes for variations

System Attribute	
<input type="text" value="color (Color)"/>	
<input type="text" value="size (Size)"/>	
<input type="button" value="Add New"/>	

Here we have:

- Enabled custom custom logic for creating configurable products;
- Decided that we want to create new Configurable Products. These products never existed at Magento 2 store, and in the Import Table, however have referenced simple products. You can disable this option, if you have a configurable product with the reference attribute in your import table, or at your product catalog;
- **Configurable Product Condition** is set as per the chapter you are currently reading;
- In the **Attribute / column name on file** we have specified 'group' as per table GIF above. As we use this column to tie simples to configurables. You can create any other column and name it the way you want. Just make sure, to put it into the **Attribute / column name of file** field, when importing the products.
- In the **Product attributes for variations** table we have specified attributes 'color' and 'size' as our simple products have these attributes for swatches. When creating configurable products on your own, - specify the attributes of your simple products.

Now, we can run the job to import these nine simple products. When the import job is processed a new configurable product with these nine simple products assigned to it will be created in your product catalog.

Create configurable products by custom rules (part of a line before/after delimiter)

This is the second and third scenarios, that differ only in that of whether extension should start scan from the beginning or the end of the cell. It allows creating configurable products using the dedicated column for simple products, where the extension should look for similarities in the submitted value before the set delimiter.

1	sku	attribute_set_code	product_type	categories	name	description
2	NEW-S-Gray	Default	simple	Default Category	Test Configurable	<p>This is a test<p>That has bee
3	NEW-S-Green	Default	simple	Default Category	Test Configurable	<p>This is a test<p>That has bee
4	NEW-S-Purple	Default	simple	Default Category	Test Configurable	<p>This is a test<p>That has bee
5	NEW-M-Gray	Default	simple	Default Category	Test Configurable	<p>This is a test<p>That has bee
6	NEW-M-Green	Default	simple	Default Category	Test Configurable	<p>This is a test<p>That has bee
7	NEW-M-Purple	Default	simple	Default Category	Test Configurable	<p>This is a test<p>That has bee
8	NEW-L-Gray	Default	simple	Default Category	Test Configurable	<p>This is a test<p>That has bee
9	NEW-L-Green	Default	simple	Default Category	Test Configurable	<p>This is a test<p>That has bee
10	NEW-L-Purple	Default	simple	Default Category	Test Configurable	<p>This is a test<p>That has bee

As per the GIF image above you can see, that we have nine simple products rows 2-10. The 'sku' column for these products contains similar product SKU, which start with NEW and then list product size and color, after a hyphen. In this case the hyphen is a delimiter, that separates one part, from another.

Let's configure an Import job, to import these simple products, and assign them to the newly created configurable product. We will omit the general and other settings and focus solely on **Custom logic for creation of configurable products**.

On ☒ Yes



Create Configurable Products ☒ Yes

Configurable Product Condition

Attribute / column name on file

Delimiter *

Product attributes for variations

System Attribute	
<input type="text" value="color (Color)"/>	
<input type="text" value="size (Size)"/>	
<input type="button" value="Add New"/>	

Here we have:

- Enabled custom custom logic for creating configurable products;
- Decided that we want to create new Configurable Products. These products never existed at Magento 2 store, and in the Import Table, however have referenced simple products. You can disable this option, if you have a configurable product with the reference attribute in your import table, or at your product catalog;
- **Configurable Product Condition** is set as per the chapter you are currently reading;
- In the **Attribute / column name on file** we have specified 'sku' as per table GIF above. As we use this column to tie simples to configurables. In this column all listed simple products have the same SKU format - NEW-size-color, where 'NEW' is a common value, that will tie these products to configurable.
You can use any other product attribute column. For example description, which you start or end with the name of a configurable product, these simples belong to, for example description that reads:
This is a Shirt: size X, color Y.
In this case, you can specify colon as a delimiter, and the extension will combine such products by the phrase "This is a Shirt".
- In the **Delimiter** field we enter '-' a hyphen. As it is the hyphen that separates common SKU value NEW from the rest of the SKU.

- In the **Product attributes for variations** table we have specified attributes 'color' and 'size' as our simple products have these attributes for swatches. When creating configurable products on your own, - specify the attributes of your simple products.

Now, we can run the job to import these nine simple products. When the import job is processed a new configurable product with these nine simple products assigned to it will be created in your product catalog.

Create configurable products by custom rules (the number of characters from the beginning of the line)

This is the fourth and fifth scenarios, that differ only in that of whether extension should start scan from the beginning or the end of the cell. It allows creating configurable products using the dedicated column for simple products, where the extension should look for similarities in the submitted value before the set delimiter.

1	sku	attribute_set_code	product_type	categories	name	description
2	NEW-S-Gray	Default	simple	Default Category	Test Configurable	<p>This is a test <p>That has bee
3	NEW-S-Green	Default	simple	Default Category	Test Configurable	<p>This is a test <p>That has bee
4	NEW-S-Purple	Default	simple	Default Category	Test Configurable	<p>This is a test <p>That has bee
5	NEW-M-Gray	Default	simple	Default Category	Test Configurable	<p>This is a test <p>That has bee
6	NEW-M-Green	Default	simple	Default Category	Test Configurable	<p>This is a test <p>That has bee
7	NEW-M-Purple	Default	simple	Default Category	Test Configurable	<p>This is a test <p>That has bee
8	NEW-L-Gray	Default	simple	Default Category	Test Configurable	<p>This is a test <p>That has bee
9	NEW-L-Green	Default	simple	Default Category	Test Configurable	<p>This is a test <p>That has bee
10	NEW-L-Purple	Default	simple	Default Category	Test Configurable	<p>This is a test <p>That has bee

As per the GIF image above you can see, that we have nine simple products rows 2-10. The 'sku' column for these products contains similar product SKU, which start with NEW and then list product size and color, after a hyphen. In this case only first three characters are similar in all nine products.

Let's configure an Import job, to import these simple products, and assign them to the newly created configurable product. We will omit the general and other settings and focus solely on **Custom logic for creation of configurable products**.

On ☒ Yes



Create Configurable Products ☒ Yes

Configurable Product Condition

Attribute / column name on file

Count Symbols *

Product attributes for variations

System Attribute	
<input type="text" value="color (Color)"/>	
<input type="text" value="size (Size)"/>	
<input type="button" value="Add New"/>	

Here we have:

- Enabled custom custom logic for creating configurable products;
- Decided that we want to create new Configurable Products. These products never existed at Magento 2 store, and in the Import Table, however have referenced simple products. You can disable this option, if you have a configurable product with the reference attribute in your import table, or at your product catalog;
- **Configurable Product Condition** is set as per the chapter you are currently reading;
- In the **Attribute / column name on file** we have specified 'sku' as per table GIF above. As we use this column to tie simples to configurables. In this column all listed simple products have the same SKU format - NEW-size-color, where 'NEW' is first three characters, that will tie these products to configurable.
You can use any other product attribute column. For example description, which you start or end with the name of a configurable product, these simples belong to, for example description that reads:
This is a Shirt size X, color Y.
In this case, you can specify '15' in the Count Symbols option, as first 15 characters will look the same for all nine simple products.
- In the **Count Symbols** field we enter '3' a hyphen. As it is only first three characters of the 'sku' column that are similar in all nine products.

- In the **Product attributes for variations** table we have specified attributes 'color' and 'size' as our simple products have these attributes for swatches. When creating configurable products on your own, - specify the attributes of your simple products.

Now, we can run the job to import these nine simple products. When the import job is processed a new configurable product with these nine simple products assigned to it will be created in your product catalog.

Import downloadable products in Magento 2

Files for downloadable products can be imported in two ways:

1. Import files from remote URL for downloadable products – you should specify a full URL address of the file and make sure the the following directory exists and is writable:
/pub/media/downloadable/links/files/
2. To import local files, upload your files to the directory you've chosen on the import page (input – **Images File Directory**).

Download CSV file example for import downloadable product

Integration with Improved Configurable Products

[Improved Configurable Products extension for Magento 2](#), besides allowing dynamic updates of the product page and URL depending on the variations selected by the customers, allows specifying the configurable product variation that should be selected by default.

I.e. when customer visits the configurable product page you can specify the variation that is pre-selected by default. For example, if this is a shirt product, you can specify that the shirt of size L and color Red should be pre-selected.

When it comes to Improved Import and Export extension, it allows you to specify which variation should be selected as default during the import process. This can be done using the *configurable_variations* attribute.

The *configurable_variations* attribute value is compound, consisting of:

```
sku={{SKU_VALUE}},attribute1={{VALUE}},attribute2={{VALUE}},default=1/0|
```

bundle_shipment_type	configurable_variations	configurable_variation_labels	associated_skus
	sku=TST-Conf-Simp-S-Gray,size=S,default=1 sku=TST-Conf-Simp-S-Green,size=S sku= color=Choose color,size=Choose		TST-Conf-Simp-S-G

Where *default=1/0* defines whether the variation assigned to the configurable product should be selected as default.

- 1 - yes, the variation is default
- 0 or missing default value - the variation is not default.

Use this attribute to your advantage if your store is heavy on configurable products. You can export all configurable products, specify the default value in the configurable_variations attribute, and import this file back. Saving probably a whole day of time managing configurables manually.

CSV, XML, Json, Excel file samples

You can download Magento 2 import CSV, XML, Json, Excel file samples here:

[Download CSV, XML, Json, Excel sample files for import to Magento 2](#)

Upcoming features

Since we always try to improve our Improved Import Export extension following community and merchant needs, here are upcoming features which will be introduced soon.

- Renewed email reporting;
- Integration with IceCat;
- Import from SOAP / REST API;
- Mapping presets management;
- Multiple files and sources input for one job.

After purchasing the extension, you will receive free upgrades during one year and get a 50% discount for the second year upgrades! Purchase the extension right now to use current features and get free updates with advanced functionality!

Compatibility with third-party extensions and modules

FireBear Studio is working hard to make sure Improved Import and Export extension works with the extensions by other development agencies. Below you can find the list of third-party software the extension has been tested with and works without any issues:

- MageWorx Advanced Product Options
- MageStore Inventory Management
- Wyomind Advanced Inventory
- Magedilight Price per Customer

List of detailed import and export guides

The manual offers a general information on how the extension operates and teaches how to use different extension features. For the full list of specialized guides on how to import and export particular entities visit

[Magento 2 import and export guide list](#)

Services and Magento 2 customization

Firebear studio offers its services at your disposal. If you want to customize the extension, perform a store migration or undertake any other custom development task - Firebear developers can assist you with it.

Our developers specialize in Magento 1 and 2, have been working on hundreds of projects for different agencies and merchants. The expertise we offer covers every store building aspect of Magento.

For further information and consultation or to request individual features use [our contact form!](#)