



**dpd**

# Info Services

## Table of contents

1	Introduction.....	3
1.1	Purpose of the document.....	3
1.2	Glossary .....	3
1.3	Definition of permissions and channels.....	3
1.4	Service address.....	4
1.5	Safety standard.....	4
2	Business description.....	4
2.1	Order of performed operations.....	4
2.2	Limitation on amount of transferred data .....	5
2.3	General description of interfaces and differences between them .....	5
2.3.1	DPDInfoServicesXmlEvents and DPDInfoServicesObjEvents interfaces .....	5
3	Description of service structure .....	8
3.1	Authorisation.....	9
3.1.1	DPDInfoServicesObjEvents i DPDInfoServicesXmlEvents .....	9
3.2	Interface method description.....	9
3.2.1	DPDInfoServicesObjEvents interface .....	10
3.2.2	DPDInfoServicesXmlEvents interface .....	24
3.3	Error codes .....	38
4	Returns and redirection .....	38

# 1 Introduction

## 1.1 Purpose of the document

The purpose of this document is to describe the mode of operation and usage of WebService Type interfaces for communication between the clients' systems and DPD systems.

The following services are available:

- Downloading the event log for parcels
- Confirmation of collection of the event log
- Collection of the client signature

Each service is authorised by login and password.

The client using the presented solution to integrate own systems with DPD Polska systems is ensured as to the compliance of prepared packages with applied standards, which improves the capacity and reliability of our services.

## 1.2 Glossary

Term	Definition
Waybill number	Number by which a specific parcel is identified
Event	An event which took place in connection of the parcel or package life cycle (e.g. dispatch or delivery).
Package	A package is an element of a parcel. The package has specified dimensions (length, width, height and weight). It is identified by the waybill number.

## 1.3 Definition of permissions and channels

Full configuration of the customer account authorized to use the webservice services takes place on the DPD Polska systems side after prior arrangements with the customer. The client retrieves events of the parcels generated with its assigned customer numbers. The information channel is defined by the payer's number (as a parameter of the API call) and a list of subordinate customer numbers on which the shipments were generated. One account can have a number of information channels. One information channel can be assigned to no more than one account.

There is also a method available which does not use a channel (getEventsForWaybill). Once the waybill number is provided, the client receives a list of events (or the last event) for the parcel with the specified number.

## 1.4 Service address

Interfaces are available at:

Interface	PROD
DPDInfoServicesObjEvents	<a href="https://dpdinfoservices.dpd.com.pl/DPDInfoServicesObjEventsService/DPDInfoServicesObjEvents?wsdl">https://dpdinfoservices.dpd.com.pl/DPDInfoServicesObjEventsService/DPDInfoServicesObjEvents?wsdl</a>
DPDInfoServicesXmlEvents	<a href="https://dpdinfoservices.dpd.com.pl/DPDInfoServicesXmlEventsService/DPDInfoServicesXmlEvents?wsdl">https://dpdinfoservices.dpd.com.pl/DPDInfoServicesXmlEventsService/DPDInfoServicesXmlEvents?wsdl</a>

## 1.5 Safety standard

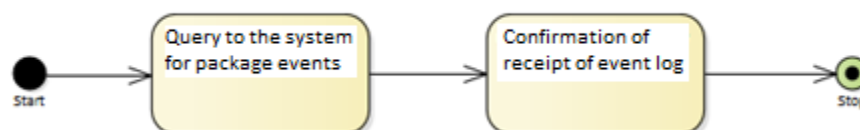
The communication is secured by SSL with 128-bite asymmetric key. Each call is secured with login and password. Client permission levels (recorded in LDAP) are also checked.

# 2 Business description

INFOServices is a universal (independent of the system type) tool for transfer of information between the client's systems and the DPD system. The tool is based on WebService Type interfaces provided to DPD clients.

## 2.1 Order of performed operations

The diagram below presents the order of operation calls within the business process.



Correct use of webservice requires the client to perform two subsequent steps:

1. Query the system for event log for packages in the client channel (in getEventsForWaybill – without including the channel)
2. Confirmation of receipt of event log.

If the user fails to perform the second step, the next attempt of query for package events (first step) will result in the same data being sent.

## 2.2 Limitation on amount of transferred data

The amount of data contained in the client's response depends on configuration setup. Default minimum data set for returned event is 100 (regardless of the „limit” parameter set up at the time of method call) and the maximum value is 1000.

## 2.3 General description of interfaces and differences between them

INFOServices has four available interfaces. Two of them differ mainly by input parameter transmission and format of returned results:

1. Object interface `DPDInfoServicesObjEvents`, which assumes and returns parameters provided directly (uncoded and uncompressed)
2. Interface `DPDInfoServicesXmlEvents`, where the parameters are coded with base64 (optionally compressed by ZIP, which has an impact on the size of transmitted requests and received replies).

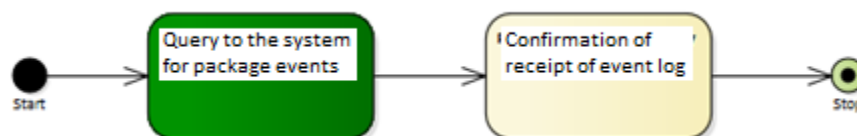
There are two variations of most methods of interface `DPDInfoServicesXmlEvents`:

- Containing prefix „X” in the version number, which receive and return data coded in the Base64 form;
- Containing prefix „C” in the version number, which receive and return data compressed by ZIP and then coded in the Base64 form

### 2.3.1 `DPDInfoServicesXmlEvents` and `DPDInfoServicesObjEvents` interfaces

The description below concerns `DPDPackagesObjServices` and `DPDPackagesXMLServices` interfaces.

Business description concerns both methods, because the difference between them are only of structural nature.



#### 2.3.1.1 Query to the system regarding package events for the client channel

Method name	<code>getEventsForCustomer</code>
-------------	-----------------------------------

<b>Available versions</b>	V1 - V4	CV1 - CV4	XV1 - XV4
---------------------------	---------	-----------	-----------

The method assumes the following input parameters:

<b>Input parameter</b>	<b>Method version</b>
The maximum number of events returned in response	1-4
A two-letter code of event description language	2, 4
Authorisation data	1-4

DPD system performs the following operations:

<b>Operation</b>	<b>Method version</b>
Validation of authorisation data	1-4
Download of event log for packages	1-4

The method returns the following values:

<b>Returned value</b>	<b>Method version</b>
Any authentication and authorisation exceptions	1-4
List of events for the packages for the specific client channel	1-4
Confirmation ID (used for confirmation of receipt of the event log)	1-4

### 2.3.1.2 Query to the system regarding an event log for a package with the specific waybill number

<b>Method name</b>	getEventsForWaybill
--------------------	---------------------

<b>Available versions</b>	V1	XV1
---------------------------	----	-----

The method assumes the following input parameters:

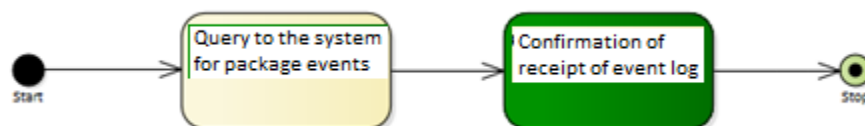
<b>Input parameter</b>
Waybill number
A two-letter code of event description language
Parameter specifying if all events should be returned (ALL) or if only the last ones ( ONLY_LAST)
Authorisation data

DPD system performs the following operations:

<b>Operation</b>
Validation of authorisation data
Download of event log for packages

The method returns the following values:

<b>Returned value</b>
Any authentication and authorisation exceptions
Event log for the package
Receipt ID to be used for confirmation of receipt of the event log



### 2.3.1.3 Confirmation of receipt of the event log

<b>Method name</b>	markEventsAsProcessed
<b>Available versions</b>	V1

The method assumes the following input parameters:

<b>Input parameter</b>
Confirmation ID
Authorisation data

DPD system performs the following operations:

<b>Operation</b>
Marking of the event log corresponding to the receipt ID as received by the client

The method returns the following values:

<b>Returned value</b>
„True” value if the call was successful, „false” in the event of an error

## 3 Description of service structure

The methods are described as follows:

- Signature – character of the method on the webservice side
- Parameter – description of all parameters present in the request structure
- Comments – additional description of parameters
- Query results – description of return information in different variants



## 3.1 Authorisation

### 3.1.1 DPDInfoServicesObjEvents i DPDInfoServicesXmlEvents

In their signature, all methods for INFOServices have a parameter authDataV1 (Typee AuthDataV1) which stores authorisation data for the service. Each recipient should have its own login parameters. AuthDataV1 consists of three elements:

- login (String) – a unique identifier assigned to the recipient by the service provider
- password (String) – password assigned to the login
- channel (String) – information channel to which the user is assigned

Parameter authDataV1 is mandatory. Lack of the parameter results in the message „Authorization failed to secure webservice method”. During validation, the following conditions of its internal structure are checked:

Field name	Mandatory field	Type
login	Yes	String
password	Yes	String
channel	Yes	String

Authentication is performed by the WebService AuthServices. AuthServices receives the instance of AuthenticationDataV1 class which contains the data provided by the user. WebService returns the response in the form of enumeration instance AuthenticationResponseStatusEnum. If the authorisation is unsuccessful (the response is different than „OK”), the following xml is returned:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <S:Fault xmlns:ns4="http://www.w3.org/2003/05/soap-envelope">
      <faultcode>S:Server</faultcode>
      <faultstring>Access denied to secured webserwis method</faultstring>
      <detail>
        <ns2:exception
class="pl.com.dpd.wscommon.exceptions.DeniedAccessWSException" note="To
disable this feature, set
com.sun.xml.ws.fault.SOAPFaultBuilder.disableCaptureStackTrace system
property to false" xmlns:ns2="http://jax-ws.dev.java.net/">
          <message>Access denied to secured webserwis method</message>
          <ns2:stackTrace>
            ...
          </ns2:stackTrace>
        </ns2:exception>
      </detail>
    </S:Fault>
  </S:Body>
</S:Envelope>
```

## 3.2 Interface method description

The service consists of two main interfaces containing the methods which serve the same business. The difference between DPDInfoServicesXmlEvents and DPDInfoServicesObjEvents Fieldga interfaces consists in the mode of formatting and transmitting information. Detailed description of all interfaces and their methods is provided below.

### 3.2.1 DPDInfoServicesObjEvents interface

#### 3.2.1.1 Query to the system regarding event log for packages in the client channel

<b>Method name</b>	<b>getEventsForCustomerV1</b>
<b>Input signature</b>	CustomerEventsResponseV1 getEventsForCustomerV1( Integer limit, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	CustomerEventsResponseV1( String confirmId, List<CustomerEventV1> eventsList)
<b>Error codes</b>	

#### Input parameters

Field	Type	Mandatory field
limit	Integer	No
authDataV1	AuthDataV1	Yes

#### Query result

XML query structure (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"  
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">  
  <soapenv:Header/>  
  <soapenv:Body>  
    <even:getEventsForCustomerV1>  
      <limit>10 </limit>  
      <authDataV1>  
        <channel>clientChannel</channel>  
        <login>user</login>  
      </authDataV1>  
    </even:getEventsForCustomerV1>  
  </soapenv:Body>  
</soapenv:Envelope>
```

```

        <password>userPassword</password>
    </authDataV1>
</even:getEventsForCustomerV1>
</soapenv:Body>
</soapenv:Envelope>

```

XML response structure for correct query in version V1 (Response):

```

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForCustomerV1Response
      xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
        <confirmId>6/YZj7qK8rzMimm3VRyYjQ==</confirmId>
        <eventsList>
          <businessCode>170101</businessCode>
          <country>PL</country>
          <depot>BIA</depot>
          <description>Parcel handed over to be delivered</description>
          <eventData1>eventData1</eventData1>
          <eventData2>eventData2</eventData2>
          <eventData3>eventData3</eventData3>
          <eventTime>2011-11-16T12:43:41</eventTime>
          <id>14123075</id>
          <objectId>46181630</objectId>
          <operationType>INSERT</operationType>
          <packageReference/>
          <parcelReference>TEL000069708012</parcelReference>
          <waybill>NG148497H090</waybill>
        </eventsList>
        <eventsList>
          <businessCode>170101</businessCode>
          <country>PL</country>
          <depot>KTW</depot>
          <description>Parcel handed over to be delivered</description>
          <eventData1>eventData1</eventData1>
          <eventData2>eventData2</eventData2>
          <eventData3>eventData3</eventData3>
          <eventTime>2011-11-16T12:48:25</eventTime>
          <id>14123116</id>
          <objectId>46183717</objectId>
          <operationType>INSERT</operationType>
          <packageReference/>
          <parcelReference>COMPANY MATERIALS</parcelReference>
          <waybill>01765003987175</waybill>
        </eventsList>
      </return>
    </ns2:getEventsForCustomerV1Response>
  </S:Body>
</S:Envelope>

```

Correctly generated response should be marked as `getEventsForCustomerV1Response` and be `CustomerEventsResponseV1` Type.

**Response data format:**

**CustomerEventsResponseV1:**

Field	Description	Type
ConfirmId	Confirmation ID	String
EventsList	Event log	List<CustomerEventV1>

**CustomerEventV1:**

Field	Description	Type
Id	Confirmation ID	Long
BusinessCode	Event code	String
Waybill	Waybill number	String
Description	Event description	String
EventTime	Time the event occurred	String
EventData1	Additional event information	String
EventData2	Additional event information	String
EventData3	Additional event information	String
Depot	Depot where the event occurred	String
Country	Country where the event occurred	String
PackageReference	Country where the event occurred	String
ParcelReference	Package reference number	String
ObjectId	Event object ID	Long
OperationType	Operation type	String

Event information may be returned with different operation types (OperationType):

- INSERT– new event (full event information is returned)
- CANCEL – cancellation of an event which was sent previously (only an object identification ObjectId is returned). It is recommended that such event is appropriately flagged in the client’s system (e.g. with the “cancelled” flag).

**Comments:**

All available descriptions which may be returned in the Description field are included in the appendix named Zdarzenia Infoervices.xlsx.

<b>Method name</b>	<b>getEventsForCustomerV2</b>
--------------------	-------------------------------

<b>Input signature</b>	CustomerEventsResponseV1 getEventsForCustomerV2 ( Integer limit, String language, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>language</b> – A two-letter code of event description language <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	CustomerEventsResponseV1( String confirmId, List<CustomerEventV1> eventsList)
<b>Error codes</b>	

### Input parameters

Field	Type	Mandatory field
limit	Integer	No
language	String	No
authDataV1	AuthDataV1	Yes

### Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForCustomerV2>
      <limit>10</limit>
      <language>PL</language>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:getEventsForCustomerV2>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for correct query in version V2 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
```

```

<ns2:getEventsForCustomerV2Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
  <return>
    <confirmId>ihtXUs/WAUU=</confirmId>
    <eventsList>
      <businessCode>230403</businessCode>
      <country>616</country>
      <depot>BIA</depot>
      <eventData1>parcel_3</eventData1>
      <eventData2/>
      <eventData3/>
      <eventTime>2014-11-26T11:39:39</eventTime>
      <id>20057</id>
      <objectId>0</objectId>
      <operationType>INSERT</operationType>
      <packageReference>ref</packageReference>
      <parcelReference>data29</parcelReference>
      <waybill>21112014111444</waybill>
    </eventsList>
    <eventsList>
      <businessCode>170301</businessCode>
      <country>616</country>
      <depot>BIA</depot>
      <eventData1/>
      <eventData2/>
      <eventData3/>
      <eventTime>2014-11-25T13:49:00</eventTime>
      <id>20058</id>
      <objectId>208750</objectId>
      <operationType>INSERT</operationType>
      <packageReference>ref</packageReference>
      <parcelReference>data10</parcelReference>
      <waybill>2111201411127K</waybill>
    </eventsList>
  </return>
</ns2:getEventsForCustomerV2Response>
</S:Body>
</S:Envelope>

```

Correctly generated response should be marked as `getEventsForCustomerV2Response` and be `CustomerEventsResponseV1` type.

Response data format is the same as in `getEventsForCustomerV1` method.

<b>Method name</b>	<code>getEventsForCustomerV3</code>
<b>Input signature</b>	<code>CustomerEventsResponseV2 getEventsForCustomerV3( Integer limit, AuthDataV1 authDataV1)</code>

<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	CustomerEventsResponseV2 ( String confirmId, List<CustomerEventV2> eventsList)
<b>Error codes</b>	

### Input parameters

Field	Type	Mandatory field
limit	Integer	No
authDataV1	AuthDataV1	Yes

### Query result

XML query structure (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForCustomerV3>
      <limit>10</limit>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:getEventsForCustomerV3>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for correct query in version V3 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForCustomerV3Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
        <confirmId>ihtXUs/WAUU=</confirmId>
        <eventsList>
          <businessCode>230403</businessCode>
          <country>616</country>
          <depot>BIA</depot>
          <eventDataList>
            <code>01</code>
            <value>&lt;![CDATA[parcel_3]]&gt;</value>
          </eventDataList>
          <eventDataList>
```

```

        <code>02</code>
        <value>&lt;![CDATA[parcel_3]]&gt;</value>
    </eventDataList>
    <eventDataList>
        <code>03</code>
        <value>&lt;![CDATA[parcel_3]]&gt;</value>
    </eventDataList>
    <eventTime>2014-11-26T11:39:39</eventTime>
    <id>20057</id>
    <objectId>0</objectId>
    <operationType>INSERT</operationType>
    <packageReference>ref</packageReference>
    <parcelReference>data29</parcelReference>
    <waybill>21112014111444</waybill>
</eventsList>
<eventsList>
    <businessCode>170301</businessCode>
    <country>616</country>
    <depot>BIA</depot>
    <eventDataList>
        <code>02</code>
        <value>parcel_1</value>
    </eventDataList>
    <eventDataList>
        <code>03</code>
        <value>parcel_1</value>
    </eventDataList>
    <eventTime>2014-11-25T13:49:00</eventTime>
    <id>20058</id>
    <objectId>208750</objectId>
    <operationType>INSERT</operationType>
    <packageReference>ref</packageReference>
    <parcelReference>data10</parcelReference>
    <waybill>2111201411127K</waybill>
</eventsList>
</return>
</ns2:getEventsForCustomerV3Response>
</S:Body>
</S:Envelope>

```

Correctly generated response should be marked as `getEventsForCustomerV3Response` and be `CustomerEventsResponseV2` type.

## Response data format

### CustomerEventsResponseV1:

Field	Description	Type
ConfirmId	Confirmation ID	String
EventsList	Event log	List<CustomerEventV2>

### CustomerEventV2:



Field	Description	Type
Id	Confirmation ID	Long
BusinessCode	Event code	String
Waybill	Waybill number	String
Description	Event description	String
EventTime	Event the time occurred	String
Depot	Depot where the event occurred	String
Country	Country where the event occurred	String
PackageReference	Country where the event occurred	String
ParcelReference	Package reference number	String
ObjectId	Event object ID	Long
EventDataList	Additional event information	List<CustomerEventDataV2>

#### CustomerEventDataV2:

Field	Description	Type
Code	Data code	String
Value	Value	String

Method name	getEventsForCustomerV4
<b>Input signature</b>	CustomerEventsResponseV2 getEventsForCustomerV4 ( Integer limit, String language, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>language</b> – A two-letter code of event description language <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	CustomerEventsResponseV2 ( String confirmId, List<CustomerEventV2> eventsList)

<b>Error codes</b>	
--------------------	--

## Input parameters

Field	Type	Mandatory field
limit	Integer	No
language	String	No
authDataV1	AuthDataV1	Yes

## Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForCustomerV4>
      <limit>10</limit>
      <language>PL</language>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:getEventsForCustomerV4>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for correct query in version V3 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForCustomerV4Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
        <confirmId>ihtXUs/WAUU</confirmId>
        <eventsList>
          <businessCode>230403</businessCode>
          <country>616</country>
          <depot>BIA</depot>
          <eventDataList>
            <code>01</code>
            <value>&lt;![CDATA[parcel_3]]&gt;</value>
          </eventDataList>
          <eventDataList>
            <code>02</code>
            <value>&lt;![CDATA[parcel_3]]&gt;</value>
          </eventDataList>
          <eventDataList>

```

```

        <code>03</code>
        <value>&lt;![CDATA[parcel_3]]&gt;</value>
    </eventDataList>
    <eventTime>2014-11-26T11:39:39</eventTime>
    <id>20057</id>
    <objectId>0</objectId>
    <operationType>INSERT</operationType>
    <packageReference>ref</packageReference>
    <parcelReference>data29</parcelReference>
    <waybill>21112014111444</waybill>
</eventsList>
<eventsList>
    <businessCode>170301</businessCode>
    <country>616</country>
    <depot>BIA</depot>
    <eventDataList>
        <code>02</code>
        <value>parcel_1</value>
    </eventDataList>
    <eventDataList>
        <code>03</code>
        <value>parcel_1</value>
    </eventDataList>
    <eventTime>2014-11-25T13:49:00</eventTime>
    <id>20058</id>
    <objectId>208750</objectId>
    <operationType>INSERT</operationType>
    <packageReference>ref</packageReference>
    <parcelReference>data10</parcelReference>
    <waybill>2111201411127K</waybill>
</eventsList>
</return>
</ns2:getEventsForCustomerV4Response>
</S:Body>
</S:Envelope>

```

Correctly generated response should be marked as `getEventsForCustomerV4Response` and be `CustomerEventsResponseV2` type.

Response data format is the same as for `getEventsForCustomerV3` method.

### 3.2.1.2 System query for the event log in connection with a package with a specified waybill number

Method name	<code>getEventsForWaybillV1</code>
Input signature	<code>CustomerEventsResponseV3 getEventsForWaybillV1(</code> <code>String waybill,</code> <code>EventsSelectTypeEnum eventsSelectType,</code> <code>String language,</code> <code>AuthDataV1 authDataV1)</code>

<b>Input parameters</b>	<b>waybill</b> – Waybill number <b>eventsSelectType</b> – specifies if all events should be returned or only the last ones <b>authDataV1</b> – Authorisation data. Format AuthDataV1 <b>language</b> – Event description language
<b>Output</b>	CustomerEventsResponseV3( String confirmId, List<CustomerEventV3> eventsList)
<b>Error codes</b>	

### Input parameters

Field	Type	Mandatory field	Additional information
waybill	Integer	Yes	
eventsSelectType	EventsSelectTypeEnum	Yes	
language	String	No	PL or EN (EN is assumed if a different value is provided)
authDataV1	AuthDataV1	Yes	

**eventsSelectType** - enumeration which specifies if all events should be returned or only the last ones:

- ALL – all events
- ONLY\_LAST – last events

### Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForWaybillV1>
      <waybill>sampleWaybill</waybill>
      <eventsSelectType>ONLY_LAST</eventsSelectType>
    </even:getEventsForWaybillV1>
  </soapenv:Body>
</soapenv:Envelope>
```

```

        <language>PL</language>
        <authDataV1>
            <channel>clientChannel</channel>
            <login>user</login>
            <password>userPassword</password>
        </authDataV1>
    </even:getEventsForWaybillV1>
</soapenv:Body>
</soapenv:Envelope>

```

XML response structure for correct query in version V1 (Response):

```

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForWaybillV1Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
        <confirmId>0</confirmId>
        <eventsList>
          <businessCode>190101</businessCode>
          <country>PL</country>
          <depot>1305</depot>
          <depotName>Warszawa</depotName>
          <description>Parcel delivered</description>
          <eventDataList>
            <code>03</code>
            <description>Recipient</description>
            <value>test</value>
          </eventDataList>
          <eventTime>2018-02-15T15:52:29.886</eventTime>
          <objectId>7305000000269747964</objectId>
          <packageReference/>
          <parcelReference/>
          <waybill>0000001094090S</waybill>
        </eventsList>
      </return>
    </ns2:getEventsForWaybillV1Response>
  </S:Body>
</S:Envelope>

```

Correctly generated response should be marked as getEventsForWaybillV1Response and be CustomerEventsResponseV2 type.

XML response structure for incorrect query in version V1 i.e. where the waybill has not been filled out by the user:

```

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForWaybillV1Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
        <confirmId>0</confirmId>
      </return>
    </ns2:getEventsForWaybillV1Response>
  </S:Body>
</S:Envelope>

```

```
</S:Body>  
</S:Envelope>
```

## Response data format

### CustomerEventsResponseV3:

Field	Description	Type
ConfirmId	Confirmation ID	String
EventsList	Event log	List<CustomerEventV3>

### CustomerEventV3:

Field	Description	Type
BusinessCode	Event code	String
Waybill	Waybill number	String
Description	Event description	String
EventTime	Time the event occurred	String
Depot	Depot where the event occurred	String
Country	Country where the event occurred	String
PackageReference	Country where the event occurred	String
ParcelReference	Package reference number	String
ObjectId	Event object ID	Long
EventDataList	Additional event information	List<CustomerEventDataV3>

### CustomerEventDataV3:

Field	Description	Type
Code	Data code	String
Value	Value	String
Description	Data description	String

### 3.2.1.3 Confirmation of receipt of event log

Method name	markEventsAsProcessedV1
-------------	-------------------------

<b>Input signature</b>	Boolean markEventsAsProcessedV1 ( String confirmId, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>confirmId</b> – Confirmation ID <b>authDataV1</b> – Data used for user authorisation
<b>Output</b>	Boolean type with "true" value when the method is executed successfully and "false" if it is unsuccessful
<b>Error codes</b>	

### Input parameters

Field	Type	Mandatory field
confirmId	String	Yes

### Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:markEventsAsProcessedV1>
      <confirmId>sampleConfirmId</confirmId>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:markEventsAsProcessedV1>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for correct query in version V1 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:markEventsAsProcessedV1Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>true</return>
    </ns2:markEventsAsProcessedV1Response>
  </S:Body>
```

```
</S:Envelope>
```

XML response structure for incorrect query in version V1 i.e. where confirmId has not been filled out or is incorrect:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:markEventsAsProcessedV1Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>false</return>
    </ns2:markEventsAsProcessedV1Response>
  </S:Body>
</S:Envelope>
```

### 3.2.2 DPDInfoServicesXmlEvents interface

#### 3.2.2.1 Query methods for event log for packages in the client channel

<b>Method name</b>	getEventsForCustomerCV1
<b>Input signature</b>	byte[] getEventsForCustomerCV1( Integer limit, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	XML object (CustomerEventsResponseV1 type) compressed by ZIP and coded with base64
<b>Error codes</b>	

#### Input parameters

Field	Type	Mandatory field
confirmId	String	Yes

#### Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForCustomerCV1>
      <limit>10 </limit>
      <authDataV1>
```



```

        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
    </authDataV1>
</even:getEventsForCustomerCV1>
</soapenv:Body>
</soapenv:Envelope>

```

XML response structure for correct query in version CV1 (Response):

```

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForCustomerCV1Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
UESDBBQACAAIAJKEVEwAAAAAAAAAAAAAAAAAAAAAAQ3VzdG9tZXJFdmVudHNSZXNwb25zZdWTX0/CM
BTFPxFpbzsYkGsT/j0QjRgc4psp46JV2Ja2mPDt7TJ1Q4X44ItJk7bnnNs295fia098viM7eaXMuz
m5Is8c3YHCUZ5tjN1N18o8+fuFY8vBYnGBrNaxKroyzod486CyPgQE5+0YWRkd7p3JyLlRviYlJI+
4RHYk4lIfVma7VQIABICoTFEUIfvQq+sSwsHBLsF0BKdBKAVE2Egq+0qOdZegyq0TWn7IN/9SqWd
QjUM0TBk0wibMRW5V8PpAFm1DA3aZ94eVAc6ZVeQDd7o9EU/0pw2ZC1LSVnaIPumhlz5rlpY1/f3y
uSxjrPVM6U+9JAj+1zjrCCrvcmz5FCQm17fTuZJ8I/U8KqvUE5R6v5ACWIuOfyCkogvz1NqJyD7Ua
/P+S1K/4EO8HN0BO/G7b9AxJq/ip38n29QSwcIvhp3Ez8BAAC6AwAAUESBAHQAFAAIAAgAkoRUTL4
adxM/AQAAugMAABYAAAAAAAAAAAAAAAAAAAAAEN1c3RvbWVvRXZ1bnRzUmVzcG9uc2VQSwUGAAAA
AAEAAQBEEAAAagwEAAAAA
      </return>
    </ns2:getEventsForCustomerCV1Response>
  </S:Body>
</S:Envelope>

```

**Data format of a response uncompressed and decoded from base64 format** is the same as for getEventsForCustomerV1 method.

Example of an archive containing a response decoded from base64 is provided in the appendix named DECODED\_CV1\_RESPONSE.zip

Method name	getEventsForCustomerCV2
Input signature	byte[] getEventsForCustomerCV2 ( Integer limit, String language, AuthDataV1 authDataV1)
Input parameters	<b>limit</b> – The maximum number of events returned in response <b>language</b> – A two-letter code of event description language <b>authDataV1</b> – data used for user authorisation
Output	XML format (CustomerEventsResponseV1 type) compressed by ZIP and coded with base64
Error codes	

## Input parameters

Field	Type	Mandatory field
limit	Integer	No
language	String	No
authDataV1	AuthDataV1	Yes

## Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForCustomerCV2>
      <limit>10</limit>
      <language>PL</language>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:getEventsForCustomerCV2>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for correct query in version CV2 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForCustomerCV2Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
        UEsDBBQACAAIAEmFVEwAAAAAAAAAAAAAAAAAAAAWAAAAQ3VzdG9tZXJFdmVudHNSZXNwb25zZdWWTX0/CM
        BTFPxFpbzsYkGsT/j0QjRgc4psp46JV2Ja2mPDt7TJlQ4X44ItJk7bnnNs295fia098viM7eaXMuz
        m5Is8c3YHCUZ5tjN1N18o8+fuFY8vBYnGBrNaxKroyzod486CyPgQE5+0YWRkd7p3JyLlRviYlJI+
        4RHYk4lIfVma7VQIABICoTFEUIfvQq+sSswsHBLsF0BKdBKAVE2Egq+0qOdZegyq0TWn7IN/9SqwD
        QjUM0TBk0wibMRW5V8PpAFm1DA3aZ94eVAc6ZVeqDd7o9EU/0pw2ZC1LSVnaIPumhlz5rlpYl/f3y
        uSxjrPVM6U+9JAj+1zjrCCrvcmz5FCQml7fTuZJ8I/U8KqvUE5R6v5ACWIuOfyCkogvz1NqJyD7Ua
        /P+S1K/4EO8HN0BO/G7b9AxJq/ip38n29QSwcIvhp3Ez8BAAC6AwAAUESBAhQAFAAIAAgASYVUTL4
        adxM/AQAAugMAABYAAAAAAAAAAAAAAAAAAAAAEN1c3RvbWVvRXZ1bnRzUmVzcG9uc2VQSwUGAAAA
        AAEAAQBEAAAAgWEAAAAA
      </return>
    </ns2:getEventsForCustomerCV2Response>
  </S:Body>
</S:Envelope>
```

Data format of a response uncompressed and decoded from base64 format is the same as for getEventsForCustomerV1 method.

Example of an archive containing a response decoded from base64 is provided in the appendix named DECODED\_CV1\_RESPONSE.zip (it is the same as for getEventsForCustomerCV1).

<b>Method name</b>	<b>getEventsForCustomerCV3</b>
<b>Input signature</b>	byte[] getEventsForCustomerCV3( Integer limit, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	Object in XML format (CustomerEventsResponseV2 type) compressed by ZIP and coded with base64
<b>Error codes</b>	

### Input parameters

Field	Type	Mandatory field
limit	Integer	No
authDataV1	AuthDataV1	Yes

### Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForCustomerCV3>
      <limit>10</limit>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:getEventsForCustomerCV3>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for correct query in version CV3 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForCustomerCV3Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
UESDBBQACAAIAI2FVEwAAAAAAAAAAAAAAAAAAAAWAAAAQ3VzdG9tZXJFdmVudHNSZXNwb25zZb1U207CQ
BT8FeWBN9xLy/2whJsJ0ajBAiaEmAIHrJa26S5E/t4tFZEC4gOabNLdOZMzPbOther73L1YYigd36
uk2BVNXaA39ieON6ukutZ1ppCqCmgspPLnGLaW6CnZQRn4nsQeFxAjt45UCVZUbe8EpzSbB6J3UF9
Ix0MpG/4EBTeoSQ0gOyD07dXIcV3BGWOCm1N/TNMEssFjOcuZ6wa6nGESw3MWYyWjqBeQbRmaGPhK
1Ns1IPEWgv7CU+FK5FgOyOYAD/b4zzZ5hB6cY6s1RhDgFsodqXjhGdwTMbGXzYsTcxeE+wNBW2k5rF
aBo3z22OhaQXTSeo6lbHHAugiP31pZQFv1r5E3Pdhco0q4qXw4azZpVGWRr6WdjOEzPVB1IzND8g9
2OS/C/1zDOJEEsxpG9zB0LYeFACFmeGpHBj0PI8zc/hzBrMaNkFkuU/nsIGT0UwtErjtV69EI+qx1
fwLnzmQjP522ys0X1lw1PB4PsPFW+N3XCufbDeVFPXUn6tW63Eilv8ESH74/eB1BLBwhZCbhTjwEA
ADUFAABQSwECFAAUAAgACACNhrVMWQm4U48BAAA1BQAAFgAAAAAAAAAAAAAAAAAAAAAAQ3VzdG9tZ
XJFdmVudHNSZXNwb25zZVBLBQYAAAAAAQABAEQAAADTAQAAAAA
      </return>
    </ns2:getEventsForCustomerCV3Response>
  </S:Body>
</S:Envelope>
```

**Data format of a response uncompressed and decoded from base64 format** is the same as for getEventsForCustomerV3 method.

Example of an archive containing a response decoded from base64 is provided in the appendix named DECODED\_CV3\_RESPONSE.zip.

<b>Method name</b>	getEventsForCustomerCV4
<b>Input signature</b>	byte[] getEventsForCustomerCV4 ( Integer limit, String language, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>language</b> – A two-letter code of event description language <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	XML object format (CustomerEventsResponseV2 type) compressed by ZIP and coded with base64
<b>Error codes</b>	

### Input parameters

Field	Type	Mandatory field
limit	Integer	No
language	String	No
authDataV1	AuthDataV1	Yes

## Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForCustomerCV4>
      <limit>10</limit>
      <language>PL</language>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:getEventsForCustomerCV4>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for correct query in version CV4 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForCustomerCV4Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
UESDBBQACAAIALmBVEwAAAAAAAAAAAAAAAAAAAAWAAAAQ3VzdG9tZXJFdmVudHNSZXNwb25zZbOxr8jNU
ShLLSrOzM+zVTLUM1BSSM1Lzk/JzEu3VQoNcd01ULK3s3EuLS7Jz00tcilLzSspDkotLsjPK04NM7
KzgYj4ZBaX6AOV5eelZRbleqbYBZS5JxbmJFVqO2fb2ugjxIFsXEYBAFBLBwiok5GDaQAAAIsAAAB
QSwECFAAUAAgACAC5gVRMqJORg2kAAACLAAAAFgAAAAAAAAAAAAAAAAAAAAAAQ3VzdG9tZXJFdmVu
dHNSZXNwb25zZVBLBQYAAAAAAQABAEQAAACtAAAAAA=
      </return>
    </ns2:getEventsForCustomerCV4Response>
  </S:Body>
</S:Envelope>
```

**Data format of a response uncompressed and decoded from base64 format** is the same as for getEventsForCustomerV3 method.

Example of an archive containing a response decoded from base64 is provided in the appendix named DECODED\_CV3\_RESPONSE.zip (it is the same as for getEventsForCustomerCV3).

<b>Method name</b>	<b>getEventsForCustomerXV1</b>
<b>Input signature</b>	byte[] getEventsForCustomerXV1( Integer limit, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	Object in XML format (CustomerEventsResponseV1 type) coded with base64
<b>Error codes</b>	

### Input parameters

Field	Type	Mandatory field
confirmId	String	Yes
authDataV1	AuthDataV1	Yes

### Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForCustomerXV1>
      <limit>10</limit>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:getEventsForCustomerXV1>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for correct query in version XV1 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForCustomerXV1Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
```

```

PEN1c3RvbWVyRXZlbnRzUmVzcG9uc2VWMT48Q29uZmlybUlkPmlodFhVcy9XQVVPVPTwvQ29uZmlybUlkPjxFdmVudHNMaXN0PjxDdXN0b211ckV2ZW50VjE+PElkPjIwMDU3PC9JZD48QnVzaW51c3NDb2RlPjIzMDQwMzwvQnVzaW51c3NDb2RlPjxXYXliaWxsPjIxMTEyMDE0MTEeNDQ0PC9XYXliaWxsPjxFdmVudFRpbWU+MjAxNC0xMS0yNlQxMT0zOT0zOTwvRXZlbnRUaW1lPjxFdmVudERhdGEePnBhcmN1bF8zPC9FdmVudERhdGEePjxFdmVudERhdGEyPjwvRXZlbnREYXRhMj48RXZlbnREYXRhMz48L0V2ZW50RGF0YTM+PERlcG90PkJJQTwwRGVwb3Q+PENvdW50cnk+NjE2PC9Db3VudHJ5PjxQYWNrYWdlUmVmZXJlbmNlPnJlZjwvUGFja2FnZVJlZmV5ZW5jZT48UGFyY2VsUmVmZXJlbmNlPmRhdGEyOTwvUGFyY2VsUmVmZXJlbmNlPjxPYmplY3RJZD4wPC9PYmplY3RJZD48T3B1cmF0aW9uVHlwZT5JT1NFU1Q8L09wZXJhdGlvb1R5cGU+PC9DdXN0b211ckV2ZW50VjE+PEN1c3RvbWVyRXZlbnRWMT48SWQ+MjAwNTg8L0lkPjxCdXNpbmVzc0NvZGU+MTcwMzAxPC9CdXNpbmVzc0NvZGU+PFdheWJpbGw+MjExMTIwMTQxMTEyN0s8L1dheWJpbGw+PEV2ZW50VGltZT4yMDE0LTExLTl1VDEzOjQ5OjAwPC9FdmVudFRpbWU+PEV2ZW50RGF0YTE+PC9FdmVudERhdGEePjxFdmVudERhdGEyPjwvRXZlbnREYXRhMj48RXZlbnREYXRhMz48L0V2ZW50RGF0YTM+PERlcG90PkJJQTwwRGVwb3Q+PENvdW50cnk+NjE2PC9Db3VudHJ5PjxQYWNrYWdlUmVmZXJlbmNlPnJlZjwvUGFja2FnZVJlZmV5ZW5jZT48UGFyY2VsUmVmZXJlbmNlPmRhdGEeMDwvUGFyY2VsUmVmZXJlbmNlPjxPYmplY3RJZD4yMDg3NTA8L09iamVjdElkPjxPcGVyYXRpb25UeXB1Pk1OU0VSVDwvT3B1cmF0aW9uVHlwZT48L0N1c3RvbWVyRXZlbnRWMT48L0V2ZW50c0xpY3Q+PC9DdXN0b211ckV2ZW50c1Jlc3Bvbmlvje+
</return>
</ns2:getEventsForCustomerXV1Response>
</S:Body>
</S:Envelope>

```

Data format of a response decoded from base64 format is the same as for getEventsForCustomerV1 method.

<b>Method name</b>	getEventsForCustomerXV2
<b>Input signature</b>	byte[] getEventsForCustomerXV2 ( Integer limit, String language, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>language</b> – A two-letter code of event description language <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	Object in XML format (CustomerEventsResponseV1 type) coded with base64
<b>Error codes</b>	

### Input parameters

Field	Type	Mandatory field
-------	------	-----------------

limit	Integer	No
language	String	No
authDataV1	AuthDataV1	Yes

## Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForCustomerXV2>
      <limit>10</limit>
      <language>PL</language>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:getEventsForCustomerXV2>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for correct query in version XV2 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForCustomerXV2Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
PEN1c3RvbWVyRXZlbnRzUmVzcG9uc2VWMT48Q29uZmlybUlkPmlodFhVcy9XQVVVPTwvQ29uZmlybUlkPjx
FdmVudHNMaXN0PjxDdXN0b211ckV2ZW50VjE+PElkPjIwMDU3PC9JZD48QnVzaW51c3NDb2
RlPjIzMDQwMzwvQnVzaW51c3NDb2RlPjxXYXliaWxsPjIxMTEyMDE0MTEeNDQ0PC9XYXliaWxsPjx
FdmVudFRpbWU+MjAxNC0xMS0yNlQxMT0zOT0zOTwvRXZlbnRUaW1lPjxmdVudERhdGEePnBhcmNl
bF8zPC9FdmVudERhdGEePjxmdVudERhdGEyPjwvRXZlbnREYXRhMj48RXZlbnREYXRhMz48L0V2Z
W50RGF0YTM+PERlcG90PkJJQTWvRGVwb3Q+PENvdW50cnk+NjE2PC9Db3VudHJ5PjxQYWNrYWdlUm
VmZXJlbnNlPnJlZjwvUGFja2FnZVJlZmV5ZW5jZT48UGFyY2VsUmVmZXJlbnNlPmRhdGEyOTwvUGF
yY2VsUmVmZXJlbnNlPjxPYmplY3RJZD4wPC9PYmplY3RJZD48T3BlcmF0aW9uVHlwZT5JTlNFU1Q8
L09wZXJhdGlvb1R5cGU+PC9DdXN0b211ckV2ZW50VjE+PEN1c3RvbWVyRXZlbnRWMT48SWQ+MjAwN
Tg8L0lkPjxCdXNpbmVzc0NvZGU+MTcwMzAxPC9CdXNpbmVzc0NvZGU+PFdheWJpbGw+MjE2PC9Db3VudHJ5P
jxQYWNrYWdlUmVmZXJlbnNlPnJlZjwvUGFja2FnZVJlZmV5ZW5jZT48UGFyY2VsUmVmZXJlbnNlPm
RhdGEeMDwvUGFyY2VsUmVmZXJlbnNlPjxPYmplY3RJZD4yMDg3NTA8L09iamVjdElkPjxPcGVyYXR
pb25UeXB1Pk1OU0VSVDwvT3BlcmF0aW9uVHlwZT48L0N1c3RvbWVyRXZlbnRWMT48L0V2ZW50c0xp
c3Q+PC9DdXN0b211ckV2ZW50c1Jlc3BvbnNlVjE+
      </return>
    </ns2:getEventsForCustomerXV2Response>
  </S:Body>
</S:Envelope>
```



**Data format of a response decoded from base64 format** is the same as for `getEventsForCustomerV1` method.

<b>Method name</b>	<code>getEventsForCustomerXV3</code>
<b>Input signature</b>	<code>byte[] getEventsForCustomerXV3( Integer limit, AuthDataV1 authDataV1)</code>
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	Object in XML format (CustomerEventsResponseV2 type) coded with base64
<b>Error codes</b>	

### Input parameters

Field	Type	Mandatory field
limit	Integer	No
authDataV1	AuthDataV1	Yes

### Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"  
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">  
  <soapenv:Header/>  
  <soapenv:Body>  
    <even:getEventsForCustomerCV3>  
      <limit>10</limit>  
      <authDataV1>  
        <channel>clientChannel</channel>  
        <login>user</login>  
        <password>userPassword</password>  
      </authDataV1>  
    </even:getEventsForCustomerCV3>  
  </soapenv:Body>  
</soapenv:Envelope>
```

XML response structure for correct query in version XV3 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">  
  <S:Body>  
    <ns2:getEventsForCustomerXV3Response  
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">  
      <return>
```

```

PD94bWwgdmVyc2lvcj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz48Q3VzdG9tZXJFdmVudHNSZXNwb
25zZVYyPjxmdmVudHNMaXN0PjxDdXN0b211ckV2ZW50VjI+PElkPjIwMDU3PC9JZD48QnVzaW51c3
NDb2RlPjIzMDQwMzwvQnVzaW51c3NDb2RlPjxXYXliaWxsPjIxMTEyMDE0MTEeNDQ0PC9XYXliaWxs
sPjxmdmVudFRpbWU+MjAxNC0xMS0yNlQxMT0zOT0zOTwvRXZlbnRUaW1lPjxEZXBvdD5CSUE8L0Rl
cG90PjxDb3VudHJ5PjYxNjwvQ291bnRyeT48UGFja2FnZVJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
WZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
9uVHlwZT5JTlNFU1Q8L09wZXJhdGlvb1R5cGU+PEV2ZW50RGF0YUxpc3Q+PEN1c3RvbWVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
EYXRhVjI+PENvZGU+MDE8L0NvZGU+PFZhbHVlPiZsdDshW0NEQVRBW3BhcmNlbF8zXV0mZ3Q7PC9W
YWx1ZT48L0N1c3RvbWVyaW51c3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
0NvZGU+PFZhbHVlPiZsdDshW0NEQVRBW3BhcmNlbF8zXV0mZ3Q7PC9WYWx1ZT48L0N1c3RvbWVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
ZlbnREYXRhVjI+PEN1c3RvbWVyaW51c3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
hW0NEQVRBW3BhcmNlbF8zXV0mZ3Q7PC9WYWx1ZT48L0N1c3RvbWVyaW51c3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
dERhdGFMaXN0PjwvQ3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
D48QnVzaW51c3NDb2RlPjE3MDMwMTwvQnVzaW51c3NDb2RlPjxXYXliaWxsPjIxMTEyMDE0MTEeNDQ0PC9XYXliaWxs
dLPC9XYXliaWxsPjxmdmVudFRpbWU+MjAxNC0xMS0yNlQxMT0zOT0zOTwvRXZlbnRUaW1lPjxEZXBvdD5CSUE8L0Rl
vdD5CSUE8L0RlcG90PjxDb3VudHJ5PjYxNjwvQ291bnRyeT48UGFja2FnZVJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
L1BhY2thZ2VSZWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
T48T2JqZWN0SWQ+MjAxNC0xMS0yNlQxMT0zOT0zOTwvRXZlbnRUaW1lPjxEZXBvdD5CSUE8L0RlcG90PjxDb3VudHJ5PjYxNjwvQ291bnRyeT48UGFja2FnZVJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
lvc1R5cGU+PEV2ZW50RGF0YUxpc3Q+PEN1c3RvbWVyaW51c3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
+PFZhbHVlPnBhcmNlbF8xPC9WYWx1ZT48L0N1c3RvbWVyaW51c3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
bnREYXRhVjI+PENvZGU+MDM8L0NvZGU+PFZhbHVlPnBhcmNlbF8xPC9WYWx1ZT48L0N1c3RvbWVyaW51c3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
XZlbnREYXRhVjI+PC9FdmVudERhdGFMaXN0PjwvQ3VzdG9tZXJFdmVudHNSZXNwbWZlcmVuY2U+PFBhcmNlbFJlZmVyaW51c3VzdG9tZXJFdmVudHNSZXNwb
48Q29uZmlybUlkPmlodFhVcy9XQVVTwvQ29uZmlybUlkPjwvQ3VzdG9tZXJFdmVudHNSZXNwb25
zZVYyPg==
    </return>
    </ns2:getEventsForCustomerXV3Response>
  </S:Body>
</S:Envelope>

```

Data format of a response decoded from base64 format is the same as for getEventsForCustomerV3 method.

<b>Method name</b>	getEventsForCustomerXV4
<b>Input signature</b>	byte[] getEventsForCustomerXV4 ( Integer limit, String language, AuthDataV1 authDataV1)
<b>Input parameters</b>	<b>limit</b> – The maximum number of events returned in response <b>language</b> – A two-letter code of event description language <b>authDataV1</b> – data used for user authorisation
<b>Output</b>	Object in XML format (CustomerEventsResponseV2 type) coded with base64
<b>Error codes</b>	

## Input parameters



```

vdD5CSUE8L0RlcG90PjxDb3VudHJ5PjYxNjwvQ291bnRyeT48UGFja2FnZVJlZmVyZW5jZT5yZWY8
L1BhY2thZ2VSZWZlcmVuY2U+PFBhcmNlbFJlZmVyZW5jZT5kYXRhMTA8L1BhcmNlbFJlZmVyZW5jZ
T48T2JqZWN0SWQ+MjaA4NzUwPC9PYmplY3RJZD48T3B1cmF0aW9uVHlwZT5JTlNFU1Q8L09wZXJhdG
lvb1R5cGU+PEV2ZW50RGF0YUxpc3Q+PEN1c3RvbWVyRXZ1bnREYXRhVjI+PENvZGU+MDI8L0NvZGU
+PFZhbHVlPnBhcmNlbF8xPC9WYWx1ZT48L0N1c3RvbWVyRXZ1bnREYXRhVjI+PEN1c3RvbWVyRXZ1
bnREYXRhVjI+PENvZGU+MDM8L0NvZGU+PFZhbHVlPnBhcmNlbF8xPC9WYWx1ZT48L0N1c3RvbWVyR
XZ1bnREYXRhVjI+PC9FdmVudERhdGFMaXN0PjwvQ3VzdG9tZXJFdmVudFYyPjwvRXZ1bnRzTG1zdD
48Q29uZmlybUlkPmlodFhVcy9XQVVVPTwvQ29uZmlybUlkPjwvQ3VzdG9tZXJFdmVudHNSZXNwb25
zZVYyPg==
    </return>
  </ns2:getEventsForCustomerXV4Response>
</S:Body>
</S:Envelope>

```

Data format of a response decoded from base64 format is the same as for getEventsForCustomerV3 method.

### 3.2.2.2 Query to the system regarding an event log for a package with the specific waybill number

Method name	getEventsForWaybillXV1
Input signature	byte[] getEventsForWaybillXV1( String waybill, EventsSelectTypeEnum eventsSelectType, String language, AuthDataV1 authDataV1)
Input parameters	<b>waybill</b> – Waybill number <b>eventsSelectType</b> – specifies if all events should be returned or only the last ones <b>language</b> – Event description language <b>authDataV1</b> – Authorisation data. Format AuthDataV1
Output	Object in XML format (CustomerEventsResponseV3 type) coded with base64
Error codes	

#### Input parameters

Field	Type	Mandatory field	Additional information
waybill	Integer	Yes	
eventsSelectType	EventsSelectTypeEnum	Yes	

language	String	No	PL or EN (EN is assumed if a different value is provided)
authDataV1	AuthDataV1	Yes	

**eventsSelectType** - enumeration which specifies if all events should be returned or only the last ones:

- ALL – all events
- ONLY\_LAST – last events

## Query result

Query structure XML (Request):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://events.dpdinfoservices.dpd.com.pl/">
  <soapenv:Header/>
  <soapenv:Body>
    <even:getEventsForWaybillXV1>
      <waybill>sampleWaybill</waybill>
      <eventsSelectType>ONLY_LAST</eventsSelectType>
      <language>PL</language>
      <authDataV1>
        <channel>clientChannel</channel>
        <login>user</login>
        <password>userPassword</password>
      </authDataV1>
    </even:getEventsForWaybillXV1>
  </soapenv:Body>
</soapenv:Envelope>
```

XML response structure for the correct query in XV1 (Response):

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:getEventsForWaybillXV1Response
xmlns:ns2="http://events.dpdinfoservices.dpd.com.pl/">
      <return>
PD94bWwgdmVyc2lvcj0iMS4wIiB1bmNvZGluZz0iVVRGLTgiPz48Q3VzdG9tZXJFdmVudHNSZXNwb
25zZVYzPjxFdmVudHNMaXN0PjxDdXN0b211ckV2ZW50VjM+PEJlc2luZXNzQ29kZT4xNzAxMDI8L0
Jlc2luZXNzQ29kZT48V2F5YmlsbD4xMzA1OTkwMDAwMDA2NDwvV2F5YmlsbD48RGVzY3JpcHRpb24
+V3lkYW5pZSBwcnplc3nFgmtPIGRvIGRvcSsZY3plbmh1PC9EZjXNjcmlwdGlvbj48RXZlbnRUaW11
PjIwMTgtMDI0MTU0MTU0MTU0MTU0MTU0MTU0MTU0MTU0MTU0MTU0MTU0MTU0MTU0MTU0MTU0MTU0
XBvdE5hbWU+V2Fyc3phd2E8L0RlcG90TmFtZT48Q291bnRyeT5QTdWvQ291bnRyeT48UGFja2FnZV
JlZmVyZW5jZT5SRUYvMS8wMDI5Mi8yMDE3PC9QYWNrYWdlUmVmZXJlbnN1PjxQYXJjZWxSZWZ1cmV
uY2U+UkVGIzAwMDAwMDAwMDAwMDAwMTIzPC9QYXJjZWxSZWZ1cmVuY2U+PE9iamVjdElkPjczMDUw
MDAwMDAyNzU2MjMwNDk8L09iamVjdElkPjxFdmVudERhdGFMaXN0Lz48L0N1c3RvbWVycXZlbnRWM
z48L0V2ZW50c0xpc3Q+PENvbmZpcmlJZD4wPC9Db25maXJtSWQ+PC9DdXN0b211ckV2ZW50c1Jlc3
BvbnN1VjM+
```

```
</return>
</ns2:getEventsForWaybillXV1Response>
</S:Body>
</S:Envelope>
```

**Data format for the response decoded from base64** is the same as for getEventsForWaybillV1 method.

### 3.2.2.3 Confirmation of receipt of event log

**markEventsAsProcessedV1** method is identical to the method included in the description of DPDInfoServicesObjEvents interface.

## 3.3 Error codes

None

# 4 Returns and redirection

Parcel waybill / tracking numbers can change sometimes and in such cases you should track the new parcel number. It can happen in the following situations:

- 230402 - parcel redirected
- 230403, 230408 - parcel to be returned to the sender

The new tracking number is being returned in the <eventDataList><value> element.