**eMAG Marketplace**

**API Documentation v4.3.5**

**08.10.2019**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date modified** | **Changes** |
| 3.9.9 | 20.07.2016 | Added Product Matching resource |
| 4.0.0 | 21.11.2016 | Added number of offers, buy button rank, best sale price & best recommended price on product\_offer/read  Added min / max sale\_price keys on API  Added Attachment Upload resource |
| 4.1.0 | 10.04.2017 | Improved AUTH method using base64 encoding  Added force\_images\_download key to product\_offer/save  Added characteristic type\_id to category/read  Added Handling Time resource  Added Courier Account resource  Added Return Requests (RMA) resource |
| 4.1.1 | 01.08.2017 | Added ownership\_id on product\_offer/read  Added is\_filter to category/read  Added EAN key to category/read, product\_offer/save and product\_offer/read  Removed barcode key from product\_offer/save and product\_offer/read |
| 4.1.2 | 25.09.2017 | Added return\_reason and return\_type description details  Added type key on order read and order filtering  Updated product validation\_status values  Added info about start\_date limitations during special campaigns |
| 4.1.3 | 13.02.2018 | Updated product stock maximum value  Updated order attachment types  Added EAN method for attaching offers on existing products  Added billing\_name and billing\_phone on order/read  Added awb\_format on awb/save |
| 4.2.0 | 01.03.2018 | Added a limit for the number of entities per request  New API endpoints available with 1st of May 2018  Removed awb\_format on awb/save |
| 4.2.1 | 20.03.2018 | Added time based throttling for save requests starting with 1st of May 2018  Added offer\_validation\_status key on product\_offer/read and filter |
| 4.2.2 | 08.05.2018 | Removed product statuses from return requests  Updated return request status change permissions  API endpoints availability change, now beginning with 1st of August 2018  Removed “Invoices”, “Details” and “Observation” keys from order fields |
| 4.3.0 | 03.10.2018 | Removed mandatory label from product url  Added AWBs key on rma/read  Extended AWB read method: AWB number, status, type, courier name, courier ID and courier account ID  Added new method for downloading AWB .zpl format types  Added type key on rma/read  Added attachments, cashed\_co and cashed\_cod keys on order/read  Added callback URLs section  Added rma\_id on awb/save |
| 4.3.1 | 05.02.2019 | Added new resource for saving volume measurments  Added other currencies option on product\_offer/save  Added locker delivery option on order/read and awb/save  Updated the time based throttling for save requests |
| 4.3.2 | 18.02.2019 | Order status matrix update  Important changes in permisions for updating orders (3.5) and removing products from an order (3.5.1)  Added supply\_lead\_time key on product\_offer/save |
| 4.3.4 | 01.10.2019 | Added rate limiting for invalid requests  Added new order cancelation reasons  Added “locker\_name” key on order/read  Adjusted the maximum number of elements returned for all read requests  Adjust measurements values decimal length |
| 4.3.5 | 08.10.2019 | Changed the warranty requirement regime |
| 4.3.6 | To be released: H1 2020 | Added product\_ranking on product\_offer/read  Added new AWB PDF file read resource |

Table of Contents

[1. eMAG Marketplace API 5](#_Toc21374700)

[1.1. Conventions 5](#_Toc21374701)

[1.2. Request, resources and actions 6](#_Toc21374702)

[1.3. Pagination and filters 7](#_Toc21374703)

[1.4. Response 7](#_Toc21374704)

[1.5. Rate limiting 8](#_Toc21374705)

[1.6. Callback URLs 8](#_Toc21374706)

[2. Publishing products and offers 8](#_Toc21374707)

[2.1. Reading categories, characteristics and family\_types 9](#_Toc21374708)

[2.2. Reading VAT rates 11](#_Toc21374709)

[2.3. Reading Handling Time values 11](#_Toc21374710)

[2.4. Sending a new product 11](#_Toc21374711)

[2.4.1. Example for a new product 16](#_Toc21374712)

[2.5. Updating existing offer 16](#_Toc21374713)

[2.6. Saving volume measurements on products 17](#_Toc21374714)

[2.7. Reading and counting products and offers 17](#_Toc21374715)

[2.8. Product validation responses 21](#_Toc21374716)

[2.9. Matching products 21](#_Toc21374717)

[2.10. Attaching offers on existing products 21](#_Toc21374718)

[3. Processing orders 24](#_Toc21374719)

[3.1. Order fields 24](#_Toc21374720)

[3.1.1. Product field in order details 27](#_Toc21374721)

[3.1.2. Customer fields in order details 27](#_Toc21374722)

[3.1.3. Order invoices 29](#_Toc21374723)

[3.2. Order notification, acknowledgment and order filters 29](#_Toc21374724)

[3.3. Order status matrix 30](#_Toc21374725)

[3.4. Order filters 30](#_Toc21374726)

[3.5. Updating orders 32](#_Toc21374727)

[3.5.1. Removing products from an order 32](#_Toc21374728)

[3.5.2. Adding products to an existing order 32](#_Toc21374729)

[3.5.3. Returned products and storno route 32](#_Toc21374730)

[4. Shipping orders 35](#_Toc21374731)

[4.1. Saving AWB 36](#_Toc21374732)

[4.2. Reading AWB 37](#_Toc21374733)

[4.3. Reading AWB PDF files 39](#_Toc21374734)

[4.4. Reading AWB ZPL type 39](#_Toc21374735)

[4.5. Counting Localities 40](#_Toc21374736)

[4.6. Reading Localities 40](#_Toc21374737)

[4.7. Reading courier accounts 41](#_Toc21374738)

[5. Processing return requests 41](#_Toc21374739)

[5.1. Return requests filters 45](#_Toc21374740)

[5.2. Status change permissions 46](#_Toc21374741)

[5.3. Return request deliveries 46](#_Toc21374742)

[5.4. Examples requests and responses 46](#_Toc21374743)

# eMAG Marketplace API

eMAG Marketplace API is developed by eMAG for Marketplace partners in order to allow them to use their own CRM’s / ERP’s. This document explains the methods for calling the API.

The API can be used in order to:

* send products and offers
* process orders

## Conventions

We define MARKETPLACE\_API\_URL constant of being the API URL of the platform (ex: <https://marketplace.emag.ro/api-3>)

We define MARKETPLACE\_URL constant of being the URL of the platform (ex: [https://marketplace.emag.ro](https://marketplace.emag.ro/))

We define DEFAULT\_CURRENCY constant of being the default currency of the platform (ex: RON).

All API parameters are key-sensitive.

|  |  |  |
| --- | --- | --- |
| **Platform** | **Romania** | **Bulgaria** |
| **MARKETPLACE\_URL** | [https://marketplace.emag.ro](https://marketplace.emag.ro/) | [https://marketplace.emag.bg](https://marketplace.emag.bg/) |
| **MARKETPLACE\_API\_URL** | <https://marketplace.emag.ro/api-3>  <https://marketplace-api.emag.ro/api-3>\* | <https://marketplace.emag.bg/api-3>  <https://marketplace-api.emag.bg/api-3>\* |
| **Protocol** | HTTPS | HTTPS |
| **Locale** | ro\_RO | bg\_BG |
| **Default currency** | RON | BGN |

*\*to be released (Aug 2018)*

|  |  |  |
| --- | --- | --- |
| **Platform** | **Hungary** | **Poland** |
| **MARKETPLACE\_URL** | [https://marketplace.emag.](https://marketplace.emag.hu/)hu | <https://marketplace.emag.pl> |
| **MARKETPLACE\_API\_URL** | <https://marketplace.emag.hu/api-3>  <https://marketplace-api.emag.hu/api-3>\* | <https://marketplace.emag.pl/api-3>  <https://marketplace-api.emag.pl/api-3>\* |
| **Protocol** | HTTPS | HTTPS |
| **Locale** | hu\_HU | pl\_PL |
| **Default currency** | HUF | PLN |

*\*to be released (Aug 2018)*

To access the API, simply make a Basic Authorization request with your username, password and a base64 computed hash. Please note that user should be granted API rights in order to access the API.

|  |
| --- |
| $hash = base64\_encode($username . ':' . $password); |

## Request, resources and actions

A Marketplace API call is represented by sending a request to API URL of platform. Every request consists of a POST to an URL like:

MARKETPLACE\_API\_URL/**resource/action**

Ex: [https://marketplace.emag.ro/api-3/product\_offer/save](https://marketplace.emag.ro/api-2/product_offer/save)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **RESOURCES AND AVAILABLE ACTIONS** | | | | | | |
| **Resource** | **Resource URL** | **Available actions** | | | | |
| **product\_offer** | MARKETPLACE\_API\_URL/product\_offer | read | save | count |  | match |
| **measurements** | MARKETPLACE\_API\_URL/measurements |  | save |  |  |  |
| **order** | MARKETPLACE\_API\_URL/api-3/order | read | save | count | acknowledge |  |
| **order/attachments** | MARKETPLACE\_API\_URL/order/attachments |  | save |  |  |  |
| **message** | MARKETPLACE\_API\_URL/message | read | save | count |  |  |
| **category** | MARKETPLACE\_API\_URL/category | read |  | count |  |  |
| **vat** | MARKETPLACE\_API\_URL/vat | read |  |  |  |  |
| **handling\_time** | MARKETPLACE\_API\_URL/handling\_time | read |  |  |  |  |
| **locality** | MARKETPLACE\_API\_URL/locality | read |  | count |  |  |
| **courier\_accounts** | MARKETPLACE\_API\_URL/courier\_accounts | read |  |  |  |  |
| **awb** | MARKETPLACE\_API\_URL/awb | read | save |  |  | read\_pdf\* |
| **rma** | MARKETPLACE\_API\_URL/rma | read | save |  |  |  |

*\*to be released (H2 2018)*

Below a code example using the resource "category" and the action "read"

|  |  |  |
| --- | --- | --- |
| **Resource** | **Example** | **Context** |
| **category/read** |  | http method: POST |

The API needs authorization and has an IP level filtering. Before testing, sellers should provide a list of whitelisted IP’s. eMAG Marketplace will only allow API calls only from those IP’s.

The POST data consist of 1 mandatory key:

|  |  |
| --- | --- |
| **REQUEST** | |
| **Key** | Description |
| **data** | Data to be passed to the API. The following document will describe keys. |

## Pagination and filters

In order to limit the number of items returned, read actions accept pagination by passing to POST data following parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| **PAGINATION** | | | |
| **Key** | **Description** | **Default value** | **Example** |
| **currentPage** | Set current page displayed | 1 | currentPage =3 |
| **itemsPerPage** | Set number of items to be displayed in one page. Maximum is set to 100. | 100 | itemsPerPage=50 |

Also, filters can be included in POST to refine result set. Filters vary depending on the resource called and are exampled on every resource section.

## Response

When an API call is made, the server MUST reply with a response. The response will ALWAYS be JSON formatted and the header 'Content-type: application/json' will always be passed.

|  |  |
| --- | --- |
| **RESPONSE** | |
| **Key** | **Description** |
| **isError** | Boolean value representing response status. |
| **messages** | Messages included in the response, like error messages, etc. |
| **results** | Results included in the response, mostly when reading resources. |

**IMPORTANT:** Every API request must have a response and the response must contain the key “isError” and its value must be “false”. For each call that does not have the key and “false” value, we recommend setting up alerts, as the call most likely was not interpreted. We also recommend logging all calls and the corresponding API response for a 30 days period.  
Every request must have at most 4000 elements. If the call surpasses this limit the call will have a response with key “isError:true” and “message: Maximum input vars of 4000 exceeded”.

In the event of an documentation error when saving a product, the API will return and “isError:true” message, but the new offer will be saved and processed.

## Rate limiting

The **product\_offer/save** resource has the following limits:

* 50 entities per request. For optimal performance we recommend using between 10 and 50 entities per request.
* Maximum 1 request every 3 seconds and maximum 20 requests every 1 minute. For optimal performance we recommend not scheduling requests at fixed hours. For example use 12:04:42 as a starting point instead of 12:00:00. The following responses are possible:

|  |  |
| --- | --- |
| **Time throttling limit is exceeded** | **Time throttling limit was not reached** |
| HTTP/1.1 429  Date: Wed, 21 Mar 2018 08:22:44 GMT  Content-Type: application/json; charset=utf-8  Transfer-Encoding: chunked  X-RateLimit-Limit-3second: 1  X-RateLimit-Remaining-3second: 0  Server: kong/0.12.1  {"message":"API rate limit exceeded"} | HTTP/1.1 200  Date: Wed, 21 Mar 2018 08:23:52 GMT  Content-Type: application/json  Transfer-Encoding: chunked  X-RateLimit-Limit-3second: 1  X-RateLimit-Remaining-3second: 0  Server: nginx  {"isError":false,"messages":[],"results":[]} |

Invalid requests will also be limited to a maximum number of 3 requests every 1 minute or 180 requests every 1 hour.

## Callback URLs

Thefollowing callback URLs can be activated from the Marketplace interface:

|  |  |
| --- | --- |
| **Context** | **Description** |
| **New order** | You will be notified for each new order |
| **Order cancellation** | You will be notified whenever an order in canceled |
| **New return & status change** | You will be notified for each new return request and select status changes (New, Acknowledged & Received) |
| **AWB status change** | You will be notified for each AWB status change |

# Publishing products and offers

We define a **product** as a list of elements displayed in in eMAG platform for a product page. The elements are:

* Name
* Brand
* Part number
* Description
* Images
* Product characteristics (and product families)
* Category
* Barcodes (optional)
* Other attachments (optional)

We define an **offer** as a list of elements required for an offer to be available for a product. These elements are:

* Price
* VAT rate
* Warranty
* Numerical stock
* Handling\_time

eMAG Marketplace API allows a seller to:

* Send new products and offers
* Send new offers for existing eMAG products (sold by eMAG or any other seller)
* Update existing own offers and/or products

## Reading categories, characteristics and family\_types

Every eMAG product has to be included in a certain category. Sellers cannot create new categories or change existing ones. Also, a seller can only post products and offers in its allowed categories list.

Reading categories without parameters will generate a response containing the first 100 categories. The read can be paginated thus obtaining a full list of categories. Only active categories will be returned.

When passing a category id, the API will return the category name and the list of available characteristics and their corresponding IDs, as well as the available product family\_types and their corresponding IDs. Reading categories one by one is important as it is the only way to identify the restrictive characteristics and corresponding allowed values.

You can read the categories and their characteristics through the API.

The resource is **category** and the available actions are **read** and **count**.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **CATEGORY – read** | | | |
| **Key – level 1** | **Key – level 2** | **Key – level 3** | **Description** | **Type** | **Example** |
| id |  |  | Category eMAG id | Integer | id=604 |
| name |  |  | Category name | String | name=”Music” |
| is\_allowed |  |  | Indicates if the seller can send products and offers in the category. In order to request access to a specific category, you can use the Marketplace interface.  0 = No 1 = Yes. | Integer | is\_allowed=0 |
| parent\_id |  |  | Id of the parent category | Integer | parent = 12 |
| is\_ean\_mandatory |  |  | Indicates if the sending an EAN is mandatory when saving products   * 1 = mandatory * 0 = optional | Integer | is\_ean\_mandatory =1 |
| is\_warranty\_mandatory |  |  | Indicates if adding warranty is mandatory when saving products   * 1 = mandatory * 0 = optional | Integer | is\_warranty\_mandatory =1 |
| characteristics |  |  | All characteristics available in category | List of arrays |  |
|  | id |  | Characteristic eMAG id | Integer | id=38 |
|  | name |  | Characteristic name | String | name=”Audio” |
|  | type\_id |  | Characteristic type. Indicates the type of values that a characteristic can accept.  **Single value characteristics**:   * 1 = Numeric (ex: 20, 1, 30, 40, etc) * 60 = Size (ex. 36 EU, XL INTL, etc) * 20 = Boolean ( Yes, No, N/A)   **Multiple values characteristics**:   * 2 = Numeric + unit of measure ( ex. 30 cm, 45 m, 20 GB, 30 inch, etc) * 11 = Text Fixed (max length 255 chars) (ex. Blue, Green, Laptop, Notebook, Copil, Man, Woman) * 30 = Resolution ( Width x Height ) (ex. 100 x 200, 640 x 480) * 40 = Volume (Width x Height x Depth - Depth 2) (ex. 30 x 40 x 50 - 10) | Integer | type\_id=11 |
|  | display\_order |  | Characteristic display order | Integer | display\_order=6 |
|  | is\_mandatory |  | Indicates if the characteristic is mandatory when sending a product. Possible values are 0 (the characteristic is not mandatory) and 1 (the characteristic mandatory). | Integer | is\_mandatory=0 |
|  | is\_filter |  | Indicates if the characteristic represents a filter in the website. Possible values are 0 (the characteristic is not filter) and 1 (the characteristic is a filter). | Integer | is\_filter=1 |
|  | allow\_new\_value |  | Indicates if the current characteristic allows you to submit new values that are automatically validated. | Integer | allow\_new\_value=0 |
|  | values |  | List the first 256 existing values of the current characteristic. | Array | 0 => 'Value 1'  1 => 'Value 2' |
| family\_types |  |  | List of all family types available in category | List of arrays |  |
|  | id |  | Family type id | Integer | Id=95 |
|  | name |  | Family name | String | name=”Quantity” |
|  | characteristics |  | All characteristics of current family type | List of arrays |  |
|  |  | characteristic\_id | Characteristic Id | Integer | characteristic\_id=44 |
|  |  | characteristic\_family\_type\_id | Can only have 3 values, each corresponding to a different display method: “1" ="Thumbnails";  "2"="Combobox";  "3"="Graphic Selection" | Integer | characteristic\_family\_type\_id=2 |
|  |  | is\_foldable | A foldable characteristic wraps all family members (with different characteristic values) as one item in the eMAG category listing | Integer | Is\_foldable=1 |
|  |  | display\_order | Characteristic display order | Integer |  |

## Reading VAT rates

When sending an offer, you have to send the VAT rate id by sending us a valid VAT id.

The resource is ***vat*** and the action is ***read***. The API will return the list of available VAT rates and their corresponding id’s.

## Reading Handling Time values

When sending an offer, you have to send the Handling Time by sending us a valid value.

The resource is **handling\_time** and the action is **read**. The API will return the list of available handling\_time values.

## Sending a new product

Sending a product for the first time requires you to send the entire product documentation and all the offer data. Please note that creating new products implies human validation, so a new product will not be displayed in eMAG platform immediately.

The products that are not compliant with eMAG Documentation Standard will not pass the human validation; in this case you will be notified by our support team. The eMAG Documentation Standard that is available upon request for each category, and it contains the best practices for documenting a product.

In order to send a new product, the resource is product\_offer and the available action is save.

| **PRODUCT OFFER – save and create/update product** | | | | |
| --- | --- | --- | --- | --- |
| **Key – level 1** | **Key – level 2** | **Description** | **Constraints** | **Example** |
| id |  | Seller internal product id. This is the primary key for identifying a product offer. | Required. Integer value between 1 and 16777215. | id=243409 |
| category\_id |  | Product category eMAG id. | Required. Integer between 1 and 65535. | category \_id=506 |
| vendor\_category\_id |  | Seller internal category id. | Integer. Optional. | vendor\_category\_id=506 |
| part\_number\_key |  | eMAG part\_number\_key. Used for attaching a product offer to an existing product in eMAG platform. If you want to create new product, don’t set this key. | Optional. String. Will be validated. | part\_number\_key=ES0NKBBBM |
| name |  | Product name. Should be consistent with eMAG Product Documentation Standard. | Required. String between 1 and 255 characters long. | name=”Test product” |
| part\_number |  | Manufacturer unique identifier of the product. | Required. String between 1 and 128 characters. | part\_number=”md788hc/a” |
| description |  | Product description. Should be consistent with eMAG Product Documentation Standard. | Optional. String between 1 and 16777215 characters. Can contain basic HTML tags. | description=”test” |
| brand |  | Brand name. Should be consistent with eMAG Product Documentation Standard. | Required. String between 1 and 255 characters. | brand=”Brand test” |
| weight |  | The weight of the product | Optional. Decimal value between 0 and 999999. Up to six decimals. | weight=12.123456 |
| force\_images\_download |  | Image attachement redownload flag. Only to be used when updating product documentation  1-forces images redownload  0-images will not be redownloaded | Optional. Integer value, 1 or 0. Default = 0 | force\_images\_download=1 |
| images |  | Product images data array. | Optional. List of arrays. |  |
|  | display\_type | Image display type.  1 – main image  2 – secondary image  0 – other images | Optional. Default value 0. Integer value between 0 and 2. | display\_type=1 |
|  | url | Seller image URL. Should be consistent with eMAG Product Documentation Standard. Max 6000px x 6000px and 8 Mb in size. | Required. String between 1 and 1024 characters. Valid URL. JPG, JPEG or PNG file type. | url=”http://valid-url.jpg” |
| characteristics |  | Characteristic data. Note that characteristics have to be category valid (be part of category template). Should be consistent with eMAG Product Documentation Standard. | Optional. List of arrays. |  |
|  | id | Characteristic eMAG id. | Required. Integer value between 1 and 65535 | id=24 |
|  | value | Characteristic value. Should be consistent with eMAG Product Documentation Standard. | Required. String between 1 and 255 characters | value=”test” |
| family |  | Family array. Used to create a new family, add a product to an existing family, or removing a product from a family. | Optional. Array. |  |
|  | id | The unique integer identifier of the family in your platform. If set to 0 (id=0), the product will be removed from its current family. | Required. Integer | Id=0 |
|  | name | Required. Seller Family name. | Required if family id is not equal to 0; | name="Test product" |
|  | family\_type\_id | Required. eMAG Family type id that can be acquired by API (the resource is **category** and the action is **read**). | Required if family id is not equal to 0. Integer. | family\_type\_id=95 |
| url |  | Product URL on the seller website. | Optional. String between 1 and 1024 characters. | url=”http://valid-url.html” |
| warranty |  | The warranty offered in months. | Required/Optional based on category.  Default value:  - 0 (no warranty) if optional  - No default if required.  Integer between 0 and 255. | warranty=24 |
| ean |  | Product barcode identifier (EAN -8, EAN-13, UPC-A, UPC-E, JAN, ISBN-10, ISBN-13, ISSN, ISMN-10, ISMN-13, GTIN-14). Please use the supplier barcode, not your internal barcodes. | Required/Optional based on category. No default value. Array of strings between 6 and 14 characters long. Only numeric figures allowed. | ean=Array('ean1', 'ean2') |
| attachments |  | Product attachments data. Max 10 Mb in size. | Optional. List of arrays. |  |
|  | id | Seller attachment internal id. | Optional. Integer value between 1 and 4294967295. | id=123 |
|  | url | Seller attachment URL. | Required. String between 1 and 1024 characters. Valid URL to document. | url=”http://valid-url” |
| status |  | Seller offer status.  1 – status active  0 – status inactive | Required. Integer value, 1 or 0. | status=1 |
| sale\_price |  | Seller offer sale price without VAT | Required. Decimal value greater than 0. Up to four decimals. | sale\_price=51.6477 |
| recommended\_price |  | Seller offer recommended retail price before discount, without VAT. If set, the offer will be displayed as promo. | Optional. Decimal value greater than 0. Up to four decimals. Must be greater than sale\_price. | recommended\_price=51.6477 |
| min\_sale\_price |  | Seller’s min offer sale price without VAT | Required on first product save. Decimal value greater than 0. Up to four decimals. | min\_sale\_price=40.6477 |
| max\_sale\_price |  | Seller’s max offer sale price without VAT | Required on first product save. Decimal value greater than 0. Up to four decimals. Must be greater than min\_sale\_price. | max\_sale\_price=60.6477 |
| currency\_type |  | Offer currency. Only send the key if it is different from the local Marketplace currency. Available options: EUR or PLN | Optional. 3 characters string. | currency\_type=”EUR” |
| stock |  | Offer available quantity array. | Required. List of arrays. | {  0=>{  warehouse\_id=1,  value=20}} |
|  | warehouse\_id | The id of the warehouse. Use warehouse\_id=1 for only one warehouse. | Required inside stock array. Integer. | warehouse\_id=1 |
|  | value | Offer available quantity. | Required inside stock array. Integer between 0 and 65535. | value=20 |
| handling\_time |  | Handling time array. If no array is sent, the products are shipped the same day they are received. | Optional. List of arrays. | {  0=>{  warehouse\_id=1,  value=1}} |
|  | warehouse\_id | The id of the warehouse. Use warehouse\_id=1 for only one warehouse. | Required inside handling\_time array. Integer. | warehouse\_id=1 |
|  | value | Handling time, in number of days counted from the day the order was received. If handling\_time = 0 the order will be shipped the same day it is received. | Required inside handling\_time array. Integer value between 0 and 255. Default value = 0. | value=0 |
| supply\_lead\_time |  | The number of days needed to restock the product, from order placement to the supplier or production order, until product reception in the warehouse.  Available values for this key are: 2, 3, 5, 7, 14, 30, 60, 90, 120 | Optional. Integer.  Default value = 14 | supply\_lead\_time=5 |
| start\_date |  | If it's a new offer, it represents the date your offer will be available from. For offer updates, it schedules value updates for the following data:   * sale\_price * recommended\_price * stock * handling\_time * vat\_id * warranty * status   All other data will be updated on the fly. Using start\_date, for example, you can schedule the inactivation of an offer, a price update, etc. | Optional. Text in YYYY-MM-DD format. Date can be as far as 60 days in the future (cannot be earlier than tomorrow). Cannot be null. | start\_date=”2014-12-31” |
| vat\_id |  | Seller offer VAT rate id. Use /vat/read to display possible values. | Required. Integer. | Ex: vat \_id=1 |

**IMPORTANT:**

* During campaings with stock in site we will not allow the following:
  + regular offer updates;
  + updates sent during campaign time that have a start\_date in the future;
  + previously scheduled updates with start\_date during campaign time.
* You can save an offer update with currency\_type different from the local Marketplace currency and start\_date before the end of the current month. Any update attempt with a start\_date greater than the last day of the current month will be rejected.
* Be aware that prices published directly in the local Marketplace currencies will be overwritten by prices published in other currencies when these are recalculated and published automatically on FX change at the beginning of every month. The automated recalculation can be disabled on request.
* Min / Max sale price keys are used for price check purposes and are mandatory for all calls used to create product/offers for the first time. As a best practice we recommend sending these keys only when you want to change their values.
* Sale price will be validated against min\_sale\_price and max\_sale\_price. Any offer that is not within the specified range will be rejected.
* In order to change a previously sent product image or attachment the url should be different from the one already sent. We reload the images only if the URL differs.
* We recommend sending the product data only upon product create/update, as there is no need to resend product unless it changed. Also we recommend sending the offer data upon changing (no matter the frequency) and at least weekly (even if the offer is the same) rather using periodical sending (crons, agents). You should program marketing campaigns using “start\_date” campaign. Also please offer the possibility for an offer to be attached to an existing eMAG product (using part\_number\_key).
* In the event of an documentation error when saving a product, the API will return and “isError:true” message, but the new offer will be saved and processed.
* When adding a product to a family
  + The category id of a product and the category id of its family type (family\_type\_id) must be the same.
  + All characteristics that define a family must be present and must have a valid value
  + All characteristics that define a family must have a single value
  + If a family is not valid, you will receive a warning response, but the product will be saved/updated
  + When moving a product from one family to another you only have to send the product with its new family type, id and name and make sure you follow the same rules as above

### Example for a new product

|  |  |  |
| --- | --- | --- |
| **Resource** | **Example** | **Context** |
| **product\_offer/save** |  | http method: POST |

## Updating existing offer

When updating an existing offer for a product, you should send only the offer, without the documentation. Mandatory when updating a product offer are the following keys:

* id
* status
* sale\_price
* vat\_id
* handling\_time
* stock

Please note that although the API permits sending the entire documentation on each offer update (price change, out-of-stock change, etc.) we do not recommend or encourage such a practice.

If you need to deactivate a valid offer on the website, you should send the offer with the “status = 0”.

## Saving volume measurements on products

In order to save volume measurements on existing products, the resource is ***measurements*** and the available action is ***save***. The measurement units for volume are millimeters and grams.

| **MEASUREMENTS – save** | | | |
| --- | --- | --- | --- |
| **Key – level 1** | **Description** | **Constraints** | **Example** |
| id | Seller internal product id. This is the primary key for identifying a product offer. | Required. Integer value between 1 and 16777215. | id=243409 |
| length | The length of the product in millimeters (mm) | Required. Decimal value between 0 and 999999. Up to two decimals. | length=100 |
| width | The width of the product in millimeters (mm) | Required. Decimal value between 0 and 999999. Up to two decimals. | width=122.50 |
| height | The height of the product in millimeters (mm) | Required. Decimal value between 0 and 999999. Up to two decimals. | height=250 |
| weight | The weight of the product in grams (g) | Required. Decimal value between 0 and 999999. Up to two decimals. | weight=1254.50 |

|  |  |  |
| --- | --- | --- |
| **Resource** | **Example** | **Context** |
| ***measurements*/save** |  | http method: POST |

## Reading and counting products and offers

In order to check the existing products (offers) and their status, the resource is ***product\_offer*** and the available action are ***read*** and ***count***.

|  | | **PRODUCT\_OFFER – read** | | | |
| --- | --- | --- | --- | --- | --- |
| **Key – level 1** | **Key – level 2** | **Description** | **Type** | **Example** | |
| part\_number\_key |  | eMAG part\_number\_key. | String | part\_number\_key=ES0NKBBBM | |
| number\_of\_offers |  | How many sellers have active offers on this product | Integer | number\_of\_offers=3 | |
| buy\_button\_rank |  | The rank of the offer in its race to win the <Add to cart> button | Integer | buy\_button\_rank=1 | |
| best\_offer\_sale\_price |  | Best selling price available in eMAG for the same Product | Decimal | best\_offer\_sale\_price=51.6477 | |
| best\_offer\_recommended\_price |  | The corresponding recommended price for the offer holding the best selling price | Decimal | best\_offer\_recommended\_price=54.6477 | |
| product\_ranking\* |  | The class of product considering its performance across category   * Super Hot * Hot * Regular * Cold * Super Cold | String | performance\_ranking=Hot | |
| ownership |  | Indicates who has ownership on the product’s documentation. Posible values:   1. Full ownership to seller 2. Full ownership to eMAG 3. Partial ownership to eMAG\*   \*Meaning that eMAG has ownership over the category and characteristic values of the product | Integer | ownership=1 | |
| category\_id |  | Product category eMAG id. | Integer | category \_id=506 | |
| vendor\_category\_id |  | Seller internal category id. | Integer | vendor\_category\_id=506 | |
| id |  | Seller internal product id. This is the primary key for identifying a product offer. | Integer | id=243409 | |
| brand |  | Product brand name. | String | brand=”Brand test” | |
| name |  | Product name. | String | name=”Test product” | |
| part\_number |  | Product part number. | String | part\_number=”md788hc/a” | |
| sale\_price |  | Seller offer sale price without VAT | Decimal | sale\_price=51.6477 | |
| recommended\_price |  | Seller offer recommended retail price before discount, without VAT. | Decimal | recommended\_price=54.6477 | |
| currency |  | Product price currency. | String | currency='RON' | |
| description |  | Product description. | String | description=”test” | |
| url |  | Product URL on the seller website. | String | url=”http://valid-url.html” | |
| warranty |  | The warranty offered in months. | Integer | warranty=24 | |
| ean |  | Product barcode identifier (EAN -8, EAN-13, UPC-A, UPC-E, JAN, ISBN-10, ISBN-13, ISSN, ISMN-10, ISMN-13, GTIN-14). | Array | ean=Array('ean1', 'ean2') |
| general\_stock |  | The sum of the stock on all seller warehouses. Is decremented and incremented when orders are processed. | Integer | general\_stock=20 | |
| estimated\_stock |  | This key takes into account the reserved stock on unacknowledged orders. | Integer | estimated\_stock=20 | |
| weight |  | The weight of the product | Decimal | weight=12.123456 | |
| status |  | Seller offer status.  1 – status active  0 – status inactive | Integer | status=1 | |
| images |  |  | List of arrays |  | |
|  | url | Seller image URL. | String | url=”http://valid-url.jpg” | |
|  | display\_type | Image display type.  1 – main image  2 – secondary image  0 – other images | Integer | display\_type=1 | |
| characteristics |  | All characteristics available in category | List of arrays |  | |
|  | id | Characteristic eMAG id | Integer | id=38 | |
|  | value | Characteristic value. | String | value=”test” | |
| vat\_id |  | Seller offer VAT rate id. | Integer | vat \_id=1 | |
| family |  | Product family. | Array |  | |
|  | id | Family id. | Integer | id=295 | |
|  | name | Family name. | String | name=”Test family” | |
| handling\_time |  |  | List of arrays |  | |
|  | warehouse\_id | The id of the warehouse. | Integer | warehouse\_id=1 | |
|  | value | Handling time, in number of days counted from the day the order was received. | Integer | value=0 | |
|  |  |  |  |  | |
| validation\_status |  | Product validation status | List of arrays |  | |
|  | value | Product validation status value | Integer | value=4 | |
|  | Description | Product validation status description | String | Description=”Rejected documentation” | |
| offer\_validation\_status |  | Offer validation status | List of arrays |  | |
|  | value | Offer validation status value | Integer | value=2 | |
|  | Description | Offer validation status description | String | Description=”Invalid price” | |

*\*to be released (H2 2018)*

The following filters are available when counting and reading products and offers:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| id | Displays the details for the corresponding ext\_id. | Optional. Integer value between 1 and 4294967295. |
| currentPage | Set current page displayed | Optional, integer. Ex: currentPage =3 |
| itemsPerPage | Set number of items to be displayed in one page. Maximum is set to 100. | Optional, integer. itemsPerPage=50 |
| status | Returns only the offers with this status. | Optional. Seller offer status. 1 – status active  0 – status inactive |
| general\_stock | Returns only offers with numerical general\_stock that have a value between 0 and the input. | Optional general\_stock = 3 |
| estimated\_stock | Returns only offers with numerical estimated\_stock that have a value between 0 and the input. | Optional reserved\_stock = 3 |
| validation\_status | Returns only the results with this validation status. | Optional.  1 = Awaiting MKTP validation 2 = Awaiting Brand validation 3 = Awaiting EAN validation 4 = Awaiting Documentation Validation 5 = Rejected Brand 6 = Rejected EAN 7 = Rejected Association 8 = Rejected Documentation  9 = Approved Documentation  10 = Blocked |
| offer\_validation\_status | Returns only the results with this validation status. | Optional.  1 = Saleable  2 = Invalid price |

Reading products for which sellers do not have ownership over the documentation will not return the values sent by sellers but the values that are displayed in the website.

## Product validation responses

After reading a product, all the elements previously sent are returned, along with the key *doc\_errors*. The key is not null for products that were rejected due to improper documentation. Below the list of possible errors, when they occur and the possible actions you need to take.

|  |
| --- |
|  |

## Matching products

In order to check if a product you sell already exists in eMAG catalog the resource is ***product\_offer*** and the available action is ***match***. The matching action is similar to the saving action so you should simply send the entire product documentation and all the offer data to the product matching resource. Please note that the matching action is available only for one product at a time therefore the request must be encapsulated in a single array.

As a response to the matching request, the following information will be returned:

|  |  |  |  |
| --- | --- | --- | --- |
| **Key – level 1** | **Description** | **Type** | **Example** |
| part\_number\_key | eMAG part\_number\_key. | String | part\_number\_key=ES0NKBBBM |
| name | eMAG product name | String | name=”Test product” |
| emag\_product\_url | eMAG product name | String | emag\_product\_url=”http://www.emag.ro/product/pd/ES0NKBBBM/” |

## Attaching offers on existing products

You can chose between using PNK (part\_number\_key) or EAN for attaching offers on existing eMAG products

If the product already exists in eMAG catalog, just add the key “part\_number\_key” with product’s part\_number\_key or the “ean” key with a single EAN.

|  |
| --- |
| The part\_number\_key is the last key found in the URL of an eMAG product. It will ALWAYS have both numbers and characters. Ex: for product http://www.emag.ro/telefon-mobil-nokia-105-black-105-black/pd/D5DD9BBBM/ the part\_number\_key is D5DD9BBBM. |

| **PRODUCT OFFER – save and create/update product** | | | | |
| --- | --- | --- | --- | --- |
| **Key – level 1** | **Key – level 2** | **Description** | **Constraints** | **Example** |
| id |  | Seller internal product id. This is the primary key for identifying a product offer. | Required. Integer value between 1 and 16777215. | id=243409 |
| name |  | Product name. Should be consistent with eMAG Product Documentation Standard. | Required. String between 1 and 255 characters long. | name=”Test product” |
| ean |  | Product barcode identifier (EAN -8, EAN-13, UPC-A, UPC-E, JAN, ISBN-10, ISBN-13, ISSN, ISMN-10, ISMN-13, GTIN-14). Please use the supplier barcode, not your internal barcodes. | Required if part\_number\_key is not present. No default value. Array of strings between 6 and 14 characters long. Only numeric figures allowed.  **OBS**: “part\_number\_key” and “ean” keys are mutually exclusive, you should use one or the other. | ean=Array('ean1') |
| part\_number\_key |  | eMAG part\_number\_key. Used for attaching a product offer to an existing product in eMAG platform. If you want to create new product, don’t set this key. | Required if ean is not present. String. Will be validated  **OBS**: “part\_number\_key” and “ean” keys are mutually exclusive, you should use one or the other. | part\_number\_key=ES0NKBBBM |
| status |  | Seller offer status.  1 – status active  0 – status inactive | Required. Integer value, 1 or 0. | status=1 |
| sale\_price |  | Seller offer sale price without VAT | Required. Decimal value greater than 0. Up to four decimals. | sale\_price=51.6477 |
| recommended\_price |  | Seller offer recommended retail price before discount, without VAT. If set, the offer will be displayed as promo. | Optional. Decimal value greater than 0. Up to four decimals. Must be greater than sale\_price. | recommended\_price=51.6477 |
| min\_sale\_price |  | Seller’s min offer sale price without VAT | Required on first product save. Decimal value greater than 0. Up to four decimals. | min\_sale\_price=40.6477 |
| max\_sale\_price |  | Seller’s max offer sale price without VAT | Required on first product save. Decimal value greater than 0. Up to four decimals. Must be greater than min\_sale\_price. | max\_sale\_price=60.6477 |
| currency\_type |  | Offer currency. Only send the key if it is different from the local Marketplace currency. Available options: EUR or PLN | Optional. 3 characters string. | currency\_type=”EUR” |
| vat\_id |  | Seller offer VAT rate id. Use /vat/read to display possible values. | Required. Integer. | Ex: vat \_id=1 |
| stock |  | Offer available quantity array. | Required. List of arrays. | {  0=>{  warehouse\_id=1,  value=20}} |
| stock | warehouse\_id | The id of the warehouse. Use warehouse\_id=1 for only one warehouse. | Required inside stock array. Integer. | warehouse\_id=1 |
| stock | value | Offer available quantity. | Required inside stock array. Integer between 0 and 65535. | value=20 |
| handling\_time |  | Handling time array. If no array is sent, the products are shipped the same day they are received. | Optional. List of arrays. | {  0=>{  warehouse\_id=1,  value=1}} |
| handling\_time | warehouse\_id | The id of the warehouse. Use warehouse\_id=1 for only one warehouse. | Required inside handling\_time array. Integer. | warehouse\_id=1 |
| handling\_time | value | Handling time, in number of days counted from the day the order was received. If handling\_time = 0 the order will be shipped the same day it is received. | Required inside handling\_time array. Integer value between 0 and 255. Default value = 0. | value=0 |
| start\_date |  | If it's a new offer, it represents the date your offer will be available from. For offer updates, it schedules value updates for the following data:   * sale\_price * recommended\_price * stock * handling\_time * vat\_id * warranty * status   All other data will be updated on the fly. Using start\_date, for example, you can schedule the inactivation of an offer, a price update, etc. | Optional. Text in YYYY-MM-DD format. Date can be as far as 60 days in the future (cannot be earlier than tomorrow). Cannot be null. | start\_date=”2014-12-31” |
| warranty |  | The warranty offered in months. | Required/Optional based on category.  Default value:  - 0 (no warranty) if optional  - No default if required.  Integer between 0 and 255. | warranty=24 |

|  |  |  |
| --- | --- | --- |
| **Resource** | **Example** | **Context** |
| **product\_offer/save** |  | http method: POST |

# Processing orders

An order consists of customer details, products and discounts from vouchers. It also has information about payment method, shipping tax. Also, each order always has a status attached. The available statuses are:

0 – canceled

1 – new

2 – in progress

3 – prepared

4 – finalized

5 – returned

The resource is ***order*** and the available actions are ***read***, ***save***, ***count*** and ***acknowledge***.

## Order fields

An order has the following properties:

| **Key – level 1** | **Key – level 2** | **Description** | **Constraints** | **Example** |
| --- | --- | --- | --- | --- |
| id |  | The number that uniquely identifies an order. | Required. Integer value between 1 and 4294967295. | id=939393 |
| status |  | The order processing status. Possible values:  0 - cancelled  1 - new  2 - in progress  3 – prepared  4 - finalized  5 - returned | Required. Integer value between 0 and 5. | status=1 |
| is\_complete |  | A flag indicating if the order is complete (has all details necessary for processing) or not. Possible values:  0 - incomplete;  1 - complete. | Optional. Integer value. | is\_complete=1 |
| type |  | A flag indicating if the order contains products fulfilled by eMAG or by seller. Possible values: 2 – fulfilled by eMAG  3 – fulfilled by seller | Optional. Integer value. | type=3 |
| payment\_mode\_id |  | The order payment method. Possible values:  1 - COD (cash on delivery)  2 - bank transfer  3 - online card payment | Required. Integer. | payment\_mode\_id=1 |
| delivery\_mode |  | The order delivery method. Possible values:  “courier” – home delivery  “pickup” – locker delivery | Optional. String. | delivery\_mode=”courier” |
| details |  | Order extra details. | Optional. Array. |  |
| details | locker\_id | The pickup point id, if the locker delivery option was selected by a customer. | Optional. String. | locker\_id=”dce0b7cf-dc38-11e8-a7d8-001a4a160153” |
|  | locker\_name | The pickup point name, if the locker delivery option was selected by a customer. | Optional. String. | locker\_name=”easybox eMAG Showroom” |
| date |  | The cart submission timestamp. | Optional. Text in YYYY-mm-dd HH:ii:ss format. | date=”1970-01-01 23:59:59” |
| payment\_status |  | The online payment status. Only used for online payment methods. Possible values:  0 - not paid  1 - paid | Required only for online payment methods. Integer. It is highly recommended to also interpret the payment status when reading orders with Card Online payment method. | payment\_status=0 |
| cashed\_co |  | The cashed amount from Card online payment | Optional. Integer. |  |
| cashed\_cod |  | The cashed amount from cash on delivery payment | Optional. Integer. |  |
| shipping\_tax |  | The shipment tax value. | Optional. Decimal. | shipping\_tax=”19.99” |
| customer |  | A list with the details about the customer, the shipping and the billing addresses. | Optional. List. | The field list is detailed below. |
| products |  | A list of arrays describing the products in the order. | List. | The field list is detailed below. |
| attachments |  | A list of arrays describing the attachments in the order. | List. | The field list is detailed below (chapter 3.1.3). |
| vouchers |  | A list describing the voucher discounts. | List. |  |
| is\_storno |  | Mandatory key when products are returned for a finalized order. Further details [here](#_Returned_products_and). | Optional. Boolean.  True indicates partial storno. | is\_storno=true |
| cancellation\_reason |  | The order cancellation reason. Possible values:  1 - Out of stock  2 - Cancelled by the client  3 - The client can not be contacted  15 - Courier delivery term is too large  16 - Transport tax, is too large  17 - Large delivery term, until the product will arrive in our warehouse  18 - Better offer in another store  19 - Payment order has not been paid  20 - Undelivered order, courier reasons  21 - Others  22 - Order Incomplete - automatic cancellation  23 - The customer changed his mind  24 - By customer request  25 - Failed delivery  26 - Late shipment  27 - Irrelevant Order  28 - Canceled by SuperAdmin on seller request  29 - Blacklisted customer  30 - No VAT invoice  31 - The eMAG Marketplace partner requested the order cancellation  32 - The delivery estimate is too long  33 - The product is no longer available in the stock of the eMAG Marketplace partner  34 - Other reasons  35 - The delivery is too expensive  36 - The customer found a better price elsewhere  37 - The customer registered another eMAG order with a similar product  38 - The customer changed his mind, does not need the product  39 - The customer can purchase the product only by installments | Optional. Integer value between 1 and 5. | cancellation\_reason=1 |

### Product field in order details

| **Key – level 1** | **Description** | **Constraints** | **Example** |
| --- | --- | --- | --- |
| id | eMAG internal order product line id. Any update on order product lines must use this id. | Required. Integer value between 1 and 9999999. id=243409 | id=123 |
| product\_id | Seller internal product id. This is the primary key for identifying a product offer. | Optional. Integer. | product\_id=3331 |
| status | The status of product of the order. Possible values: 0 - cancelled 1 - active | Required. Integer. | status=1 |
| part\_number | Manufacturer unique identifier for the product. | Optional. String between 1 and 128 characters. | part\_number='682133frs' |
| created | The date when the order product line was created. | Optional. Text in YYYY-mm-dd HH:ii:ss format. | created='2014-07-24 12:16:50' |
| modified | The date when the order product line was last modified. | Optional. Text in YYYY-mm-dd HH:ii:ss format. | modified='2014-07-24 12:18:53' |
| currency | Product price currency. | Optional. String. | currency='RON' |
| quantity | Product quantity. | Required. Integer. Positive, different than 0. | quantity=2 |
| sale\_price | The sale price without VAT. | Required. Integer. | sale\_price=12.1234 |
| details | Additional product notes. | Optional. Text. | details=”text” |
| status | The status of product of the order. Possible values: 0 - cancelled 1 - active | Required. Integer. | status=1 |
| sale\_price | The sale price without VAT. | Required. Integer. | sale\_price=12.1234 |
| details | Additional product notes. | Optional. Text. | details=”text” |

### Customer fields in order details

The customer field has the following properties:

| **Key** | **Description** | **Constraints** | **Example** |
| --- | --- | --- | --- |
| id | The number that uniquely identifies a customer. | Optional. Integer value between 1 and2147483647. | id=1 |
| name | The customer's name. | Optional. Text. | name=”Surname Name” |
| email | This is a hash that uniquely identifies the customer’s email. | Optional. Text. | email=”1243536@emag.ro” |
| company | The name of the company. For physical person it has the same value as name. | Optional. Text. | company=”Company name ltd.” |
| gender | The customer gender. Possible values:  M - male  F – female | Optional. Text. | gender=”M” |
| code | The company registration code. | Optional. Text. | code=”14399840” |
| registration\_number | The company registration number. | Optional. Text | registration\_number=” 40/372/2002” |
| bank | The bank name. | Optional. Text. | bank=”Bank name” |
| iban | The bank account. | Optional. Text. | iban=” RO24BACX0000000031430310” |
| fax | The customer's fax number. | Optional. Text. | fax=”4021123123” |
| legal\_entity | A flag indicating if the customer is physical or juridical entity. Possible values:  0 - private entity;  1 - legal entity. | Optional. Integer value. | legal\_entity=1 |
| is\_vat\_payer | A flag indicating it the customer is vat payer. Possible values:  0 - the customer is not vat payer;  1 - the customer is vat payer. | Optional. Integer value. | is\_vat\_payer=0 |
| phone\_1 | The customer's first phone number. | Optional. Text. | phone\_1=”4021123123” |
| phone\_2 | The customer's second phone number. | Optional. Text. |  |
| phone\_3 | The customer's third phone number. | Optional. Text. |  |
| billing\_name | The customer's invoice name. | Optional. Text. | billing\_name=”Surname Name” |
| billing\_phone | The customer's invoice phone. | Optional. Text. | billing\_phone=”4021123123” |
| billing\_country | The customer's invoice country. | Optional. Text. | billing\_country=”Romania” |
| billing\_suburb | The customer's invoice county. | Optional. Text. | billing\_suburb=”Suburb” |
| billing\_city | The customer's invoice city. | Optional. Text. | billing\_city=”City” |
| billing\_street | The customer's invoice address. | Optional. Text. | billing\_street=”Street Name” |
| billing\_postal\_code | The customer's invoice postal code. | Optional. Text. | billing\_postal\_code=”23125” |
| shipping\_contact | The name of the contact person that will pick up the parcel. Should be printed on the AWB. | Optional. Text. | shipping\_contact=”Name Surname” |
| shipping\_phone | The phone used by the courrier to contact the shipping person. Should be printed on the AWB. | Optional. Text. | shipping\_phone=”23125” |
| shipping\_country | The customer's shipping country. | Optional. Text. | shipping\_country=”Romania” |
| shipping\_suburb | The customer's shipping county. | Optional. Text. | shipping\_suburb=”Suburb” |
| shipping\_city | The customer's shipping city. | Optional. Text. | shipping\_city=”City name” |
| shipping\_street | The customer's shipping suburb. | Optional. Text. | shipping\_street=”Street name” |
| shipping\_postal\_code | The customer's shipping postal code. | Optional. Text. | shipping\_postal\_code=”23125” |
| billing\_locality\_id | This field uniquely identifies a locality in the eMAG database.  It represents the billing locality. | Integer value between 1 and 4294967295. | billing\_locality\_id=”23” |
| shipping\_locality\_id | This field uniquely identifies a locality in the eMAG database.  It represents the shipping locality. | Integer value between 1 and 4294967295. | shipping\_locality\_id=”23” |

### Order invoices

When pushing orders into finalized status, you should also send the invoice PDF file location for the specific order.

The resource is **order/attachments** and the available action is **save**.

The following keys should be sent in attachments array in order to display an invoice in the customer’s order details: name, url, type.

An attachment has the following properties:

| **Key** | **Description** | **Constraints** | **Example** |
| --- | --- | --- | --- |
| order\_id | The number that uniquely identifies an order. | Required. Integer value between 1 and 4294967295. | id=939393 |
| name | The name of the attachment displayed to the customer (in order history or in email) | Optional. String between 1 and 60 characters | name='Invoice title' |
| url | Attachment URL. | Required. String between 1 and 1024 characters. Valid URL to document. | url=”<http://valid-url>/invoice.pdf” |
| type | The type of document attached to the order. Possible values are:   * 1 - invoice * 3 - warranty * 4 - user manual * 8 - user guide * 10 - AWB * 11 - proforma | Integer. Optional. Default 1=”Invoice”. Only .pdf files are accepted | type=1 |
| force\_download | Flag used in order to force attachment download restrictions. If value is 0 and the attachment URL has not changed, the attachment will not be downloaded again. | Integer. Optional. Default value 0. Possible values: 0,1. | force\_download=0 |

## Order notification, acknowledgment and order filters

When a new order is placed in eMAG Marketplace for the first time, it’s status is 1 (new) and a GET request with the order id is automatically made to an URL you provide (call-back URL). Ex: <http://valid_url/path?order_id>=123

In the next step, you should read the order passing the id previously mentioned and after successfully saving the order in your database you should notify us by calling back the API using the route MARKETPLACE \_URL/api-3/order/acknowledge/[orderId]. This stops the order notification system for the mentioned order. Unless acknowledged, we will notify the new orders for up to 48 hours.

**IMPORTANT**:

* Order acknowledge is the only method of marking the order status as “in progress”.
* Clients may ask for an order to be canceled, this will be done by eMAG only if the order was not acknowledged by the seller, thus some of the orders may be read directly with status 0 (canceled).

## Order status matrix

The following matrix defines the order flow in eMAG Marketplace:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **New status** | | | | | |
| **Actual status** | 1 - New | 2 - In progress | 3 – Prepared | 4 - Finalized | 0 - Canceled | 5 - Returned |
| 1 - New | No | Yes by ACK only | No | No | No | No |
| 2 - In progress | No | Yes | Yes | Yes | Yes | No |
| 3 - Prepared | No | No | Yes | Yes | Yes | No |
| 4 - Finalized | No | No | Yes in 48h max | Yes | Yes in 48h max | Yes in RT\* + 5 days max |
| 0 - Canceled | No | Yes in 48h max | Yes in 48h max | Yes in 48h max | Yes | No |
| 5 - Returned | No | No | No | No | No | No |

*\*RT = return time allowed to customers*

**IMPORTANT**:

* We recommend setting up a periodical /order/read (cron, agent) that should identify orders that were not acknowledged. By default on /order/read we expose the last 100 orders, but you can request up to 1000 or use pagination. Do not forget to test the order status matrix against your internal order workflow. As a best practice you should either acknowledge the order prior the read or re-read the order after acknowledging it; an order can be modified by eMAG employees upon the client’s request as long as it is not acknowledged.
* You can only edit the order (add/remove products modify quantity or price) when in status 2 (in progress) or 3 (prepared).
* Once an order is finalized, you can change its status back to status 3 (prepared) or 0 (canceled) only in the first 48 hours since finalization.
* Order status “finalized” will be set automatically when issuing the first AWB for that order. See chapter [Saving AWB’s](#_Saving_AWB's).
* The order status “returned” is set automatically when all the products from the initial invoice are marked as returned. The change is permitted only within the maximum return timeframe allowed to the customer.

## Order filters

You can read all your orders though the API, using filters. The following are available when counting orders:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| id | Only the order with this value. | Optional. Integer value between 1 and4294967295. |
| createdBefore | Only the orders created before the specified date. Can only be set if “createdAfter” is present. Maximum 1 month difference. | Optional. Text in YYYY-mm-dd HH:ii:ss format. |
| createdAfter | Only the orders created after the specified date. Can only be set if “createdBefore” is present. Maximum 1 month difference. | Optional. Text in YYYY-mm-dd HH:ii:ss format. |
| modifiedBefore | Only the orders modified before the specified date. Can only be set if “modifiedAfter” is present. Maximum 1 month difference. | Optional. Text in YYYY-mm-dd HH:ii:ss format. |
| modifiedAfter | Only the orders after before the specified date. Can only be set if “modifiedBefore” is present. Maximum 1 month difference. | Optional. Text in YYYY-mm-dd HH:ii:ss format. |
| status | Only the orders with the specified status. It is a single value or a list of values. | Optional. Integer or list. |
| payment\_mode\_ id | Only the orders with the specified payment method id. It is a single value or a list of values. | Optional. Integer or list. |
| is\_complete | Only the orders with the specified completion status. | Optional. Order completion status. 1 – complete orders  0 – incomplete orders |
| type | Only the orders with the specified type | Optional. Default value = 3.  2 – fulfilled by eMAG  3 – fulfilled by seller |

The following filters are available when reading orders:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| itemsPerPage | The maximum number of orders to return. | Optional. Integer value between 1 and 100. |
| currentPage | The page offset. | Optional. Integer value between 1 and 65535. |
| id | Only the order with this value. | Optional. Integer value between 1 and4294967295. |
| createdBefore | Only the orders created before the specified date. Can only be set if “createdAfter” is present. Maximum 1 month difference. | Optional. Text in YYYY-mm-dd HH:ii:ss format. |
| createdAfter | Only the orders created after the specified date. | Optional. Text in YYYY-mm-dd HH:ii:ss format. |
| modifiedBefore | Only the orders modified before the specified date. Can only be set if “modifiedAfter” is present. Maximum 1 month difference. | Optional. Text in YYYY-mm-dd HH:ii:ss format. |
| modifiedAfter | Only the orders after the specified date. | Optional. Text in YYYY-mm-dd HH:ii:ss format. |
| status | Only the orders with the specified status. It is a single value or a list of values. | Optional. Integer or list. |
| payment\_mode\_id | Only the orders with the specified payment method id. It is a single value or a list of values. | Optional. Integer or list. |
| is\_complete | Only the orders with the specified completion status. | Optional. Order completion status. 1 – complete orders  0 – incomplete orders |
| type | Only the orders with the specified type | Optional. Default value = 3.  2 – fulfilled by eMAG  3 – fulfilled by seller |

## Updating orders

You cannot create new orders through the API, you can only read and update them. When updating an order, the seller should send ALL the fields initially read.

**IMPORTANT**:

* Updating products by reducing their quantities for orders with Online Card payment method is no longer possible.
* Updating product prices is no longer available
* A canceled order can no longer be reactivated if more than 48 hours have passed since cancelation

|  |  |  |
| --- | --- | --- |
| **Resource** | **Example** | **Context** |
| **order/save** |  | http method: POST |

### Removing products from an order

To remove a product from the order send the status=0 for the product or do not send it at all. Products can be removed from an order only while in status 2 or 3 (in progress or prepared) for orders with payment methods different than Online card. For returned products (the order is in status 4, finalized), please use the storno route.

**IMPORTANT**: Removing products for orders with Online Card payment method is no longer possible.

### Adding products to an existing order

To add a new product to an existing order, add it to the order by sending the product id (mandatory), name, status and sale price.

**IMPORTANT:** virtual products such as internal discounts can be inserted in an order, even if they were not previously sent to eMAG. Adding these products to an order will not make them available for purchase in the eMAG Marketplace platform.

### Returned products and storno route

A finalized order cannot be modified, it can be fully reversed by changing the order status from finalized (4) to returned (5) or have only some of the products reversed using a call with is\_storno key true.

The following conditions must be met in order for a partial storno to occur:

* Order must be in status 4
* At least one product quantity was reduced

The following scenarios can be used as a guideline for returning products (partial storno) from a finalized order:

| **Current order status** | **Request** | **isError** | **Order read** |
| --- | --- | --- | --- |
| status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), | status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 1,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ),  'is\_storno'=true | FALSE | status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 1,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), |
| status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), | status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 0,  'sale\_price' => '123.4567',  'status' => 1,  ),  ),  'is\_storno'=true | FALSE | status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 0,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), |
| status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), | status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 0,  ),  ),  'is\_storno'=true | FALSE | status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 0,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), |
| status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), | status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 1,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), | TRUE | The request will be discarded, as you are trying to modify a finalized order without is\_storno key. |
| status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), | status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ),  'is\_storno'=true | TRUE | The request will be discarded, as you are sending is\_storno key without any change to an order line |
| status' => 3,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), | status' => 3,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ),  'is\_storno'=true | TRUE | The request will be discarded, as you are sending is\_storno key for an order with a status different than 4 |
| status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  ), | status' => 4,  'products' =>  array (  0 =>  array (  'id' => 1,  'product\_id' => '1',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  1 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => 2,  'sale\_price' => '123.4567',  'status' => 1,  ),  2 =>  array (  'id' => 2,  'product\_id' => '2',  'quantity' => -1,  'sale\_price' => '123.4567',  'status' => 1,  ),  ),  'is\_storno'=true | TRUE | The request will be discarded, as you are trying to send a negative quantity for a product |

# Shipping orders

For electronic deliveries and downloadable goods, please skip this section. Shipping an eMAG Marketplace order requires the seller to issue an AWB using eMAG Marketplace API.

The resource is ***AWB*** and the available actions are ***read*** and ***save***

## Saving AWB

To save an AWB just call the API with the following parameters:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| order\_id | Identifies the order | Required. Integer value between 1 and 4294967295.  Must be a valid order in the eMAG database, and must be owned by the seller. |
| rma\_id | Identifies the return request | Optional. Integer value between 1 and 4294967295.  Must be a valid return request in the eMAG database, and must be owned by the seller. |
| sender | \*Array explained [below](#_AWB_sender/receiver_fields) |  |
| receiver | \*Array explained [below](#_AWB_sender/receiver_fields) |  |
| locker\_id | The pickup point id. Should be filled in with the pickup point id received on the order. If filled in, the courier will deliver the parcel in the designated locker. | Optional. String value between 3 and 255 characters |
| is\_oversize | If set to 1, marks the delivery as containing oversized products | Required. Value can only be 0 or 1. |
| insured\_value | The insured value | Optional. Double value between 0 and 999999999 |
| weight | The weight of delivery | Optional. Double value between 0 and 99999 |
| envelope\_number | Number of envelopes to be delivered | Required. Integer value between 0 and 9999. If parcel\_number is 0, this parameter cannot be 0 |
| parcel\_number | Number of parcels to be delivered | Required. Integer value between 0 and 999. If envelope\_number is 0, this parameter cannot be 0 |
| observation | Observation text | Optional. String value between 0 and 255 |
| cod | Cash on delivery | Required. Double value between 0 and 999999999 |
| courier\_account\_id | Unique identifier for seller’s courier account. If not provided, a default account will be used. | Optional. Integer. |
| pickup\_and\_return | If set to 1, sender expects something in return to this expedition (documents, buy-back products, etc). | Optional. Value can only be 0 or 1. |
| saturday\_delivery | If set to 1, sender requests the package to be delivered on Saturday. | Optional. Value can only be 0 or 1. |
| sameday\_delivery | If set to 1, sender requests the package to be delivered the same day. | Optional. Value can only be 0 or 1. |

An AWB S/R (sender/receiver) has the following properties:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| name | S/R's name | Required. String value between 3 and 255 |
| contact | Receiver's contact person name | Required. String value between 1 and 255 |
| phone1 | S/R first phone number | Required. String value between 8 and 11 digits (only '+' character is allowed at the beginning of the string) |
| phone2 | S/R second phone number | Optional. String value between 8 and 11 digits (only '+' character is allowed at the beginning of the string) |
| legal\_entity | If Receiver is legal entity (applicable only to receiver) | LEGAL\_ENTITY\_NO = 0 LEGAL\_ENTITY\_YES = 1 |
| locality\_id | S/R's locality\_id | Required. Integer value between 1 and 4294967295.  Must be a valid locality in the eMAG database. |
| street | S/R's street | Required. String value between 3 and 255 |
| zipcode | S/R's zipcode | Optional. String value between 1 and 255 |

**IMPORTANT:**

* For orders with “pickup” as a delivery method if you do not change the locker id that is already included in the “shipping\_street” field the AWB will be issued as a locker delivery using the proper courier account regardless of the actual courier account you specified when issuing the AWB.

## Reading AWB

The following filters are available when reading AWBs:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| emag\_id | The eMAG internal barcode id | Integer value between 1 and 4294967295.  Must be a valid AWB in eMAG database. |
| reservation\_id | The eMAG internal AWB reservation id | Integer value between 1 and 4294967295.  Must be a valid AWB in eMAG database. |

An AWB has the following properties:

| **AWB – read** | | | | | |
| --- | --- | --- | --- | --- | --- |
| **Key – level 1** | **Key – level 2** | **Description** | **Constraints** | **Example** |
| emag\_id |  | The eMAG internal AWB id | Integer. | emag\_id=243409 |
| order\_id |  | The id of the order on which the AWB was issued | Integer. | order\_id=243409 |
| rma\_id |  | The id of the return request on which the AWB was issued | Integer. | rma\_id=243409 |
| weight |  | The weight of delivery | Integer. | weight=1 |
| awb\_type |  | The type of delivery. Possible values:  1 – delivery to customer  2 – pickup from customer | Integer. | awb\_type=1 |
| awb |  | The AWB | List of arrays. |  |
|  | emag\_id | The eMAG internal AWB barcode id | Integer | emag\_id=243409 |
|  | awb\_number | The AWB number | String | awb\_number = “2EMG00011012” |
|  | awb\_barcode | The AWB barcode | String | awb\_barcode = “2EMG00011012001” |
| status |  | The status of the delivery | List of arrays. |  |
|  | code | The code status of the delivery | String. | code=”DLV” |
|  | name | The name of the status of the delivery | String. | name=”Delivered” |
|  | description | The description of the status of the delivery | String. | description=”AWB delivered” |
| courier |  | The courier used for issuing the AWB | List of arrays. |  |
|  | courier\_account\_id | The eMAG internal courier account id used for issuing the AWB | Integer. | courier\_account\_id=5186 |
|  | courier\_name | The eMAG internal courier name used for issuing the AWB | String. | courier\_name="SAMEDAY" |

## Reading AWB PDF files

To download an AWB PDF file call the MARKETPLACE\_API\_URL/awb URL as in the example below

|  |
| --- |
| <html>  Running...<br>  <?  $username = 'user';  $password = 'pass';  $hash = base64\_encode($username . ':' . $password);  $headers = array(  'Authorization: Basic ' . $hash  );  $ch = curl\_init();  curl\_setopt($ch, CURLOPT\_URL,'**https://marketplace.emag.ro/awb/read\_pdf?emag\_id=9755945&awb\_format=A4**';  curl\_setopt($ch, CURLOPT\_SSL\_VERIFYPEER, false);  curl\_setopt($ch, CURLOPT\_FOLLOWLOCATION, 1);  curl\_setopt($ch, CURLOPT\_HEADER, 0);  curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, 1);  curl\_setopt($ch, CURLOPT\_CUSTOMREQUEST, 'GET');  curl\_setopt($ch, CURLOPT\_HTTPHEADER, $headers);  $result = curl\_exec($ch);  echo $result . "\n";  ?>  </html> |

**Optional:** You can set the paper format on PDF download link by using the parameter "awb\_format=A4" in the link. The possible values are A4, A5, A6.

The following method will be released in the near future (H2 2019)

To download an AWB just call the API with the following parameters:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| emag\_id | The AWB's eMAG id. | Integer value between 1 and 4294967295.  Must be a valid AWB in eMAG database. |
| awb\_format | The paper format on PDF download | The possible values are A4, A5, A6 and ZPL |

## Reading AWB ZPL type

To read an AWB in ZPL type call the MARKETPLACE\_API\_URL/awb URL as in the example below

|  |
| --- |
| <html>  Running...<br>  <?  $username = 'user';  $password = 'pass';  $hash = base64\_encode($username . ':' . $password);  $headers = array(  'Authorization: Basic ' . $hash  );  $ch = curl\_init();  curl\_setopt($ch, CURLOPT\_URL,'**https://marketplace.emag.ro/awb/read\_zpl?emag\_id=9755945**';  curl\_setopt($ch, CURLOPT\_SSL\_VERIFYPEER, false);  curl\_setopt($ch, CURLOPT\_FOLLOWLOCATION, 1);  curl\_setopt($ch, CURLOPT\_HEADER, 0);  curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, 1);  curl\_setopt($ch, CURLOPT\_CUSTOMREQUEST, 'GET');  curl\_setopt($ch, CURLOPT\_HTTPHEADER, $headers);  $result = curl\_exec($ch);  echo $result . "\n";  ?>  </html> |

Using this request will return a base64 encoded content of the ZPL format as in the example below

|  |  |  |
| --- | --- | --- |
| **Resource** | **Example** | **Context** |
| **awb/read\_zpl** |  | http method: POST |

## Counting Localities

In order to issue an AWB you need to submit the correct locality id. You can also use the id from the order.

The resource is ***locality*** and the available actions are ***read*** and ***count***.

The following filters are available when counting localities:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| emag\_id | The locality with this id | Integer |
| name | All localities with this name | String of length between 0 and 60 |
| modified | All localities modified after this date | Date with the 'Y-m-d H:i:s' format |

## Reading Localities

The following filters are available when reading localities:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| emag\_id | The locality with this id | Integer |
| name | All localities with this name | String of length between 0 and 60 |
| modified | All localities modified after this date | Date with the 'Y-m-d H:i:s' format |
| itemsPerPage | The maximum number of localities to return. | Optional. Integer value between 1 and 100. |
| currentPage | The page offset. | Optional. Integer value between 1 and 65535. |

A locality has the following properties:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| emag\_id | The id of the locality | Integer |
| name | The name of the locality | String of length between 0 and 60 |
| region([1-4]+) | Region name | String of length between 0 and 60 |
| region([1-4]+)\_latin | Region name latin version | String of length between 0 and 60 |
| geoid | Geographic id of the location | Integer |
| modified | Last modification date | Date with the 'Y-m-d H:i:s' format |

## Reading courier accounts

In order to issue an AWB you need to submit the correct courier account id.

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| account\_id | The id of the account | Integer |
| account\_display\_name | The name of the account | String of length between 0 and 60 |
| courier\_account\_type | The type of the account | Integer. Possible values: 1 - RMA; 2 - Order; 3 - RMA & Order; 4 - Non Marketplace |
| courier\_name | The name of the courier | String of length between 0 and 60 |
| courier\_account\_properties | The courier account properties | Array. Possible values: 1 – Regular; 4 – Locker delivery |
| created | The account creation date | Date with the 'Y-m-d H:i:s' format |
| status | Account status | Integer. Possible values: 1 - Active; 0 - Inactive |

# Processing return requests

A return request consists of customer details, products and reason for returning products. Each return request always has a status attached. The available statuses are:

1 - incomplete

2 - new

3 - acknowledged

4 - refused

5 - canceled

6 - received

7 - finalized

The resource is ***RMA*** and the available actions are ***read***, ***save***.

Return requests fields

The message structure for both read and save actions is detailed below:

| **RMA – read and save/update** | | | | |
| --- | --- | --- | --- | --- |
| **Key – level 1** | **Key – level 2** | **Description** | **Constraints** | **Example** |
| emag\_id |  | Return request eMAG system ID | Required.  Type: bigint |  |
| id |  | Seller internal return request ID | Type: bigint |  |
| order\_id |  | The id of the order in which the product to be returned was included | Required.  Type: bigint |  |
| type |  | A flag indicating if the return request contains products fulfilled by eMAG or by seller. Possible values: 2 – fulfilled by eMAG  3 – fulfilled by seller | Optional.  Type: Integer. | type=3 |
| invoice\_number |  | Invoice of the order in which the product to be returned was included | Optional.  Type: varchar  Default value: NULL |  |
| customer\_name |  | Customer name | Required.  Type: varchar |  |
| customer\_company |  | Customer company info | Optional.  Type: varchar  Default value NULL |  |
| customer\_phone |  | Customer phone no. | Required.  Type: varchar |  |
| products |  | Product to be returned info | Array |  |
|  | id | eMAG internal return product line id. Any update on return product lines must use this id. | Required. Integer value between 1 and 9999999. | id=123 |
|  | product\_emag\_id | Product to be returned eMAG ID |  |  |
|  | product\_id | Product to be returned seller internal ID | Required.  Type: int |  |
|  | product\_name | Product name | Required.  Type: varchar |  |
|  | quantity | Product quantity | Required  Type: int |  |
|  | diagnostic | Diagnostic after product analysis | Optional.  Type: int  Default value NULL |  |
|  | reject\_reason | Reason of return rejection | Optional.  Type: int  Default value NULL |  |
|  | refund\_value | Refund value | Optional.  Type: varchar |  |
| awbs |  | Issued AWBs | List of arrays. |  |
|  | reservation\_id | eMAG internal AWB reservation id. Use this id to read an AWB | Optional.  Type: int | reservation\_id= 4528511 |
| pickup\_country |  |  | Required.  Type: Varchar |  |
| pickup\_suburb |  |  | Required.  Type: Varchar |  |
| pickup\_city |  |  | Required.  Type: Varchar |  |
| pickup\_address |  |  | Required.  Type: Varchar |  |
| pickup\_address\_id |  | Id of address already saved on the customers' account | Optional.  Type: Int  Default value: NULL |  |
| pickup\_zipcode |  |  | Optional.  Type: Varchar  Default value: NULL |  |
| pickup\_date |  | Returned product pickup date (in case of vendor pickup from customer) | Optional.  Type: datetime  Default value NULL |  |
| pickup\_locality\_id |  | The internal eMAG ID of the pickup address city/locality | Required.  Type: int |  |
| pickup\_method |  | The product pickup method selected by the customer when inserting the return request. Possible values:   * 1 - eMAG courier * 2 - Seller’s own courier * 3 - Sent by client | Required.  Type: int |  |
| return\_reason |  | It holds the return reason selected by the customer. Possible values:  1 - I recorded the order by mistake  2 - I received my order with delays  3 - I found a better price elsewhere  4 - I received another product than ordered  5 - I was recommended a wrong product  6 - I received the product with no parts / accessories  7 - I'm not satisfied with how the product works  8 - I have received a defective product  9 - The product was damaged during shipping  10 - The product does not correspond with the site presentation  11 - I changed my mind, I do not need the product  12 - The size selected is small  13 - The size selected is large  14 - I don't like how it fits  15 - Product color is different from site presentation  16 - Product material is different from what I expected  17 - The product has defective pixels  100 - Unspecified | Required.  Type: Int |  |
| return\_type |  | It holds the return type selected by the customer. Possible values:  1 - Replacement with same product  2 - Replacement with a different product  3 - Refund  4 - Cancel payment contract for this product  5 - Voucher | Required.  Type: Int |  |
| return\_address\_id |  | It will hold the id of the return address selected by the vendor in RMA UI | Optional.  Type: Int  Default value: id of the address set as default in vendor profile address page. |  |
| return\_tax\_value |  | Shipping tax for refused returned products (VAT included)  The currency used will be the platforms' default | Optional.  Type: Float Default value: NULL |  |
| customer\_account\_iban |  |  | Type: Varchar  Default value: NULL |  |
| customer\_account\_bank |  |  | Type: Varchar  Default value: NULL |  |
| customer\_account\_beneficiary |  |  | Type: Varchar  Default value: NULL |  |
| replacement\_product\_emag\_id |  | The eMAG ID of the replacement product | Type: Int  Default value: NULL |  |
| replacement\_product\_id |  | The seller internal ID of the replacement product | Type: Int  Default value: NULL |  |
| replacement\_product\_name |  |  | Type: Varchar  Default value: NULL |  |
| replacement\_product\_quantity |  |  | Type: Int  Default value: NULL |  |
| observations |  | Free text notes field | Optional.  Type: text  Default value: NULL |  |
| date |  | RMA request insertion date | Required.  Type: datetime |  |
| request\_status |  | RMA request status. Possible values:  1 - Incomplete  2 - New  3 - Approved  4 - Refused  5 - Canceled  6 - Received  7 - Finalized | Optional.  Type: Int |  |

## Return requests filters

When reading return requests the following filters are available:

| **Key** | **Description** | **Constraints** |
| --- | --- | --- |
| id | Seller internal return request ID |  |
| emag\_id | eMAG return request ID |  |
| order\_id | Order on which the product to be returned was included |  |
| product\_id | Seller internal returned product ID |  |
| product\_emag\_id | eMAG returned product ID |  |
| requests\_status | Return request status ID |  |
| date | Return request insertion date |  |

## Status change permissions

The following matrix defines the return request processing flow in eMAG Marketplace:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **New status** | | | | | | |
| **Actual status** | 2 - New | 3 - Acknowledged | 4 - Rejected | 5 - Canceled | 6 - Received | 7 - Finalized |
| 2 - New | Yes | Yes | No | Yes | No | No |
| 3 - Acknowledged | No | Yes | No | Yes | Yes | No |
| 4 - Rejected | No | No | Yes | No | No | No |
| 5 - Canceled | No | No | No | Yes | No | No |
| 6 - Received | No | No | Yes | No | Yes | Yes |
| 7 - Finalized | No | No | No | No | No | Yes |

*\*Some of the statuses were left out by design; these should not be used in any seller implementation*

## Return request deliveries

There are two types of possible deliveries for the return requests:

* pick-up requests - courier picks up the returned product(s) from the customer and delivers them to the seller
* regular deliveries - courier delivers the returned/replaced product back to the customer

 The delivery requests will be generated using the ***AWB save*** resource.

## Examples requests and responses

|  |  |  |
| --- | --- | --- |
| **Resource** | **Example** | **Context** |
| **RMA/read** |  | http method: POST  Seller has return requests |
| **RMA/read** |  | http method: POST  Seller does not have return requests |
| **RMA/save** |  | http method: POST |