beslist.nl

Shopitem API

A technical guide to the REST API for managing updates of shopitems

Date: 07-12-2018

Version: 3.4

Index

Introduction and background

The Beslist.nl platform imports webshop inventory items using feeds. The process of feed creation, downloading and processing all items in a feed is by nature not real-time. Therefore an API has been developed enabling near real-time updates to a number of properties of inventory items. Within the context of this API those webshop inventory items are referred to as ‘shopitems’ and the API is called the ‘Shopitem API’.

The Shopitem API is published with built-in technical API documentation aimed at API client developers. This built-in documentation defines the contracts for the REST endpoints, by specifying input and output and documenting resulting status codes.

This document gives an overview of the conventions used for he Shopitem API and gives a few code examples of how to use the API. It is not exhaustive but should give enough information to start using the API.

1. How to get access to the API and its online docs

Location of online API documentation

The Shopitem API has a single host where both the API and the online documentation can be found. The API is versioned, and the version number is in the urls. The documentation URL is a good starting point because it contains the urls to the API endpoints. All API urls, including documentation use https.

Shopitem API docs: https://shopitem.api.beslist.nl/swagger

Documentation for different versions of the API endpoints

API endpoints can co-exist in different versions, meaning that multiple versions of a single endpoint can exist. The most recent version is always the default version. Older versions are labeled as deprecated and will be removed in the future. The online documentation is also available in different versions. The default view that is displayed at the doc location url always shows the documentation for the most recent versions of all API endpoints:

Current docs: https://shopitem.api.beslist.nl/swagger

Besides being shown in the default view, every version of the API has its own view. The different versions can be selected in the top right corner.

Endpoint urls

The API has several endpoints that are documented in the online documentation. The urls are all based on a versioned url with an endpoint specific suffix. All endpoints require an apiKey header to be sent along and the **Content-Type header should be set to application/json**. We will always respond with json formatted data.

Before we start explaining the different endpoints it's important to agree on some terms and on the datamodel used for updating:

Term	Explanation
{shopId}	Replace this with the shop id you are currently sending updates for. You can retrieve your shop id by accessing /auth/v1/shops.
{externalId}	Replace this with your unique identifier of the product you are updating.
{batchId}	Replace this with the batch id as returned when sending batch updates.

Datamodel property	Explanation
price.regularPrice	The current selling price in decimal format.
price.previousPrice	The previous selling price in decimal format.
shipping	An array of shipping information. The primary key is the destinationCountry.
shipping / destinationCountry	The country where the product can be shipped to. Either "nl" or "be".
shipping / price	The costs for shipping the product to the corresponding destinationCountry in decimal format.
shipping / deliveryTime	A text representation of the time it will take the product to reach the customer when placing an order.
stock.level	An uint32 representation of the number of products that may be sold. This cannot be a negative number.
externalId	Your unique identifier of the product you are updating.
error	A message describing the error.
errorDetails	A dictionary containing all properties and corresponding error messages.

Authentication:

[GET] https://shopitem.api.beslist.nl/auth/v1/shops
Content-Type: application/json

You can retrieve information about the shops you can update by sending a GET request.

Possible http responses:

200	Successful request. Response body (1) contains a list of shops.
401	Returned when authentication failed (e.g. when an invalid or expired API key was used).
404	Returned when no shops are found.
500	Returned when the request failed.

Response body:

```
[
  {
    "shopId": 0,
    "name": "string"
  }
]
```

Single updates:

[PUT] https://shopitem.api.beslist.nl/v3/offer/shops/{shopId}/offers/{externalId}
Content-Type: application/json

You can update a single product by sending a PUT request containing a json formatted body as follows:

```
{
  "price": {
    "regularPrice": 0,
    "previousPrice": 0
  },
  "shipping": [
    {
      "destinationCountry": "nl",
      "price": 0,
      "deliveryTime": "string"
    }
  ],
  "stock": {
    "level": 0
  }
}
```

You can send partial data. i.e. only price.regularPrice. Or only shipping to destinationCountry “nl”.

Datamodel:

```
OfferV3Request {
  description:        Offering of a product
  price               Price {
                        regularPrice        number($double)
                                            Price in Euros incl. VAT
                        previousPrice       number($double)
                                            Previous Price (before discount) in Euros incl. VAT (valid in combination with 'RegularPrice' only))
                      }
  shipping            [
                      Shipping information per country
                      Shipping {
                        destinationCountry* Country  string
                                            ISO 3166-1 alpha-2 (two-letter) country code

                                            Enum:

                                            [ nl, be ]
                        price               number($double)
                                            Shipping cost to destination country in Euros incl. VAT
                        deliveryTime        string
                                            Shipping time specification to destination country
                      }]
  stock               Stock {
                        level               integer($int32)
                                            Number of products currently in stock
                      }
}
```

Validation:

- RegularPrice should be greater than 0 when it's not null.
- RegularPrice should have a value when PreviousPrice is not null.
- PreviousPrice should be greater than 0 when it's not null.
- If shipping is present:
  - Shipping.Price should be greater than or equal to 0 when it's not null.
  - Shipping.DestinationCountry cannot be null.

Possible http responses:

200	Successful request.
400	Returned when invalid arguments are supplied. The response body (1) will explain what properties are incorrect.
401	Returned when authentication failed (e.g. when an invalid or expired API key was used).
404	Returned when the external id is unknown.
500	Returned when the request failed.

Response body:

```
{
  "error": "string",
  "errorDetails": {
    "additionalProp1": "string",
    "additionalProp2": "string",
    "additionalProp3": "string"
  }
}
```

[GET] https://shopitem.api.beslist.nl/v3/offer/shops/{shopId}/offers/{externalId}

Content-Type: application/json

You can retrieve the information you have just updated once it has been processed by our backend system. Note that updates are not available instantaneously.

Possible http responses:

200	Successful request. Response body contains the current data for your external id.
401	Returned when authentication failed (e.g. when an invalid or expired API key was used).
404	Returned when the external id is unknown.
500	Returned when the request failed.

NOTE: in a GET request for a single offer, only the fields available in the document will be returned. This means that (for example), when stock is not updated in a PUT, it will not be returned when using the GET endpoint for the same offer.

Batch updates:

[PUT] https://shopitem.api.beslist.nl/v3/offer/shops/{shopId}/batch
Content-Type: application/json

You can update multiple offers by sending a PUT request containing a json formatted body as follows:

```
[
  {
    "externalId": "string",
    "price": {
      "regularPrice": 0,
      "previousPrice": 0
    },
    "shipping": [
      {
        "destinationCountry": "nl",
        "price": 0,
        "deliveryTime": "string"
      }
    ],
    "stock": {
      "level": 0
    }
  }
]
```

You can send partial data. i.e. only price.regularPrice. Or only shipping to destinationCountry “nl”.

The response to this call will either return http status code 201 along with a http header “location” property when successful. Send a GET request to the given location to verify the status of the batch update after a while.

Datamodel:

Validation:

- ExternalId cannot be empty.
- RegularPrice should be greater than 0 when it's not null.
- RegularPrice should have a value when PreviousPrice is not null.
- PreviousPrice should be greater than 0 when it's not null.
- If shipping is present:
    - Shipping.Price should be greater than or equal to 0 when it's not null.
    - Shipping.DestinationCountry cannot be null.

Possible http responses:

201	Successful request. Response headers contain the location where the batch status can be downloaded.
400	Returned when invalid arguments are supplied. The response body (1) will explain what properties are incorrect.
401	Returned when authentication failed (e.g. when an invalid or expired API key was used).
404	Returned when the shop was not found.
500	Returned when the batch update failed.

Response body 1

```
{
  "error": "string",
  "errorDetails": {
    "additionalProp1": "string",
    "additionalProp2": "string",
    "additionalProp3": "string"
  }
}
```

[GET] https://shopitem.api.beslist.nl/v3/offer/shops/{shopId}/batch/{batchId}

Content-Type: application/json

You can retrieve the status of a batch update by sending a GET request.

Possible http responses:

200	Successful request. Response body contains the current data for your batch.
400	Returned when invalid arguments are supplied. The response body (1) will explain what properties are incorrect.
401	Returned when authentication failed (e.g. when an invalid or expired API key was used).
404	Returned when the batch and/or shop are not found.
500	Returned when the request failed.

Response body:

```
{
  "error": "string",
  "errorDetails": {
    "additionalProp1": "string",
    "additionalProp2": "string",
    "additionalProp3": "string"
  }
}
```

API access keys

Every call to an endpoint url must be authenticated with an API access key. This key must be sent in a header “apiKey”. An access key can be generated for you, but is not stored and cannot be recovered or reverse engineered. If you lose your key you will need to have a new one generated for you. Keep it secured, because it enables making changes to product of your shop!

Getting an API key

You may contact productfeed@beslist.nl and request an API key for production.

Before publishing the data, we will check if all updates are correct.

For shops who are still or separately live on a cost per click business model it is important to prevent corrupting live data.

Contact productfeed@beslist.nl for receiving credentials for a test shop. They will fill the test-shop with actual data for your web shop and provide you with an API key.

Once products in your data feed have been updated successfully for the test shop, your request for a live key will be considered.

2. Code examples

The following code examples are not to be used in a real environment, but can help as an example of how to consume the Shopitem API.

Getting a list of shops associated with an API key

```
curl -X GET -H "application/json" -H "apikey:
12345678abcdef134353738923929020202390 3abcdabc356748494903023abcfde"
https://shopitem.api.beslist.nl/auth/v1/shops
```

This curl request can either return a 401/Unauthorized status code with a json response body if the apikey is invalid, or a 200/OK status code with a list of shops for which the apikey can be used. The list of shops are key-value pairs in json, where the key is the shop id that must be used in other API calls:

```
{
  "12345": "Debesteshopvoordingen.eu",
  "75634": "Goedkopespullenkopen.nl"
}
```

Two things should be noted:

1. This API endpoint is a bit of an exception, as it does not return a list of shop resources, but a list of key-value pairs.

2. The keys in this list are the shop id values that must be used in the other endpoints

Get your current data

```
curl -X GET -H "application/json" -H "apikey:
12345678abcdef134353738923929020202390 3abcdabc356748494903023abcfde"
https://shopitem.api.beslist.nl/v3/offer/shops/12345/offers/12abcd13
```

This curl request will return a 200/OK status code on a successful request and a json representation of the shopitem of “Debesteshopvoordingen.eu” with unique identifier 12abcd13.

```
{
  "shopId": 12345,
  "externalId": "12abcd13",
  "price": {
    "regularPrice": {
      "value": 99.95,
      "lastUpdate": "2018-05-03T09:35:43.431Z"
    },
    "previousPrice": {
      "value": 110.95,
      "lastUpdate": "2018-05-03T09:35:43.431Z"
    }
  },
  "shipping": [
    {
      "destinationCountry": "nl",
      "price": {
        "value": 5.95,
```

```
        "lastUpdate": "2018-05-03T09:35:43.431Z"
      },
      "deliveryTime": {
        "value": "1 werkdag",
        "lastUpdate": "2018-05-03T09:35:43.431Z"
      }
    },
    {
      "destinationCountry": "be",
      "price": {
        "value": 9.95,
        "lastUpdate": "2018-05-03T09:35:43.431Z"
      },
      "deliveryTime": {
        "value": "2 – 4 werkdagen",
        "lastUpdate": "2018-05-03T09:35:43.431Z"
      }
    }
  ],
  "stock": {
    "level": {
      "value": 546,
      "lastUpdate": "2018-05-03T09:35:43.431Z"
    }
  }
}
```


Updating the price

```
curl -X PUT -H "application/json" -H "apikey:
12345678abcdef134353738923929020202390 3abcdabc356748494903023abcfde" -d { "price": {
"regularPrice": 119.95 } }
```

Updating the price and previous price

```
curl -X PUT -H "application/json" -H "apikey:
12345678abcdef134353738923929020202390 3abcdabc356748494903023abcfde" -d { "price": {
"regularPrice": 119.95, "previousPrice": 139.95 } }
```

Updating the stock

```
curl -X PUT -H "application/json" -H "apikey:
12345678abcdef134353738923929020202390 3abcdabc356748494903023abcfde" -d { "stock": {
"level": 200 } }
```

Updating the deliveryTime

```
curl -X PUT -H "application/json" -H "apikey:
12345678abcdef134353738923929020202390 3abcdabc356748494903023abcfde" -d { "shipping": [ {
"destinationCountry": "nl", "price": 2.95, "deliveryTime": "1 werkdag" } ] }
```

Any combination of updates can be created.