ROCKET WEB

475 River Bend Rd, Ste 200C Naperville IL 60540 USA
Phone: (855) 624 3686
Email: info@rocketweb.com
Support: http://help.rocketweb.com

Google Shopping Feed User's Guide

GSF User's Guide

This guide covers features of the Google Shopping Feed magento extension developed by Rocket Web Inc.

Magento versions

- Enterprise 1.14 latest
- Community 1.9 latest

CURRENT VERSION IS 1.6.4

How to use this guide

This guide assumes that you already have a Google Merchant Account set up, and your Magento Store running on a live or testing environment, set up with minimum catalog data like products and product categories.

If you're just starting out with this product, you should follow the steps at Getting Started; otherwise, if you are looking for details on a specific configuration, use the left side navigation on this page to find the appropriate information.

Troubleshooting

"My feed isn't working!" First steps to take or see our Troubleshooting & FAQs section.

> **Get Support**
> If the solution you are looking for cannot be found here, please submit a request using our Service Desk.

Features

Unencrypted code and multi-language support.

- There are **no encrypted** files

  This makes it possible to modify the code to custom needs. The extension should work fine without changes, but if custom logic is needed we offer technical advice.

- **Multi-language** and **Apparel** support.

  Products are being processed and identified as apparel for any of the google supported target countries:

  - en-US, en-GB, en-AU, es-ES, it-IT, nl-NL, pt-BR, cs-CZ, de-DE, fr-FR, ja-JP, ru-RU, da-DK, no-NO, pl-PL, sv-SE, tr-TR

  Apparels are been identified as products under Apparels & Accessories google category. The extension allows setting google categories against magento categories using a set of rules, making it easier to match products as apparels.

  Allows appropriate information for apparels, like: size, color, age_grop, gender, etc. Those feed columns can have multi-attribute source, so that for example, size column is filled with shirt_size attribute for shirts, and shoe_size attribute for shoes.

For any catalog and automated.

- Allows for **real-time microdata item updates**

  Implements google Automatic Item Updates feature by adding hidden microdata on your products pages to keep your products up to date during the day. Useful for regular promotions and special offers, or low-stock items that sell fast.

- Support for **large catalogs** by processing small chunks at once.

When server memory is limited, the generator can be set to process items in batches, so that it does not run out of memory. Dep ending on the server resources available and complexity of your catalog, it can process 400k products in under one hour.

- Supports all **product types** defined by default in magento, plus AheadWorks Subscription types, and even some of the custom built product types.

  Complex product types like configurable, grouped and bundles can be set to output as single line items in the feed or detailed items, or both.

  Simple products having color and size variations as Custom Options are supported, and those can be set to generate as multiple rows in the feed, detailing all combinations of those options.

- The feed file is **auto-generated nightly**,

  By default the extension uses magento's cron schedules to run each night at 1am. But it also features a **shell script** that can be scheduled to run on a custom schedule free from a more error prone Magento crontab.

- Allows for **customized feed location.**

  The path on the server to save your feed file can be customized so that it's easily accessed. By default the file is web accessible at a public URL but you may easily change to a protected directory.

Versatile column definitions.

- Feed can be configured **differently by store view**, having a different feed file for each one of them.
- Comes with **predefined columns** map, to meet the basic feed requirements, and starting from it, **you can add / remove columns** d epending on the products you sell.
- Supports **cascading map rules.** When products are missing data in any of the column you can define **replace empty** rules to add layers of mappings so that your feed never misses data.
- Feed columns can be **mapped to any attribute** in your catalog, but also to a set of **directives** which bring special logic in collecting product data.

  Some of the important Directives are:
  - Static Value - all products get the same text value.
  - Product Parent Id - useful for apparel associated items.
  - Product category Image URL - when you want to output the image of the category instead of the product image.
  - Toybanana External image URL - when your images are stored outside Magento using a flexible external image extension.
  - Product ID & Product URL special logic to output the correct information of your product as it appears on your public website.
  - Price, Special Price and Special Price Date Range - considers how your promotions are being set in your catalog.
  - Shipping and Shipping Weight - considers your shipping rules and computes the correct price and weight of your products.
  - Variant Attributes - collects data from multiple attributes into one value. Very handy for apparel columns like size, detecting appropriate value through shoe_size attribute for shoes, and shirt_size attribute for shirts.
  - Product Review Count and Average - will compute reviews of your products.
  - Product Type using Magento Category Path - takes all product categories and converts them into a comma separated value list of your category paths, useful to output the product_type column.
  - Identifier Exists - detects if your product has the minimum identifier columns filled in. If you do not have a unique product identifier, this will be set to FALSE
  - Concatenate Attributes - allows to use multiple base attributes in an expression like format to give robust advanced flexibility.

Only the data you want.

- The feed can be set to **filter the catalog** so that you only export the products you want.

  This can be done by: product type, product category, attribute sets, out of stock, and missing product data (empty columns).

- Supports **find and replace** rules which are applied at the feed output to replace any bad column values with correct ones.
- Has built in **data validation** for bad products and duplicates, to assure that only quality data is been included in the feed, or products can optionally be skipped. We will let you include them as well if you plan to do some feed post processing elsewhere.
- You get the chance to **test and troubleshoot** single products before generating the full feed.

Tracking and campaigns

- Has customizable **tracking information** added to product URLs so that you can track your feed campaigns with analytics.
- Allows you to categorize your product by **price buckets,** to helps better manage your feed items through your campaigns.
- Supports **custom labels** to hold data which may help manage your campaigns.

Third Party support

We have adjusted our extension to work with third party extensions like:

- Simple Configurable Products

- AyaSoftware's Simple Product
- Amasty simple configurable price
- Aitoc_Aitcbp Cost-based automatic pricing
- MagicToolbox_Magic360
- Toybanana Extenral Images

In addition we also include support for:

- Weee Eco tax in product pricing
- Magento 1.9.x configurable swatches
- Updates for existing Google Remarketing Tag

Included support.

We can help provide direction to all of your feed related questions at http://help.rocketweb.com/

Continuous development

The extension was designed primarily with Google Shopping in mind, it can generate a single feed file for any type of tab delimited feed file to be consumed by any shopping service in the world. Currently we only let you generate a single feed per store view but we're working hard to enable multiple feeds per store view soon.

Installation

Backing up
This extension will alter your store database by installing new catalog attributes and new tables, so it is best to have a **backup of your database** before installing.

Upgrading from a previous version of this extension may alter your feed configuration to reflect new functionalities. Since your feed configuration is stored in the database, a **database backup** may be handy in this case as well.

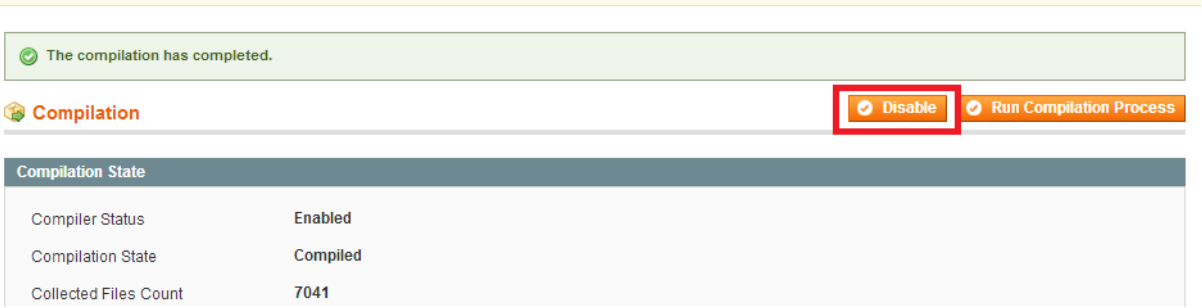
Requirements

- min PHP 5.3.x
- Magento CE > 1.4.2, EE > 1.9.2

Step-by-step guide

Follow these steps to install the extension, or to upgrade from an older version of the extension.

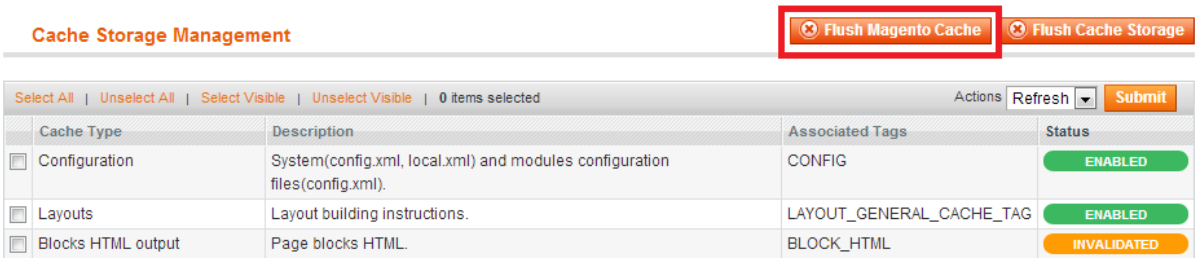
1. **Disable Compilation**
   In the Magento admin panel, go to System >Tools > Compilation. In case Compiler Status is "Enabled", click on the Disable button. This step is for Magento 1.4+ versions, so If you are running an older version, this step can be skipped. In case the status is disabled you can skip this step.



2. **Download and Extract**
   Download and extract (unzip) the extension's contents on you computer. Navigate inside the extracted folder.

3. **Upload files**
   Using a FTP client, upload the content of the extension directory to the store's document root, so that the app directory in the extension folder merges with the app directory on the server. If asked to replace any files, select "Yes".

4. **Clear the cache**
   In the store admin panel, go to System > Cache Management and press Flush Magento Cache button.



Feed Admin

After the extension is installed, you can visit System > Configuration > Google Shopping Feed in the store admin panel to see the main configuration options.

Troubleshooting

1. I don't see Google Shopping Feed in the admin.
   Please verify that you have turned off compilation. If compilation is off, make sure you have uploaded all the files in the store's web root.
2. I get 404 when trying to access the feed's admin.
   Logout & Login back into the shop admin panel to refresh admin permissions that allow access to the new installed section.
3. My site is broken.
   Our extension is significantly modularised and we have never yet seen it break any site when it's fully uploaded and compilation is off. If your site does break, try reverting from a backup and contact us for support though http://help.rocketweb.com

Uninstall

If you do want to completely uninstall, open a shell console to your server and run shell/**gsf_uninstall.sh** script. This script will attempt to remove all extension files and also cleanup your database *(this script is available in versions 1.6.0 and newer).*

Alternatively, you could manually remove all files from your server, by following the archive structure to find out where files are located on your store.

Related articles

- Installation
- What's the difference between SSH and FTP installations?
- Additional Installation resources

Getting Started

Getting your products submitted is a 3 step process and could take several iterations to get right. The amount of effort you need to put in depends on the quality of your catalog data and the amount of errors you'll be getting when Google reviews your feed.

We advise you to review the Products Feed Specification before setting up the feed, in order to streamline this process.

1. Install & Configure

- Follow the Installation steps.
- Set up your Product Categories.
- Set up Shipping and Tax.
- Review the Feed Configuration to better match your catalog content.

2. Generate the Feed

- Generate a test feed and review results.
- Check your automatic update schedule, and adjust it according to your needs.

3. Submit the Feed

- Set up Scheduled feed upload in you Google Merchant Account.
- Set and Improve your AdWords Campaigns

Get Support
If you're having issues with setting up the feed, use our Service Desk to get your questions answered.

Related articles

- Setup microdata to enable automatic pricing updates
- Getting Started
- Feed Configuration
- File Settings
- Submitting your feed to Google

16 related results

Categorize Products in the Feed

Google requires you to use a set of taxonomies to classify your products, and will not accept products without this information. Read more about google requirements on Product Categorization.

Your should already have your products categorized in your shop, so first thing you need to do is to compare your categories with the google taxonomies and see how they would match. Most of the times they don't match so you need to set correspondent values for your magento categories under Product Category By Category Map.

Here's the list of taxonomies for each target country / language available:

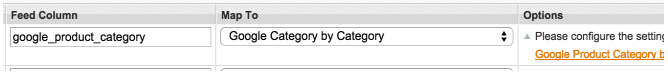
- US: http://www.google.com/basepages/producttype/taxonomy.en-US.txt
- AU: http://www.google.com/basepages/producttype/taxonomy.en-AU.txt
- BR: http://www.google.com/basepages/producttype/taxonomy.pt-BR.txt
- CZ: http://www.google.com/basepages/producttype/taxonomy.cs-CZ.txt
- DE: http://www.google.com/basepages/producttype/taxonomy.de-DE.txt
- DK: http://www.google.com/basepages/producttype/taxonomy.da-DK.txt
- ES: http://www.google.com/basepages/producttype/taxonomy.es-ES.txt
- FR: http://www.google.com/basepages/producttype/taxonomy.fr-FR.txt
- GB: http://www.google.com/basepages/producttype/taxonomy.en-GB.txt
- IT: http://www.google.com/basepages/producttype/taxonomy.it-IT.txt
- JP: http://www.google.com/basepages/producttype/taxonomy.ja-JP.txt
- NL: http://www.google.com/basepages/producttype/taxonomy.nl-NL.txt
- NO: http://www.google.com/basepages/producttype/taxonomy.no-NO.txt
- PL: http://www.google.com/basepages/producttype/taxonomy.pl-PL.txt
- RU: http://www.google.com/basepages/producttype/taxonomy.ru-RU.txt
- SW: http://www.google.com/basepages/producttype/taxonomy.sv-SE.txt
- TR: http://www.google.com/basepages/producttype/taxonomy.tr-TR.txt

> *The taxonomies language you choose, should match the language set on your Feed Localization setting under File Settings .*

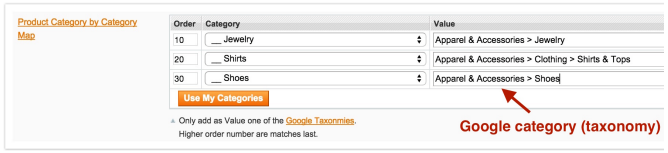
Step by step guide

Google Product Category column:

1. Make sure your Columns Map have a column called google_product_category, and is mapped to Google Category By Category directive.
2. Under Product Category by Category Map setting from Columns Map section, define mapping rules for corresponding taxonomies to your magento categories.
    a. Use the Add Category button to create a new rule, choose a category and fill in the Value with the most suitable google category name from taxonomy file.
    b. If you can't find a suitable taxonomy correspondence, do not add a rule for that category, and relay on replace empty function described below in this page.
    c. Make sure you define deeper categories first, and set low order numbers for them so they have priority. High level categories should be defined with higher order numbers so that they match last.
    d. IMPORTANT: Each rule should have an unique order number, don't add tow rules with same order number as it would overwrite one another.



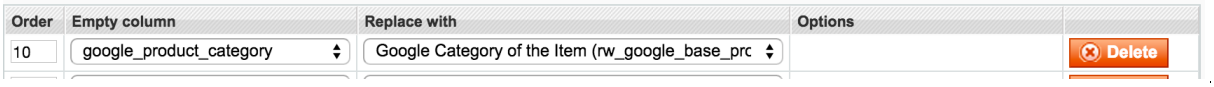
Categories mapping:



A fast approach to fill in the category mappings is to first add a rule for each of your magento category without filling in any Value, than use the button Use My Categories. This would pre-fill the taxonomy values with a text version of your category path. Seeing your full category path pre-filled, it's going to make it easier to look for suitable taxonomies to replace with.

> Take note of category tree inheritance and rule ordering, make sure you understand how rules apply. Each rule should have an unique order number. Check out the Google Category By Category directive.
>
> If you filter out the categories you include in the feed under Include products only from these categories from Product Filters, make sure you map all those categories to google taxonomies here.

Optionally, you can set Replace Empty rules under the Product Filters section to cover any products that might not be assigned to a category in your catalog, or to fill in products that belong to categories missing from Product Category by Category Map.



The replace empty rule can be set to the specially created attribute called Google Category of the Item (rw_google_base_product_categ), which you must fill for those products missing magento categories. Of course this attribute needs to have values from google taxonomies.

Once your mappings are defined, you can **save the configuration** and give it a few test runs by using the individual SKU testing feature, to make sure google_product_category column is been filled correctly.

Google modifies their taxonomies from time to time, so it's a good idea to review your classification few times per year.

Related Articles

- Categorize Products in the Feed
- Columns Map
- Changing the number of levels of categories
- Getting Started
- Feed Configuration

10 related results

Setting up Shipping and Tax

> Both shipping and tax requirements depend on the feed target country, please read Submitting tax & shipping google docs to understand what the requirements are for your feed.

- Tax setup
- Shipping setup

Tax setup

For most of the countries, value added tax needs to be included in the price. This is needed on product pages as well as in the price field in the feed. Exception to this rule apply to US, Canda and India stores.

To see the up to date list of restrictions please read google's Tax Policy.

1. **Including tax in the price** (non-US feeds)
   First make sure your products pages display the price including tax. You should check the setting under Admin > System > Configuration > Sales > Tax > Price Display Settings, should be set to Including Tax.

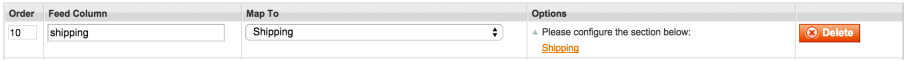
   The feed has to be configured to follow it by setting the price and sale_price columns in the Columns Map to the appropriate Price a nd Sale Price directives, having the option **Add Tax = Yes**.

2. **Explicit tax submission** (US feeds)
   Define tax information by setting it up in the Merchnat Account. Under Settings > Tax define the same tax rules you have running on your store.
   Another option to submit the tax is by creating a custom attribute on your products to hold tax information in google format, i.e. US:94114:8.75, than add the tax column to your feed under Columns Map and mapi it to the custom attribute you created.

Shipping setup

Depending on the complexity of your store shipping definition, you can provide the rates to google in multiple ways.

1. **Basic shipping methods**: flat_rate, table_rates, free_shipping
   If your store is setup with one of those basic rates, you could include the shipping information in your feed by adding the shipping col umn to Columns Map and map it to the Shipping directive. Than turn **ON** the Shipping section in the feed configuration and set the desired methods and regions to be included. The feed will automatically compute the rates based on your shipping methods definitions in magento, and provide the feed with the appropriate google shipping format.



   To understand how this section can be configure, see Shipping guide.
   If for any reason the rates do not come through properly, you can always define them in merchnat center instead of providing them through the feed. See guide on Merchant account-level shipping.

2. **Carrier calculated rates:** UPS, DHS, FedEx
   Those rates cannot be computed through the feed, and those need to be setup in the Merchnat Account under Settings > Shipping. Please follow the guide on Merchant account-level shipping.
   In order for account-level shipping rates to apply on your products, the feed should not include shipping, so make sure you have shipping tuned **OFF** under Shipping section of feed configuration.

3. **Rates by individual product:** i.e. free shipping items
   If your catalog include a set of products that apply for free shipping, and the rest of the products have regular rates or carrier calculated rates, the setup has to be a defined both in your merchnat account as well in your feed configuration as follows:
   a. Create a custom attribute for your products a call it for example custom_shipping_rate, than fill in your free shipping items with US::Free Shipping:0 USD, leaving this attribute blank for the other products.
   b. Map this custom_shipping_rate attribute to the shipping column in the Columns Map.
   c. Set the catalog wide rates in the Merchnat Account, similar to how it's defined for carrier calculated rates, bullet 2.

   This setup has the following effect: the rates defined at bullet c) will apply for all products, but the ones having a value defined in the feed.

   Another option for setting catalog wide rates from bullet c), is when you have basic shipping methods defined in magento and you don't want to define them in the merchnat center. For this case you could fill in the appropriate rates in the feed by setting a replace empty rule under Product Filters section to replace empty shipping values with ones computed through the Shipping directive.

Related articles

- Setting up Shipping and Tax
- Add Tax to Price option
- Shipping
- Adding Free Shipping label

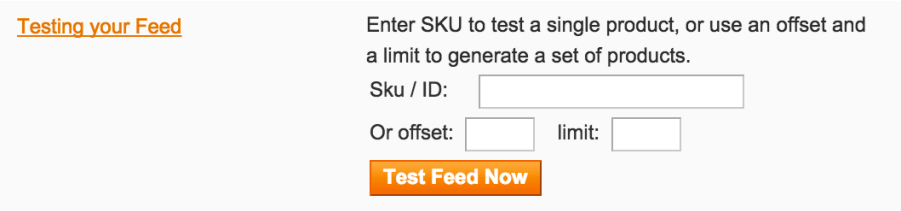
- Getting Started

11 related results

Testing & Generating the Feed

Testing individual SKUs

Quickly assessing a feed configuration change is vital, and this is done pretty easy using the Test Feed Now from the admin.



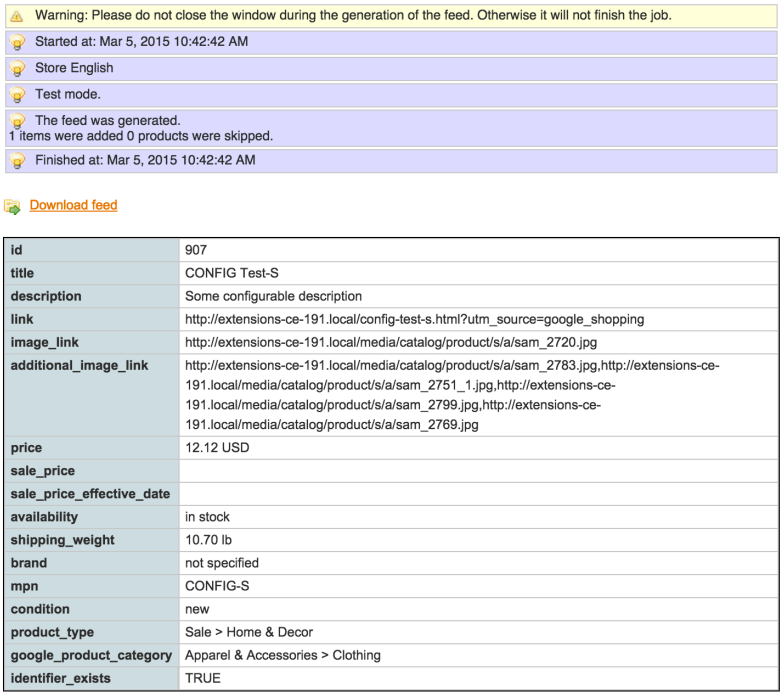
There are 3 additional fields which next to the button, telling which product(s) to test:

- Sku / ID - for testing individual products.
- Offset and limit - for testing a range of products.

When testing individual products, you only have to fill in the Sku/ID field and hit Test Feed Now button.

The resulting column values are printed in a popup window.

> When choosing products to test, use SKUs that are enabled, visible in catalog and assigned to the website your are testing against. If you want to test an apparel variant, use the configurable parent SKU.



id	907
title	CONFIG Test-S
description	Some configurable description
link	http://extensions-ce-191.local/config-test-s.html?utm_source=google_shopping
image_link	http://extensions-ce-191.local/media/catalog/product/s/a/sam_2720.jpg
additional_image_link	http://extensions-ce-191.local/media/catalog/product/s/a/sam_2783.jpg,http://extensions-ce-191.local/media/catalog/product/s/a/sam_2751_1.jpg,http://extensions-ce-191.local/media/catalog/product/s/a/sam_2799.jpg,http://extensions-ce-191.local/media/catalog/product/s/a/sam_2769.jpg
price	12.12 USD
sale_price	
sale_price_effective_date	
availability	in stock
shipping_weight	10.70 lb
brand	not specified
mpn	CONFIG-S
condition	new
product_type	Sale > Home & Decor
google_product_category	Apparel & Accessories > Clothing
identifier_exists	TRUE

Using this button won't write into the main file. It will write in an alternate file with the format: *test_google_base_[store code].txt*

Generating the Feed

There are several ways you can fully generate the feed on demand:

1. The recommended way is by running the included shell script in a SSH console.

shell run

```
php [MAGENTO_ROOT]/shell/gsf_generate.php --verbose
```

Replace [magento_root] with the appropriate directory path. See Custom Crontab System article for more details about the command options.

2. If you don't have SSH access to your server, but you have CPanel access, use the same shell command to set up a cronjob that will run in the next timeframe.
Most of the times you can set an email alert in your crontab utility to deliver the output of crontab commands, and that can be useful to know when the command has completed.

   Alternatively, you could monitor the generator log file to see when the feed is been run.
[magento_root]/var/log/google_base_feed_default.log

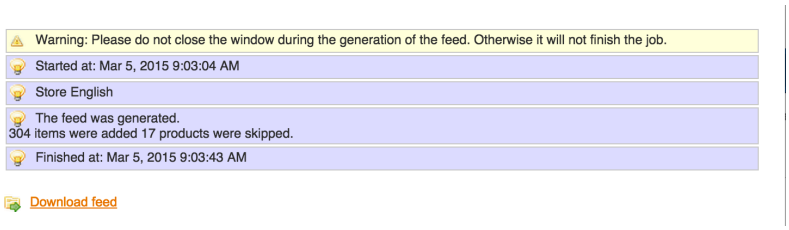
3. The most accessible way of running the feed is through the admin, using Generate Feed Now button. But this button is limited to work only for small catalogs for less than 1000 products. To change the button limit, edit the config xml file:

app/code/community/RocketWeb/GoogleBaseFeedGenerator/config.xml

```
<button_max_products>1000</button_max_products>
```

The Generate Feed Now button will open a new window, displaying result.

> **Debug**
> If for some reasons the new window is blank or throws errors, review your apache / web server error log.



> **Limitations**
> The success of Generate Feed Now button is highly depended on your apache / php server configuration, and most of the time it is necessary to adjust your php max_execution_time, memory_limit and apache TimeOut.

Related articles

- Submitting your feed to Google
- Testing & Generating the Feed
- Making images appear for all products in Test Feed
- Getting Started
- Feed Configuration

13 related results

Submitting your feed to Google

> **Schedule Updates**
> Your file is automatically generated at 1AM each night. That, if your magento install has cron.php running each 5 min or so. For multi-store setup, or if store code is different than "default" you may need to adjust the automatic update schedule.

Once the feed file has generated, access the feed URL in your browser and make sure it is accessible.

> **File access**
> After generating the feed, your file is web accessible at http://your_domain.com/media/feeds/google_base_default.txt, that is if your store code is "default".
>
> Similarly, if your store code is differently named, the file would be google_base_[sore_code].txt

Step By Step guide

Login to Google Merchant Center: http://www.google.com/merchants/merchantdashboard

1. Access **Feeds** and hit the button to add new feed "**+Data Feed**"
2. Select mode: **Standard** for a live feed or **Test** for testing your file and evaluating errors.
3. Select your **target country** which should match the "Feed Localisation" of your feed file.
4. Fill in a feed file name, used to identify your feed in the list. and click **Continue**
5. Select the upload type as been **Automatic upload (scheduled fetch)** and click **Continue**
6. Fill in a file name to upload, for your reference.

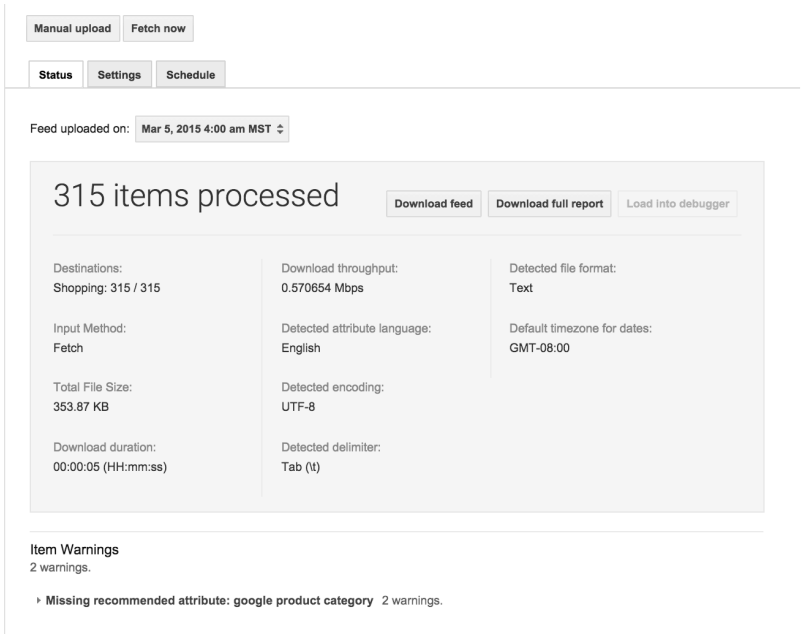
- Fetch frequency: **Daily**,
- Fetch time: **3am**, should be 1-2hrs after the feed file is generated.
- Timezone should be the timezone of your server.
- Feed URL: http://your_domain.com/media/feeds/google_base_default.txt

Hit **Save**

Google will grab the feed file from your server and then fetch it regularly according to the schedule you picked. It will also accept ftp or sftp urls if you'd like to give Google your ftp password and keep the feed in a non web accessible directory.

Validating the feed

To test your feed file, selecting **Test** for the feed mode at step 2. After the feed is set, use the **Fetch Now** button on your feed in google account and wait till the feed file is parsed. It could take few minutes for Google to process your files, so try reloading the page after few moments.



After processing the feed, **Download full report** to see which products did not pass validation. Use that list of products to test individual SKUs in admin, adjust your product data and feed config till you eliminate most of the errors.

Only submit a live feed after most of the errors are cleared.

Fixing errors

When reviewing the error report, a good practice is to take one product of each error type, and test it using Test Feed Now button for running individual SKU and asses the output. Than if the output has indeed wrong values for that product, review Columns Map for that store and than review your product to have all attributes with correct values.

Common errors:

- *Missing values* - Check that the attribute on the product has a value. You can see which attribute to review in the Columns Map.
- *Missing two out of three identification (mpn, brand, gtin)* - usually assuring SKU for mpn column, and a value for brand is enough. Check your products to have the manufacturer / brand filled in.
- *Duplicate items* - you could ignore this for now, it's mostly caused by having same item assigned to multiple complex products and not all of them can be filtered.
- *Missing google_product_category* - review Categorize Products in the Feed, and add missing mapping rules to your categories, or define a Replace empty values rule to cover missing ones with a default value.
- *Missing shipping or tax* - Usually that get's cleared out if you define Shipping and Tax under your merchant center.
- *Encoding issues in description or title* - Assuming data has been filled in in proper character set at your products, a good practice is to also to configure Apache to append UTF-8 in response headers. It has been proved that in many cases this solution worked where special characters were previously not displaying properly in the feed file.
  Add or edit the **.htaccess** file to you feed output folder [magento_root]/media/feeds, and add the following lines to the .htaccess file:

```
AddDefaultCharset UTF-8
AddType 'text/css; charset=UTF-8' txt
```

- *Missing attributes* - Create these attributes and add them to attribute set in magento admin. Also every attribute must have only values, that match google requirements. After creating the attributes you should to add the values of attributes to each product in this category. T hen go to feed configuration and add these attributes to Columns Map block.

Every time you do corrections on your product attributes or in the feed configuration, generate the feed again and download it in merchnat center, to see if the issues have been fixed.

Related articles

- File Settings
- Setup microdata to enable automatic pricing updates
- Categorize Products in the Feed
- Adjust automatic update schedule
- Getting Started

12 related results

Running AdWords Shopping Campaigns

> Shopping campaigns offer a simple and flexible way to organize your Google Merchant Center product inventory within AdWords.
>
> To understand how google campaigns work please follow Setting up shopping campaigns.

> In October 2014 Google converted all Product Listing Ads to Shopping Campaigns; therefore, adwords_labels is deprecated and shouldn't be used any more.

Once your feed is submitted and approved, create an AdWords account and connect it to your Mer chant Account under Settings > AdWords. Doing so, your products should become available for shopping campaigns.

> If your products are approved in Merchant Center but they are not visible in AdWords campaigns, you may need to contact a Google representative to help.

The basic product campaign is to bid the same way for all your products, but you can improve your ads by bidding differently on groups of products.

Edit your campaign in AdWords, and go to Product Groups and click the "+" sign to find out how you can group them. Those groups can be defined using the following columns in the feed:

- **Category** - is the google_product_category column in the feed and also the main way to organize your products into groups. Check the Categorize Products in the Feed guide to understand how that's set in the feed.
- **Brand** - is your brand column in the feed, mapped by default to manufacturer attribute on your products.

- **Item Id** - is your item_id column from the feed and should be mapped by default to your product ID from magento.
- **Condition** - is the condition columns from the feed, by default all products having value "new". If you want to change that, create a new attribute on your products, set the accepted values on your products and map it under Columns Map.
- **Product type** - is the product_type column in the feed, and should be correspond to magento categories. That if, product_type is mapped to Product Type by Category directiv e in the Columns Map.
- **Custom label 0-5** - should correspond to columns in the feed custom_lable_0 to custom_label_5, and allow you to specify custom data on which you can group products. Those columns are not mapped by default in your feed, so if you will need to add them.

If you want to group your products for use in a Shopping campaign, you'll need to setup custom labels in your feed.

Setting up custom labels

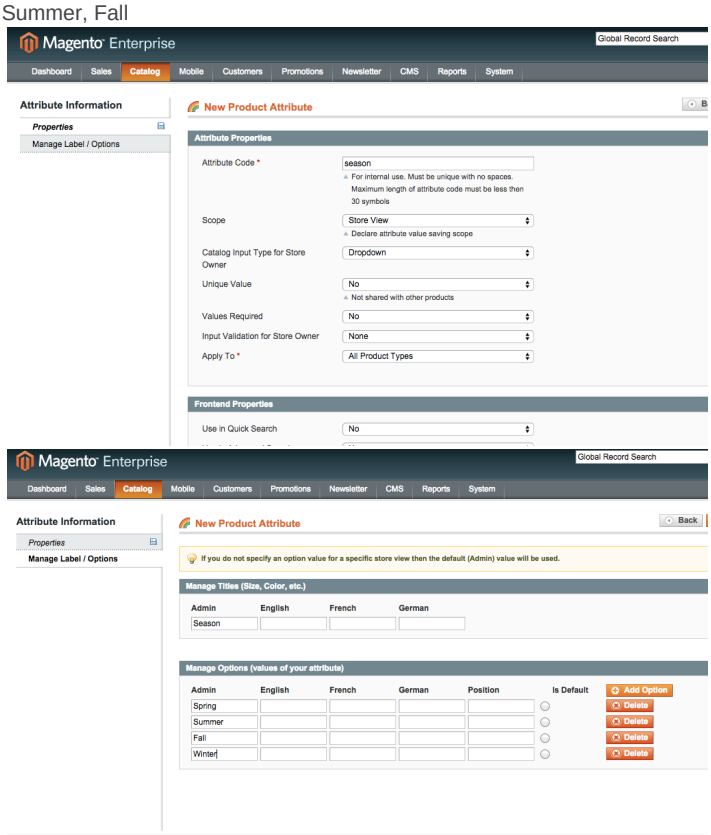
To use custom labels, you'll need the following:

1. **Create new product attributes**
   To create new product attributes in Magento, one for each label. If you already have an attribute, you can use that. If it's not a drop-down, that's ok too, but please be aware of its limits, and make sure it has consistent values, otherwise your grouping won't work.

   > We suggest you make these attributes *dropdowns*, and add the values you intend to use. This will give you one place to manage their values: if you have a typo, instead of editing each product, you just edit the attribute. This will also help staying consistent with Google requirements (see below)

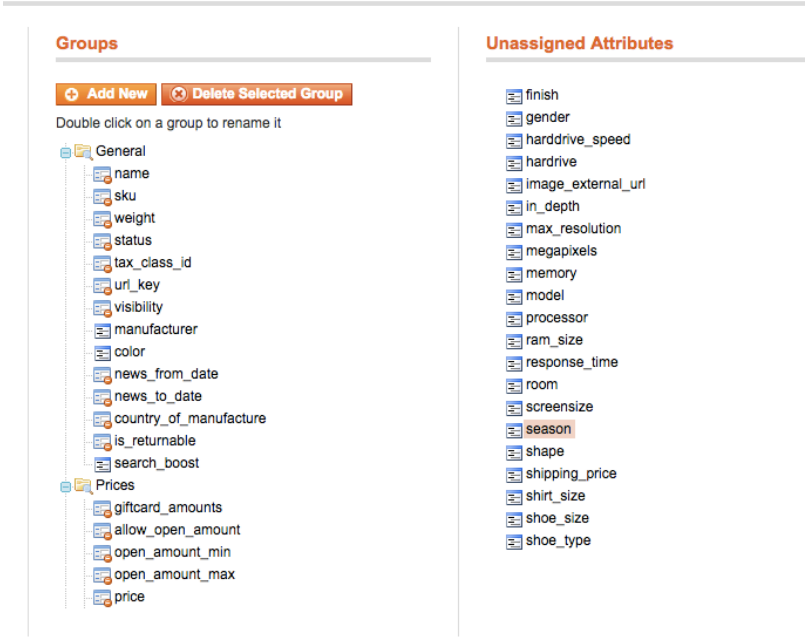
   As an example, let's say you want to add a label named "season" to all your products:

   a. Add a Magento attribute named Season, of type *dropdown*. Add all your label values to the attribute: Winter, Spring, Summer, Fall



   b. Add the Season attribute to all the relevant Attribute Sets. If you don't use Attribute Sets, you will still have to add it to the Default set.

      Go to Catalog > Attributes > Manage Attribute Sets, pick your set from the list, then use Drag-and-drop to take Season from the right, and place it in one of the groups on the left. You'll have to do this for all relevant attribute sets.



After this is done, you can edit your products and fill in Season values.

2. **Add Custom Labels to the feed columns**

   To include the new attributes in the feed as a custom labels:
   a. Go to the feed settings, under Columns Map and add a new column,
   b. Name it custom_label_0, and select the new attribute ("season" in our example). You'll only need to do this once.
   c. Save and you're done! The new feed that generates overnight should include the new attribute.

   You can do those steps for up to 5 custom_labels. The attributes you set can have any value you'd like to use in AdWords.

Google requirements

> *You can create up to five custom labels, numbered 0 through 4, for each item in your feed. You may submit one value per item for each custom label attribute. You should assign a specific definition for each of the five custom labels and specify the possible values for each. Then, you use these custom labels consistently across the products in your Merchant Center account, assigning appropriate values to each product according to your definition. Each of the five custom labels can only have one value per product.*
>
> *Source: https://support.google.com/merchants/answer/188494?hl=en*

Limitations

- You can only have 5 custom labels
- one product can have only one value per label, so in total, a product can only have 5 labels
- and you cannot group more than 1,000 products with the same labels.

> *As custom labels are meant to allow grouping of items, only 1,000 unique values are supported for each custom label attribute across the items in your account. Products with a custom label submitted after the limit was exceeded won't be included in product groups using this custom label. To correct the issue, update your product data to reduce the number of unique values for the custom label to less than 1,000.*
>
> *Source: https://support.google.com/merchants/answer/188494?hl=en*

Setting up Price Buckets

Price buckets is a concept to help target campaign bids by product price ranges. I.e. products with price intervals:

- < $50
- $50 - $100
- $100 - $500
- > $500

To set them up, you would need to define a set of rules under Adwords Price Buckets feed setting and insert a new custom_label_0 column and map it to the Adwords Price Buckets directive in the feed Columns Map

To view how this looks, see the Adwords Price Buckets directive instructions.

Related articles

- Columns Map
- File Settings
- Setup microdata to enable automatic pricing updates
- Categorize Products in the Feed
- Running AdWords Shopping Campaigns

10 related results

Feed Configuration

Preconfigured columns
The extension comes preconfigured with basic feed columns that allows generating a basic feed for your catalog. You'll have to extend this definition by adding new columns in the Columns Map to better match the information available on your products.

For example see Apparels section to find out which columns you should add if your store sells apparels. Similarly look for available columns under Google Shopping Specifications and add them as you see fit.

The feed configuration is split into sections, each dedicated to control specific actions of the catalog processor.

File Settings

Columns Map

Directives

Product Filters

Product Options

Configurable products

Grouped products

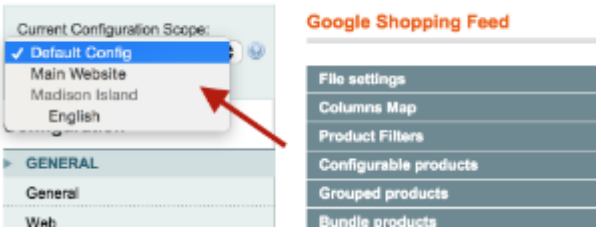
Bundle products

Apparels

Shipping

Configuration Scope

You can have as many feeds as you have stores views. You can configure all your stores the same way by leaving the configuration in the global scope.



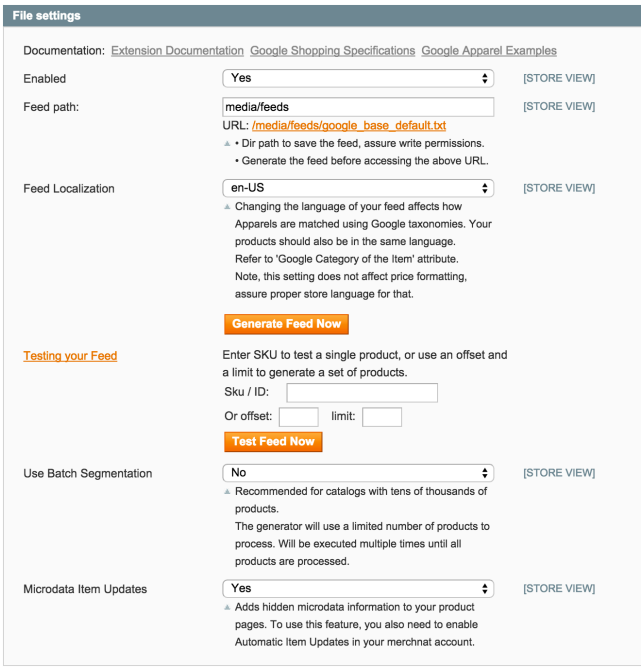
If stores need to have slightly different configuration, for example different language / localisation setting, change the configuration scope and adjust the needed setting.

Related articles

- Become an extension reseller
- Error: Base table rw_gfeed_processes not found
- Setting up products with Minimum Advertised Pricing
- Running AdWords Shopping Campaigns
- Setup microdata to enable automatic pricing updates

9 related results

File Settings



Enabled:

Option to enable or disable the feed generator. If you would like to disable this extension on a specific store then switch to store view and save this option as "No".

Feed path:

"Feed path" is relative to the web root like: *[magento_root]/media/feeds/.* Allows you to to set an alternate save path. Make sure the feed path has enough write permissions for both web server's user and cron user if they are different. Ask your web hosting company for help if needed.

Feed files can be created for every store. Files will have following file name format: *google_base_[store code].txt* Magento comes by default in single store mode with the store code default. If unchanged, this feed file will be in: *[magento_root]/media/feeds/google_base_default.txt*

Feed Localisation:

Changing the language of your feed affects how Apparel products are matched using Google taxonomies. Your products should also be in the same language.

Note, this setting does not affect price formatting. Price formatting depends on what is set for your store under System > Configuration > Locale Options > Locale.

Generate Feed Now button:

This will generate the feed on demand for your full catalog.

> **Limitations**
> The success of this button is highly depended on your apache / php server configuration, and most of the times it is necessary to adjust your php / apache configuration:
>
> - php *max_execution_time* and *memory_limit*
> - apache *TimeOut*

Because generating a full catalog can be demanding on your server, it has been limited to only function on small catalogs, for less than 1000 products. To change this limit, edit *app/code/community/RocketWeb/GoogleBaseFeedGenerator/config.xml*, *<button_max_products>1000</button_max_products>*

Test Feed Now button:

We highly recommend using this button to see if data fetched from a selection of products is well formed. There are 3 additional fields which specify which product(s) to test:

- Sku / ID - for testing individual products. Use products that are visible in catalog and enabled.
- Offset and limit - for testing a range of products. The offset can be from 0 to the total number of products. Limit can be maximum of 100 and will instruct the test to return that number of products.

Using this button won't write into the main file. It will write in an alternate file with the format: *test_google_base_[store code].txt* in the feed save path.

more info on testing and generating the feed

Use Batch Segmentation:

This feature is meant to help with large catalogs or to spread out the server load over time. A concern may be memory consumption when loading large collection of products. The script is optimized to load the minimum data in the collection of products by Magento's native model, but memory leaks will keep consuming more memory as the script runs longer.

With this feature active you can specify how many products should be processed in one batch. To make sure the right Batch Limit is set, run a full feed in console and see how many products it processes without crashing out of memory. When processing a batch and less than 10% of allowed memory is available, the batch terminates early and postpones the remaining products for the next batch, increasing the number of batches needed. So make sure you have the appropriate cronjob schedule in place to complete the feed.

Microdata Item Updates:

Implements google Automatic Item Updates feature by adding hidden microdata on your products pages to keep your products up to date during the day. Useful for regular promotions and special offers, or low-stock items that sell fast.

Follow the steps in automatic pricing updates tutorial to make sure all goes well.

Related articles

- Setup microdata to enable automatic pricing updates

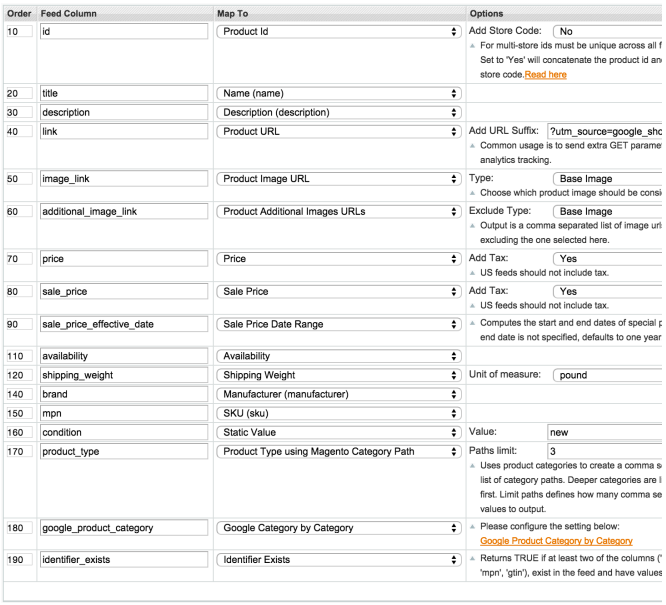
- Getting Started
- Feed Configuration
- File Settings
- Submitting your feed to Google

16 related results

Columns Map

Columns Map allows you to associate any column of the feed with product attributes or directives implementing special logic.

> **Feed Requirements**
> Your feed depends on your store configuration, types of products and available data stored in your products.
>
> Please carefully read Google Shopping Specification to configure this extension properly.



The Column Map grid features the following:

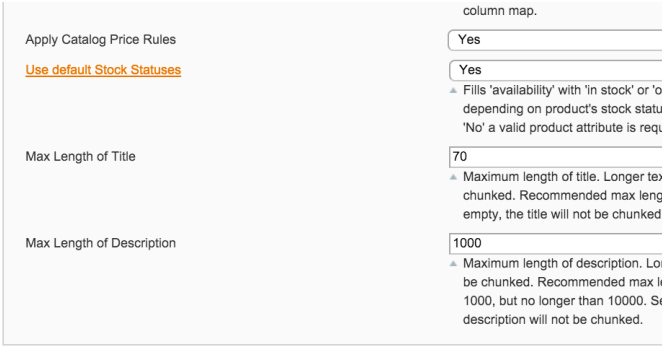
- **Order** - allows you to define the columns order in the feed. Higher numbers go last.
- **Feed Column** - is the name of the field as listed in the Google Shopping Specifications.
- **Map To** - can be either Directive (first options in the dropdown), or a product Attribute. Directives have special logic. Attributes simply grab the exact value from your products.
- **Options** - most Directives accept parameters that can be specified here.

Generally it is recommended to associate directives (the first grouping in the dropdowns) to corresponding columns. Column names must be exactly as Google specifies. English column names are acceptable for any locale; you may also specify column names in the target country's language, but these values must be approved by Google (any errors will show in Merchant).

By default the extension comes configured with a minimal set of column associations. Most countries now require a valid **mpn** or **gtin** for products along with the **brand**. You should at least add one of these 2 columns before submitting your feed. To configure apparel products you should add these additional columns: color, age_group, gender, size, item_group_id and optionally material and/or pattern.

Order will sort the columns in the feed. Google Shopping does not need Feed Columns to be in any order, use this mostly for your own readability as you review the feed output.

Apply Catalog Price Rules



Whether to include Magento catalog rules when computing prices on products. This has direct influence on how the Sale Price is being computed, in addition to product Special Price attribute, when this option is set to yes, promotions will affect whether Sale Price gets populated or not.

Max Length of Title and Description

Title and Description of the products are limited in Google Data to 70 and respectively 1000 characters. Those settings allow you to change those values if needed.

Use default Stock Statuses

Defaults to yes, uses regular magento stock information to compute whether a products is out of stock or in stock. When set to No, will allow you to specify a custom attribute which has to hold valid values for stock status, under setting Alternate Stock/Availability Attribute.

The attribute's values can be: 'in stock', 'out of stock', 'preorder'. Other values will be replaced by 'out of stock'.

Related articles

- Directives
- Columns Map
- Bundle products
- Grouped products
- Configurable products

Directives

Directives are functions performing specific logic that can be applied in the Feed Columns. Navigate through the list below to learn what their logic is.

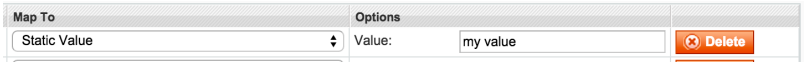
All Directives

Static Value

This directive can be used to set a string value for all your products. The parameter accepted is a string value.

Most of the times when all your products have the same constant information, you would want to use this directive instead of setting the same value in an attribute for all your products. One good example is setting the brand information for stores selling only products of the same brand.

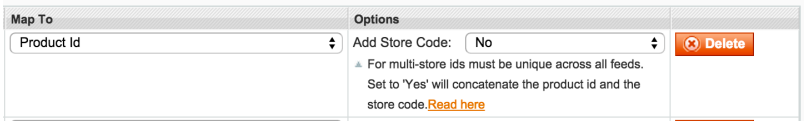
In the default set of columns, condition is set to be new using this directive.



Product Id

Primarily this directive will output the product entity_id, and accepts as parameter Add Store Code Yes/No.

For multi-store, ids must be unique across all feeds, and to make them unique, you have the option to concatenate the product id and the store code together.



Item Group ID

This directive has been renamed in v1.6.1 from "Parent Product Id", which used to simply output the Id of parent configurable product.

Will output the Parent SKU for associated product that vary by one of the columns: color / material / pattern / size. For all others products, it will output empty. Most common usage for this directive is to output google's item_group_id column.

This directive is designed to make variant products appear as they are separate products in case one of the configurable attribute varies and is not part of the color / material / pattener and size, set.

For example if you have a pair of jeans that vary by size and fit, you would want to have Slim and Regular as different products even if all variants are part of the same configurable parent. The item_group_id would be :

- SKU1-slim
- SKU1-regular

and each of them would have different sizes.

The directive is doing this by concatenating variant attributes values other than color / material / patener or size to the Parent product SKU, in such way that they are unique per size.

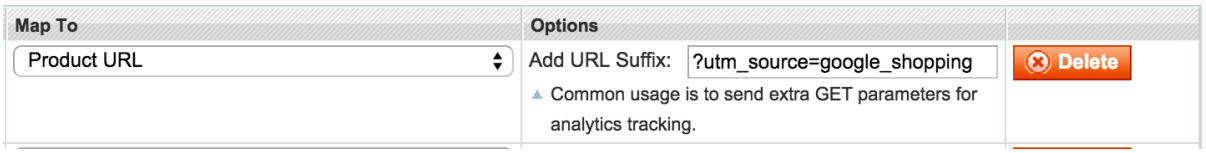
The output limit is at 50 chars, so computed values longer than that will be trimmed.

Product URL

Processes product urls so that they match the type of product and listing in the feed. Set the additional parameter Add URL Suffix to customise further the end of a product URL with custom string.

The suffix can be used to track referring URLs in web analytics software. Setting this to ?utm_source=google_shopping will generate links like: http://mystore.com/product-url.html?utm_source=google_shopping. Products without assigned categories will not use the SEO friendly version of the URL.

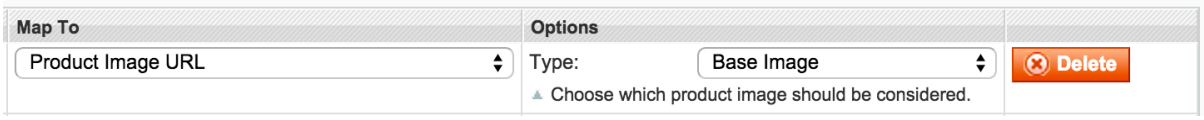
Configurable associated products can append to this URL parameters to target their specific variant, see Unique urls for associated products not visible setting under Configurable Products section of the feed config.



Product Image URL

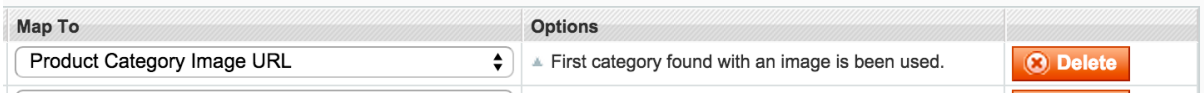
Simply pulls the product's main image URL. Allows the option to set under the Type parameter, which images is the main image, by default it's set to Base Image.

If your product does not have an image assigned as the Base Image, your product may be skipped if "image_link" is under "Skip product with empty". If you notice problems, try removing it from skip rule.



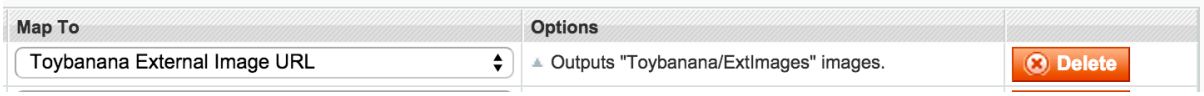
Product Category Image URL

Returns category image of the first category assignment found for your product. This should only be used for special cases where product images are defined on the category level.



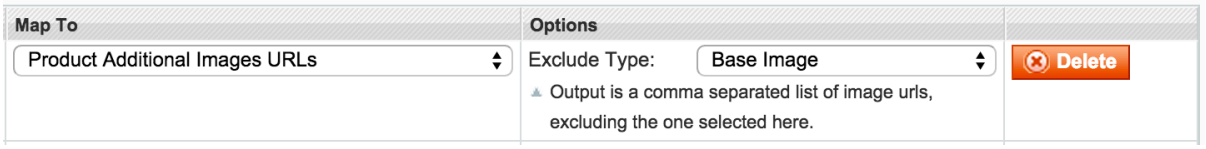
Toybanana External Image URL

Returns the value of product's "image_external_url", which comes packed with Toybanana_ExtImages extension. If you do not have the extension, you can simply create the image attribute and fill in the data for products.



Product Additional Images URLs

This directive outputs a comma separated list of active image URLs for the product. Under the Exclude Type parameter should be set to the main image of the product, so that only the rest of images are added through this directive.



Product Image URL 360_Magic

This directive outputs the main image managed by MagicToolbox_Magic360 extension. Under the Main Image no (0001) parameter can be specified which number of the image is the main image used with that extension.

This directive is available starting with v1.6.1.

Product Additional Image URL 360_Magic

This directive outputs other images managed by MagicToolbox_Magic360 extension, up to 10 images in a comma separated value. Under the Main Image no (0001) parameter can be specified which number of the image is the main image used with that extension so it can be excluded from the list.

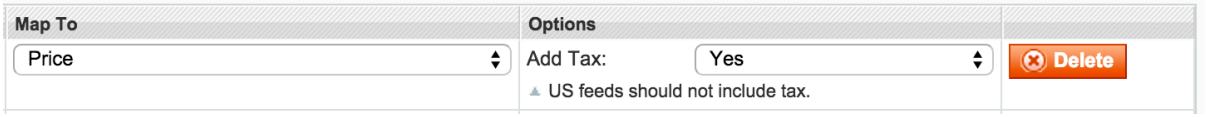
This directive is available starting with v1.6.1.

Price

The price directive aims to match the price displayed on the product page, so it takes into account currency conversions, tax, promotions, MSRP and eco tax. Configurable, bundle and grouped items will have the minimal price of an item. Setting Add Tax option below to Yes will add tax to this price column, for US all prices should not include tax, the appropriate setting should be made here. The price output includes the currency information.

Msrp price is used in this directive when the product has msrp enabled and that value is bigger than the price itself, forcing Sale Price directive to display the regular price.

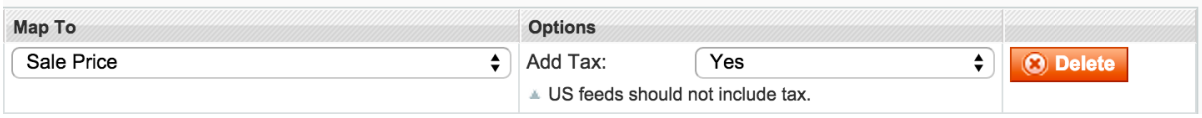
For products having qty increments active it will multiply the amount with the minimum number of items to purchase.



Sale Price

This directive will output the lowest value for which the product can be purchased at. It computes the Special Price of the product, and any catalog promotion rules according to the Apply Catalog Price Rules setting from Columns Map section. Setting Add Tax option below to Yes will add tax to this price column, for US all prices should not include tax, the appropriate setting should be made here.

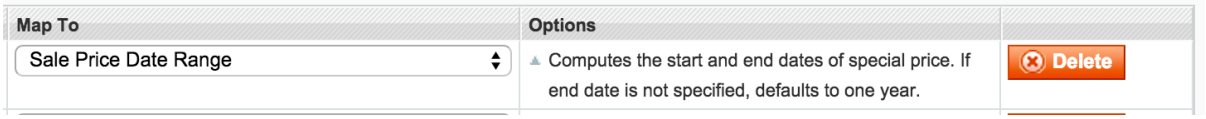
When Msrp is enabled, the value of regular price is used here if there are no running promotions at a lower value. Similar to Price directive it takes into account the qty_increments set at the product.



Sale Price Date Range

The date interval is been taken primarily from products Start / End dates of the special price. This interval will never output empty, so when no end date is been specified at the product, the range will be set for a full year. If the start date is not specified on the product, the current processing date is been considered.

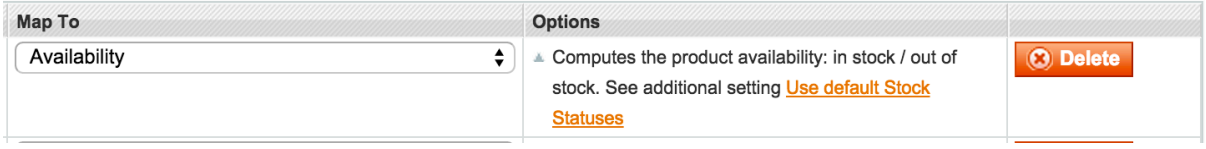
This date range is mostly useful in the feed along with the Sale Price, specifying when this sale apply.



Availability

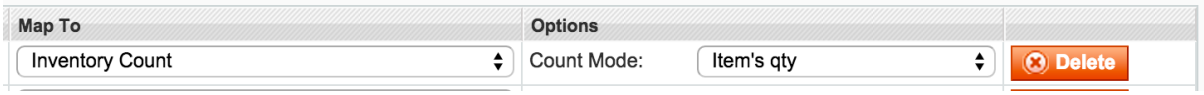
Uses regular magento stock information to compute whether a products is out of stock or in stock. When the additional setting Use default Stock Statuses is set to No, a custom attribute specified under Alternate Stock/AvailabilityAttribute, will be used to specify the availability. See more settings under Columns Map section.

Accepted values for this directive are: 'in stock', 'out of stock', 'preorder'. Other values will be replaced by 'out of stock'.



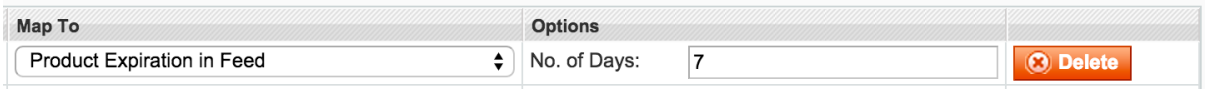
Inventory Count

This is a legacy directive used to map the 'quantity' column which is no longer required by google. It simply pulls the stock inventory qty information. The Count Mode defines how the qty count should be performed in case of complex products like bundle.



Product Expiration in Feed

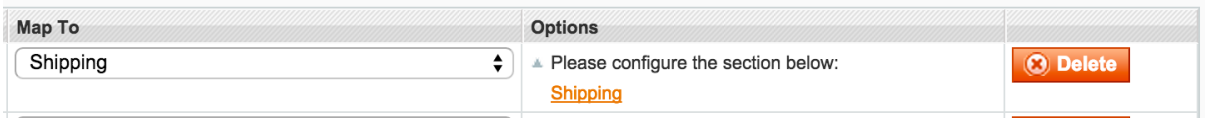
Computes an expiration date based on product creation and the "No. of Days" option below. Set "No. of Days" option to increase or decrease this date limit. This field is not longer required by Google.



Shipping

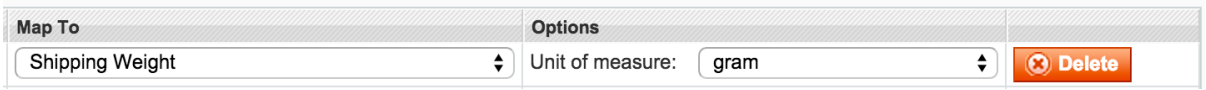
This directive computes shipping information into the Google accepted format based on the Shipping Methods available for your products. To use this directive, Shipping section has to be enabled and configured on your feed. To specify which Shipping Methods and which regions will be considered please review Shipping section.

If Shipping is not enabled and not shipping methods are specified this directive will output empty.



Shipping Weight

Outputs the weight attribute of your products along with the unit of measure specified here under options. For bundle products, there's an extra setting called Combined weight found under the Bundle Products section.



Variant Attributes (Apparels)

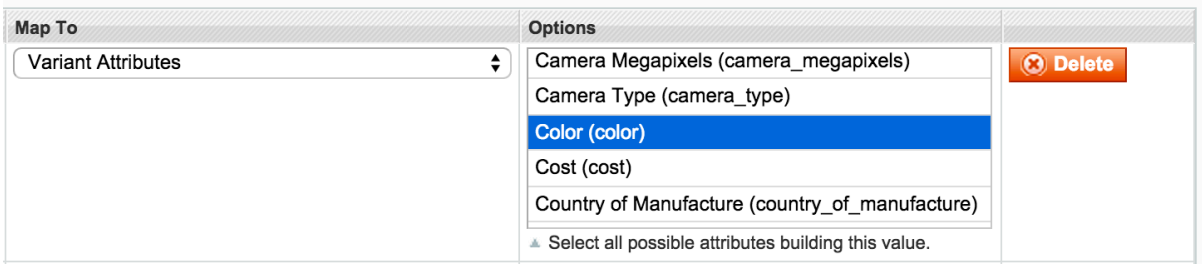
Allows for using multiple product attributes to specify a different value systems for the same property. It comes handy to map apparel specific properties.

Example of **size** property:

- shirts - shirt_size attribute, with values: Small, Large, X-Large, etc
- shoes - shoe_size attribute, with values: 8, 9, 10, etc

Using this directive you can select both attributes, and when processing the product, the appropriate value will be considered depending on whether a value is specified in one of the attributes. First attribute found having a value for the product is been used.

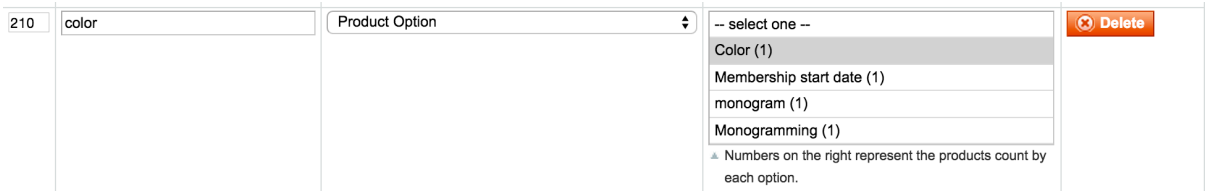
For configurable items, all values will be computed into a list of values delimited by comma, or a delimiter specified under Associated Product Attribute Value Separator setting under Configurable products section of the config.



Product Options

This directive is used to output values of product custom options. This directive is used for apparel catalogs which old the color and size variations as custom options instead of having those properties as configurable products items. It takes as argument the list of custom options names which should hold the values needed for the column you're setting up.

Each custom option in the argument has the number of products affected in the right (x).

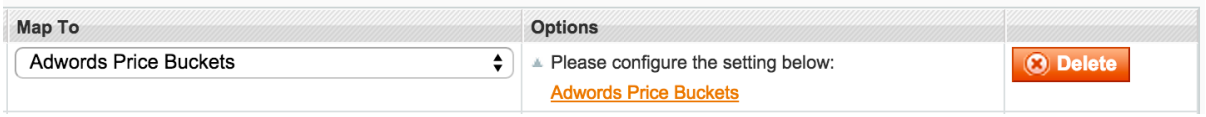


The output of this directive also depends on how the Product Options section is configured, and it can be a list of possible values as comma separated, or it can be a single value in case the products is set to detail each options combination as separate row in the feed.

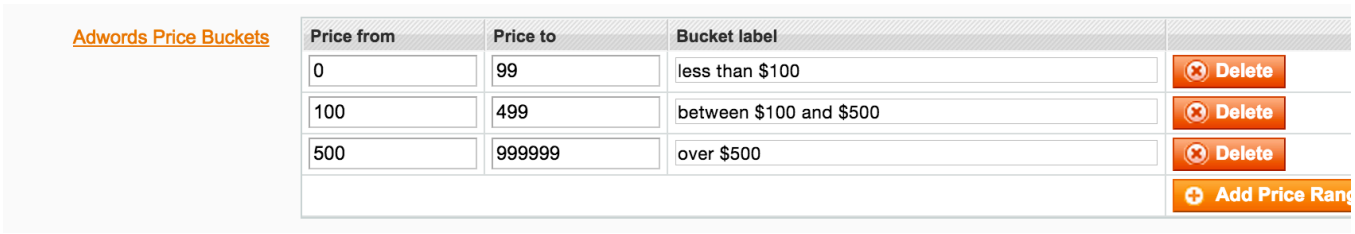
Adwords Price Buckets

This directive uses a set of rules defined under Adwords Price Buckets settings to output labels for different price ranges of your products.

A common usage is to add the google **custom_label_0** column which is handy to group your products when running AdWords Shopping Campaigns.



To specify the upper limit of your prices, just use some big number like 99999999. See below example image.



Product Review Average

Output product's average rating in the interval 1 - 5. Returns 0 if no reviews have been approved on your product.

Product Review Count

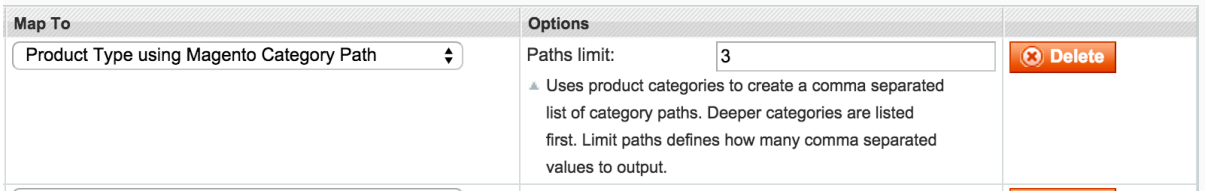
Outputs the count of product reviews. Returns 0 if no reviews have been approved on your product.

Product Type using Magento Category Path

This method is the default way of filling a categorised column of your products that can be used to fill product_type.

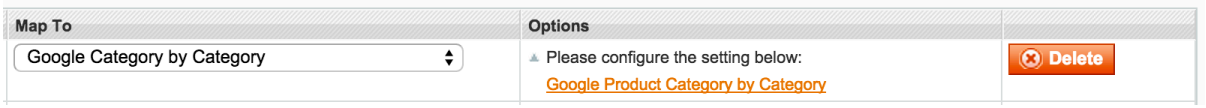
Your product category paths are converted into a string revealing the hierarchy like: Root > Categ A > Categ B. Product been assigned to Categ B. All product categories will be used to build a list of comma separated paths, and they are limited to the Paths limit value.

Deeper categories are listed first, so that when the limit is set the most relevant ones are revealed.



Google Category By Category

Uses a set of rules defined under the Google Product Category By Category Map setting to output category taxonomies.



Products will be matched through the set of rules using their assigned categories. The first category found with the most important category rules below will be considered. The rule importance is given by the Order rule column, lower values matching first.

Product categories do not necessarily need to have a direct correspondent rule to cover products deeper in the tree. High level categories set here would dictate inheritance for products deeper in the category tree.

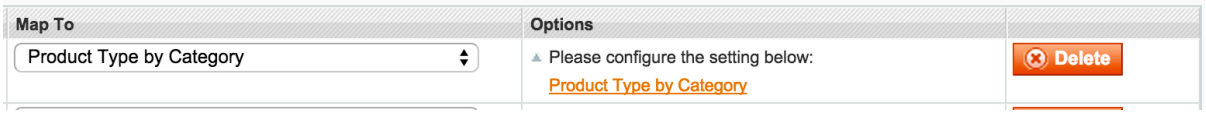
The more categories defied here, the granular your taxonomy definition becomes giving google a better and detailed view of your products.

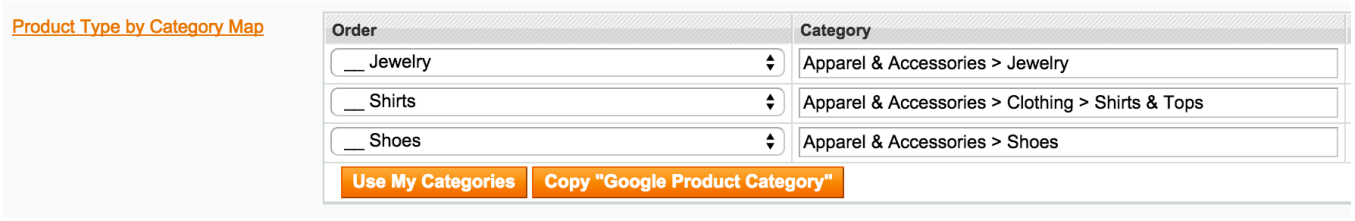
> We recommend that you set all rules with an order value. Setting higher Order values for high level categories, and lower Order value for deeper categories will make deeper categories will match first, dictating the output value.

It is advise to also set a Replace empty values rule to cover all missed categories from this set, and fill the values with a static default taxonomy or some product attribute holding the taxonomy.

Product Type by Category

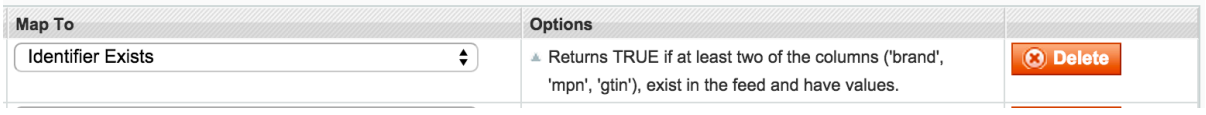
Similar to Google Category By Category a set of rules can be defined under Product Type by Category Map setting.





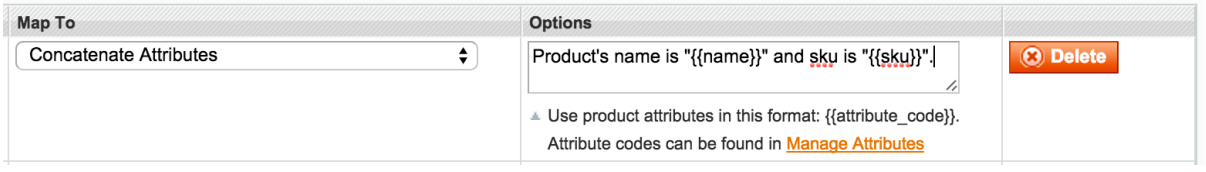
Identifier Exists

This directive looks at your feed columns to see if two out of three of the following: brand, mpn, gtin columns have a value. Those columns are been considered as identification information of products, and if at least two of them are missing, the value returned here will be FALSE.



Concatenate

Allows you to compose a string using attributes of your products.



Use product attributes in this format: {{attribute_code}}. Attribute codes available for your products can be looked up under Catalog > Attributes > Manage Attributes. This method works like a replace in string function, and looks for the {{}} marker through the string template provided and will replace it with the attribute code between the marker characters.

The following placeholders:

- {{name}}
- {{description}}
- {{url}}
- {{image}}

will inherit values as defined in the 'Fetch ... from' settings from Configurable and Grouped configuration sections.

If all placeholders bing an empty value when computed, the column output itself will be empty.

If all {{attribute_code}} values are empty, the return value will also be empty. So if we set the Options field to:

Concatenate of {{name}} and {{size}}

and {{name}} and {{size}} values in product are empty, the Directive will return empty string. Those values can be caught using Product Filters - Replace .

Is Bundle

Outputs TRUE if the item is a bundle item and FALSE otherwise.

Related articles

- Directives
- Columns Map

Product Filters

> Product filters section mostly allows you o filter out products you don't want in the feed, but also replace values generated with Columns Map definition.

Allow Out of Stock

This filter is applied to standalone simple products or configurable products. Associated simple products have other rules defined in "Configurable products" and "Grouped products"

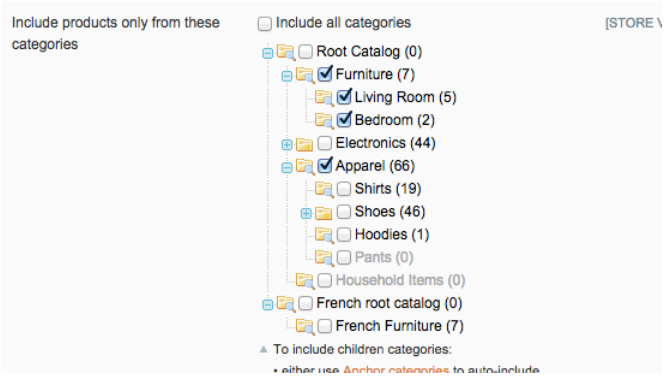
Include products only from those categories

This feature enables you to select which categories will be included in the feed. By default, products from all categories are included. If you want to only include products from some categories, please un-check "Include all categories" and select the categories you wish to include by checking the boxes.

This feature also respects anchor categories. In Magento, Anchor categories work like this: if you set a category as *Is Anchor*, then all products from its children categories will also appear on the parent category page. This is useful in your website: for example to create a parent category page to include all *Apparel* products - if the visitors want to filter results, they can use the layered navigation to go to *Men* or *Women*.

The **benefit** of using anchors is that we'll include all products of all sub-categories automatically.
This means that:

- you don't need to select all sub-categories here
- if you add more sub-categories to an anchor category, you



In the screenshot above, we only want to include Furniture and Apparel.

Since Furniture is not an anchor, we select its children too. Later, if I want a *Bathroom* subcategory, I'll have to come back here and

don't need to worry about including them here

Note: *to set a category as Anchor, edit the category in Catalog > Categories > Manage Categories , Display Settings tab.*

include it in the feed. On the other hand, Apparel is an anchor, so there's no need to select its children. What's more, if I later want to add *Dresses* to Apparel, I don't need to come back here.

Submit only products of these types

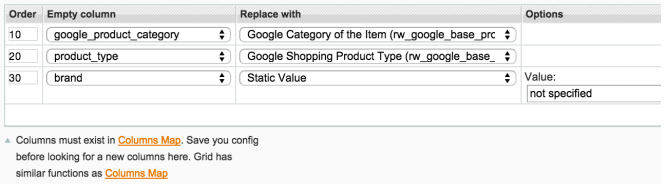
You can specify by store what types of products should be included in the feed.

Adding configurable or bundle types will actually pull also associated simple products if it's specified in the Configurable products or Apparel sections. Custom product types like AheadWorks subscription types will do fine.

Submit only products that have these attribute sets

If you want to include All Attribute Sets, you can select only the first option; selecting the first option along with a few other sets won't include all sets.

Replace empty values



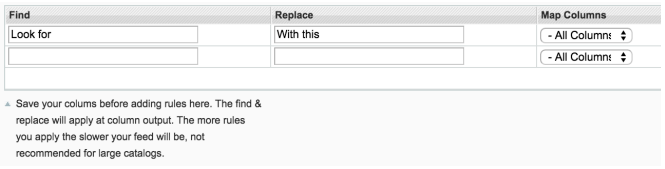
This set of rules have similar function as Columns Map, the only difference been that columns defined here have to exist. Make sure you save your column map first.

This setting gives the opportunity to stop the feed from delivering empty columns when products are lacking respective data. You can set your empty data to be replaced with Static Values or any other regular Directive or Attributes.

The rules defined here will be applied last in the catalog processing, right before the output, and they will be applied in the order defined by Order column in this grid.

You can actually define cascading rules for the same column to minimise the risk of having empty values in the feed.

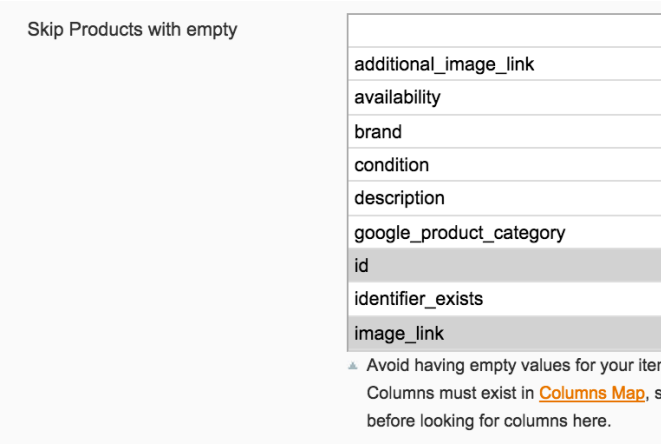
Find and replace



The find & replace will apply at column output, similarly how replace empty values does. The more rules you apply the slower your feed will be, not recommended for large catalogs.

Save your columns before adding rules here.

Skip product with empty



If you want to avoid entries in the feed with empty values, you can skip them here.

This filter comes predefined with required columns for a google shopping feed: id, image_link, link and price. Any product, having any of the columns defined here empty, would be skipped.

This filter will apply at the end of Replace empty values rules, so that products with values from replace empty rules would still be in the feed.

If you can't find the your column in the dropdown, it's because you haven't saved the new feed columns after the column has been added. A Save should solve it.

Related articles

- Product Filters
- Bundle products

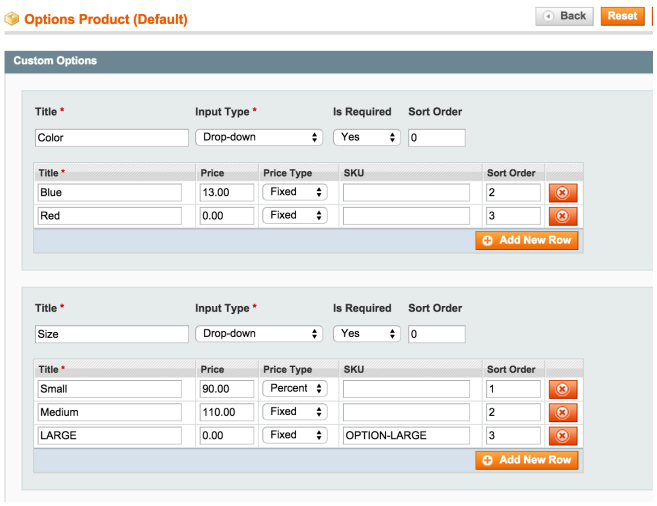
- Grouped products
- Configurable products

Product Options

Product Options support is available starting with **version 1.6.2**

> Current version of the feed extension supports Custom Options for Simple products, and not for complex products like configurable or grouped.

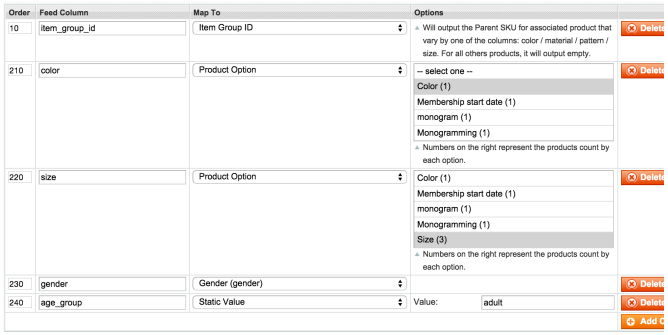
This section is handy when your catalog have apparels that hold size or color properties as product Custom Options, which may have different pricing or different SKUs.



Example of custom options product.

> In order for your custom options to be added in the feed, you'll have to use the Custom Option directive in your Columns Map definition.

In case the catalog contains both apparels as simple products with custom options, and configurable products, the recommended setup is to have size and color columns set in the Columns Map as Product Options directive, and this will process custom options, than under Replace Empty, set the size and color columns to Variant Attributes directive, and this will process configurable attributes.



Example of using the Product Options directive in the Columns Map.

How to add product options

There are two ways you can process those product into the feed:

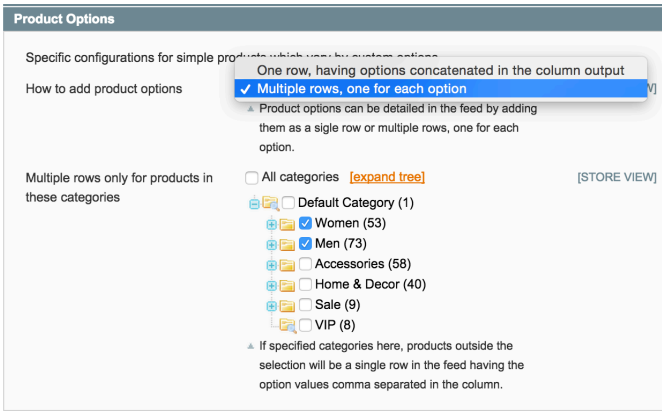
- One row, having options concatenated into the column output.
- Multiple rows, one for each option combination

In case the One row option is selected, the output is on row having all option values as comma separated. I/e/:

- color = red, blue
- size = S, M

If the multipele rows is set, the output is a row for each options combination. I.e.

- color = red, size = S
- color = red, size = M
- color = blue, size = S
- color = blue, size = M



Multiple rows only for products in those categories

This option lets you select which of the product categories should be treated as apparels, so hose will be added as multiple rows in the feed with respect to custom options. The rest of the products will simply have comma separated option values.

Related articles

- Configurable products
- Directives
- Columns Map
- Bundle products
- Grouped products

6 related results

Configurable products

How to add associated products

This option specifies how configurable associated items will be processed. It has three options:

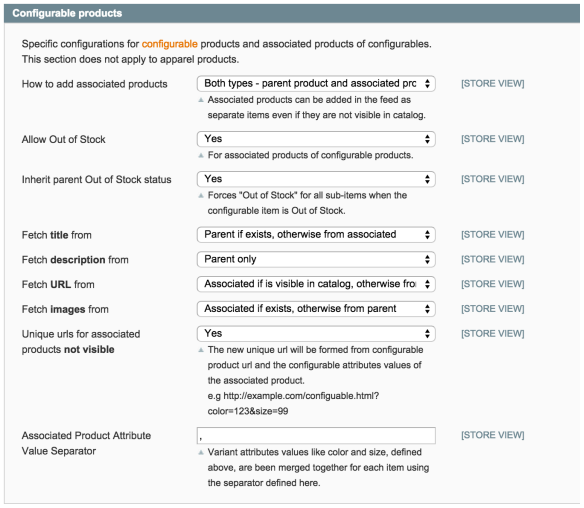
- Only parent / No associated products
- No parent product / Only associated products
- Both types - parent product and associated products

The recommended option for a google shopping feed is to use the option No parent product / Only associated products, which would list all associated items of the products but miss the configurable itself.

> In order for this option to work, configurable type has to be selected under the Submit only products of these types under the Product Filters section, regardless of the option you choose here.

Allow out of stock

This option can filter out associated items that are out of stock, regardless of what's been set under Allow Out of Stock setting from Product Filters section. The goal is to allow you to remove out of stock associated items even if you want to include regular out of stock products.



Inherit Out of Stock Status

Dictates how the parent configurable item will behave when all associated items of a configurable are **out of stock**, the configurable itself will become out of stock. It will not do the same for **in stock** items.

Fetch title from

Tells where to grab product name of associated items when they are included in the feed. Has the following options:

- Parent only
- Associated only
- Associated if exists, otherwise from parent
- Parent if exists, otherwise from associated

The recommended option Associated if exists, otherwise from parent should cover most cases of configurable product setup, doing a failover to configurable parent when no value is found on the associated item itself.

Fetch description from

Tells where to grab description of associated items when they are included in the feed. Same options and conditions apply as for the Fetch title from setting.

Fetch URL from

This allows you to dictate which URL to present to associated item with depending on it's status in the catalog. Has the following options:

- Parent only
- Associated if visible in catalog, otherwise from parent

In most cases grabbing the parent URL is enough, but when the associated items are also visible in catalog on their own it;s probably best to use their own URL so that the appropriate product information is displayed on the page when accessed.

Fetch image from

Tells what images should be used when associated items are included in the feed.

Has the following options:

- Parent only
- Associated only
- Associated if exists, otherwise from parent
- Parent if exists, otherwise from associated

Most of the times the images are set directly to the configurable item and those are sufficient, but sometimes when you want to feature out some variant characteristics, you would set individual images on the variant items.

> **Fetching price from associated products**
> Some stores are using extensions to bypass default Magento behavior when it comes to Configurable Product Price (CP Price). It shows Associated Simple Configurable Product Price (Associated SCP Price) instead of Configurable Product Price. With version 1.6.4 we gave support for this behavior, so if CP Price is empty or 0, then it fetches minimal Associated SCP Price.

Unique URLs for associated products not visible

When associated products are not visible in catalog on their own, they are accessed through the configurable item URL, making it hard to tell which products we're targeting. Activating this option will append extra parameters to the URL, making it unique.

Ex: http://example.com/configuable.html?color=123&size=99

When accessing this kind of URL, there's a special javascript logic on products page automatically selecting drop-downs or radio buttons to reveal the intended product details: image, price, etc.

Associated Product Attribute Value Separator

When parent products are added to the feed, the variant information like **color** and **size** is been composed using all possible values that their associated items could take. To list all of them, a separator is been used, and it can be specified here.

Ex: color = blue, yellow, brown

> This option only works for columns using the Variant Attributes directive in the Column Map.

Related articles

- Configurable products
- Directives
- Columns Map
- Bundle products
- Grouped products

6 related results

Grouped products

How to add associated products

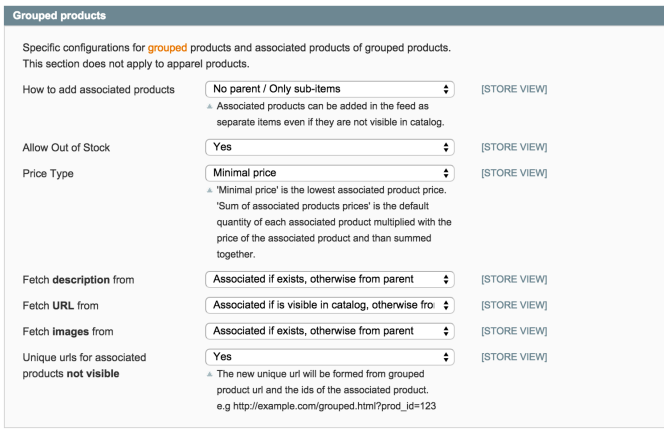
This option specifies how items of grouped products will be processed. It has three options:

- Only parent / No sub-items
- No parent product / Only sub-items
- Both types - parent and sub-items

> In order for this option to work, grouped type has to be selected under the Submit only products of these types under the Product Filters section, regardless of the option you choose here.

Allow Out of Stock

This option can filter out sub-items that are out of stock, regardless of what's been set under Allow Out of Stock setting from Product Filters section.



Price Type

Defines how the price of the grouped product will be computed. It can be:

- Minimal price
- Sum of associated product prices

Minimal price is the lowest associated product price.
Sum of associated products prices is the default quantity of each associated product multiplied with the price of the associated product and than summed together.

Fetch title from

Tells where to grab product name of associated items when they are included in the feed. Has the following options:

- Parent only
- Associated only
- Associated if exists, otherwise from parent
- Parent if exists, otherwise from associated

The recommended option Associated if exists, otherwise from parent should cover most cases of grouped product setup.

Fetch description from

Tells where to grab description of associated items when they are included in the feed. Same options and conditions apply as for the Fetch title from setting.

Fetch URL from

This allows you to dictate which URL to present to associated item, depending on it's status in the catalog. Has the following options:

- Parent only
- Associated if visible in catalog, otherwise from parent

In most cases grabbing the parent URL is enough, but when the associated items are also visible in catalog on their own it's probably best to use their own URL so that the appropriate product information is displayed on the page when accessed.

Fetch image from

Tells what images should be used when associated items are included in the feed.

Has the following options:

- Parent only
- Associated only
- Associated if exists, otherwise from parent
- Parent if exists, otherwise from associated

Most of the times the images are set directly to the grouped item itself, but sometimes when you want to feature out some variant characteristics, you would set individual images on the sub-items.

Unique URLs for associated products not visible

Activating this option will append extra parameters to the URL, making it unique and indicating the product we want to point out.

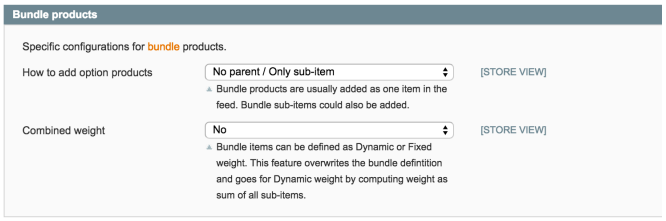
Ex: http://example.com/grouped.html?prod_id=123

When accessing this kind of URL, there's a special javascript logic on products page automatically selecting drop-downs or radio buttons to reveal the intended product details: image, price, etc.

Related articles

- Grouped products
- Directives
- Columns Map
- Bundle products
- Configurable products

Bundle products



How to add option products

This setting dictates how items that make up the bundle would be processed into the feed. Has the following options:

- Only parent / no sub-item
- No parent / only sub-item
- Both types / parent and all sub-items

Most of the times you would want to all items making the bundle, but is some cases you may want to output them as a single product.

Combine weight

Bundle items can be defined as Dynamic or Fixed weight. This feature overwrites the bundle defintition and goes for Dynamic weight by computing weight as sum of all sub-items.

> This option applies to the output of Shipping Weight directive from Columns Map.

Related articles

- Bundle products
- Directives
- Columns Map
- Grouped products
- Configurable products

Apparels

> To understand apparel requirements, please review the Submitting Apparels document from google.

To make use of the apparel functionality, your products must matched under the Apparel & Accessories or the equivalent for your feed locale. So to assure the minimum requirements you'll have to do the following:

1. Define your Google Categories to match your appropriate categories under Apparel & Accessories or subcategories of this taxonomy.
2. Add the apparel specific columns to your Columns Map.

When using apparels, specific columns must be manually added to Columns Map. Some of the apparel specific columns are:

- color, size, age_group, gender, material, pattern, item_group_id, size_type, size_system

Always consider using the Variant Attributes directive when mapping any of the above columns, to ensure appropriate and well formed data to your feed.

Columns like age_group and gender may be mapped to the Static Value directive if all your products address the same client base like adults or females.

Column item_group_id is mandatory for apparel variants as it offers a way of grouping them back with the parent product. Its value is the

configurable product id, and should be mapped to Parent Product Id directive.

> **Where is the special Apparels section?**
> For users upgrading from older version, you will notice that the Apparels section is now gone.
>
> Apparel detection is now turned on by default, and all the settings can be found under Configurable products. Since apparel products are Configurable products, we simplified the configuration screen by merging the two sections.

Related articles

- Bundle products
- Grouped products
- Configurable products

Shipping

> The configuration section fine tunes the output of Shipping directive from Columns Map. Make sure you add a column called "shipping" and map it to the Shipping directive.

The options here are helping to construct the output format required by Google Shopping Specification docs.

- Only table rates and static shipping methods are available for use here.
- Simple rates by price and weight can be defined in Google Merchant Center as well, so you may wish to check those options and skip this section.
- Setting up real time shipping methods must be done within Google Merchant Center, under Settings > Shipping > + Shipping Method

> The address used in rates computations is been pulled from Shipping Origin under System > Configuration > Sales > Shipping Settings > Origin. Make sure that address is been set correctly.

Enable Shipping Feature

If the shipping is disabled here, no output will be generated for the directive.

Methods

Lets you choose which of the shipping methods you would want to apply on your products in the feed. Only table rates and static shipping methods are available for use here.

Countries

This option should be correspond to the feed's target country you are choosing when setting up the feed in your Merchant Center. Through your are free to choose multiple countries here, google only allows one feed per country, so it's advise to use only one country in this setting.

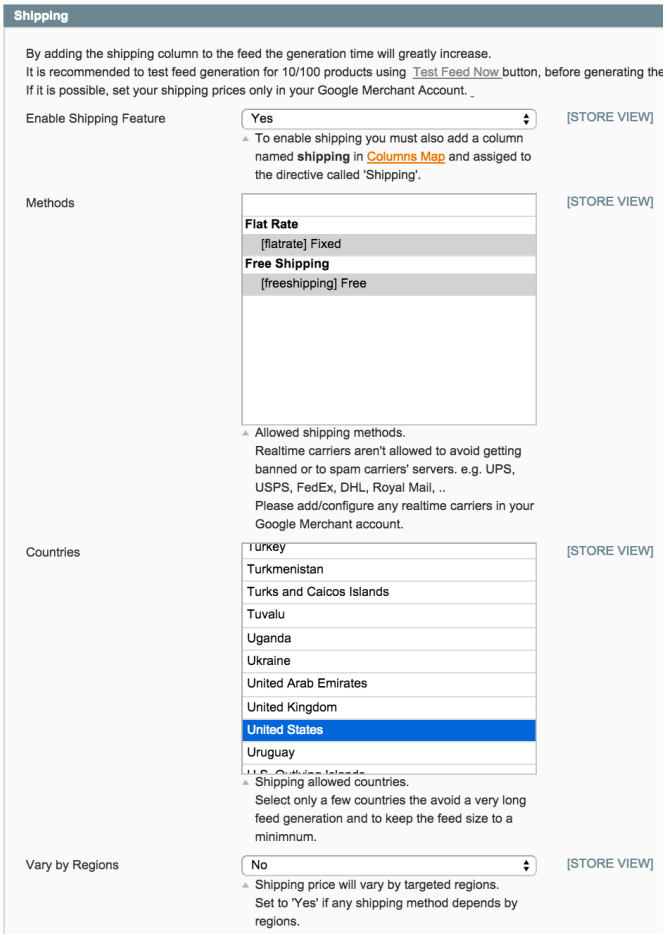
Vary by Regions

This setting is useful with table rates where you have different rates by shipping locations. It is recommended to have this option turned on regardless of how many methods you set in the Shipping Metods .

Countries with Regions

This option becomes active when Vary by Regions is turned on,

and lets you choose which of the Countries should have variable rates.



Only Minimum Price

If more than one methods apply, you can output only the smallest rate from those that apply.

Only Free Shipping

When more than one rate applies to the same product, and one of them sets free shipping, you can output Free Shipping alone by having this option turned on.

Add Tax to Shipping Price

Will include tax rate calculation if it applies to the shipping rate. Most of the times - and especially for US feeds - this should be off.

Format Shipping Price

Whether to format shipping price according to the store's locale settings. It will format the prices according to the locale set under System > Configuration > Locale Options > Locale

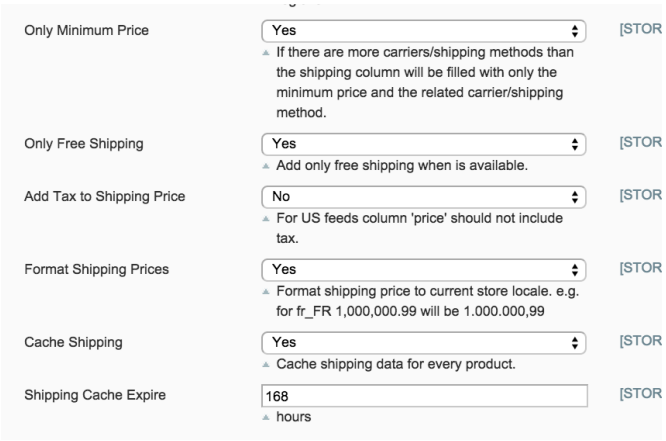
It does not follow the feed's localisation, or the general Format Prices from Columns Map section.

Cache Shipping

Because computing shipping rates for each product can bring significant server overhead, the values are being cached throughout feed generation. It's advised to keep this option turned on, especially for large catalogs.

Shipping Cache Expire

Defines the time interval when computed values will expire. Most of the times this value should be more than a few hours to make sure cache is being used from one product to another in the same generation cycle. Setting this value to 0 is equivalent to turning off caching.



Related articles

- Directives
- Shipping
- Setting up Shipping and Tax
- Adding Free Shipping label

How-to articles

Add how-to article

Error rendering macro 'content-report-table' : In template Confluence.Templates.User.userLinkUrl: When evaluating "contextPath()": Error while computing function "contextPath()": null

Adjust automatic update schedule

There are two ways automatic feed updates can be set:

1 Magento Cron System
2 Custom Crontab System
    2.1 Setting crontab with batch mode

The first one comes pre-installed with the extension, but for more control we recommend using the second option.

Magento Cron System

Schedule Updates
Your file is automatically generated at 1AM each night by default. For this to work, your Magento install need to have it's cron.php running each 5 minutes or so.

Example of how magento's cronjob should be set on the server:

```
*/5 * * * * /usr/bin/php MAGENTO_PATH/cron.php
```

Without this crontab command set, none of the majecto cron schedules will run, so it's important to have this set.

The extension installs with just one schedule to update the feed nightly, for store code default. If you store code is different or you have multiple store views, you'll have to adjust this setting under:

[magento_root]/app/code/community/RocketWeb/GoogleBaseFeedGenerator/etc/config.xml

config.xml

```
<crontab>
        <jobs>
            <googlebasefeedgenerator_generate>
                <!-- In batch mode (use_batch_segmentation==1) must be set after
midnight otherwise will not finish the queue. -->
                <!-- each 1 am -->
                <schedule>
                    <cron_expr>0 1 * * *</cron_expr>
                </schedule>
                <run>
                    <model>googlebasefeedgenerator/observer::generateFeed</model>
                </run>
                <!--<store>default</store>-->
            </googlebasefeedgenerator_generate>
            <!--
            Setting generator to run for other stores in a multistore configuration.
            <googlebasefeedgenerator_generate_store_2>
                <schedule><cron_expr>0 2 * * *</cron_expr></schedule>

<run><model>googlebasefeedgenerator/observer::generateFeed</model></run>
                <store>store_code_2</store>
            </googlebasefeedgenerator_generate_store_2>-->
        </jobs>
    </crontab>
```

Custom Crontab System

Sometimes Magento cron can be a bit finicky, and to avoid missing out feed updates, we recommend to use the included shell script for setting up a crontab on your server. If your store is hosted using Cpanel, you can add the shell command under crontab section of Cpanel

Example:

Crontab example

```
0 1 * * * php MAGENTO_PATH/shell/gsf_generate.php --verbose --store_code [STORE_CODE]
```

Permissions
Whenever cron scripts are run, they should run under the same user UUID which your web server (apache) is running. This will allow proper file permissions for the feed and lock files.

Before setting up the crontab, try running the shell script directly to see how it works and make sure the path to MAGENTO_PATH and extra parameters like [STORE_CODE] are correct. Running the script with --help argument will print out all available options:

gsf_generate.sh

```
php MAGENTO_PATH/shell/gsf_generate.php --help
Usage:  php gsf_generate.php -- [options]
store_code <string>       Store Code (e.g. [STORE_CODE] or default). Store must exist
and should be enabled. By default uses 'default' value.
batch_mode <int>          Segment the feed generation. Values accepted: 0 or 1.
Default is 0.
test_mode <int>           Enable test mode or not. Default is 0.
test_sku <string>         Generate the feed only for a product sku. To be used for
tests and debuging.
test_limit <int>          Sql limit parameter in test mode. Is applied to the select
of the collection of products.
test_offset <int>         Sql offset parameter in test mode. Is applied to the select
of the collection of products.
verbose                   Outputs skus and memory during processing
help                      This help
e.g. php gsf_generate.php --store_code [STORE_CODE] --batch_mode 1 --verbose
```

Setting crontab with batch mode

If you have 15,000 products and you build the file in batches of 1,000 try running gsf_generate.php on the command line once to see how long it takes to run. If it takes 2 minutes for example, then setup cron to run at least 15 times every 5 minutes for a couple hours like:

Crontab example

```
*/5 2,3 * * * php MAGENTO_PATH/shell/gsf_generate.php --store_code [STORE_CODE]
--batch_mode 1 > /dev/null 2>&1
```

Something like the above command will run the script 24 times and should work for up to 24,000 skus. When the script runs for the 16th time in this example, it will see it has found all the products and will immediately quit. So it's OK to run the script several more times than needed each night.

Related articles

- Adjust automatic update schedule
- Configure the cron to automatically generate the feed every night
- Submitting your feed to Google
- Testing & Generating the Feed
- "My feed isn't working!" First steps to take

14 related results

Set up stores and feeds for multiple countries

Magento supports the very useful feature of creating multiple stores that target different countries. You can setup storeviews that have a different language or currency, while re-using the same product catalog, categories and promotions, even the same customer base.

There are many ways to achieve this setup: some are easier to configure, but more restrictive, while others are a bit more involved, but offer more flexibility.

However, to make sure you adhere to the Google Shopping policies, there's one particular way to setup Magento that has proven to be the most efficient.

Conditions for product pages

A shopping feed must be submitted to a specific Country. Every feed will naturally contain links to the products on your website, links that will apply to the feed's Country: this means that products in an U.S. feed will be found when searching in Google Shopping U.S., with the links from the U.S. feed.

Google requires all Product pages to adhere to some standard rules. The most important are:

1. The Page must show the exact product as in the feed, with the same name, image and description
    a. this information must be visible at all times, and must not require a login
2. The price and currency must match exactly the values submitted in the feed
    a. Specifically, javascript or cookies cannot be used to change the currency on page load
3. The page language must match the country of the feed

> Here's the link to the full Landing Page policy: https://support.google.com/merchants/answer/2700318?hl=en

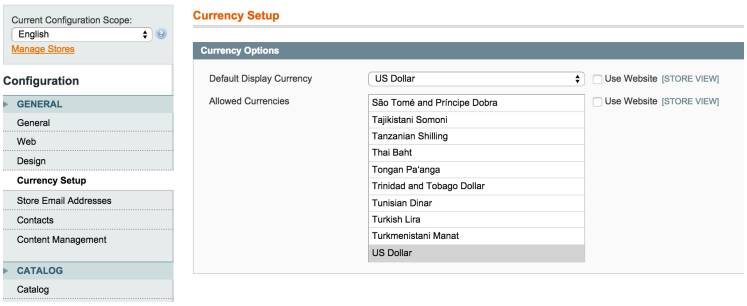
Magento setup

Let's take an example: we have a clothing store that sells the same products in *the U.S*. and *Canada*, and wants to also launch in *the UK* and *France*.

The best way to set this up is to have **2 Magento websites**, and **5 store views**.

- one Magento website with US and Canada, since they are already an established business there
    - US has one store view
    - Canada has two store views - one for English, and one for French
- and another website for the UK and France, to allow for different products and special promotions targeting this new Market
    - the UK has one store view
    - and France another
- (if you don't need special treatment for the EU stores, a good alternative is to have just one website for all storeviews)

Each country must have their own store view, so you can safely and **easily do these settings required by Google Shopping**:

1. **Languages** - France will require product pages in French; for Canada, you can choose between French and English, or better yet have both, one in each store view
2. **Currency** - every country needs a different currency
    a. Go to the Store View level to set this up for each of the store views
    b. For example, For US, set USD as the Default Display currency, and the only Allowed currency.
    c. 

3. **Tax Settings**
    a. US and Canada pages must not include taxes in the price
    b. UK and French however, do need to display prices including tax (you can choose to display both)
    c. remember to do this for the feed settings as well, for the price and sale_price columns

> Here's a comprehensive list of allowed countries, languages and currencies: https://support.google.com/merchants/answer/160637?hl=en

Submitting to Google Shopping

Once you have this setup, you can generate one feed per store-view, with the correct language, currency and tax setup.
Here's how to setup our extension to generate more than one feed: Adjust automatic update schedule

You can then submit your feeds to Shopping, one per Country-Language pair. Go to Data feeds, and Add feed.
For Canada, once you select the country, you can select the language as well.

Frequently asked questions

1. Can I submit the same feed to Shopping to more than one country?
    a. No - This is because of the differences in pricing (tax/no tax), currency and language. Any mismatch in any one of the 3 will not

work.
   b. Even if you do convert prices manually, after the feed is generated, you still need to have Product pages that use that currency / language
   c. Switching the currency with javascript / ajax / cookie (default currency switcher) will not work as Google will not understand it.
2. Can I have multi-currency on one store view?
   a. No - Magento uses cookies to switch currencies. While this is acceptable for users, the Google crawler will not understand it.

Related articles

- Adjust automatic update schedule
- Use pricing from simple child products
- Set up stores and feeds for multiple countries
- Submitting your feed to Google
- Testing & Generating the Feed

9 related results

Setup microdata to enable automatic pricing updates

Automatic item updates is a Google Merchant feature which improves pricing and stock information updates in Google Shopping. It uses microdata from your Product pages to maintain fresh price and availability info, by updating this several times a day.

> Activating Microdata item updates on your feed config will add hidden microdata information on your store product pages. Make sure to clear cache and check your product pages to make sure they still render properly.

AVAILABLE FROM VERSION 1.6.0

Benefits

If you update prices during the day, regularly hold promotions and special offers, or have low-stock items that sell fast, then microdata will help you:

- keep prices, availability and condition fresh in Google
- validate your products in Google's strict pricing consistency rules
- improve user experience, traffic to your product listings, and higher conversion rates because users already see the correct price on Google Shopping

In contrast, if you choose not to enable automatic item updates in your account, items with mismatched price and availability will be temporarily disapproved rather than updated.

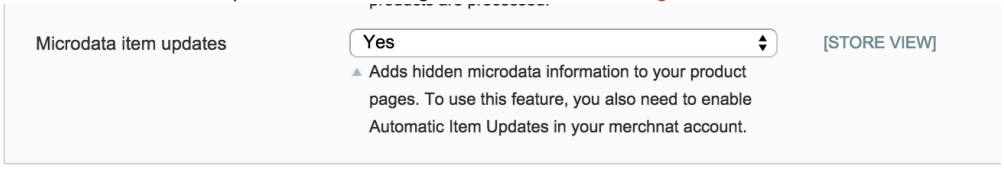
> If you don't update prices more than once per day, you can still benefit from Microdata for stock information:
>
> - Google will be able to see when a product goes out-of-stock, or **back in stock**, and mark it in Shopping.
> - This will prevent item rejection due to mismatched stock info.

Prerequisite

You'll need to have price and availability columns in the feed, these are enabled and set up by default.

Step-by-step guide

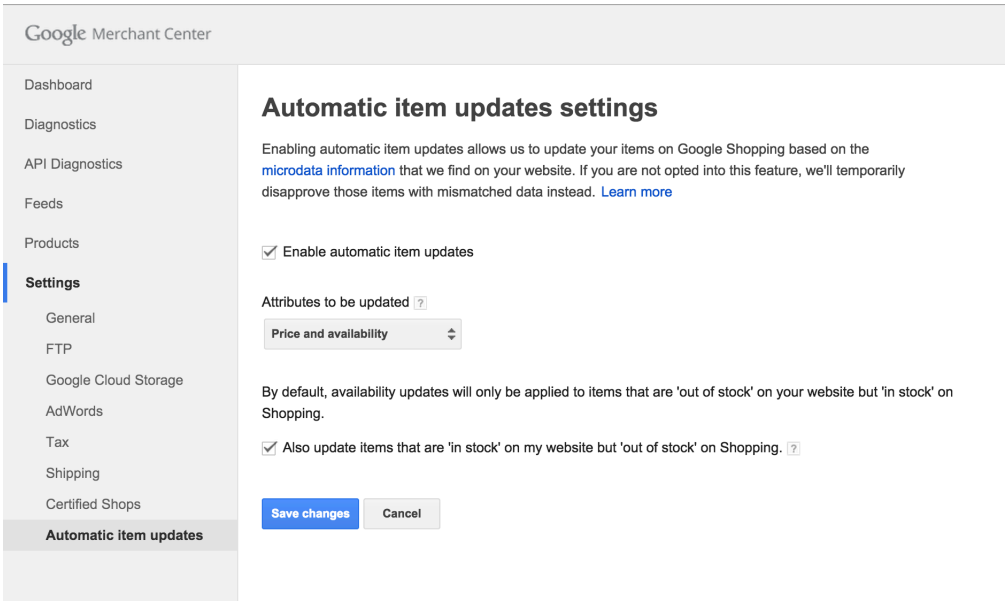
1. Activate Microdata item updates in feed configuration under File Settings section.



2. Enable automatic item updates in Google Merchant.

- Go to "Settings" in the left-hand menu, then "Automatic item updates"
- Click "Edit Settings"
- Check "Enable automatic item updates"

- Select "Price and Availability" and check "Also update items that are 'in stock' on my website but 'out of stock' on Shopping."
- Save settings



Troubleshooting

Google provides a handy tool to see what microdata is enabled on your product pages.
Just paste the URL to a Product Page and you should see all the items on that page with their respective price and stock status.

http://www.google.com/webmasters/tools/richsnippets

> If you use another extension that outputs microdata in the Product Page, you should consider disabling it, or removing price and stock info from its output.
> Our extension will always output the exact **Product ID**, **availability**, **price** and **currency** that you send in the feed - this is important for successful Google Merchant validation.

Microdata missing?

You should first verify that this particular product is included in the feed.

- if the product is not in the feed, then microdata may not appear correctly
- if the product is missing Stock and Price columns in the feed, then microdata will be missing them as well
- and that all caches are cleared - you'll need to see a fresh Product Page for the new info to work. So a full clear of Full Page Cache / Block Cache / 3rd party Varnish caches will be needed before your product pages will contain microdata. The "Flush Cache Storage" button will clear all caches in a default Magento installation

If you see any errors, or need us to troubleshoot, don't hesitate to reach out at http://help.rocketweb.com/

Related articles

- File Settings
- Setup microdata to enable automatic pricing updates
- Categorize Products in the Feed
- Getting Started
- Submitting your feed to Google

9 related results

Use pricing from simple child products

If you use configurable products and have different pricing for each option, then you might want to set prices on the Simple products associated with your configurables.

By default, Magento will use the price you set on the configurable product, for all pricing: normal, special price, tier price etc. If you want to set different prices for more expensive product options, you have to go to the "Associated Products" tab of the configurable product, and set extra pricing

Supported extensions

To use this feature, your store needs to have one of the following extensions:

for the options, that will get added to the configurable price - see more details in the Magento documentation.

1. SCP (since 1.5.18)
2. Simple Product Pricing (since 1.6.0)

Step-by-step guide

The default Magento setup described above may prove tedious if you have many different prices. Instead, you can choose to disregard the price of the configurable product, and use "price" and "special price" from the simple children.

1. You need to install one of these two extensions:
    a. SCP http://www.magentocommerce.com/magento-connect/simple-configurable-products.html
    b. Simple Product Pricing http://www.magentocommerce.com/magento-connect/simple-product-pricing.html

> Installing these extensions will affect your live site - all prices for configurable products in catalog and checkout will be changed to use prices set on the simple children only!

2. All configurable pricing in the website and shopping feed will now be meaningless; you need to set price (and special price if needed) on each simple product
3. You don't need to configure anything else; just re-generate the feed and the new pricing will be in place
    a. simple children will have their own separate prices
    b. if you include the configurable in the feed, it will have the minimum price from its children, but it's own special price

> If you use Configurable Product overwrites in the Apparel section, please make sure you do **not** select Special Price or Price. It will overwrite the SCP price.

Related articles

- Use pricing from simple child products
- Setting up products with Minimum Advertised Pricing
- Configurable products
- Set up stores and feeds for multiple countries

Pushing from staging to production store

The feed extension does not usually conflict with other extensions and it's designed to minimize the risk of breaking stores. Still, sometimes it makes sense to configure the feed on a staging site to get that risk as low as possible, but also it can be useful to asses catalog data without making any changes on live store.

> **Backing up**
> Pushing the feed to production store will alter the live store database, so it is highly important that a backup is created before taking any action on production.

Step-by-step guide

1. Install the extension on live store using steps from Installation guide. You may skip this step you're only pushing feed configuration changes.
2. Push product changes you may have done on staging, for filling in feed missing data.
3. Copy feed configuration from staging to live store.
4. If you have setup custom cronjobs on staging, you'll have to redo them on live store. Follow update schedule checklist to not miss things.

Copy feed configuration

This can be achieves either by manually redoing all the configuration through the admin, the same way it was done on staging, or you could copy over the database records of it.

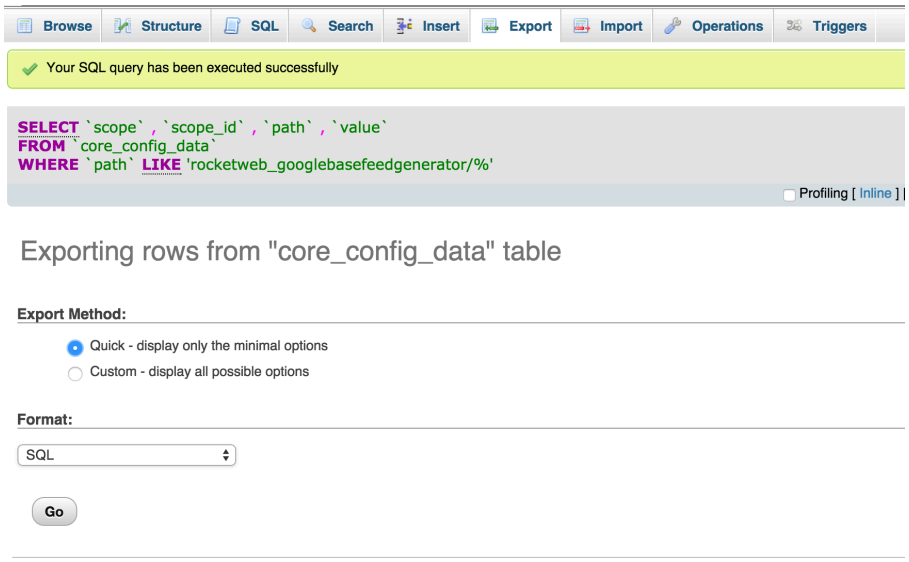
> **Versions**
> When copying configuration through sql export, make sure the same extension version is installed on both servers.

Feed configuration resides in the store's database, and to extract it, run the following sql query:

export

```
SELECT `scope`, `scope_id`, `path`, `value` FROM `core_config_data` WHERE `path` LIKE
'rocketweb_googlebasefeedgenerator/%';
```

If you have access to phpmyadmin, run the above query on the staging DB, than export the result into SQL format.



You should be obtaining a sql file similar to this example:

core_config_data.sql

```
INSERT INTO `core_config_data` (`scope`, `scope_id`, `path`, `value`) VALUES
('default', 0, 'rocketweb_googlebasefeedgenerator/settings/is_turned_on', '1'),
('default', 0, 'rocketweb_googlebasefeedgenerator/settings/locale', 'en_US'),
('default', 0, 'rocketweb_googlebasefeedgenerator/settings/alternate_feed_dir', NULL),
('default', 0, 'rocketweb_googlebasefeedgenerator/settings/category_tree_include', '3,
4, 5, 20'),
```

To import the exported configuration on the live server, first delete any pre-existing configuration on live:

```
DELETE FROM `core_config_data` WHERE `path` LIKE
'rocketweb_googlebasefeedgenerator/%';
```

Than run exported sql file from staging on live database.

> When importing the configuration though SQL export, make sure both staging and live have the same store views defined, with same store IDs.

For submitting the live feed to merchnat center, please see Submitting your feed to Google.

Related articles

- File Settings
- Setup microdata to enable automatic pricing updates
- Categorize Products in the Feed

- Getting Started
- Submitting your feed to Google

8 related results

Setting up products with Minimum Advertised Pricing

> Make sure your MSR price display settings in magento comply with Google Shopping Policies. Your products would be disapproved if the price from the feed differs from the one on product pages, so you probably have to display the real price aside msrp on the pages. The most appropriate option to do that is by setting Display Actual Price to gestures on your products.

Step-by-step guide

1. In Columns Map, set the price column to map on msrp attribute.
2. Under Replace empty values from Product Filters, add a rule to replace column price with the directive called Price - gives you the normal price.

This has the following consequences:

- For simple products, it will use MSRP if exists, if not it will use the normal price.
- For configurable, it will use the configurable MSRP. If empty, will use the configurable normal price.
- For children of configurable, it uses MSRP if exists. If empty, will get the price from the configurable + options. If you're using SCP (simple configurable products), it will in fact use simple pricing.

The sale_price will not take MSRP into consideration as promos won't be ever applied to msrp. You could output MSRP in lieu of sale_price if you'd like, just do the same 2 steps outlined above.

> You shouldn't overwrite msrp or price attributes in the Configurable attribute overwrite section. This may lead to weirdness. Set MSRP on configurable products and all their children.

Related articles

- GSF User's Guide
- Directives
- Columns Map
- File Settings
- Features

27 related results