Shipmall eLogist webservice specification

Version 1.26

Contents

Change log

1.12	More information in InventoryChangesGet response, possibility to state purchase prices in storage orders
1.13	New parameter `suspended` in `DeliveryOrder` method
1.14	New parameter `deliveryDate` in `StorageOrder` method
1.15	Shipping details extended (`DeliveryOrderStatusGet` and others)
1.16	New parameter `packingInstruction`, reporting of prices for processing and shipping
1.17	New field `beforeDateTime` added for entering a time range in the past. Methods `DeliveryOrderHistoryGet` and `StorageOrderHistoryGet` added
1.18	New parameters for methods `ProductUpdate` and `StockInventoryGet`
1.19	Added methods `ProductLabelSet`, `ProductLabelDelete` and `ReceivedPaymentsGet`
1.20	New parameter `service` for method `BranchListGet`
1.21	Result pagination added to selected methods, `StockInventoryGet`(`News`) extended with a new parameter `disregardVariants` and a new value `available`
1.22	New value `demanded` in stock information. Listing of incoming and outgoing serial numbers.
1.23	Added `food` parameter in product specification
1.24	Added methods `ProjectListGet`, `CarrierListGet`, `DeliveryOrderDocumentAdd`, `StorageOrderGet`. New parameter `purpose` for documents. Reporting of date when last item was received at warehouse for storage order (`lastReceiptDate`).
1.25	Added `problem` field to delivery order status information (`DeliveryOrderStatusGet` a další)
1.26	Added `lotTracking` parameter in product specification, expanded display of expiration dates and lot numbers in inventory status and movements

Technology

Protocol

The service provides an interface with SOAP 1.2 protocol. Formal description (WSDL) is attached. Application libraries for working with SOAP protocol are available for most platforms – Java, PHP, .NET and others. A sample implementation of connection with PHP is attached as well.

Data is trasported in a standard manner over HTTP 1.1 protocol.

Service URL

For testing purposes there is a service at https://elogist-demo.shipmall.cz/api/soap.

Production version runs at https://elogist.shipmall.cz/api/soap.

Authentication

The interface uses HTTP Basic method (username and password) for authentication. Encryption is provided by HTTPS. Username and password will be assigned by our sales representative.

Regrettably, implementation on WCF/.NET platform is not completely straightforward, but still possible using instructions at https://msdn.microsoft.com/en-us/library/ms733775.aspx

Upgrading to newer protocol versions

New versions may appear over time, introducing new functionality. Previous versions of the service continue to work without changes. If you are fine with your last used version, you can keep using it, nothing changes for you.

The version used for communication is determined with the version of WSDL definition you use. To use the latest version, get the current WSDL file at https://elogist.shipmall.cz/api/soap?wsdl and use that in your implementation.

You can get older versions of WSDL definition on addresses with form
`https://elogist.shipmall.cz/api/soap?wsdl-v<version.number>`
e.g. https://elogist.shipmall.cz/api/soap?wsdl-v1.5.

System overview

With the service you can

- send in delivery orders
- send in storage orders
- check order statuses
- cancel orders
- watch stock movements and get current quantities in stock
- obtain details of payments sent to client’s bank account

Delivery orders

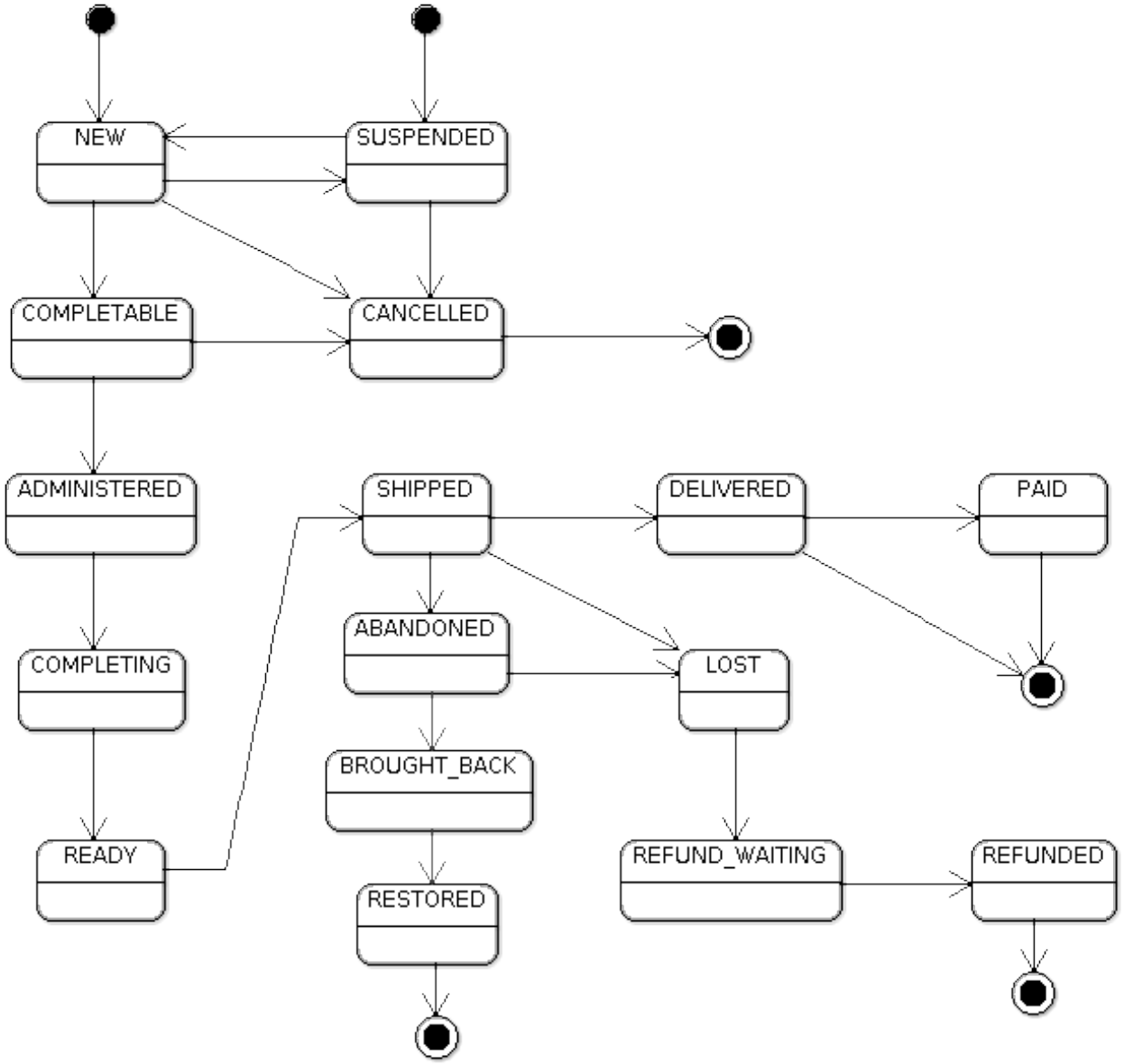
Delivery order is an instruction to send goods to a particular customer, usually based on an order in an e-shop. The order includes the recipient address, items list, shipping service, accompanying documents and other optional data.

An order can be sent even if the respective stock is out of stock. The order waits in a queue and its processing will continue automatically on restock.

Order statuses

NEW	Order was accepted and waits for further processing
SUSPENDED	Order was accepted and waits for confirmation or missing data entries
CANCELLED	Order was cancelled
COMPLETABLE	All order items needed for the order are in stock
ADMINISTERED	Order is getting checked and prepared at the backoffice
COMPLETING	Order is getting collected in store and packed
READY	Order is ready to be handed over to the carrier
SHIPPED	Order is on its way to the recipient
DELIVERED	Order has been delivered the recipient
ABANDONED	The carrier failed to deliver the order
PAID	A payment (COD) was sent to client’s account
RESTORED	Order returned and items were accepted back to store
LOST	The carrier lost or damaged the shipment during transport
REFUND_WAITING	Awaiting client's documents needed for the transport claim
REFUNDED	The compensation was sent to the client's account

State diagram



Only orders with cash on delivery go to PAID status. For others, the last status is normally DELIVERED.

Status changes can be tracked using methods described further in the document.

Storage orders

The purpose of a storage order is to announce client's intention to bring or send new stock to the warehouse. It is assumed that the goods will arrive in a single shipment.

It may be a clone of your order at the supplier. Based on your needs, the goods can be sent either in larger batches to make a reserve, or just the quantities needed for current delivery orders.

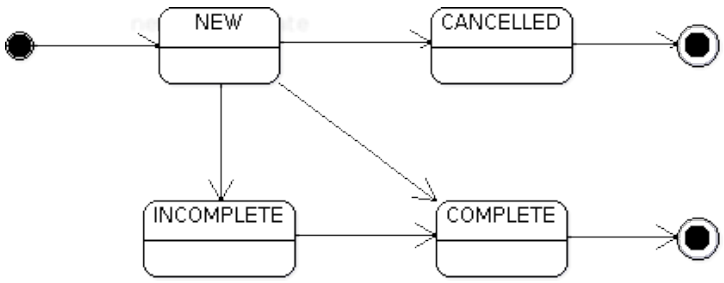
Having a storage order helps considerably with the identification of goods received.

Order statuses

NEW	Order was accepted and a shipment is expected
CANCELLED	Order was cancelled

INCOMPLETE	Part of the goods was received
COMPLETE	All announced quantities were received

State diagram



Order status can be tracked using appropriate methods described further in the document. Order can be cancelled in NEW status.

Product catalog

The system keeps a partial copy of e-shop product catalog, at least for the items in stock. New items can be added in three ways:

1. importing a CSV file in the web application at the beginning of cooperation and at further updates. You can do it yourself (Klient – Katalog menu) or we carry it out for you.
2. as a part of delivery and storage orders – a detailed product specification can be included in the items list
3. sending product specifications separately using `ProductUpdate` method

Products with variants

When a particular item is supplied in multiple variants, while their properties don't affect price etc. and all the variants are filed as a single product in the e-shop (with a common product ID) only with various combinations of attributes attached, it is possible to specify it that way in communication with the webservice too.

To give an idea, let's imagine a t-shirt with ID 1234, supplied in different sizes and colors. A product definition includes not only the ID and a name, but also a list of attributes (size, color). In an order there will be written that 1pc of ID 1234 is required with a variant (size=M,color=yellow). It is of course possible to request multiple variants of the same product within a single order.

It is however not a preferred solution. If the total number of variants is small, we recommend to register them (or at least communicate them towards us) as separate products with their individual IDs.

Stock status

Goods physically present in stock is divided into three groups

- reserved – taken by not-yet-sent orders
- available – usable for future orders
- blocked – not ready for use at the moment (e.g. damaged)

Beside that, "expected" count based on unfulfilled storage orders is calculated, giving numbers still on the way from the supplier.

When a shipment arrives from the supplier and new items are added to stock, the system automatically founds delivery orders waiting in queue and reserves quantities needed for them to be dispatched. These quantities are added directly as "reserved" while the remainder counts in as "available".

Projects

When the client runs multiple e-shops with the same products, they can still be serviced using a single implementation instance.

Orders origin can be stated in two ways.

When order numbers from different e-shops overlap, it is possible to create separate projects that will then serve as namespaces. An order is assigned to a specific project using `projectId` parameter. You can also define a text for each project, that should identify the e-shop on shipping labels.

Alternative way to set e-shop name on labels for each order individually is using the `sender.label` parameter. So, if all order numbers are unique over all e-shops, it is possible to manage with a single project.

You will find a list of your projects on the page with supplemental developer information: https://elogist.shipmall.cz/klient/info/developer

You can get a list of your projects by calling the ProjectListGet method, or you can find it on the page with supplemental developer information: https://elogist.shipmall.cz/klient/info/developer

Order processing simulation

To test responses of the e-shop to order status changes you can use these functions in application https://elogist-demo.shipmall.cz:

1) simulate new stock reception (in the list at *Příchozí – Objednávky* use link *Simulovat naskladnění* in respective lines)

2) simulate outgoing shipment (in the list at *Odchozí – Objednávky* check one or more orders and use button *Simulovat odeslání* under the table)

Results pagination

Selected methods support result pagination, a way to obtain a larger result set successively in smaller steps. The desired page size (number of records) is set using `pageSize` parameter.

Request example:

```
<StockInventoryGet>
  <subset>IN_STOCK</subset>
  <pageSize>200</pageSize>
  <page>1</page>
</StockInventoryGet>
```

Response example:

```
<StockInventoryGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
    <totalItems>340</totalItems>
    <totalPages>2</totalPages>
    <pageSize>200</pageSize>
    <page>1</page>
  </result>
  <stockInventory>
    <product productId="13515">
      <stored>10</stored>
      <expected>30</expected>
<!-- first 200 records follow... -->
```

By default (when parameters `page`/`pageSize` are not given), paging is turned off in order to maintain compatibility with older versions of the protocol.

Interface methods

ProjectListGet

A list of existing projects for the client (see Projects).

Request format

```
<ProjectListGet/>
```

Method has no parameters.

Response format

```
<ProjectListGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <project>
    <projectId>muj-eshop-cz</projectId>
    <name>Můj eshop CZ</name>
  </project>
  <project>
    <projectId>muj-eshop-sk</projectId>
    <name>Můj eshop SK</name>
  </project>
</ProjectListGetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
projectId	text	project ID (used as parameter in further method calls)
name	text	project name

CarrierListGet

List of available carriers and their shipping services.

Requst format

```
<CarrierListGet/>
```

Method has no parameters.

Response format

```
<CarrierListGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <carrier>
    <carrierId>CPOST</carrierId>
    <name>Česká pošta</name>
    <service>
      <name>Balík Do balíkovny</name>
      <hasBranches>true</hasBranches>
    </service>
    <service>
      <name>Balík Do ruky</name>
      <hasBranches>false</hasBranches>
    </service>
  </carrier>
  <carrier>
    <carrierId>DPD-CZ</carrierId>
    <name>DPD CZ</name>
    <service>
      <name>DPD Private</name>
      <hasBranches>false</hasBranches>
    </service>
    <service>
      <name>Pickup</name>
      <hasBranches>true</hasBranches>
    </service>
  </carrier>
</CarrierListGetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
carrierId	text	carrier code (used in orders for `carrierId` field)
name	text	carrier name
service.name	text	shipping service name (used in orders for `service` field)
hasBranches	boolean	delivery to pickup points (a list can be obtained by calling BranchListGet

DeliveryOrder

Send in a new delivery order

Request format

```
<DeliveryOrder>
  <projectId>klient.cz</projectId>
  <orderId>C1624</orderId>
  <customerOrderId>651651651</customerOrderId>
  <paymentId>388438</paymentId>
  <orderDateTime>2010-12-23T11:58:01+02:00</orderDateTime>
  <suspended>false</suspended>
  <packingInstruction>opatrně!!!</packingInstruction>
  <sender>
    <label>Klient CZ</label>
  </sender>
  <recipient>
    <name>František Omáčka</name>
    <address>
      <company>FO Servis, s.r.o.</company>
      <street>Osadní 14</street>
      <city>Praha 7</city>
      <postcode>170 00</postcode>
      <country>CZ</country>
    </address>
    <phone>+420777666555</phone>
    <email>omacka@foservis.cz</email>
  </recipient>
  <shipping>
    <carrierId>PPL</carrierId>
    <service>PPL Parcel CZ Private</service>
    <branchId></branchId>
    <cod currency="CZK">2830.0</cod>
    <insurance currency="CZK">2830.0</insurance>
    <comment>zvoňte na zvonek „Správce“</comment>
    <sendAt>2010-12-23</sendAt>
    <option>
      <name>evening_delivery</name>
      <value>true</value>
    </option>
  </shipping>
  <orderItems>
    <orderItem>
      <productSheet>
        <productId>1234597</productId>
        <barcode>123457</barcode>
        <productNumber>A00DD3561</productNumber>
        <name>BDR-202</name>
        <description>Blu-ray/DVD/CD zapisovací mechanika</description>
        <vendor>Pioneer</vendor>
        <attributeSet>
          <attribute>barva</attribute>
          <attribute>připojení</attribute>
        </attributeSet>
        <quantityUnit>PC</quantityUnit>
      </productSheet>
      <variant attribute="barva" value="černá"/>
      <variant attribute="připojení" value="SATA"/>
      <quantity>1</quantity>
    </orderItem>
    <orderItem>
```

```
      <productId>455435</productId>
      <quantity>2</quantity>
    </orderItem>
  </orderItems>
  <documents>
    <document>
      <title>Faktura</title>
      <type>application/pdf</type>
      <purpose>INSERT</purpose>
      <content md5checksum="48dbb8ee55311848967f56ead404e6ee">
        cGVsYS
        Bob3Bh
      </content>
    </document>
  </documents>
</DeliveryOrder>
```

projectId	text	E-shop/project identification (see https://elogist.shipmall.cz/klient/info/developer for assigned project IDs)
orderId	text	order ID – a reference number for further communication and status querying. It is mandatory and must be unique at least within a single project (see projects). It is possible but not required to match order number in e-shop (the next field can be used for that purpose)
customerOrderId	text	order ID in the e-shop – ie. order number seen by end user in the e-shop
paymentId	integer	will be used as variable symbol for outgoing payment of collected cash on delivery (when sending as individual payments is preferred). Probably same symbol as the one used in invoice. Optional.
orderDateTime	dateTime	Date and time when order was created in the e-shop. Optional
suspended	boolean	Don't send for now, wait for order confirmation. Optional (default value is `false`).
packingInstruction	text	Instructions and notes for order processing in warehouse. Optional
label	text	Text to use as e-shop name on shipping label. Optional, when not provided, project name will be used.
name	text	Recipient name
company	text	Recipient company (optional)
street	text	Recipient street
city	text	Recipient city
postcode	text	Recipient postcode; with *Balík Na poštu*: code of receiving post office
country	text	Two-letter country code (ISO 3166-1), e.g. CZ, SK
phone	text	Recipient phone number
email	text	Recipient e-mail (optional)
carrierId	text	Carrier, e.g. PPL, DPD-CZ, DHL, CPOST, WEDO, ZASILKOVNA
service	text	Shipping service, more information below
branchId	text	Branch code for pickup points of *Uloženka* or *Zásilkovna* carriers and PPL ParcelShop service. Optionally, the postcode for Czech Post's *Balík Do balíkovny.* Can be omitted altogether with other carriers. Get branch lists with BranchListGet method
cod	decimal	Cash on delivery amount. Optional. Stated in destination country currency.
currency	text	Three-letter currency code (ISO 4217), e.g. CZK, EUR.

insurance	decimal	Package value. Optional. Stated in destination country currency.
comment	text	Note for driver. Optional
sendAt	date	Hold order processing until specified date. Optional.
option	name/value pair	Request additional shipping options (like e.g. evening delivery). Described in greater detail in section „Additional shipping options“.
productId	text	product ID. An identifier visible to e-shop operator is preferred, ideally also matching the identifier used by product supplier – helps recognizing the product in the warehouse and communication with the client.
barcode	text	Product barcode (optional)
productNumber	text	Producer product number (optional)
name	text	Product name
description	text	short description, optional, provide only when considered helpful in product identification (not a multi-paragraph description)
vendor	text	Producer (optional)
attribute	text	Name of the property that defines a variant (see products with variants)
quantityUnit	text	Quantity unit - PC = piece, KG, METER, SQUARE_METER, PALLET
attribute	text	Name of variant property (matches attributeSet.attribute)
value	text	Value of variant property
quantity	int	Quantity ordered
title	text	Document title (e.g. „Invoice F080624“, „Delivery note“ etc.)
type	text	File MIME type, currently the only expected option is „application/pdf“
purpose	text	document purpose (INSERT, SHIPPING_LABEL, PRODUCT_LABEL). Optional.
content	text	File contents, base64-encoded
md5checksum	text	File checksum using MD5 algorithm, 32-character hexadecimal form

Response format

```
<DeliveryOrderResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <deliveryOrderStatus>
    <projectId>klient.cz</projectId>
    <orderId>1624</orderId>
    <changed>2010-12-23T12:00:00+02:00</changed>
    <status>NEW</status>
    <carrierId>PPL</carrierId>
    <service>PPL Parcel CZ Private</service>
    <sysOrderId>135153</sysOrderId>
    <itemCount>2</itemCount>
    <unitCount>3</unitCount>
    <destinationCountry>CZ</destinationCountry>
    <cod currency="CZK">2830.0</cod>
    <dateCreated>2010-12-23</dateCreated>
  </deliveryOrderStatus>
</DeliveryOrderResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
Remaining fields with order details are described under DeliveryOrderStatusGet		

Postcode format

Postcode can be written including eventual spaces and dashes. Real existence is not checked (except for *Balík Na poštu*), however the code must adhere to formal rules, namely correct number of digits. List of rules is given on page https://elogist.shipmall.cz/klient/info/developer.

Balík Na poštu (Czech Post – Delivery To Post Office)

With this service the field `postcode` does not carry recipient adress postcode, expected is postcode of the receiving office selected by the customer. A sample implementation of office selection and a list of available offices is provided by Czech Post at http://www.postovnibaliky.cz.

This service also requires providing a mobile phone number or an email address.

Balík Do balíkovny (Czech Post – Delivery To Parcel Pickup Outlet)

With this service, either the `postcode` or `branchId` field will hold the postcode of receiving office chosen by the customer (if both are used, `branchId` takes precedence). A sample implementation of office selection and a list of available offices is provided by Czech Post at http://www.postovnibaliky.cz. In adition, it is possible to obtain a list of available pickup outlets using the BranchListGet method.

This service requires the recipient's email address and it is recommended to state a mobile number for SMS notifications.

Phone number format

Phone number is mandatory and must be specified in the international format (E.164) without spaces or other delimiters. Examples of correct format: +420602123456 or +421987654321.

Carrier and shipping service

The list of carriers given before may change over time, an up to date list is available in the web application at https://elogist.shipmall.cz/klient/info/developer.

When specifying a particular shipping service (e.g. *PPL Parcel CZ Business* with PPL), use the name itself. Providing one is not required though; when omitted, a default service is selected automatically. You'll find an updated list of services for each carrier at the address above as well.

Order price and Cash on delivery

It is customary to provide total order value in the element `insurance`, some carriers request this information. We don't keep prices of individual order items. In orders with cash on delivery the two amounts (`insurance` and `cod`) are usualy equal. In orders without cash on delivery, `cod` is zero while `insurance` is still non-zero.

Requirements for cash on delivery:

- it must be in currency used in destination country (e.g. in EUR to Slovakia)
- the amount must respect the smallest coins (e.g. CZK in whole crowns, HUF in multiples of 5 Ft)

Cash on delivery is supported by most carriers in Czechia and in some cases abroad too. An overview of supported combinations is listed again on https://elogist.shipmall.cz/klient/info/developer.

Additional shipping options

Optionally, a few special services can be requested with some carriers. Currently supported options are listed on page https://elogist.shipmall.cz/klient/info/developer. Example:

Service	name	value	Available with
Evening delivery (after 5pm)	evening_delivery	true false	PPL Parcel CZ Private
Delivery on weekend	weekend_delivery	true false	GW
Delivery at specified hour	delivery_time_window	9-12 12-14 14-17 false	GW
Delivery to specified floor	deliver_to_floor	Floor number (>0) false	GW

Shipping options are entered as one or more `option` elements in `shipping` section, e.g.

```
<shipping>
  <option>
    <name>weekend_delivery</name>
    <value>true</value>
  </option>
  <option>
    <name>delivery_time_window</name>
    <value>14-17</value>
  </option>
</shipping>
```

If a particular option is not desired, it is posible to either omit the respective `option` or to leave it with `false` value, whatever is easier to implement.

Order items

There are two ways of specifying order items:

1) if products are already created in system catalog (see product catalog), it is sufficient to enter only product ID and quantity, e.g.

```
<orderItem>
  <productId>231344</productId>
  <quantity>5</quantity>
</orderItem>
```

2) the other way is to repeat full product specification in every order using `productSheet` element. Minimal form could be:

```
<orderItem>
  <productSheet>
    <productId>231344</productId>
    <name>Nokia 3310</name>
    <quantityUnit>PC</quantityUnit>
  </productSheet>
  <quantity>5</quantity>
</orderItem>
```

Which way to choose depends mainly on your expected frequency of changes in catalog. If you plan to add new products often, you might consider the second alternative.

Complete list of fields of `productSheet` section is specified under `ProductUpdate` method documentation. This method parameter structure is identical to `productSheet` structure.

If it helps your implementation in any way, it is possible to include one product more than once in an order (as multiple order items). The resulting order will contain a sum for every product.

Accompanying documents

You can attach one or more documents in PDF format. They can be printed and attached to the shipment or otherwise used during its processing. Document data need to be inserted in base64 encoding.

In situations where you need to attach a document in a separate request and not as part of the order creation, you can use the DeliverOrderDocumentAdd method.

You can specify how the document is handled by entering one of the following values in the `purpose` field. If the field is not filled in, we will try to estimate the purpose from the file name.

INSERT	a document (e.g. invoice) to be printed and enclosed with the contents of the shipment
PACKAGE_LABEL	shipping label (e.g. for Alza courier service), we expect A6 format, if there are more labels then they must be on separate pages
PRODUCT_LABEL	labels to be attached to products – intended for BCX codes when sending to Alza

If you can control the version of PDF format used, we prefer version 1.4 or lower for easier batch processing. Although no particular limit on file size is imposed, big files may cause problems. Please try to reduce included fonts, use font subsetting, consider omitting images. It is usually possible to keep file size under 100 kB.

Implementation note if you are using PHP: the `SoapClient` class performs base64 encoding for you automatically due to the fact that `content` element is defined in WSDL schema as type `xsd:base64Binary`. Therefore, avoid calling the `base64_encode()` function yourself, encoding would be doubled.

Unconfirmed order

The recommended practice is to place only orders allowed to be processed and shipped. And, for example, to withhold an order waiting for a payment from the customer. In that case, it is possible to ignore (omit) the `suspended` parameter or set it to `false`.

In special cases however it may be appropriate to place the order in advance, even if it is not to be processed right away. If the `suspended` parameter is set to `true`, the order will be kept in SUSPENDED status and will not be processed until client approves it in the web application.

Invoice data

Under special circumstances and based on previous agreement it is possible to include customer invoice data together with the order. Example:

```
<DeliveryOrder>
  ... order data as usual ...
  </documents>
  <invoice>
    <number>FA20108516</number>
    <varSymbol>20108516</varSymbol>
    <currency>CZK</currency>
    <dateIssued>2010-12-23</dateIssued>
    <dateTax>2010-12-23</dateTax>
    <dateDue>2011-01-23</dateDue>
    <totalSum>2830.00</totalSum>
    <paymentType>delivery</paymentType>
    <address>
      <name>František Omáčka</name>
      <company>FO Servis, s.r.o.</company>
      <street>Osadní 14</street>
      <city>Praha 7</city>
      <postcode>170 00</postcode>
      <country>CZ</country>
      <phone>+420777666555</phone>
      <email>omacka@foservis.cz</email>
      <ico>12345678</ico>
      <dic>CZ12345678</dic>
```

```
      <icDph></icDph>
    </address>
    <invoiceItems>
      <invoiceItem>
        <code>1234597</code>
        <name>BDR-202</name>
        <quantity>1</quantity>
        <unitPrice>1541.67</unitPrice>
        <price>1541.67</price>
        <rateVAT>20.0</rateVAT>
        <priceIncVAT>1850.00</priceIncVAT>
      </invoiceItem>
      <invoiceItem>
      ... další položky ...
      </invoiceItem>
    </invoiceItems>
  </invoice>
</DeliveryOrder>
```

number	text	Invoice number
varSymbol	text	Variable symbol
currency	text	Invoice currency (three-letter code, ISO 4217, e.g. CZK, EUR)
dateIssued	date	Date of issue of the invoice
dateTax	date	Date of supply
dateDue	date	Due date
totalSum	decimal	Total sum
paymentType	text	Payment type (see list below)
ico	text	person identification number (optional)
dic	text	tax identification number (optional)
icDph	text	IČ DPH (VAT number in Slovakia)
code	text	item code (optional)
name	text	name of the item
quantity	decimal	quantity
unitPrice	decimal	unit price without VAT
price	decimal	total item price without VAT (tax base)
rateVAT	decimal	vat rate (e.g. 21)
priceIncVAT	decimal	total item price including VAT

Payment types

draft	Bank transfer
cash	Cash payment
delivery	Cash on delivery

creditcard	Card payment

DeliveryOrderStatusGet

Get current status of a delivery order.

Package tracking number is included in the response if it is known. With PPL, Geis and most other carriers the number is available at the time the package is handed over to the carrier. With some carriers, the number can be added later or may not be known altogether.

In special cases, the package can use a different shipping service than the one specified at creating the order. For that reason, carrier and shipping service is included in the response as well.

Request format

```
<DeliveryOrderStatusGet>
  <projectId>klient.cz</projectId>
  <orderId>1624</orderId>
</DeliveryOrderStatusGet>
```

projectId	text	e-shop/project identification
orderId	text	ID of the checked order

Response format

```
<DeliveryOrderStatusGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <deliveryOrderStatus>
    <projectId>klient.cz</projectId>
    <orderId>1624</orderId>
    <changed>2008-10-10T12:00:00+02:00</changed>
    <status>SHIPPED</status>
    <carrierId>GEIS</carrierId>
    <service>Firemní adresa</service>
    <branchId>321</branchId>
    <carrierWeb>www.zasilkovna.cz</carrierWeb>
    <trackingNo>05093958882</trackingNo>
    <trackingNo>05093958883</trackingNo>
    <sysOrderId>135153</sysOrderId>
    <itemCount>4</itemCount>
    <unitCount>18</unitCount>
    <destinationCountry>CZ</destinationCountry>
    <cod currency="CZK">2830.0</cod>
    <dateCreated>2008-09-27</dateCreated>
    <packageSet>
      <dateProcessed>2008-10-01</dateProcessed>
      <processingFee currency="CZK">32.00</processingFee>
      <shippingFee currency="CZK">65.00</shippingFee>
      <paidByCard>false</paidByCard>
      <package>
        <sysPackageId>654663</sysPackageId>
        <trackingNo>05093958882</trackingNo>
        <weight>2.54</weight>
        <length>84</length>
        <width>46</width>
        <height>20</height>
        <packaging>karton</packaging>
      </package>
      <package>
        <sysPackageId>654664</sysPackageId>
        <trackingNo>05093958882</trackingNo>
```

```
      </package>
      <serialNumberOut>
        <productId>231344</productId>
        <serialNumber>93957246381</serialNumber>
      </serialNumberOut>
      <serialNumberOut>
        <productId>231344</productId>
        <serialNumber>93957244816</serialNumber>
      </serialNumberOut>
    </packageSet>
  </deliveryOrderStatus>
</DeliveryOrderStatusGetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
orderId	text	order ID
changed	dateTime	Date when the change ocurred
status	text	Status code (see delivery order statuses)
carrierId	text	Carrier used for shipping
service	text	Shipping service used
branchId	text	Branch code for pick-up (if set)
carrierWeb	text	Carrier website. Omitted when tracking number is not known.
trackingNo	text	Package tracking number. If the shipments consists of multiple packages and each of them has its own number, all numbers will be listed. Element is omitted when tracking number is not known.
sysOrderId	int	Shipmall system order ID
itemCount	int	Number of items in the order
unitCount	decimal	Number of units (pieces total) in the order
destinationCountry	text	Destination country
amount	decimal	COD amount
currency	text	COD currency
dateCreated	date	Date when order was created
dateProcessed	date	Date when order was processed (determines the month of billing)
processingFee	decimal	Order processing fee (completing, packaging), if already established
shippingFee	decimal	Shipping fee, if already established
paidByCard	boolean	COD was/was not paid by card
sysPackageId	int	Shipmall system package ID
weight	decimal	Package weight in kg (if known)
length	decimal	Package length in cm (if known)
width	decimal	Package width in cm (if known)
height	decimal	Package height in cm (if known)
packaging	text	Package type
serialNumberOut		list of serial numbers included in the package
productId	text	- product
serialNumber	text	- serial number
problem	text	Description of an error that is currently prevening the order from being shipped (e.g. an unavailable pickup branch, a newly detected problem in the

		address, etc.

DeliveryOrderStatusGetNews

Get statuses of changed delivery orders.
Method supports results pagination.

Request format

```
<DeliveryOrderStatusGetNews>
  <projectId>klient.cz</projectId>
  <afterDateTime>2008-10-01T05:00:00+02:00</afterDateTime>
  <beforeDateTime>2008-10-01T05:30:00+02:00</beforeDateTime>
</DeliveryOrderStatusGetNews>
```

projectId	text	e-shop/project identification. Optional (when not specified, all projects are included)
afterDateTime	dateTime	Return only changes after specified moment. Usual value will be the time of previous method call shifted by a few minutes back to be sure.
beforeDateTime	dateTime	Optional. Return changes before specified moment. Used on rare occasions when the intention is to check a time window in the past.

Response format

```
<DeliveryOrderStatusGetNewsResponse>
  <result>
    <code>0</code>
    <description>OK</description>
  </result>
  <deliveryOrderStatus>
    <projectId>novypocitac.cz</projectId>
    <orderId>1628</orderId>
    <changed>2008-10-01T09:00:00+02:00</changed>
    <status>SHIPPED</status>
    <carrierId>PPL</carrierId>
    <service>PPL Parcel CZ Private</service>
    <carrierWeb>http://www.ppl.cz</carrierWeb>
    <trackingNo>1651515</trackingNo>
    <trackingNo>1651516</trackingNo>
    <sysOrderId>135153</sysOrderId>
    <itemCount>1</itemCount>
    <unitCount>3</unitCount>
    <destinationCountry>CZ</destinationCountry>
    <cod currency="CZK">2830.0</cod>
    <dateCreated>2008-09-27</dateCreated>
    <packageSet>
      <dateProcessed>2008-10-01</dateProcessed>
      <processingFee currency="CZK">32.00</processingFee>
      <shippingFee currency="CZK">65.00</shippingFee>
      <paidByCard>false</paidByCard>
      <package>
        <sysPackageId>654663</sysPackageId>
        <trackingNo>1651515</trackingNo>
        <weight>2.54</weight>
        <length>84</length>
        <width>46</width>
        <height>20</height>
        <packaging>EUR paleta</packaging>
      </package>
      <package>
        <sysPackageId>654664</sysPackageId>
        <trackingNo>1651516</trackingNo>
      </package>
```

```
      <serialNumberOut>
        <productId>231344</productId>
        <serialNumber>93957246381</serialNumber>
      </serialNumberOut>
      <serialNumberOut>
        <productId>231344</productId>
        <serialNumber>93957244816</serialNumber>
      </serialNumberOut>
    </packageSet>
  </deliveryOrderStatus>
  <deliveryOrderStatus>
    <projectId>novypocitac.cz</projectId>
    <orderId>1629</orderId>
    <changed>2008-10-01T10:00:00+02:00</changed>
    <status>CANCELLED</status>
    <carrierId>ZASILKOVNA</carrierId>
    <service>Osobní odběr</service>
    <branchId>351</branchId>
    <sysOrderId>135214</sysOrderId>
    <itemCount>2</itemCount>
    <unitCount>16</unitCount>
    <destinationCountry>CZ</destinationCountry>
    <cod currency="CZK">2830.0</cod>
    <dateCreated>2008-09-18</dateCreated>
  </deliveryOrderStatus>
</DeliveryOrderStatusGetNewsResponse>
```

Meaning of all data entries is the same as in the response to DeliveryOrderStatusGet method.

Parameter `afterDateTime` applies not only to order status changes, but also to the moment when tracking number was set. In case it is set additionally (ie. not simultaneously with order status change), current status is reported again together with the tracking number. Timestamp of the status in `changed` element stays unchanged.

An order may pass multiple states during the specified period. When that happens, all states are listed from the oldest to the newest (meaning, a single order can be listed multiple times in the response).

Because orders can be created manually using the web application in addition to calling DeliveryOrder method, the response may include orders not known to the querying system. It is a good idea to anticipate this situation in your implementation.

DeliveryOrderHistoryGet

Get complete history of statuses of a delivery order.

Request format

```
<DeliveryOrderHistoryGet>
  <projectId>klient.cz</projectId>
  <orderId>1624</orderId>
</DeliveryOrderHistoryGet>
```

projectId	text	e-shop/project identification.
orderId	text	ID of the checked order

Response format

```
<DeliveryOrderHistoryGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <deliveryOrderStatus>
    <projectId>klient.cz</projectId>
    <orderId>1624</orderId>
    <changed>2008-10-10T09:15:46+02:00</changed>
    <status>NEW</status>
  </deliveryOrderStatus>
  <deliveryOrderStatus>
    <projectId>klient.cz</projectId>
    <orderId>1624</orderId>
    <changed>2008-10-10T12:00:00+02:00</changed>
    <status>CANCELLED</status>
  </deliveryOrderStatus>
</DeliveryOrderHistoryGetResponse>
```

Meaning of all data entries is the same as in the response to DeliveryOrderStatusGet method.

DeliveryOrderStatusSet

Change delivery order status.

It can be used to cancel an order in NEW or COMPLETABLE status, before its processing has begun. To cancel an order later please contact our staff.

Request format

```
<DeliveryOrderStatusSet>
  <projectId>klient.cz</projectId>
  <orderId>1624</orderId>
  <status>CANCELLED</status>
</DeliveryOrderStatusSet>
```

projectId	text	e-shop/project identification.
orderId	text	order ID
status	text	Desired order status (currently only CANCELLED is accepted)

Response format

```
<DeliveryOrderStatusSetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <deliveryOrderStatus>
    <projectId>klient.cz</projectId>
    <orderId>1624</orderId>
    <changed>2008-10-11T14:32:00+02:00</changed>
    <status>CANCELLED</status>
    <carrierId>PPL</carrierId>
    <service>PPL Parcel CZ Business</service>
    <sysOrderId>135153</sysOrderId>
    <itemCount>2</itemCount>
    <unitCount>3</unitCount>
    <destinationCountry>CZ</destinationCountry>
    <dateCreated>2008-10-04</dateCreated>
  </deliveryOrderStatus>
</DeliveryOrderStatusSetResponse>
```

Meaning of all data entries is the same as in the response to DeliveryOrderStatusGet method. The returned status shows situation after the change.

DeliveryOrderDocumentAdd

Adding a document to the order.

Request format

```
<DeliveryOrderDocumentAdd>
  <projectId>klient.cz</projectId>
  <orderId>1624</orderId>
  <document>
    <title>Faktura</title>
    <type>application/pdf</type>
    <purpose>INSERT</purpose>
    <content md5checksum="48dbb8ee55311848967f56ead404e6ee">
      cGVsYS
      Bob3Bh
    </content>
  </document>
</DeliveryOrderDocumentAdd>
```

The structure of the `document` parameter is identical to that of the documents supplied when creating an order. Further details are given in the Accompanying documents section.

projectId	text	e-shop/project identification.
orderId	text	order ID
title	text	Document title (e.g. „Invoice F080624“, „Delivery note“ etc.)
type	text	File MIME type, currently the only expected option is „application/pdf“
purpose	text	document purpose (INSERT, SHIPPING_LABEL, PRODUCT_LABEL). Optional.
content	text	File contents, base64-encoded
md5checksum	text	File checksum using MD5 algorithm, 32-character hexadecimal form

Response format

```
<DeliveryOrderDocumentAddResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
</DeliveryOrderDocumentAddResponse>
```

StorageOrder

Send in a new storage order

Request format

```
<StorageOrder>
  <orderId>159753</orderId>
  <supplier>AAA Computer</supplier>
  <deliveryDate>2010-08-03</deliveryDate>
  <note>Přivezu osobně</note>
  <orderItems>
    <orderItem>
      <productSheet>
        <productId>455435</productId>
        <name>UTP kabel cat.5 - 3m</name>
        <description>Nestíněný síťový kabel</description>
        <vendor>OEM</vendor>
        <quantityUnit>PC</quantityUnit>
      </productSheet>
      <quantity>25</quantity>
    </orderItem>
    <orderItem>
      <productId>324234</productId>
      <quantity>1</quantity>
      <unitValue>25.00</unitValue>
    </orderItem>
  </orderItems>
</StorageOrder>
```

orderId	text	order ID
supplier	text	Name of supplier – sender of the goods
deliveryDate	date	Estimated date of arrival. Optional
note	text	Note. Optional
unitValue	decimal	Unit purchase price. Ignored unless recording of purchase prices is enabled (ask your sales representative).

Stucture of the order items list is virtually identical as with delivery orders (see DeliveryOrder). The only difference is the possibility to include purchase prices.

Response format

As a confirmation that order was created successfully, current order status is included in the response.

```
<StorageOrderResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <storageOrderStatus>
    <orderId>159753</orderId>
    <changed>2008-10-08T12:00:00+02:00</changed>
    <status>NEW</status>
  </storageOrderStatus>
</StorageOrderResponse>
```

code	int	Return code (see return codes)

description	text	Operation result description
orderId	text	order ID
changed	dateTime	Date when the change ocurred
status	text	Status code (see storage order statuses)

Order items

If it helps your implementation in any way, it is possible to include one product more than once in an order (as multiple order items).

Purchase prices

Providing purchase price is optional, `unitValue` field may be omitted. When the client has not recording of purchase prices enabled (ask your sales representative), field will be ignored. It is possible to include one product multiple times with different prices within an order.

StorageOrderStatusGet

Get current status of a storage order.

Request format

```
<StorageOrderStatusGet>
  <orderId>1624</orderId>
</StorageOrderStatusGet>
```

orderId	text	ID of the checked order

Response format

```
<StorageOrderStatusGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <storageOrderStatus>
    <orderId>1624</orderId>
    <changed>2008-10-08T12:00:00+02:00</changed>
    <status>COMPLETE</status>
    <lastReceiptDate>2008-10-08</lastReceiptDate>
    <serialNumberIn>
      <productId>231344</productId>
      <serialNumber>93957246381</serialNumber>
    </serialNumberIn>
    <serialNumberIn>
      <productId>231344</productId>
      <serialNumber>93957244816</serialNumber>
    </serialNumberIn>
  </storageOrderStatus>
</StorageOrderStatusGetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
orderId	text	order ID
changed	dateTime	Date when the change ocurred
status	text	Status code (see storage order statuses)
lastReceiptDate	date	Date of last receipt of goods into the warehouse.
extrasReceived	boolean	The flag may appear for orders with COMPLETE status, when some goods was accepted in a greater quantity than announced.
serialNumberIn		list of serial numbers noticed on delivered goods
productId	text	- produkt
serialNumber	text	- sériové číslo

StorageOrderStatusGetNews

Get statuses of changed storage orders.
Method supports results pagination.

Request format

```
<StorageOrderStatusGetNews>
  <afterDateTime>2008-10-01T05:00:00+02:00</afterDateTime>
  <beforeDateTime>2008-10-01T06:00:00+02:00</beforeDateTime>
</StorageOrderStatusGetNews>
```

afterDateTime	dateTime	Return only changes after specified moment
beforeDateTime	dateTime	Optional. Return changes before specified moment. Used on rare occasions when the intention is to check a time window in the past.

Response format

```
<StorageOrderStatusGetNewsResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <storageOrderStatus>
    <orderId>129753</orderId>
    <changed>2008-10-01T13:06:00+02:00</changed>
    <status>COMPLETE</status>
  </storageOrderStatus>
</StorageOrderStatusGetNewsResponse>
```

Meaning of all data entries is the same as in the response to StorageOrderStatusGet call.

An order may pass multiple states during the specified period. When that happens, all states are listed from the oldest to the newest (meaning, a single order can be listed multiple times in the response).

Because orders can be created manually using the web application in addition to calling StorageOrder method, the response may include orders not known to the querying system. It is a good idea to anticipate this situation in your implementation.

StorageOrderHistoryGet

Get complete history of statuses of a storage order.

Request format

```
<StorageOrderHistoryGet>
  <orderId>1624</orderId>
</StorageOrderHistoryGet>
```

orderId	text	ID of the checked order

Response format

```
<StorageOrderHistoryGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <storageOrderStatus>
    <orderId>1624</orderId>
    <changed>2008-10-08T12:00:00+02:00</changed>
    <status>NEW</status>
  </storageOrderStatus>
  <storageOrderStatus>
    <orderId>1624</orderId>
    <changed>2008-10-09T10:35:41+02:00</changed>
    <status>COMPLETE</status>
  </storageOrderStatus>
</StorageOrderHistoryGetResponse>
```

Meaning of all data entries is the same as in the response to StorageOrderStatusGet call.

StorageOrderStatusSet

Change storage order status.

It can be used to cancel an order in NEW status.

Request format

```
<StorageOrderStatusSet>
  <orderId>159753</orderId>
  <status>CANCELLED</status>
</StorageOrderStatusSet>
```

orderId	text	order ID
status	text	Desired order status (currently only CANCELLED is accepted)

Response format

```
<StorageOrderStatusSetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <storageOrderStatus>
    <orderId>159753</orderId>
    <changed>2008-10-08T12:00:00+02:00</changed>
    <status>CANCELLED</status>
  </storageOrderStatus>
</StorageOrderStatusSetResponse>
```

Meaning of all data entries is the same as in the response to StorageOrderStatusGet call. The returned status shows situation after the change.

StorageOrderGet

Get storage order data with the addition of a list of received goods.

Request format

```
<StorageOrderGet>
  <orderId>159753</orderId>
</StorageOrderGet>
```

orderId	text	order ID

Response format

```
<StorageOrderGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <orderId>159753</orderId>
  <supplier>AAA Computer</supplier>
  <deliveryDate>2010-08-03</deliveryDate>
  <note>Přivezu osobně</note>
  <orderItems>
    <orderItem>
      <productId>455435</productId>
      <quantity>25</quantity>
    </orderItem>
    <orderItem>
      <productId>324234</productId>
      <quantity>1</quantity>
    </orderItem>
  </orderItems>
  <receivedItems>
    <receivedItem>
      <date>2010-08-03</date>
      <productId>455435</productId>
      <quantity>20</quantity>
    </receivedItem>
    <receivedItem>
      <date>2010-08-04</date>
      <productId>324234</productId>
      <quantity>1</quantity>
    </receivedItem>
  </receivedItems>
</StorageOrderGetResponse>
```

The response contains a copy of the data entered at order creation, followed by a list of the received goods. Since the reception process can span over multiple days (for example, when the goods are split into into several deliveries), the items are differentiated by date. Which in effect means that one product may be listed more than once.

orderId	text	order ID
supplier	text	Name of supplier specified when placing the order.
deliveryDate	date	Estimated date of arrival specified when placing the order
note	text	Note specified when placing the order
productId	text	product ID
quantity	decimal	product quantity
date	date	date when the item was received

StockInventoryGet

Get stock counts – either for selected products only or for all items together.
Method supports results pagination.

Request format

Alternative 1 – get complete stock status

```
<StockInventoryGet/>
```

Alternative 2 – get information about one or more selected products

```
<StockInventoryGet>
  <filter>
    <product productId="13515" />
    <product productId="13517" />
  </filter>
</StockInventoryGet>
```

Alternative 3 – get status of a distinct product variant (see products with variants). If no exact variants are specified in filter, response will contain a separate record for every variant currently in stock.

```
<StockInventoryGet>
  <filter>
    <product productId="13517">
      <variant attribute="barva" value="černá" />
      <variant attribute="připojení" value="SATA" />
    </product>
  </filter>
</StockInventoryGet>
```

Alternative 4 – get status for one of predefined subsets (e.g. only products in stock)

```
<StockInventoryGet>
  <subset>IN_STOCK</subset>
</StockInventoryGet>
```

Additional parameter to return sums by products while ignoring individual product variants. Suitable e.g. when the client has no interest in detailed information on counts of goods with various expiration dates.

```
<StockInventoryGet>
  <disregardVariants>true</disregardVariants>
</StockInventoryGet>
```

Listed filter options may be combined as needed.

filter		Optional. List of products to check
productId	text	Product ID
attribute	text	Variant property name
value	text	Variant property value
subset	text	Optional. Narrow results down to one of following subsets: IN_STOCK_OR_INCOMING – products in stock or on the way to warehouse IN_STOCK – only products in stock RUNNING_SHORT – products with unsufficient quantity EXPIRED – products with expiration date passed NEAR_EXPIRATION – products with expiration date within next 60 days ALL – all products defined in catalog, even if it has never been in stock

		If the parameter is not set, the response will contain products that are or recently were in stock or are expected. For other products, all the quantities (`stored`, `expected` etc.) would be zero.
disregardVariants	bool	Optional . `false` (default) – result records will include information on individual variants `true` – results will be summed up by products, disregarding variants

Response format

```
<StockInventoryGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <stockInventory>
    <product productId="13515">
      <stored>10</stored>
      <expected>30</expected>
      <allocated>0</allocated>
      <aside>0</aside>
      <demanded>0</demanded>
    </product>
    <product productId="13517">
      <variant attribute="barva" value="černá" />
      <variant attribute="připojení" value="SATA" />
      <stored>2</stored>
      <expected>0</expected>
      <allocated>2</allocated>
      <aside>0</aside>
      <demanded>0</demanded>
    </product>
  </stockInventory>
</StockInventoryGetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
productId	text	Product ID
attribute	text	Variant property name
value	text	Variant property value
stored	int	Total quantity in stock
expected	int	quantity on the way from supplier (announced products that have not yet arrived)
allocated	int	quantity reserved for current delivery orders
available	int	quantity available for new orders
aside	int	quantity not ready for shipping (e.g. damaged items)
demanded	int	total quantity needed to process all current delivery orders
value	decimal	Total value of the product in stock. The field is included only when client has recording of purchase prices enabled.

Records in the result are ordered by product ID.

StockInventoryGetNews

Get changes on stock after the specified moment.
Method supports results pagination.

Request format

```
<StockInventoryGetNews>
  <afterDateTime>2008-10-08T12:00:00+02:00</afterDateTime>
</StockInventoryGetNews>
```

afterDateTime	datetime	Only products with stock changes after this moment.
disregardVariants	bool	Optional. `false` (default) – result records will include information on individual variants `true` – results will be summed up by products, disregarding variants

Response format

Response format is identical to StockInventoryGet method.

InventoryChangesGet

Get a list of stock accounting documents (receipts, expeditions, returns) for a given period.

```
<InventoryChangesGet>
  <afterDateTime>2011-02-3T08:00:00+02:00</afterDateTime>
  <restrictType>receipt</restrictType>
</InventoryChangesGet>
```

afterDateTime	dateTime	Only documents created after specified moment will be returned
beforeDateTime	dateTime	Optional. Only documents created before specified moment will be returned. Used on rare occasions when the intention is to check a time window in the past.
restrictType	text	Optional. When set, output will be restricted to items of specified type; accepted values: `receipt`, `expedition`, `return`

Document types are:

- receipt – goods received from supplier
- expedition – goods sent to customer
- return – undelivered goods returned back

```
<InventoryChangesGetResponse>
  <result>
    <code>0</code>
    <description>OK</description>
  </result>
  <receipt>
    <receiptId>201102111</receiptId>
    <date>2011-02-03</date>
    <orderId>A13313</orderId>
    <supplier>Dunlop CZ</supplier>
    <item>
      <productId>1234597</productId>
      <quantity>2</quantity>
    </item>
    <item>
      <productId>1234588</productId>
      <quantity>2</quantity>
    </item>
  </receipt>
  <expedition>
    <expeditionId>2011021561</expeditionId>
    <date>2011-02-04</date>
    <projectId>klient.cz</projectId>
    <orderId>79879897987</orderId>
    <item>
      <productId>1234597</productId>
      <quantity>2</quantity>
    </item>
  </expedition>
  <expedition>
    <!-- etc... -->
  </expedition>
  <return>
    <returnId>201102111</returnId>
    <date>2011-02-07</date>
    <expeditionId>2011021561</expeditionId>
    <projectId>klient.cz</projectId>
    <orderId>79879897987</orderId>
    <reporter>příjemce</reporter>
```

```
    <reason>vrácení do 14 dnů</reason>
    <description>vrací kvůli nevyhovujícímu konektoru</description>
    <contact>tel. 724125125</contact>
    <bankAccount>651651651/0100</bankAccount>
    <withLetter>true</withLetter>
    <item>
      <productId>1234597</productId>
      <quantity>2</quantity>
      <condition>UNPACKED</condition>
      <note>bez původního obalu</note>
    </item>
  </return>
</InventoryChangesGetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
receiptId	text	Receipt number
expeditionId	text	Expedition number
returnId	text	Return number
date	date	Date when document was created
projectId	text	Project ID of the respective delivery order
orderId	text	In a receipt: storage order ID Otherwise: delivery order ID
supplier	text	supplier name
reporter	text	sender of returned goods (carrier, customer)
reason	text	return type (rejected, returning within 14 days etc.)
description	text	other details of returned package
contact	text	additional contact information given by the customer
bankAccount	text	bank accound for money refund, if told by the customer
withLetter	boolean	whether or not a covering letter was attached to the returned goods
productId	text	product ID
quantity	int	product quantity
condition	text	condition of returned goods
note	text	additional information to the status of the returned goods

`condition` values

OK	mint condition
UNPACKED	product was unpacked
DAMAGED	product is damaged
PARTIAL	part of package content is missing
UNSTATED	condition not entered (for older returns)

PaymentListGet

Get list of payments sent to client's bank account.
Method supports results pagination.

Request format

```
<PaymentListGet>
  <afterDate>2010-08-03</afterDate>
</PaymentListGet>
```

afterDate	date	Return only payments entered on this date or later

Response format

```
<PaymentListGetResponse>
  <result>
    <code>0</code>
    <description>OK</description>
  </result>
  <payment>
    <account>85-123456/0300</account>
    <paymentId>1300</paymentId>
    <date>2010-08-11</date>
    <amount>1851.00</amount>
    <currency>CZK</currency>
  </payment>
  <payment>
    <account>85-123456/0300</account>
    <paymentId>1301</paymentId>
    <date>2010-08-17</date>
    <amount>953.00</amount>
    <currency>CZK</currency>
  </payment>
</PaymentListGetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
account	text	Bank account number to which the payment was sent
paymentId	int	Variable symbol
date	date	Date when the payment was submitted
amount	decimal	Amount remitted
currency	text	Currency code

PaymentDetailGet

Get a list of orders covered by a certain payment to client's account.

Request format

```
<PaymentDetailGet>
  <account>85-123456/0300</account>
  <paymentId>1300</paymentId>
</PaymentDetailGet>
```

account	text	Client bank account number where the payment was sent
PaymentId	int	Payment variable symbol

Response format

```
<PaymentDetailGetResponse>
  <result>
    <code>0</code>
    <description>OK</description>
  </result>
  <paidOrder>
    <projectId>klient.cz</projectId>
    <orderId>123456</orderId>
    <amount>133.00</amount>
    <currency>CZK</currency>
  </paidOrder>
  <paidOrder>
    <projectId>klient.cz</projectId>
    <orderId>123457</orderId>
    <amount>105.00</amount>
    <currency>CZK</currency>
  </paidOrder>
</PaymentDetailGetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
projectId	text	e-shop/project identification
orderId	text	order ID
amount	decimal	amount that belongs to the order
currency	text	currency code

ReceivedPaymentsGet

Get a list of COD amounts received by Shipmall from carriers.

Request format

```
<ReceivedPaymentsGet>
  <afterDate>2014-07-25</afterDate>
</ReceivedPaymentsGet>
```

afterDate	date	Return only amounts received on this date or later

Response format

```
<ReceivedPaymentsResponse>
  <result>
    <code>0</code>
    <description>OK</description>
  </result>
  <payment>
    <projectId>klient.cz</projectId>
    <orderId>123458</orderId>
    <date>2014-07-29</date>
    <amount>133.00</amount>
    <currency>CZK</currency>
  </payment>
  <payment>
    <projectId>klient.cz</projectId>
    <orderId>123459</orderId>
    <date>2014-07-30</date>
    <amount>105.00</amount>
    <currency>CZK</currency>
  </payment>
</ReceivedPaymentsResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
projectId	text	e-shop/project identification
orderId	text	order ID
date	date	Date when payment was received on Shipmall account
amount	decimal	Amount received for this order
currency	text	Currency code

ProductUpdate

Update product information in the catalog. Or create product if it does not already exist.

Parameter structure is identical to `productSheet` element used in DeliveryOrder a StorageOrder methods.

Request format

```
<ProductUpdate>
  <productId>231344</productId>
  <barcode>5-901234-123457</barcode>
  <productNumber>A00DD3561</productNumber>
  <name>BDR-202</name>
  <description>Blu-ray/DVD/CD zapisovací mechanika</description>
  <vendor>Pioneer</vendor>
  <attributeSet>
    <attribute>barva</attribute>
    <attribute>připojení</attribute>
  </attributeSet>
  <quantityUnit>PC</quantityUnit>
  <enabled>true</enabled>
  <expirable>false</expirable>
  <minimalSupply>50</minimalSupply>
  <valuable>false</valuable>
  <labelIncoming>false</labelIncoming>
  <labelOutgoing>false</labelOutgoing>
</ProductUpdate>
```

productId	text	required	product ID. An identifier visible to e-shop operator is preferred, ideally also matching the identifier used by product supplier – helps recognizing the product in the warehouse and communication with the client.
barcode	text	optional	barcode
productNumber	text	optional	producer product number
name	text	required	product name
description	text	optional	short description, provide only when considered helpful in product identification (not a multi-paragraph description)
vendor	text	optional	producer
attribute	text	optional	name of the property that defines a variant (see products with variants)
quantityUnit	text	required	quantity unit - PC = piece, KG, METER, SQUARE_METER, PALLET
enabled	boolean	optional	the product is currently on sale. Used for filtering in selected reports.
expirable	boolean	optional	the product has expiration date and it should be worked with*
food	boolean	optional	to product is food requiring adequate storage
lotTracking	boolean	optional	the product has lot numbers and they should be tracked*
minimalSupply	decimal	optional	Minimum stock. Used for filtering in selected reports.
valuable	boolean	optional	Valuable goods to be stored in a special section with restricted access*
labelIncoming	boolean	optional	Product is to be labeled upon receipt at the warehouse*
labelOutgoing	boolean	optional	Product is to be labeled before shipping to the customer*

*settings marked with an asterisk will be taken into account only after approval with our sales representative

When any optional parameter is omitted, its current value in catalog remains valid.

Response format

```
<ProductUpdateResponse>
```

```
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
</ProductUpdateResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description

ProductLabelSet

Set a description to be printed and attached on a product label (on demand service). A single product can have multiple descriptions in different languages.

Request format

```
<ProductLabelSet>
  <productId>231344</productId>
  <language>cs</language>
  <content>&lt;b&gt;Složení: &lt;/b&gt; voda, aroma, přírodní barvivo
    &lt;br&gt;Vyrobeno v ČR</content>
</ProductLabelSet>
```

productId	text	Product ID
language	text	Descritption language code (ISO 639-1), e.g. cs, sk, en
content	text	Description text. Basic formatting using HTML tags `<b>`, `<i>` a ` ` is allowed

Response format

```
<ProductLabelSetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
</ProductLabelSetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description

ProductLabelDelete

Delete a description that was set eariler using ProductLabelSet.

Request format

```
<ProductLabelDelete>
  <productId>231344</productId>
  <language>cs</language>
</ProductLabelDelete>
```

productId	text	Product ID
language	text	Description language code (ISO 639-1), e.g. cs, sk, en

Response format

```
<ProductLabelDeleteResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
</ProductLabelDeleteResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description

BranchListGet

Get a list of branches for pick-up. Applicable with Uloženka and Zásilkovna, PPL ParcelShops and Czech post's service *Balík Do balíkovny*. Lists are updated daily at about 5:00am, it is therefore pointless to call the method more than once per day for any carrier/service.
Method supports results pagination.

Request format

```
<BranchListGet>
  <carrierId>ZASILKOVNA</carrierId>
  <service>Osobní odběr</service>
</BranchListGet>
```

carrierId	text	carrier ID – ZASILKOVNA, PPL, CPOST, WEDO-POINT etc.
service	text	service – same as in DeliveryOrder

Response format

```
<BranchListGetResponse>
  <result>
    <code>1000</code>
    <description>OK</description>
  </result>
  <branch>
    <branchId>10</branchId>
    <name>Hradec Králové, Náměstí 5. Května</name>
    <street>Náměstí 5. Května 285/6</street>
    <city>Hradec Králové</city>
    <country>CZ</country>
    <url>https://www.zasilkovna.cz/point/hradec-kralove-namesti-5-kvetna</url>
  </branch>
  ...
  <branch>
    <branchId>100</branchId>
    <name>Žatec, Tyršova</name>
    <street>Tyršova 2896</street>
    <city>Žatec</city>
    <country>CZ</country>
    <url>https://www.zasilkovna.cz/pobocky/zatec-tyrsova</url>
  </branch>
</BranchListGetResponse>
```

code	int	Return code (see return codes)
description	text	Operation result description
branchId	Text	Branch code for use in DeliveryOrder calls
name	Text	Branch name to be displayed in e-shop
street	Text	Street and street number
city	Text	City
country	Text	Country code (e.g. CZ, SK)
url	Text	Website with more branch details

Return codes

Return codes that may appear in response element „code“.

1000 – OK

1001 – unknown project ID (projectId)

1002 – unknown product

1003 – invalid product variant entry

1004 – unknown order number

1005 – duplicate order number

1006 – status change not allowed

1011 – unknown quantity unit

1012 – document contents don‘t match supplied md5 sum

1013 – unknown shipping service

1014 – invalid post code

1015 – bad document structure

1017 – unknown currency

1018 – carrier not available in destination country

1019 – shipping service not available for supplied address

1020 – mandatory contact information missing (differs for various shipping services)

1021 – bank account not found

1022 – bank statement not found

1023 – invalid product ID format

1024 – incorrect shipping option declaration

1025 – unknown country code

1026 – unknown branch code

1027 – branch code required

1028 – carrier doesn’t collect COD in recipient country

1029 – COD currency does not match recipient country

1030 – invalid COD amount

1034 – variant attribute specified more than once

1035 – recipient phone missing, required for selected shipping service

1036 – invalid recipient phone number

1037 – recipient email address is missing (required for selected shipping service)

1038 – a part of recipient address is missing (eg. house number)

1039 – unsupported language code

1040 – invalid email address

1046 – COD amount exceeds allowed limit

1047 – mobile phone number required

1048 – unexpected document purpose

1049 – changes to order are not accepted any more (processing started already)

1099 – not implemented