**eMAG Marketplace**

**Product Families API Best Practices v1.1**

**05.05.2015**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date modified** | **Changes** |
| 1.0 | 08.04.2015 | First draft |
| 1.1 | 05.05.2015 | Small examples added |

Table of Contents

[1. Product families 2](#_Toc418499326)

[2. Reading product family information 2](#_Toc418499327)

[3. Family characteristics 4](#_Toc418499328)

[3.1. ID 4](#_Toc418499329)

[3.2. Name 4](#_Toc418499330)

[3.3. Family\_type\_id 4](#_Toc418499331)

[4. Family examples 4](#_Toc418499332)

[4.1. Product has color-size variation 4](#_Toc418499333)

[4.2. Product has only size variation 5](#_Toc418499334)

[4.3. Produsul este disponibil doar in variatii de culoare 5](#_Toc418499335)

[5. Best practices 6](#_Toc418499336)

# Product families

Product families are very useful for grouping same products with variations of one or more attributes into one single page. The translation used in other e-commerce platforms are product variants, grouped products, product variations.

Example: <http://www.emag.bg/mobilen-telefon-apple-iphone-6-16gb-space-grey-iphone-6-16-gb-space-grey/pd/DM51RBBBM/> (the product iPhone 6 has variations based on color and capacity)

A product family has the following characteristics:

*id* – static, unique key for each family, numeric only

*name* – family title or name, unique for each family

*family\_type\_id* – family type (color-size, color, size)

# Reading product family information

Using **category/read** route through API, you can get information about the available family types for a specific category.

For example, for *Men sneakers* category, using **category/read** route, you will get the following information regarding families:

Table 1 -

|  |
| --- |
| 'family\_types' =>  array (  0 =>  array (  'id' => '419',  'name' => 'color-size,  'characteristics' =>  array (  0 =>  array (  'characteristic\_id' => '5401',  'characteristic\_family\_type\_id' => '1',  'is\_foldable' => '1',  'display\_order' => '2',  ),  1 =>  array (  'characteristic\_id' => '6506',  'characteristic\_family\_type\_id' => '3',  'is\_foldable' => '0',  'display\_order' => '1',  ),  ),  ),  1 =>  array (  'id' => '420',  'name' => color,  'characteristics' =>  array (  0 =>  array (  'characteristic\_id' => '5401',  'characteristic\_family\_type\_id' => '1',  'is\_foldable' => '0',  'display\_order' => '1',  ),  ),  ),  2 =>  array (  'id' => '421',  'name' => size,  'characteristics' =>  array (  0 =>  array (  'characteristic\_id' => '6506',  'characteristic\_family\_type\_id' => '3',  'is\_foldable' => '1',  'display\_order' => '1',  ),  ),  ),  ), |

As you can see, for the above category, we have 3 types of product families available:

* Color-size, with id 419. This family groups products by Color (id 5401) and Size (id 6506) attributes.
* Color, with id 420. This family groups products only by Color (id 5401) attribute.
* Size, with id 421. This family groups products only by Size (id 6506) attribute.

# Family characteristics

## ID

The id of the family is an unique, static, identification key. It can be a number, bigger than 0 and must be unique for each family. All products grouped in the same family must have the same family id in the API call. The id of the family must not be changed between two product updates (import/API update).

Example: id=219

## Name

The name (title) of the family must be unique for each family. You cannot create two different families (with different IDs) with the same name. The name of the family must be related to the name of the product that is included in it, but it must exclude variations (size, color). All products part of the same family must have the same family name on API calls. The name of the family must not be changed between two product updates (import/API update).

Example: name=Sport shoes Nike Dual Fusion Trail for men

## Family\_type\_id

The ID of the family, returned by making a category/read call, for each category. All products part of the same family must have the same family\_type\_id on API calls. The family type id must not be changed between two product updates (import/API update).

Example: family\_type\_id=419

# Family examples

## Product has color-size variation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Product id** | **part\_number** | **Product name** | **color** | **size** |
| 20110 | PNTF031\_Green\_35 | Green Sport shoes , Nike Dual Fusion Trail for men, 35 | Green | 35 EU |
| 20111 | PNTF031\_Green\_36 | Green Sport shoes , Nike Dual Fusion Trail for men, 36 | Green | 36 EU |
| 20112 | PNTF031\_Green\_37 | Green Sport shoes , Nike Dual Fusion Trail for men, 37 | Green | 37 EU |
| 20113 | PNTF031\_Green\_38 | Green Sport shoes , Nike Dual Fusion Trail for men, 38 | Green | 38 EU |
| 20114 | PNTF031\_Green\_39 | Green Sport shoes , Nike Dual Fusion Trail for men, 39 | Green | 39 EU |
| 20115 | PNTF031\_Green\_40 | Green Sport shoes , Nike Dual Fusion Trail for men, 40 | Green | 40 EU |
| 20116 | PNTF031\_Red\_35 | Red sport shoes, Nike Dual Fusion Trail for men, 35 | Red | 35 EU |
| 20117 | PNTF031\_Red\_36 | Red sport shoes, Nike Dual Fusion Trail for men, 36 | Red | 36 EU |
| 20118 | PNTF031\_Red\_37 | Red sport shoes, Nike Dual Fusion Trail for men, 37 | Red | 37 EU |
| 20119 | PNTF031\_Red\_38 | Red sport shoes, Nike Dual Fusion Trail for men, 38 | Red | 38 EU |
| 20120 | PNTF031\_Red\_39 | Red sport shoes, Nike Dual Fusion Trail for men, 39 | Red | 39 EU |

All these products are the same model, but with different color and size variations. They are part of the same color-size family so the API call on product\_offer/save must have the following family array:

|  |
| --- |
| [family] => Array  (  [id] => 201  [name] => Sport shoes , Nike Dual Fusion Trail for men  [family\_type\_id] => 419  ) |

where [id] – unique identifier of the family

[name] – product title, excluding color size variation

[family\_type\_id] – the id for the type of the family for this category

## Product has only size variation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Productid** | **part\_number** | **Product name** | **color** | **size** |
| 30110 | PNTF045\_Blue\_35 | Blue sport shoes, Nike Downshifter for men, 35 | Blue | 35 EU |
| 30111 | PNTF045\_Blue\_36 | Blue sport shoes, Nike Downshifter for men, 36 | Blue | 36 EU |
| 30112 | PNTF045\_Blue\_37 | Blue sport shoes, Nike Downshifter for men, 37 | Blue | 37 EU |
| 30113 | PNTF045\_Blue\_38 | Blue sport shoes, Nike Downshifter for men, 38 | Blue | 38 EU |
| 30114 | PNTF045\_Blue\_39 | Blue sport shoes, Nike Downshifter for men, 39 | Blue | 39 EU |
| 30115 | PNTF045\_Blue\_40 | Blue sport shoes, Nike Downshifter for men, 40 | Blue | 40 EU |

All these products are the same model, but with different size variations. They are part of the same size family so the API call on product\_offer/save must have the following family array:

[family] => Array

(

[id] => 301

[name] => Sport shoes Nike Downshifter for men

[family\_type\_id] => 421

)

Where

* [id] – unique identifier of the family
* [name] – product title, excluding size variation
* [family\_type\_id] – the id for the type of the family for this category

## Product has only color variation

This case can be found in fashion items (accessories, jewelry).

# Best practices

* You cannot create two or more families with the same title. We recommend that the titles of the products to be created in such a way that there will be no possibility to create two families with the same name, but with different products. To be sure you have unique titles, you can also use in the title the product codes.
  + Product title: Casual blue shoes, natural leather, PNTF012, for men, 35
  + Family name: Casual shoes, natural leather, PNTF012, for men
* Please make sure you double check the family names, to be certain you do not create two titles that look identical, but are in fact different because of extra spaces between words.
* The family IDs don’t have to be changed between two product updates. Changing these values between two updates on the same products will lead to removing the products from the families initially created.
* The family IDs must be the same for all products part of the same family. If you send a different family ID for one product, it will not group with the rest of the products.
* For color-size families, the attribute *color* cannot be identical for two or more products that have the same size. Example: you cannot group in the same color family two dresses (same model/design) if the color attribute has the value Multicolor for all sizes.

 

In this case, in order to display the products correctly, you must create two separate families (id, name).