AUTODOC

Technical Guideline for Dropshipping Integration of Suppliers

Inhaltsverzeichnis

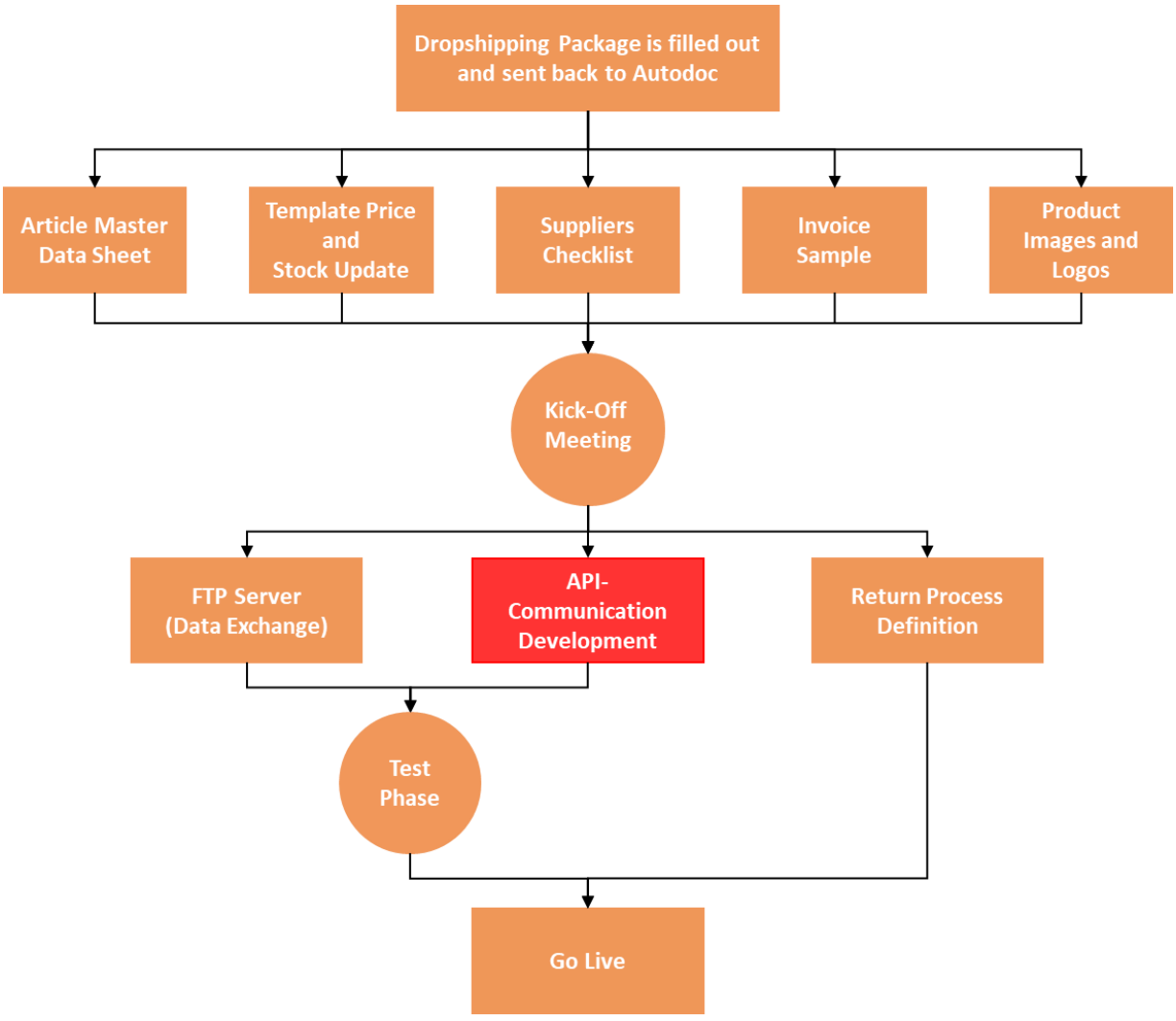
Overview

This document provides technical specifications to help guide your successful integration into the drop-shipping process. This includes required information, and the format that information is expected to be in, with examples of implementation where appropriate.

ATTENTION!

Order communication between supplier system and Autodoc is done only via the (developed) API interface. We do not accept any other alternatives. Please give this documentation to your IT-Provider.

The Integration process is as follows:



FTP Server: Price and Stock Updates

The Integration steps: Price and Stock Updates

Step 1. Send us one file in CSV format, Template Price and Stock Update.

Step 2. We will examine if the file remains the required structure and headers to our template.

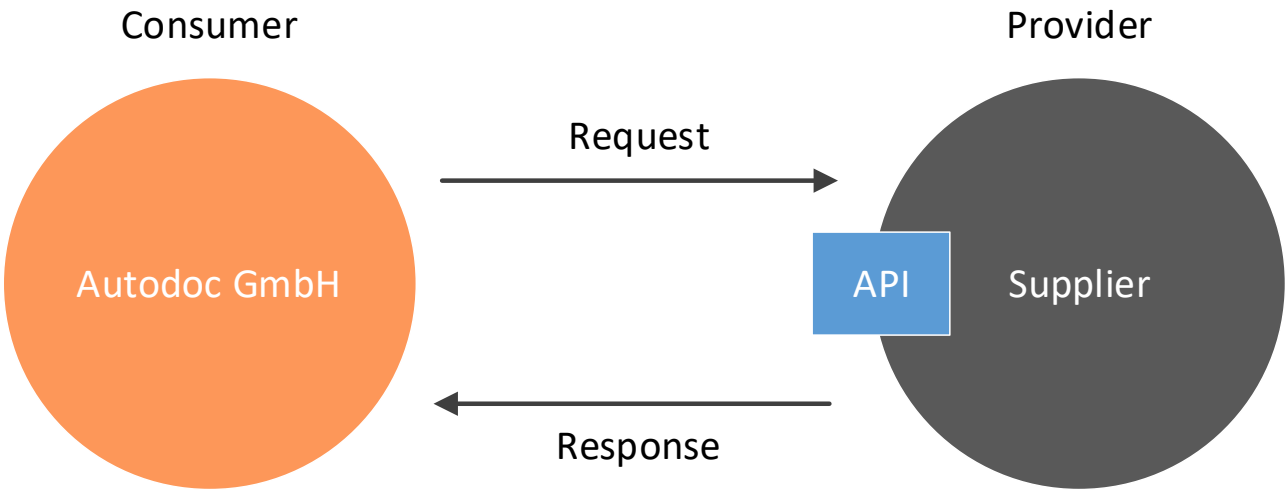
Step 3. If the file is ok, Autodoc will arrange an FTP account for you and send you all required access credentials.

Step 4. Supplier uploads File on FTP server.

- ➔ The Price and Stock File must be updated **hourly.**
- ➔ The Price and Stock File must always come to FTP Server with the same **Name**, **Format,** and **Structure**. It is mandatory that the name of the file for a specific country indicates this country. For example, ***SupplierName_DE.csv***.
  A file with a date in the title will not be processed. The reason is that the date changes and, accordingly, the names will be different and will not be processed.

API Communication Development

Overview



As the API consumer - **Autodoc** has specific requirements for a Web Service to successfully make business transactions with the Supplier.

It is required that the Supplier provides access to his existing API or the Supplier is developing his API at the moment.

Integration Process: API communication

Step 1. Supplier implements the required methods and sends API documentation.

Step 2. After a documentation research, Autodoc initiates a testing phase. Supplier creates account and provides information: access key, port number, Test Order ID and data feed file with products (example file).

Step 3. Autodoc confirms a successful testing.

In the following chapters are listed all API methods that are mandatory.

API documentation from Supplier

Documentation WSDL or WADL and additional documentation that describes the Suppliers API would be very helpful for the communication development.

The following API methods are **mandatory**:

- **get_stock** – to get stock of the products.
  This method must return availability and price of the product, by country.

- **create_order** – to place order on supplier's side.
  In case of any error with order this method must return code of the error and text of the error, accordingly.

- **get_tracking_numbers** – to get order tracking number.

- **get_order_status** – to check other order statuses.
  The following statuses are mandatory: ordered, shipped, delivered to customer.

- **cancel_order** – to cancel order.

Credentials for connection with Supplier API

Data required for each separate country:

- API login;
- API password;

Autodoc API template for Dropshipping Suppliers

The Web Service should be designed to provide the following features: stock enquiry, tracking numbers receiving, order manipulations (creation, cancellation, status checking).

Supplier should provide logins and passwords for authentication for each delivery country.

As a request/response data the most preferred format is JSON.

The presence of a test environment (sandbox) will be very helpful during API implementation. The required methods are provided below.

API-Method: get_stock

The stock enquiry allows to see if an item is available, its quantity and price. The query parameter can be the supplier's product ID or EAN. In addition, it should be possible to differentiate prices by country if the supplier operates in several countries.

URL: <Example: /api/getstock/:productId,
/api/getstock/:ean,
/api/getstock/:productId&country>

Method: `GET`

URL Parameters:

```
productId = [integer|string] OR ean = [string]
```

URL Parameters by contry:

```
productId=[integer|string] OR ean=[string]&country=[string]
```

Request example:

```
api/getstock/productId=WEKR0093&country=FR
```

Response example:

```
{
     "product-id":  "12345",
     "amount":  "5",
     "price":  "9.99",
     "success":  1|0,
     "error-message":  "error message if success == 0"
}
```

API-Method: create_order

Several order items can be sent at the same time in one request. A unique order ID must be generated for each order and sent in the response. The API must report any errors that may have occurred (address data incomplete, item not available, etc.) and sends an error code and error message in the response. Note: Address should have 3 address lines (street, number, address-description) except zip, city, country, phone and email.

URL: <Example: /api/createorder>

Method: `POST`

Request example:

```
{
   "items" : [
      {
        "product-id" : "12345",
        "product-amount" : "3",
        "product-price" : "204.83"
      },
      {
        "product-id" : "54321",
        "product-amount" : "2",
        "product-price" : "232.60"
      }
   ],
   "autodoc-order-id" : "28838105",
   "delivery-firstname" : "John",
   "delivery-lastname" : "Doe",
   "delivery-street" : "Zagumnie",
   "delivery-number" : "27",
   "address-description" : "Floor 4, Door 16",//
   "delivery-zip" : "32-440",
   "delivery-city" : "Berlin",
   "delivery-country" : "DE",
   "delivery-phone-number" : "+49880501277",
   "delivery-email" : "user@autodoc.de"
}
```

Request example:

```
{
   "items" : [
       {
         "ean" : "12345",
         "product-amount" : "3",
         "product-price" : "104.83"
       }
   ],
   "autodoc-order-id" : "28838105",
   "delivery-firstname" : "John",
   "delivery-lastname" : "Doe",
   "delivery-street" : "Zagumnie",
   "delivery-number" : "27",
   "address-description" : "Floor 4, Door 16",
   "delivery-zip" : "32-440",
   "delivery-city" : "Berlin",
   "delivery-country" : "DE",
   "delivery-phone-number" : "+49880501277",
   "delivery-email" : "user@autodoc.de"
}
```

Success response example:

```
{
     "order-id":  "S00000001", //supplier order id
     "order-items": [
         {
             "product-id":  "12345",
             "product-amount" : "3",
             "product-price" : "204.83"
         },
         {
             "product-id":  "54321",
             "product-amount" : "2",
             "product-price" : "232.60"
         }
     ],
     "success": 1
}
```

Error response example:

```
{
   "success" : 0,
   "error-message" : "Not enough stock",
   "error-code" : "10008"
}
```

API-Method: get_tracking_numbers

Return tracking info for each order id, which was generated on supplier's side and sent to AutoDoc in create_order response

NOTE: If there is no possibility to have `get_tracking_numbers` method in API, supplier should give an alternative way to get tracking numbers for orders. For example to send CSV files on FTP account.

URL: <Example: /api/trackinfo/:orderId>

Method: `GET`

URL Parameters: `orderId = [integer|string]`

Response example:

```
{
   "order-id":  "S00000001", //supplier order id
   "parcels": [
      {
        "date-estimated-delivery":  "03.03.2020",
        "date-shipped":  "28.02.2020",
        "parcel-number":  "123456788",
        "service":  "GLS",
        "tracking": "http://www.gls-group.eu?txtRefNo=123456788"
      },
      {
        "date-estimated-delivery":  "03.03.2020",
        "date-shipped":  "28.02.2020",
        "parcel-number":  "123456789",
        "service":  "GLS",
        "tracking": "http://www.gls-group.eu?txtRefNo=123456789"
      }
   ],
   "success":  1|0,
   "error-message":  "error message if success == 0"
}
```

API-Method: get_order_status

Return current order status (For example: 1 = in progress, 2 = order shipped, 3 = order canceled, 4 = returned, 5 = ordered, 6 = delivered to customer) by id, which was generated on supplier's side and sent to AutoDoc in create_order response

NOTE: This method is mandatory. The following statuses are mandatory:

- Ordered
- Canceled
- Shipped
- Delivered to customer.

URL: <Example: /api/orderstatus/:orderId>

Method: `GET`

URL Parameters: `orderId = [integer|string]`

Response example:

```
{
   "order-id":  "S00000001" //supplier order id
   "success" : 1|0,
   "status" : 3, // or "shipped"
   "error-message" : "error message if success == 0"
}
```

API-Method: cancel_order

Cancel the exist order by id, which was generated on supplier's side and sent to AutoDoc in create_order response.

NOTE: If there is no possibility to have `cancel_order` method in API, supplier should give an alternative way to cancel orders.

URL: <Example: /api/cancelorder>

Method: `POST`

Request example:

```
{
    "order-id":  "S00000001", //supplier order id
    "comment":  "Double order"
}
```

Response example:

```
{
    "success":  1|0,
    "error-message": "error message if success == 0"
}
```

FAQ

Dropshipping price calculation

- Is it possible to invoice separately for shipping?

No, the prices must be DDP to the customer. We need to have the DDP price to guarantee the comparability of the prices in our system.

- Do we have separate purchasing prices for Dropshipping and Just in Time deliveries?

Yes, that is why we work with different price files. The system compares the prices from the Dropshipping suppliers and gives the order to the best price supplier for the corresponding country.

- Is a price- & stock file needed for each country or can we put it all together in one file?

We need **one file for each country** with all articles, stocks, and prices. The reason is that the configuration of one separate file is more flexible. Many suppliers start with a few countries and then add countries as they develop. With the files for each country, we can also easily block a country without having to block all countries in case of a change of plans.