zanox.

ERP WEB SERVICES

DEVELOPER MANUAL

zanox.

TABLE OF CONTENTS

zanox.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 zanox ERP Web Services

The zanox ERP Web Services were developed by zanox for the automated exchange of tracking information data. The ERP Web Services can either be fully integrated into the advertiser system or, alternatively, may be used with the help of the zanox ERP Client.

Output format and data volume are defined in a programme-specific ERP profile. This profile will be drawn up in collaboration between the advertiser and zanox.

This manual provides an explanation of the structure and operation of the application programming interface of zanox ERP Web Services.

ERP stands for "Enterprise Resource Planning".

Further information on the ERP Client can be found in the ERP Client manual at http://www.zanox.com/en/advertiser/help/start-package.html.

1.2 Conventions for the description of XML data

The XML-format data described in this manual will be illustrated with the aid of tables. The XML description tables contain the following columns:

- Object
- Data type
- Characters
- Description

Object column

The Object column represents the structure of the XML data. Elements are noted in angle brackets < >; attributes are without brackets. For an improved overview, elements which possess sub-elements or attributes are shown in grey-shaded lines.

Data type column

The data type of the element/attribute is noted in the Data type column in accordance with XML scheme definitions.

Characters column

The Characters column indicates the maximum length of an object of the data type "normalizedstring" or "String".

Description column

The Description column contains a brief description of the data, which are contained in the respective element or attribute.

2. GENERAL INFORMATION

2.1 Services

zanox ERP Web Services provide all information and functionalities required for the processing of tracking information on leads, sales and shopping baskets, which has been captured via the zanox system.

The zanox ERP Web Services encompass the following categories:

Category	Service module	Description
User management	userservice.asmx	Log-in and log-out
Export	exportservice.asmx	Export of tracking information from the zanox system
Import	importservice.asmx	Import of tracking information into the zanox system
General	commonservice.asmx	General functions of zanox Web Services

A detailed description of the function of each category can be found in the section "Description of the functions."

zanox ERP Web Services are available via the following URLs by means of http or https (the latter SSL-encrypted):

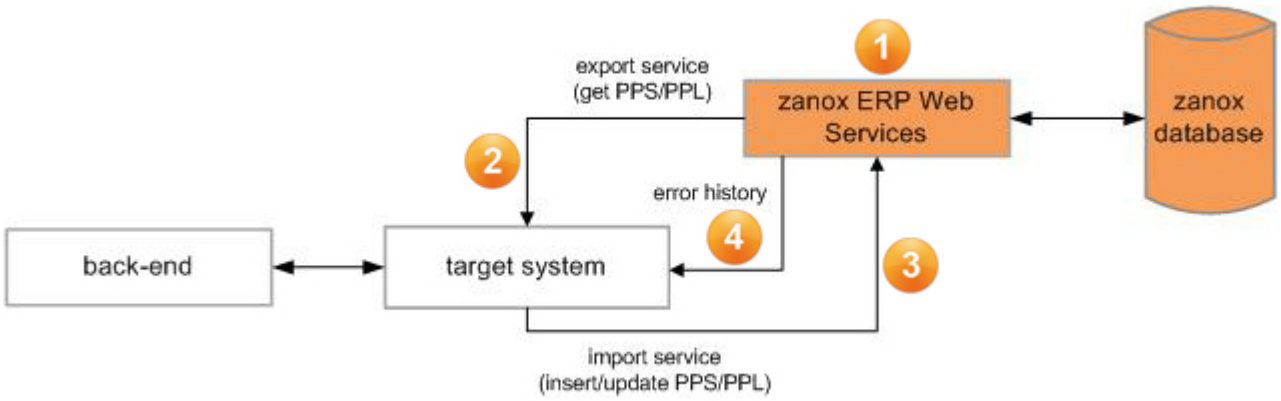
http://services.zanox.com/erp/v2

https://services.zanox.com/erp/v2

SOAP messages are used for two-way communication with the zanox ERP Web Services. These contain all relevant data and internal information.

2.2 Process

The diagram shows the general sequence for the processing of leads and sales via zanox ERP Web Services:



Step 1: Login

You authenticate yourself vis-à-vis zanox ERP Web Services with UserService login method. This method returns a ticket string with which you authenticate yourself for every further SOAP request. The ticket becomes invalid after 20 minutes of inactivity or if the logout method is activated.

Step 2: Export

The export service enables the export of existing leads, sales or shopping baskets. A zanox-wide unique tracking ID is delivered for each dataset. Based on this tracking ID, the desired amendments can be carried out on the relevant lead, sale or shopping basket by means of a subsequent update process.

Step 3: Import

During import, a distinction is drawn between the import types "Update" and "Insert." Amended tracking information is designated as "Update," newly-imported tracking information as "Insert".

The import service provides all necessary methods for the Update. Status changes (confirmations, rejections) can be carried out on existing leads and sales. In addition to this, detail information can be added, such as a processing note or the total price (only for sales). In this regard, the UpdatePpl, UpdatePps and UpdateBasket methods are used.

On the other hand, new sales, which are unknown to the zanox system can be inserted using the Insert methods. In the event of an Insert, the tracking ID will be generated at the time of import.

The import occurs in two phases. In phase 1, the leads and sales are saved by the ERP Web Services at zanox. Phase 2 runs as an automatic job independent of phase 1. In doing so, the data from phase 1 is regularly processed and the changes made visible to advertisers and publishers. zanox guarantees that the processing and visualisation of imported datasets will occur at least once a day.

During import, every Insert or Update process is assigned a so-called history ID. You will receive this ID as a return value of your Insert or Update process. You can obtain access to the current processing status via the history ID. In this regard, you should use the GetImportLog method.

Step 4: Error output

Any error messages from import phase 1 will be transmitted to you directly after the import. In addition to this, you will receive details regarding how many leads and sales were received and how many leads and sales were transferred to the import job. To this end, for each non-transferred lead/sale, the corresponding error will be transmitted. Ideally, you should retrieve the error messages from import phase 2 by means of a cronjob or similar. Here, an error code will be stated for each failed lead/sale.

Possible error codes:

Ignored: The lead/sale is already in the desired state. Consequently, an update will be ignored.

Rejected: The lead/sale has already been confirmed or the partner ID is incorrect (i.e. invalid/inapplicable to the programme/stems from a cancelled partnership, etc.). Consequently, an update will be rejected.

The error messages can then be sent to the responsible person, e.g. by email, so that the problem can be escalated.

Overall sequence

The general sequence of events can be summarised as follows:

1. Export of the leads/sales for a defined period
2. Review of the leads/sales using ones own system, setting of values for Review_State (Status), Review_Note (Note) and, where required, amendment of the total price (only for sales)
3. Import of the leads/sales and saving the history ID
4. In the event that there are further sales in your system, which should be assigned to publishers (bonus transactions, etc.), these may be added by means of Insert.
   > Save history ID
5. Retrieval of the status of import and update errors and, where applicable, escalation
6. 24 hours later: Retrieval of the status of processing and, where applicable, escalation

In the following sections, the above-mentioned steps are explained on the basis of the methods of the zanox ERP Web Services. Particular attention is to be paid to the stipulated required parameters for an Insert and Update.

2.3 Login

To be able to use zanox ERP Web Services, your affiliate programme must be accordingly enabled by a zanox employee. In this initial process, a so-called ERP profile will be set up, which defines the data that can be exported and imported. Following this, the ERP Web Services interface can be used with a valid advertiser account.

A ticket system is used to enable tracking of the user's session, following login, throughout several function calls. A unique ticket is generated at the time of login. The user is known to the zanox ERP Web Services under this ticket. The ticket must be stated in the SOAP header of the SOAP message for every function call, so that the ERP Web Services can identify the respective user.

If the user makes no function calls with a period of 20 minutes (standard), the ticket automatically expires. In order to continue using zanox ERP Web Services, the user will need to login again.

The following information is cached during each session:

- Number of registered programmes and their names
- Assigned ERP profile of a programme
- Assigned lock wait time of a programme
- Access authorisations for the import and export of data for each programme

NOTE: Changes to these parameters only come into effect after the next login.

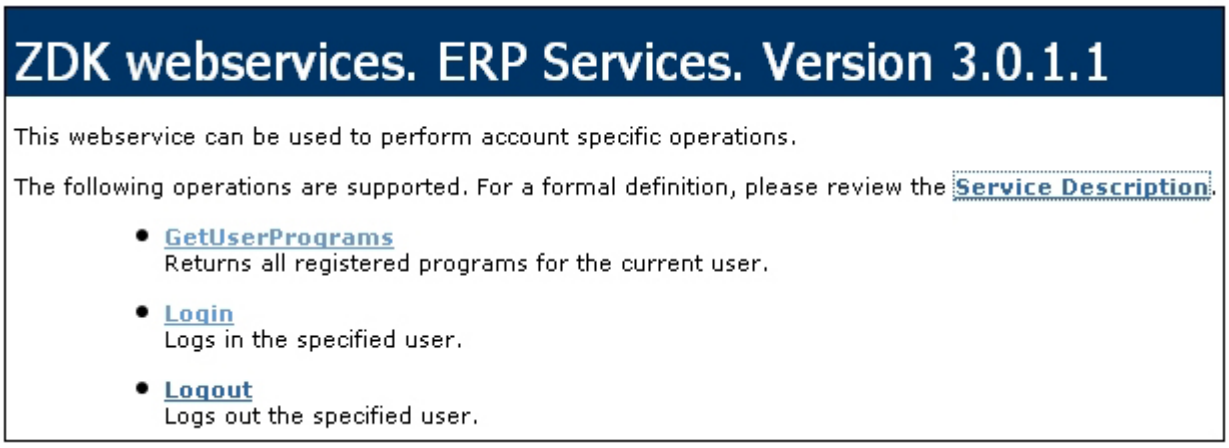
2.4 WSDL description

The WSDL description of the individual zanox ERP Web Services can be accessed online.

To do so, proceed as follows:

1. Access the current version of zanox ERP Web Services via the URL http://services.zanox.com/erp/v2.

2. Click on one of the listed services, in order to view the description page.



3. Click on the "Service Description" link, in order to view the WSDL description of the selected service.

   You can also call up the WSDL description directly by adding the parameter "?wsdl" to the URL.

Examples: http://services.zanox.com/erp/v2/UserService.asmx or http://services.zanox.com/erp/v2/UserService.asmx?wsdl

With the aid of the WSDL description, a proxy or a proxy class can be created, via which a client can easily communicate with the zanox ERP Web Services.

In principle, the required SOAP messages can also be manually generated and sent in XML format.

2.6 Namespace

The requests sent and received by zanox ERP Web Services will be output in XML format and require the definition of a namespace. In principle, all requests lie in the namespace http://services.zanox.com/erp.

In addition to this, two further namespaces are used for export and import functions:

- Export: http://services.zanox.com/erp/Export
- Import: http://services.zanox.com/erp/Import

Please note that, besides a valid XML syntax, the correct use of the namespaces is to be considered. Requests with incorrect namespace declarations cannot be properly processed by zanox ERP Web Services.

2.7 ERP profile

Using the information stored in the ERP profile, self-developed clients can - when converting the received data into CSV format - define which fields should be written into the file, as well as their order and scope.

zanox has assigned a profile to every programme whose ERP data will be exported. The GetPrograms function delivers the profile description in the element `<erpprofile>` in XML format, with the following information:

- Data format for the client-side storage of data
- Number and order of CSV columns
- Names of the individual columns
- Constant values for individual columns

The profile will be delivered in simple XML format.

Example

```
<erpprofile>

<![CDATA[<?xml version="1.0" encoding="utf-8"?>

<profile saveas="csv">

   <fieldlist trackingtype="3" csvdelimiter=";">

     <field intern="prog_id" extern="Programm ID" export="1"/>

     <field intern="tcat_cid" export="1"/>

     <field intern="update_flag" extern="Update" value="0" export="1"/>

   </fieldlist>

   <fieldlist trackingtype="4">
```

```
      <field intern="prog_id" extern="Programm ID" export="1"/>
      <field intern="tcat_cid" export="1"/>
      <field intern="update_flag" extern="Update" value="0" export="1"/>
    </fieldlist>
    <fieldlist trackingtype="23" csvdelimiter=",">
      <field intern="prog_id" extern="Programm ID" export="1" />
      <field intern="tcat_cid" export="1" />
      <field intern="update_flag" extern="Update" value="0" export="0" />
    </fieldlist>
</profile>]]>
</erpprofile>
```

Element <profile>

Object	Data type	Description
<profile>		Contains n elements <fieldlist> as children
saveas	normalizedstring	Format in which the data should be saved on the client-side
<fieldlist>		Contains n elements <field> as children
trackingtype	unsignedbyte	ID of the tracking type assigned to the element <fieldlist>
csvdelimiter	normalizedstring	Field delimiter, which should be used for the CSV export for the stated tracking type
<field>		Element with different attribute values, represents a data field. The order of the individual attribute values must correspond with the order of the individual CSV columns.
intern	normalizedstring	Internally-used field name
extern	normalizedstring	Column name for the CSV export. If this is not stated, the internal field name (attribute "intern") should be used.
export	unsignedbyte	Indicates whether the field should be exported (0 = yes, 1 = no)
value	normalizedstring	Fixed value, which should always be used for this field

The meta information that can be retrieved with the help of the GetFieldInfos function also contain the attribute "intern" for each field. Accordingly, the name stated here can be used to map both information sources to each other.

2.8 Updating of ERP data

2.8.1 Data types

The data type of every field, whose metadata can be retrieved with the help of the GetFieldInfos function, is noted in each instance in the "type" attribute. In this regard, this involves the standard names of the standard types of the Microsoft SQL server. This information can be used by self-developed clients for the purposes of precise validation of the data prior to import.

Internal data type	Value range/format
bit	0 or 1
tinyint	0 to 255
smallint	-32,768 to 32,767 (-2^15 to 2^15-1)
int	-2,147,483,648 to 2,147,483,647 (-2^31 to 2^31-1)
bigint	-9223372036854775808 to 9223372036854775807 (-2^63 to 2^63-1)
smallmoney	-214,748.3648 to 214,748.3647
money	-922,337,203,685,477.5808 to 922,337,203,685,477.5807
decimal	-10^38 + 1 to 10^38 - 1
float	-1.79E + 308 to 1.79E + 308
real	-3.40E + 38 to 3.40E + 38
datetime	1 January 1753 up to 31 December 9999
smalldatetime	1 January 1900 up to 6 June 2079 Accuracy: 1 minute
char, varchar, text	String
nchar, nvarchar, ntext	Unicode string
uniqueidentifier	Format: xxxxxxxx–xxxx–xxxx–xxxx–xxxxxxxxxxxx Characters: 0 – 9, A – F

2.8.2 Validity criteria

During an Insert or Update, zanox ERP Web Services carries out a plausibility and format check for each dataset. Should a problem occur during an Insert, it will be rejected and a corresponding entry is added to the error log (refer to the following section).

In this respect, a dataset must meet the following criteria:

- The data of all transmitted errors must correspond with both the expected XML data type and the possibly still-restricted internal data type (refer to the previous section):

XML data type	Value range/format
boolean	0 or 1
unsignedbyte	0 to 255
short	0 – 65535
int	0 – 2147483647
decimal	0 – 922337203685477.58
normalizedstring	Any character Exceptions: • Carriage return • Line feed • Tabulator
string	Any character
GUID	Format: xxxxxxxx–xxxx–xxxx–xxxx–xxxxxxxxxxxx Characters: 0 – 9, A – F
Characters	0 – 9, A – F
datetime	Format: YYYY-MM-DDThh:mm:ss (Y = year, M = month, D = day, h = hour, m = minute, s = second)

!

NOTE: For "decimal," the English numeral format must be used (with a point rather than a comma).

!

NOTE: Don't forget to put a "T" in front of the hour format hh:mm:ss.

- The maximum string length may not be exceeded.
- All required fields (marked with x) must be present.
- There should not be multiple occurrences of the unique ID of a dataset within one import process.
  - > Dataset ID for Updates: PPS ID, PPL ID or basket ID
  - > Dataset ID for Inserts: Order ID
- Percentage values for commission data must be between 0 and 100.
- Currency amounts may not be negative.
- Currency abbreviations (elements `<currency>`) must be three characters long and may only assume certain values. The list of currently-valid currency abbreviations can be retrieved using the GetSupportedCurrencyCollection method of the "commonservice.asmx" service.
- The programme ID at dataset level in the element `<program>` must correspond with the programme ID of the function parameter.
- The review status must be either 0 (open), 1 (approved) or 2 (rejected).
- Based on the current date, the expiration date must lie in the future.

2.8.3 Error log

If problems occur during an Insert or Update of ERP datasets as a result of incorrect or incomplete data, an error number and an error message will be noted in the error log for each rejected dataset.

This log can be retrieved for a specific import process with the help of the GetImportErrorLog function. The log will be delivered in simple XML format.

Example

```
<log>
   <entries>
     <error recordnr="1" recordid="5521" code="120001">
        Program id must be ‚{0}‘.
     </error>
     <message recordnr="5" recordid="22121" code="120002">
        Review state specified must be between ‚{0}‘ and ‚{1}‘.
     </message>
   </entries>
</log>
```

Element <log>

Object	Data type	Description
<log>		Contains n elements <entries> as children
<entries>		Contains n elements <error> and <message> in any order as children
<error>	string	Error message
recordnr	int	Consecutive number of the dataset in the data package, beginning with 1
recordid	normalizedstring	Unique dataset ID
code	int	Error number
<message>	string	Error message
recordnr	int	Consecutive number of the dataset in the data package, beginning with 1
recordid	normalizedstring	Unique dataset ID
code	int	Message number

In the case of an Update, the GUID of each dataset will be used for the "recordid" attribute (PPS ID, PPL ID or basket ID). The respective order ID will be used for an Insert.

3. DESCRIPTION OF THE FUNCTIONS

In this chapter, all available functions for each service module will be listed and described. In some parts, reference is made to the "current user." This refers to the user who is currently logged in to the zanox ERP Web Services and whose session ticket is stated in the SOAP header.

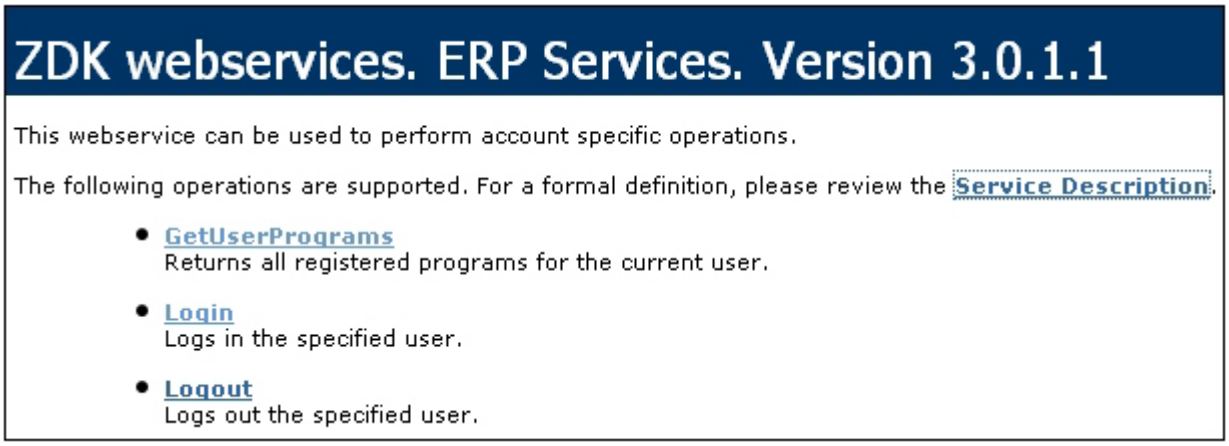
Access to the function descriptions

SOAP messages are sent and received during the course of interaction with zanox ERP Web Services. In order to call up a particular function, the message must be sent to zanox ERP Web Services in a function-specific, XML-like format. The required message format can be viewed online.

To do so, proceed as follows:

1. Access the current version of zanox ERP Web Services via the URL http://services.zanox.com/erp/v2

2. Click on one of the listed services, in order to view the description page.



3. Click on one of the listed functions (e.g. Login), in order to view the message format of the selected function.

You can also call up the message format directly by adding the parameter "?op=<Function_name>" to the URL.

Example: http://services.zanox.com/erp/v2/UserService.asmx?op=Login

SOAP header ticket

In principle, all functions demand a valid ticket in the SOAP header, with the exception of:

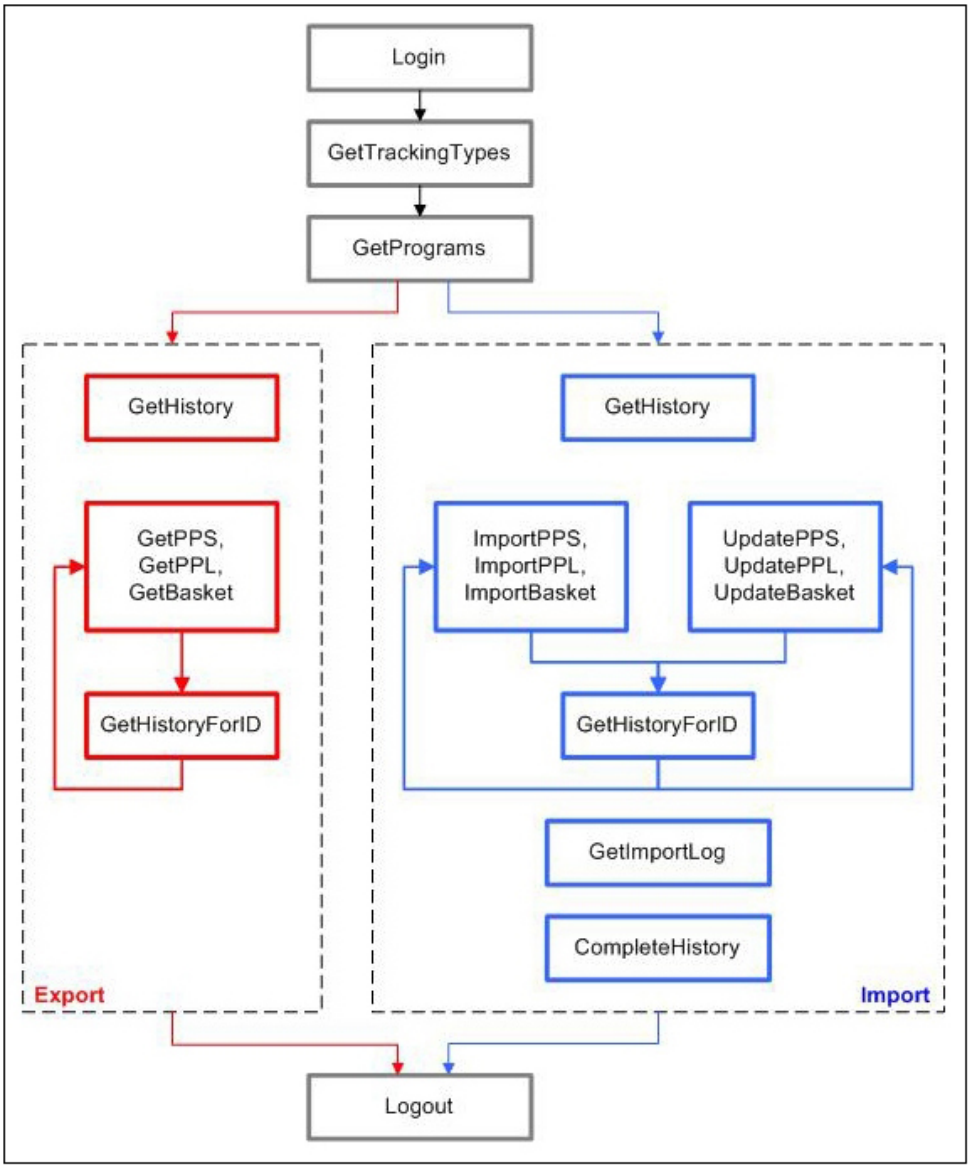
- Login
- GetVersions

The ticket must be stated in the SOAP header.

```
<soap:Header>
   <erp:zanox>
     <erp:ticket>415C6B10D538B1703987CF5436B62DA48DD109C2</erp:ticket>
   </erp:zanox>
</soap:Header>
```

> The namespace to be stated in the XML header is: xmlns:exp="http://services.zanox.com/erp/Export". Where applicable, 'export' must be replaced with 'import.'

Sequence of a general function call



3.1 General functions - Service module "CommonService.asmx"

3.1.1 GetFieldInfosCommon

Delivers meta information on all data fields, which occur during import or export for every tracking type. These data can be used by the client, in order to conduct plausibility and validity checks prior to the import.

Input

None

Output

```
<GetFieldInfosCommonResponse xmlns="http://services.zanox.com/erp">
 <fieldinfos>
   <fieldlist trackingtype="3">
     <field intern="currency_alias" type="String" maxlen="3">
        <xpath>amount/price/@currency</xpath>
        <use insert="1" update="0" />
     </field>
     ...
   </fieldlist>
   ...
 </fieldinfos>
</GetFieldInfosCommonResponse>
```

Element <fieldinfos>

Object	Data type	Description
<fieldinfos>		Contains n elements <fieldlist> as children
<fieldlist>		Contains n elements <field> as children
trackingtype	unsignedbyte	ID of the tracking type assigned to the element <fieldlist>
<field>		
intern	normalizedstring	Internally-used field name Is used in the ERP profile (refer to the section ERP profile) during the assignment of individual fields to one another

Object	Data type	Description
type	normalizedstring	Precise data type or value range of the field
maxlen	int	Maximum length of the field (only if "String" or "normalizedstring" is stated as the type)
<xpath>	normalizedstring	XPath printout, with which the date can be directly extracted from the XML dataflow, provided it is available
<use>		
insert	unsignedbyte	0 > Field will be disregarded in the case of an Insert 1 > Optional field in the case of an Insert 2 > Required field in the case of an Insert
update	unsignedbyte	0 > Field will be disregarded in the case of an Update 1 > Optional field in the case of an Update 2 > Required field in the case of an Update

3.1.2 GetTrackingTypesCommon

Returns a list of all available tracking types.

Input

None

Output

```
<GetTrackingTypesCommonResponse xmlns="http://services.zanox.com/erp">
 <trackingtypes>
   <trackingtype id="3" name="PPS" />
  ...
 </trackingtypes>
</GetTrackingTypesCommonResponse>
```

Element <trackingtypes>

Object	Data type	Description
<trackingtypes>		Contains n elements <trackingtype> as children
<trackingtype>		
id	unsignedbyte	ID of the tracking type
name	normalizedstring	Name of the tracking type

3.1.3 GetSupportedCurrencyCollection

Returns a list of all supported currencies and currency abbreviations.

Input

None

Output

```
<GetSupportedCurrencyCollectionResponse xmlns="http://services.zanox.com/erp">

 <GetSupportedCurrencyCollectionResult xmlns="http://services.zanox.com/erp/Common">

  <currency id="1" name="Euro" alias="EUR"/>

  ...

 </GetSupportetCurrencyCollectionResult>

</GetSupportedCurrencyCollectionResponse>
```

Element <GetSupportedCurrencyCollectionResult>

Object	Data type	Description
<GetSupported-CurrencyCollec-tionResult>		Contains n elements <currency> as child-ren
<currency>		
id	unsignedbyte	Currency ID
name	normalizedstring	Currency name
alias	normalizedstring	Currency abbreviation, e.g. USD or EUR

3.1.4 GetApplicationVersions

Returns information regarding the version of the zanox ERP Web Services.

Input

None

Output

```
<GetApplicationVersionsResponse xmlns="http://services.zanox.com/erp">
 <versions>
   <client>
     <minimum major="1" minor="0" build="230"/>
     <newest major="1" minor="1" build="230"/>
   </client>
   <service major="1" minor="0" build="230"/>
 </versions>
</GetApplicationVersionsResponse>
```

Element <versions>

Object	Data type	Description
<versions>		Contains one child element <client> and one child element <service>
<client>		
<minimum>		
major	int	Main version number of the zanox ERP Client, deemed as a minimum requirement
minor	int	Ancillary version number of the zanox ERP Client, deemed as a minimum requirement
build	int	Build version number of the zanox ERP Client, deemed as a minimum requirement
<newest>		
major	int	Main version number of the latest zanox ERP Client
minor	int	Ancillary version number of the latest zanox ERP Client
build	int	Build version number of the latest zanox ERP Client
<service>		

Object	Data type	Description
major	int	Main version number of the latest zanox ERP Web Services
minor	int	Ancillary version number of the latest zanox ERP Web Services
build	int	Build version number of the latest zanox ERP Web Services

3.2 User management - Service module "UserService.asmx"

3.2.1 GetUserPrograms

Delivers a list of all programmes, which are registered for the current user and for which he/she possesses access rights.

Input

None

Output

```
<GetUserProgramsResponse xmlns="http://services.zanox.com/erp">
 <programs>
   <program id="10">
     <name>Zanox</name>
     <locktime>60</locktime>
     <permissions>
       <export trackingtype="3" />
       ...
       <import trackingtype="3" />
       ...
     </permissions>
     <erpprofil><xml></xml></erpprofil>
   </program>
   ...
 </programs>
</GetUserProgramsResponse>
```

Element <programs>

Object	Data type	Description
<programs>		Contains n elements <program> as children
<program>		
id	int	Programme ID
<name>	normalizedstring	Name of the programme
<locktime>	int	Lock wait time of the programme, in minutes
<permissions>		List of the current user's access rights to the available tracking types, differentiated by import and export lists
<export>		
trackingtype	unsignedbyte	ID of the tracking type for which programme data may be exported
<import>		
trackingtype	unsignedbyte	ID of the tracking type for which programme data may be imported
<erpprofil>		ERP profile of the programme (refer to the section ERP profile)

The lock wait time indicates the minimum temporal intervals in which individual data packages may be called up. By default, this is possible once per hour. Upon request, the lock wait time can be changed by the zanox Service.

A list of all available tracking types can be retrieved with the help of the GetTrackingTypes function.

3.2.2 Login

Logs a user into zanox ERP Web Services. In doing so, definition of a valid and active advertiser account is required. The advertiser account must have a programme, which has been explicitly enabled for the use of zanox ERP Web Services.

To have your advertiser account activated for the zanox ERP Web Services, please contact the zanox Service.

Input

```
<Login xmlns="http://services.zanox.com/erp">

   <username>zanox</username>

   <password>zanox</password>

</Login>
```

Object	Data type	Description	
<username>	normalizedstring	User's login name	x
<password>	normalizedstring	User's password	x

x = mandatory parameter

Output

```
<LoginResponse xmlns="http://services.zanox.com/erp">
 <user>
   <name>zanox</name>
 </user>
</LoginResponse>
```

Object	Data type	Description
<user>		Contains one element <name> as child
<name>	normalizedstring	Name of the logged-in user

Upon successful login, a unique session ticket will be returned to identify the user in the SOAP header.

3.2.3 Logout

Logs the current user out of zanox ERP Web Services. All information from the ongoing session, including the session ticket, will be discarded.

In the event that the logout function is not called up at the end of the session, the session will be automatically terminated after 20 minutes.

Input

None

Output

Returns `True`, if the user has been successfully logged out. Returns `False`, if the session with the stated ticket was already terminated at an earlier time.

3.3 Export – Service module "ExportService.asmx"

3.3.1 GetBasket

Delivers the shopping basket datasets of the stated programme, which correspond with the consigned filter, as well as the history ID of the newly-created export entry.

Input

```
<GetBasket xmlns="http://services.zanox.com/erp">
 <programid>10</programid>
 <basketfilter xmlns="http://services.zanox.com/erp/Export">
  <period from="2004-12-01T00:00:00+02:00"
   to="2004-12-31T23:00:00+02:00"/>
  <reviewstate negate="1">1<reviewstate>
```

```
  <category status="1"/>

  <orderid negate="1">TestKat</orderid>

</basketfilter>

</GetBasket>
```

Object	Data type	Charac-ters	Description	
<programid>	int		ID of the programme whose data will be retrieved	x
<basketfil-ter>			Filter criteria, which will be applied during the export	x
<period>			Export period	x
from	datetime		Most recent time stamp of the dataset	x
to	datetime		Oldest time stamp of the data-set	x
<reviewsta-te>	int		Review status of the sales to which the shopping basket appertains	
negate	boolean		Criterion <reviewstate> should • 0 = not be negated • 1 = be negated	
<category>			Tracking category of the dataset at zanox	
status	byte		Status of the tracking category is • 1 = known • 2 = unknown	
<orderid>	normalized-string	50	Order ID of the dataset	
negate	boolean		Criterion <orderid> should • 0 = not be negated • 1 = be negated	

x = mandatory parameter

If the value of the "negate" attribute of the orderid element is ‚1,' all datasets that do not correspond with the condition will be exported.

Output

If the lock wait time has not yet expired and the user wishes to carry out a new data export, the request will be terminated with an error (refer to <locktime> element in the section UserService – GetUserPrograms).

Object	Data type	Description
<historyid>	int	History ID that was created for the export process

Otherwise, zanox ERP Web Services delivers the following data:

```
<GetBasketResponse xmlns="http://services.zanox.com/erp">
 <basketdata>
   <basket id="a5028acb-23b2-4b42-9edf-70ff0ac04167" trackingtype="3">
     <click>
       <ulp>string</ulp>
     </click>
     <request timestamp="2004-06-18T12:16:28"/>
     <program id="10"/>
     <order id="1396525"/>
     <category id="Ihre-Kategorie"/>
     <affiliate>
       <commission currency="EUR">0.66</commission>
     </affiliate>
     <product number="1234" quantity="1">
       <name>productname</name>
       <price currency="EUR">8.15</price>
     </product>
     <commission fix="0.20" percent="0.15"/>
  </basket>
  ...
 </basketdata>
<GetBasketResponse>
```

If the value for <historyid> is '0', no data have been exported. For instance, this might happen because the filter has been set too restrictively).

Object	Data type	Description
<basketdata>		Contains n elements <basket> as children
<basket>		Contains information on the shopping basket dataset
id	normalizedstring	Unique ID of the shopping basket dataset (GUID)
trackingtype	unsignedbyte	Tracking type for which the dataset was tracked
<click>		
<ulp>	normalizedstring	Transmitted ULP
<request>		
timestamp	datetime	Time stamp of the request
<program>		
id	int	ID of the programme to which the dataset appertains
<order>		
id	normalizedstring	Order ID
<category>		
id	normalizedstring	Category ID
<affiliate>		
<commission>	decimal	Publisher commission
currency	normalizedstring	Currency of the publisher commission
<product>		
number	normalizedstring	Product number
quality	short	Product quantity
<name>	normalizedstring	Product name
<price>	decimal	Product price
currency	normalizedstring	Currency of the product price
<commission>		
fix	decimal	Fixed commission in the programme currency
percent	decimal	Variable commission in percent

3.3.2 GetHistoryExport

Delivers a list of all export processes carried out (export history) for the stated programme and the tracking type.

Input

```
<GetHistoryExport xmlns="http://services.zanox.com/erp">
 <programid>10</programid>
 <trackingtype>3</trackingtype>
 <rowcount>20></rowcount>
</GetHistoryExport >
```

Object	Data type	Description	
<programid>	int	ID of the programme whose history will be retrieved	x
<trackingtype>	unsignedbyte	ID of the tracking type for which the data will be retrieved	x
<rowcount>	int	Number of displayed entries (with 0, all will be displayed)	

x = mandatory parameter

Output

```
<GetHistoryExportResponse xmlns="http://services.zanox.com/erp">
 <historydata>
   <history id="1" programid="10" trackingtype="3">
     <created>2005-05-01 12:05:33</created>
     <records>1234</records>
     <filteredby>
       <period from="2004-12-01T00:00:00+02:00"
   to="2004-12-31T23:00:00+02:00"/>
       <reviewstate negate="1">1</reviewstate>
       <categoryid>TestKat</categoryid>
       <orderid>22</orderid>
     </filteredby>
   </history>
   ...
 </historydata>
```

```
</GetHistoryExportResponse>
```

Object	Data type	Description
<historydata>		Contains n elements <history> as children
<history>		
id	int	ID of the export process
programid	int	ID of the programme for which the export was executed
trackingtype	unsignedbyte	ID of the tracking type for which the export was executed
<created>	datetime	Time stamp of the export
<records>	int	Number of exported datasets
<filteredby>		Filter criteria, which were applied during the export
<period>		
from	datetime	Most recent time stamp of the dataset
to	datetime	Oldest possible time stamp of the dataset
<reviewstate>	datetime	Review status of the dataset (only stated if the history entry was created via GetPps or GetPpl)
negate	boolean	Criterion <reviewstate> was • 0 = not be negated • 1 = negated
<categoryid>	normalizedstring	Category ID of the dataset (only stated if the history entry was created via GetPps or GetPpl)
negate	boolean	Criterion <categoryid> was • 0 = not be negated • 1 = negated
<orderid>	normalizedstring	Order ID of the dataset (only stated if the history entry was created via GetBasket)
negate	boolean	Criterion <orderid> was • 0 = not be negated • 1 = negated

If the value of the "negate" attribute of the <reviewstate>, <categoryid> or <orderid> elements is ‚1,' all datasets that did not correspond with the condition were exported.

3.3.3 GetHistoryForId

Delivers the data for a selected history entry, e.g. if a new history entry was created by an export process and only these data should be retrieved.

Input

```
<GetHistoryForId xmlns="http://services.zanox.com/erp">

   <historyid>10</historyid>

</GetHistoryForId>
```

Object	Data type	Description	
<historyid>	int	ID of the history entry whose data should be retrieved	x

x = mandatory parameter

Output

```
<GetHistoryForIdResponse xmlns="http://services.zanox.com/erp">

 <historydata>

   <history id="1" programid="10" trackingtype="3">

     <created>2005-05-01 12:05:33</created>

     <records>1234</records>

     <filteredby>

       <period from="2004-12-01T00:00:00+02:00"
     to="2004-12-31T23:00:00+02:00"/>

       <reviewstate negate="1">1</reviewstate>

       <categoryid>TestKat</categoryid>

       <orderid>22</orderid>

     </filteredby>

   </history>

 </historydata>

</GetHistoryForIdResponse>
```

Object	Data type	Description
<historydata>		Contains n elements <history> as children
<history>		
id	int	ID of the export process
programid	int	ID of the programme for which the export was executed
trackingtype	unsignedbyte	ID of the tracking type for which the export was executed
<created>	datetime	Time stamp of the export
<records>	int	Number of exported datasets
<filteredby>		Filter criteria, which were applied during the export
<period>		
from	datetime	Most recent time stamp of the dataset
to	datetime	Oldest possible time stamp of the dataset
<reviewstate>	datetime	Review status of the dataset (only stated if the history entry was created via GetPps or GetPpl)
negate	boolean	Criterion <reviewstate> was • 0 = not be negated • 1 = negated
<categoryid>	normalizedstring	Category ID of the dataset (only stated if the history entry was created via GetPps or GetPpl)
negate	boolean	Criterion <categoryid> was • 0 = not be negated • 1 = negated
<orderid>	normalizedstring	Order ID of the dataset (only stated if the history entry was created via GetBasket)
negate	boolean	Criterion <orderid> was • 0 = not be negated • 1 = negated

If the value of the "negate" attribute of the <reviewstate>, <categoryid> or <orderid> elements is ‚1,' all datasets that did not correspond with the condition were exported.

3.3.4 GetPpl

Delivers the PPL datasets of the stated programme, which correspond with the consigned filter criteria, as well as the ID of the newly-created export history entry.

Input

```
<GetPpl xmlns="http://services.zanox.com/erp">

 <programid>10</programid>

 <pplfilter xmlns="http://services.zanox.com/erp/Export">

   <period from="2004-12-01T00:00:00+02:00"
    to="2004-12-31T23:00:00+02:00"/>

   <reviewstate negate="1">0</reviewstate>

   <categoryid negate="1">TestKat</categoryid>

 </pplfilter>

</GetPpl>
```

Object	Data type	Description	
<programid>	int	ID of the programme whose data will be retrieved	x
<pplfilter>		Filter criteria, which will be applied during the export	x
<period>			x
from	datetime	Most recent time stamp of the dataset	x
to	datetime	Oldest possible time stamp of the dataset	x
<reviewstate>	datetime	Review status of the dataset	
negate	boolean	Criterion <reviewstate> should • 0 = not be negated • 1 = be negated	
<categoryid>	normalizedstring	Category ID of the dataset	
negate	boolean	Criterion <categoryid> should • 0 = not be negated • 1 = be negated	

x = mandatory parameter

If the value of the "negate" attribute of the <reviewstate> or <categoryid> elements is ‚1,' all datasets that do not correspond with the condition will be exported.

Output

If the lock wait time has not yet expired and the user wishes to carry out a new data export, the request will be terminated with an error (refer to element <locktime> in the section UserService – GetUserPrograms).

Object	Data type	Description
<historyid>	int	History ID that was created for the export process

If the value for <historyid> is '0', no data have been exported. For instance, this might happen because the filter has been set too restrictively).

Otherwise, zanox ERP Web Services delivers the following data:

```
<GetPplResponse xmlns="http://services.zanox.com/erp">
 <historyid>1</historyid>
 <ppldata>
   <ppl id="a5028acb-23b2-4b42-9edf-70ff0ac04167" trackingtype="3">
     <click timestamp="2004-06-18T12:20:58">
        <ulp>string</ulp>
     </click>
     <request timestamp="2004-06-18T12:16:28" ipaddress="111.111.111.
111"/>
     <program id="10"/>
     <order id="1396525"/>
     <customer id="285796"/>
     <category id="Ihre-Kategorie"/>
     <session id="Sess01"/>
     <expiration date="2005-02-12T12:00:12"/>
     <affiliate code="829057C1355136413S66656">
        <commission currency="EUR">0.66</commission>
        <username>Affiliate</username>
        <website>http://www.affiliate.de</website>
     </affiliate>
     <subaffiliate id="21">
        <commission currency="EUR">0.20</commission>
     </subaffiliate>
     <review state="0" date="2005-01-02 21:22:00">ReviewText</review>
   </ppl>
   ...
 </ppldata>
</GetPplResponse>
```

Object	Data type	Description
<ppldata>		
<ppl>		
id	normalizedstring	Unique ID of the PPL dataset (GUID)
trackingtype	unsignedbyte	Tracking type for which the dataset was tracked
<click>		
timestamp	datetime	Time stamp of the click
<ulp>	normalizedstring	Transmitted ULP
<request>		
timestamp	datetime	Time stamp of the request
ipaddress	normalizedstring	IP-address of the request
<program>		
id	int	ID of the programme to which the dataset apper-tains
<order>		
id	normalizedstring	Order ID
<customer>		
id	normalizedstring	Customer ID
<category>		
id	normalizedstring	Category ID
<session>		
id	normalizedstring	Session ID
<expiration>		
date	datetime	Expiration date
<affiliate>		
code	normalizedstring	Complete partner ID in xxxCxxxSxxx format ("Sxxx" is optional)
<commission>	decimal	Publisher commission
currency	normalizedstring	Currency of the publisher commission
<username>	normalizedstring	User's login name
<website>	normalizedstring	User's homepage
<subaffiliate>		
id	normalizedstring	Sub-affiliate ID
<commission>	decimal	Sub-affiliate commission
currency	normalizedstring	Currency of the sub-affiliate commission
<review>	string	Review text

Object	Data type	Description
state	int	Review status • 0 = open • 1 = approved • 2 = rejected
date	datetime	Review date

3.3.5 GetPps

Delivers the PPS datasets of the stated programme, which correspond with the consigned filter criteria, as well as the ID of the newly-created export history entry.

Input

```
<GetPps xmlns="http://services.zanox.com/erp">
 <programid>10</programid>
 <ppsfilter xmlns="http://services.zanox.com/erp/Export">
   <period from="2004-12-01T00:00:00+02:00"
    to="2004-12-31T23:00:00+02:00"/>
   <reviewstate negate="1">0</reviewstate>
   <categoryid negate="1">TestKat</categoryid>
 </ppsfilter>
</GetPps>
```

Object	Data type	Description	
<programid>	int	ID of the programme whose data will be retrieved	x
<ppsfilter>		Filter criteria, which will be applied during the export	x
<period>			x
from	datetime	Most recent time stamp of the dataset	x
to	datetime	Oldest possible time stamp of the dataset	x
<reviewstate>	datetime	Review status of the dataset	
negate	boolean	Criterion <reviewstate> should • 0 = not be negated • 1 = be negated	

Object	Data type	Description	
<categoryid>	normalizedstring	Category ID of the dataset	
negate	boolean	Criterion <categoryid> should • 0 = not be negated • 1 = be negated	

x = mandatory parameter

If the value of the "negate" attribute of the <reviewstate> or <categoryid> elements is ‚1,' all datasets that do not correspond with the condition will be exported.

Output

If the lock wait time has not yet expired and the user wishes to carry out a new data export, the request will be terminated with an error (refer to element <locktime> in the section UserService – GetUserPrograms).

Object	Data type	Description
<historyid>	int	History ID that was created for the export process

Otherwise, zanox ERP Web Services delivers the following data:

If the value for <historyid> is '0', no data have been exported. For instance, this might happen because the filter has been set too restrictively).

```
<GetPpsResponse xmlns="http://services.zanox.com/erp">
 <historyid>1</historyid>
 <ppsdata>
   <pps id="a5028acb-23b2-4b42-9edf-70ff0ac04167" trackingtype="3">
     <click timestamp="2004-06-18T12:20:58">
        <ulp>string</ulp>
     </click>
     <request timestamp="2004-06-18T12:16:28" ipaddress="111.111.111.
111"/>
     <program id="10"/>
     <order id="1396525"/>
     <suborder id=""/>
     <customer id="285796"/>
     <category id="Ihre-Kategorie"/>
     <session id="Sess01"/>
     <expiration date="2005-02-12T12:00:12"/>
```

```
      <amount>
         <price currency="ANG">2.65</price>
         <tracked currency="EUR">4.50</tracked>
         <intern currency="EUR">4.50</intern>
      </amount>
      <affiliate code="829057C1355136413S66656">
         <commission currency="EUR">0.66</commission>
         <username>Affiliate</username>
         <website>http://www.affiliate.de</website>
      </affiliate>
      <subaffiliate id="21">
         <commission currency="EUR">0.20</commission>
      </subaffiliate>
      <review state="0" date="2005-01-02T21:22:00">ReviewText</review>
      <commission fix="0.20" percent="0.15"/>
   </pps>
   ...
 </ppsdata>
</GetPpsResponse>
```

Object	Data type	Description
<ppsdata>		
<pps>		
id	normalizedstring	Unique ID of the PPS dataset (GUID)
trackingtype	unsignedbyte	Tracking type for which the dataset was tracked
<click>		
timestamp	datetime	Time stamp of the click
<ulp>	normalizedstring	Transmitted ULP
<request>		
timestamp	datetime	Time stamp of the request
ipaddress	normalizedstring	IP-address of the request
<program>		
id	int	ID of the programme to which the dataset appertains

Object	Data type	Description
<order>		
id	normalizedstring	Order ID
<suborder>		
id	normalizedstring	Sub-order ID
<customer>		
id	normalizedstring	Customer ID
<category>		
id	normalizedstring	Category ID
<session>		
id	normalizedstring	Session ID
<expiration>		
date	datetime	Expiration date
<amount>		
<price>	decimal	Price
currency	normalizedstring	Desired currency of the price
<tracked>	decimal	Recorded price
currency	normalizedstring	Currency of the recorded price
<intern>	decimal	Internal price
currency	normalizedstring	Currency of the internal price
<affiliate>		
code	normalizedstring	Complete partner ID in xxxCxxxSxxx format ("Sxxx" is optional)
<commission>	decimal	Publisher commission
currency	normalizedstring	Currency of the publisher commission
<username>	normalizedstring	User's login name
<website>	normalizedstring	User's homepage
<subaffiliate>		
id	normalizedstring	Sub-affiliate ID
<commission>	decimal	Sub-affiliate commission
currency	normalizedstring	Currency of the sub-affiliate commission
<review>	string	Review text
state	int	Review status • 0 = open • 1 = approved • 2 = rejected)

Object	Data type	Description
date	datetime	Review date
<commission>		
fix	decimal	Fixed commission in the programme currency
percent	decimal	Variable commission in percent

3.4 Import - Service module "ImportService.asmx"

3.4.1 CompleteHistory

Concludes the import process with the stated ID. As a result, it is no longer output by default in the zanox ERP Client when retrieving the import history data. An import process is normally concluded automatically, if it contains no erroneous datasets.

Input

```
<CompleteHistory xmlns="http://services.zanox.com/erp">
   <historyid>10</historyid>
</CompleteHistory>
```

Object	Data type	Description	
<historyid>	int	ID of the history entry, which should be concluded	x

x = mandatory parameter

Output

Returns `True`, if the import process has been successfully concluded. `False` is returned in all other cases.

3.4.2 GetHistoryImport

Delivers a list of all import processes carried out (import history) for the stated programme and the tracking type.

Input

```
<GetHistoryImport xmlns="http://services.zanox.com/erp">
   <historyid>10</historyid>
   <trackingtype>3</trackingtype>
   <rowCount>3</rowCount>
</GetHistoryImport>
```

Object	Data type	Description	
<historyid>	int	ID of the history entry whose data should be retrieved	x
<trackingtype>	unsignedbyte	ID of the tracking type for which the data will be retrieved	x
<rowCount>	int	Number of displayed entries (with 0, all will be displayed)	

x = mandatory parameter

Output

```
<GetHistoryImportResponse xmlns="http://services.zanox.com/erp">
 <historydata>
   <history id="1" importtype="1" programid="10" trackingtype="3"
status="received">
     <uploaded>2005-01-01 01:12:01</uploaded>
     <records>
        <received imported="100" rejected="5"/>
        <processed imported="97" rejected="2" ignored="1"/>
     </records>
     <completed loginid="50">2005-01-03T21:22:00</completed>
   </history>
   ...
 </historydata>
</GetHistoryImportResponse>
```

Object	Data type	Description
<historydata>		Contains n elements <history> as children
<history>		
id	int	ID of the import process
importtype	unsignedbyte	Indicates whether it involved an Update or an Insert process • 1 = Insert • 2 = Update
programid	int	ID of the programme for which the import was executed

Object	Data type	Description
trackingtype	unsignedbyte	ID of the tracking type for which the import was executed
status	byte	Status of the history entry: • received = Data have been received by zanox ERP Web Services • processed = Data have been processed • rejected = Data have been completely rejected • completed = Data have been registered
<uploaded>	datetime	Time stamp of the import
<records>	int	Number of imported datasets
<received>		
inserted	int	Number of datasets, which were inserted during import
rejected	int	Number of datasets, which were rejected during import as a result of data errors
<processed>		
inserted	int	Number of datasets, which were inserted during the subsequent registration process (or -1, in the event that the registration process has not yet occurred)
rejected	int	Number of datasets, which were rejected during the subsequent registration process as a result of data errors (or -1, in the event that the registration process has not yet occurred)
ignored	int	Number of datasets, which were ignored during the subsequent registration process as a result of data errors (or -1, in the event that the registration process has not yet occurred)
<completed>	datetime	Date on which the process was concluded
loginid	int	Login ID with which the process was concluded

3.4.3 GetHistoryForId

Delivers the data for a selected history entry, e.g. if a new history entry was created by an import process and only these data should be retrieved.

Input

```
<GetHistoryForId xmlns="http://services.zanox.com/erp">
   <historyid>10</historyid>
</GetHistoryForId>
```

Object	Data type	Description	
<historyid>	int	ID of the history entry whose data should be retrieved	x

x = mandatory parameter

Output

```
<GetHistoryForIdResponse xmlns="http://services.zanox.com/erp">
 <historydata>
   <history id="1" importtype="1" programid="10" trackingtype="3" status="1">
     <uploaded>2005-01-01 01:12:01</uploaded>
     <records>
        <received imported="100" rejected="5"/>
        <processed imported="97 "rejected="2" ignored="1"/>
     </records>
     <completed loginid="50">2005-01-03T21:22:00</completed>
   </history>
   ...
 </historydata>
</GetHistoryForIdResponse>
```

Object	Data type	Description
<historydata>		Contains n elements <history> as children
<history>		
id	int	ID of the import process

Object	Data type	Description
importtype	unsignedbyte	Indicates whether it involved an Update or an Insert process • 1 = Insert • 2 = Update
programid	int	ID of the programme for which the import was executed
trackingtype	unsignedbyte	ID of the tracking type for which the import was executed
status	byte	Status of the history entry: • 1 - received = Data have been received by zanox ERP Web Services • 2 - processed = Data have been partially processed • 3 - rejected = Data have been completely rejected • 4 - completed = Data have been completely processed
<uploaded>	datetime	Time stamp of the import
<records>	int	Number of imported datasets
<received>		
inserted	int	Number of datasets, which were inserted during import
rejected	int	Number of datasets, which were rejected during import as a result of data errors
<processed>		
inserted	int	Number of datasets, which were inserted during the subsequent registration process (or -1, in the event that the registration process has not yet occurred)
rejected	int	Number of datasets, which were rejected during the subsequent registration process as a result of data errors (or -1, in the event that the registration process has not yet occurred)

Object	Data type	Description
ignored	int	Number of datasets, which were ignored during the subsequent registration process as a result of data errors (or -1, in the event that the registration process has not yet occurred)
<completed>	datetime	Date on which the process was concluded
loginid	int	Login ID with which the process was concluded

3.4.4 GetImportLog

Returns both error logs for the actual upload process and the subsequent registration process appertaining to a specific import process. The latter runs asynchronously at defined times within the zanox system and updates the corresponding log section in the event of occurring errors.

Input

```
<GetImportLog xmlns="http://services.zanox.com/erp">

   <historyid>10</historyid>

</GetImportLog>
```

Object	Data type	Description	
<historyid>	int	ID of the history entry for which the log will be retrieved	x

x = mandatory parameter

Output

```
<GetImportLogResponse xmlns="http://services.zanox.com/erp">
 <importlog>
   <received><log>…</log></received>
   <processed><log>...</log></processed>
 </importlog>
</GetImportLogResponse>
```

Object	Data type	Description
<importlog>		Contains the elements <received> and <processed> as children
<received>	string	Error log for the actual upload process
<processed>	string	Error log for the actual registration process

The returned strings are XML documents whose format is described in greater detail in the section "Error log."

3.4.5 InsertBasket

Serves for the import of new shopping basket datasets for the stated programme.

Input

```
<InsertBasket xmlns="http://services.zanox.com/erp">
 <programid>10</programid>
 <basketinsert xmlns="http://services.zanox.com/erp/Import">
   <basket>
     <program id="10"/>
     <order id="55215412"/>
     <category id="55215412"/>
     <commission fix="4.05" percent="3.00"/>
     <product number="1234" quantity="1">
        <name>productname</name>
        <price>8.15</price>
     </product>
     <click>
        <ulp>string</ulp>
```

```
      </click>
    </basket>
  ...
</InsertBasket>
```

Object	Data type	Characters	Description	
<programid>	int		ID of the programme for which new shopping basket datasets will be imported	x
<basket-insert>			Shopping basket datasets, which will be newly imported	x
<basket>				x
<program>				x
id	int		ID of the programme to which the datasets will be newly imported	x
<order>				x
id	string	Maximum 50	Order ID	x
<category>				
id	normalized-string	Maximum 10	Category ID	
<commission>				
fix	decimal		Fixed commission in the programme currency	
percent	decimal		Variable commission in percent	
<product>				
number	normalized-string	Maximum 50	Product number	
quantity	unsignedbyte		Product quantity	
<name>	normalized-string	Maximum 100	Product name	
<price>	decimal		Product price	
<click>				
<ulp>	normalized-string	Maximum 255	Transmitted ULP	

x = mandatory parameter

3.4.6 InsertPpl

Serves for the import of new PPL datasets for the stated programme.

Input

```
<InsertPpl xmlns="http://services.zanox.com/erp">
 <programid>10</programid>
 <pplinsert xmlns="http://services.zanox.com/erp/Import">
   <ppl>
     <request timestamp="2004-04-20T21:00:25"/>
     <program id="10"/>
     <order id="55215412"/>
     <suborder id="4453C33"/>
     <customer id="Cust02"/>
     <category id="Ihre-Kategorie"/>
     <session id="TTF44FF434444FFD"/>
     <expiration date="2004-05-21T11:00:22"/>
     <affiliate code="5565C623007S22031"/>
     <review state="0">ReviewText!</review>
     <imp:commission fix="4.05"/>
      <imp:amount>
      <imp:price currency="EUR">0</imp:price>
      </imp:amount>
     <cookie id="12345"/>
   </ppl>
 ...
 </pplinsert>
</InsertPpl>
```

Object	Data type	Characters	Description	
<programid>	int		ID of the programme for which new PPL datasets will be imported	x
<pplinsert>			PPL datasets, which will be newly imported	x

Object	Data type	Characters	Description	
<ppl>				x
<request>				x
timestamp	datetime		Time stamp	x
<program>				x
id	int		ID of the programme to which the datasets will be newly imported	x
<order>				x
id	normalized-string	Maximum 50	Order ID	x
<suborder>				
id	normalized-string	Maximum 10	Sub-order ID	
<customer>				x
id	normalized-string	Maximum 50	Customer ID	x
<category>				
id	normalized-string	Maximum 10	Category ID	
<session>				
id	normalized-string	Maximum 255	Session ID	
<expiration>				
date	datetime		Expiration date	
<affiliate>				x
code	normalized-string	Maximum 64	Complete partner ID in xxx-CxxxSxxx format ("Sxxx" is optional)	x
<review>	string	Maximum 255	Review text	
state	int		Review status 0 = open 1 = confirmed, approved	x
<commission>				
fix	decimal		Fixed commission in the pro-gramme currency	
<cookie>				

Object	Data type	Characters	Description	
id	int		Cookie ID, which was internally generated by zanox	

x = mandatory parameter

3.4.7 InsertPps

Serves for the import of new PPS datasets for the stated programme.

Input

```
<InsertPps xmlns="http://services.zanox.com/erp">
 <programid>10</programid>
 <ppsinsert xmlns="http://services.zanox.com/erp/Import">
   <pps>
     <request timestamp="2004-04-20T21:00:25"/>
     <program id="10"/>
     <order id="55215412"/>
     <suborder id="4453C33"/>
     <customer id="Cust02"/>
     <category id="Ihre-Kategorie"/>
     <session id="TTF44FF434444FFD"/>
     <expiration date="2004-05-21T11:00:22"/>
     <amount>
        <price currency="EUR">3.50</price>
     </amount>
     <review state="0">ReviewText!</review>
     <commission fix="4.05" percent="3.00"/>
     <cookie id="12345"/>
   </pps>
 ...
 </ppsinsert>
</InsertPps>
```

Object	Data type	Characters	Description	
<programid>	int		ID of the programme for which new PPS datasets will be imported	x
<ppsinsert>			PPS datasets, which will be newly imported	x
<pps>				x
<request>				x
timestamp	datetime		Time stamp	x
<program>				x
id	int		ID of the programme to which the datasets will be newly imported	x
<order>				x
id	normalized-string	Maximum 50	Order ID	x
<suborder>				
id	normalized-string	Maximum 10	Sub-order ID	
<customer>				x
id	normalized-string	Maximum 50	Customer ID	x
<category>				
id	normalized-string	Maximum 10	Category ID	
<session>				
id	normalized-string	Maximum 255	Session ID	
<expiration>				
date	datetime		Expiration date	
<amount>				x
<price>	decimal		Price	x
currency	normalized-string	Maximum 3	Desired currency of the price	x
<affiliate>				x
code	normalized-string	Maximum 64	Complete partner ID in xxx-CxxxSxxx format ("Sxxx" is optional)	x
<commission fix="4.05"/>	string	Maximum 255	Review text	

Object	Data type	Characters	Description	
state	int		Review status 0 = open 1 = confirmed, approved	x
<commission>				
fix	decimal		Fixed commission in the programme currency	
percent	decimal		Variable commission in percent	
<cookie>				
id	int		Cookie ID, which was internally generated by zanox	

x = mandatory parameter

3.4.8 UpdateBasket

Serves for the update of existing shopping basket datasets for the stated programme.

Input

```
<UpdateBasket xmlns="http://services.zanox.com/erp">
 <programid>10</programid>
 <basketupdate xmlns="http://services.zanox.com/erp/Import">
   <basket id="a1a1a1a1-a1a1-a1a1-a1a1-a1a1a1a1a1a1">
     <program id="10"/>
     <product quantity="10">
        <price currency="EUR">3.50</price>
     </product>
   </basket>
   ...
 </basketupdate>
</UpdateBasket>
```

Object	Data type	Characters	Description	
<programid>	int		ID of the programme whose shopping basket datasets will be updated	x
<basketupdate>			Shopping basket datasets, which will be updated	x

Object	Data type	Characters	Description	
<basket>				x
id	normalized-string		Unique ID of the shopping basket (GUID)	x
<program>				x
id	int		ID of the programme for which the datasets will be updated	x
<product>				
quantity	unsignedbyte		Product quantity	
<price>	decimal		Product price	
currency	normalized-string	Maximum 3	Desired currency of the pro-duct price	

x = mandatory parameter

3.4.9 UpdatePpl

Serves for the update of existing PPL datasets for the stated programme.

Input

```
<UpdatePpl xmlns="http://services.zanox.com/erp">
 <programid>10</programid>
 <pplupdate xmlns="http://services.zanox.com/erp/Import">
   <ppl id="a1a1a1a1-a1a1-a1a1-a1a1-a1a1a1a1a1a1">
     <program id="10"/>
     <expiration date="2004-05-21T11:00:22"/>
     <review state="0">ReviewText</review>
     <imp:commission fix="4.05"/>
     <imp:amount>
     <imp:price currency="EUR">0</imp:price>
     </imp:amount>

   </ppl>
   ...
 </pplupdate>
</UpdatePpl>
```

Object	Data type	Characters	Description	
<programid>	int		ID of the programme whose PPL datasets will be updated	x
<pplupdate>			PPL datasets, which will be updated	x
<ppl>				x
id	normalized-string		Unique ID of the PPL dataset (GUID)	x
<program>				x
id	int		ID of the programme for which the datasets will be updated	x
<expiration>				
date	datetime		Expiration date	
<review>	string	Maximum 255	Review text	
state	normalized-string		Review status 0 = open 1 = confirmed, approved 2 = rejected	x
<commission>				
fix	decimal		Fixed commission in the pro-gramme currency	

x = mandatory parameter

3.4.10 UpdatePps

Serves for the update of existing PPS datasets for the stated programme.

Input

```
<UpdatePps xmlns="http://services.zanox.com/erp">
 <programid>10</programid>
 <ppsupdate xmlns="http://services.zanox.com/erp/Import">
   <pps id="a1a1a1a1-a1a1-a1a1-a1a1-a1a1a1a1a1a1">
     <program id="10"/>
     <expiration date="2004-05-21T11:00:22"/>
     <amount>
        <price currency="EUR">3.50</price>
     </amount>
```

NOTE: You can undo the status "Rejected" and set a transaction back to "Open" once. Please note, that the transaction will be set to "Approved" by zanox after 72 hours.

```
      <review state="0">ReviewText</review>
      <commission fix="4.05" percent="3.00"/>
    </pps>
    ...
  </ppsupdate>
</UpdatePps>
```

Object	Data type	Characters	Description	
<programid>	int		ID of the programme whose PPS datasets will be updated	x
<ppsupdate>			PPS datasets, which will be updated	x
<pps>				x
id	normalized-string		Unique ID of the PPS dataset (GUID)	x
<program>				x
id	int		ID of the programme for which the datasets will be updated	x
<expiration>				
date	datetime		Expiration date	
<amount>				
<price>	decimal			
currency	normalized-string	Maximum 3		
<review>	string	Maximum 255	Review text	
state	int		Review status 0 = open 1 = confirmed, approved 2 = rejected	x
<commission>				
fix	decimal		Fixed commission in the programme currency	
percent	decimal		Variable commission in percent	

x = mandatory parameter

NOTE: You can undo the status "Rejected" and set a transaction back to "Open" once. Please note, that the transaction will be set to "Approved" by zanox after 72 hours.

4. ERROR MESSAGES

When inserting new tracking datasets the following error messages may occur:

Error code	Error message	Description
274	zTRC_ERR_DISABLED_CID	The tracking category is deactivated.
302	zTRC_ERR_CHECK_PROG	The programme ID is invalid (e.g. wrong check_num or inexistent prog_id) or the tracking type (HTML or script) has not been activated for the programme.
307	zTRC_ERR_INVALID_PROGRAM	The programme does not exist (e.g. inexistent prog_id).
320	zTRC_ERR_INVALID_TIMESTAMP	The timestamp is invalid or in the future.
326	zTRC_ERR_EMPTY_CURRENCY_SYMBOL	No currency symbol has been specified.
327	zTRC_ERR_EMPTY_CUSTOMER_ID	No customer ID has been specified.
331	zTRC_ERR_INVALID_PARTNER_PARAM	The partner ID is invalid.
332	zTRC_ERR_EMPTY_COMMISSION	A currency symbol has been specified but the commission amount (commission_fix) is missing.
333	zTRC_ERR_INVALID_CURRENCY_SYMBOL	The specified currency symbol is invalid.
335	zTRC_ERR_INVALID_CID	The category ID is invalid for the programme or tracking type (PPS/PPL).
382	zTRC_ERR_LOG_PPS	An internal database error occurred.

5. GLOSSARY

Abbreviation	Meaning
CSV	Comma Separated Value Text file in which data in a row are separated by a delimiter (e.g. comma or semicolon) and which is used for the display of a table
ERP	Enterprise Resource Planning Software solutions that control and / or evaluate the course of business
ERP Client	Client for the data exchange of tracking information between zanox and the advertiser's system.
ERP Web Services	Web Services that enable the automated exchange of tracking information between zanox and the advertiser's system.
GUID	Global Unique IDentifier World-wide unique identification number
SOAP	Simple Object Access Protocol Protocol for the exchange of information in disseminated systems
SSL	Secure Socket Layer Encryption procedure that serves for secure communication between webhost and client
Web service	Disseminated application that uses World Wide Web technologies for communication
WSDL	Web Service Definition Language XML-based language for the description of Web Services
XML	eXtensible Markup Language Extensible markup language for the description of data contents or structures

zanox.

zanox AG | Stralauer Allee 2 | 10245 Berlin | Deutschland

+49 (0)30 509691-0 | +49 (0)30 509691-99 | info@zanox.com

www.zanox.de | blog.zanox.com | help.zanox.com