Paynet System – Document Web Services (DWS)

Manual for developers

Document history

Version	Date	Modifications
1.0	11.05.2011	First version
1.1	18.05.2011	Spelling errors corrected
1.2	21.11.2012	Hint for limit of 500 added
1.3	17.01.2013	Added/changed new URL
1.4	26.04.2013	Adaptations (wording)
1.5	14.07.2015	More information for document-types
1.6	10.08.2015	Changed ShipmentPriority

Table of contents

1. Introduction

SIX Paynet Ltd offers different interfaces for exchanging documents with the Paynet system, which are intended to meet the various needs of senders and recipients. The Paynet system's Document Web Services (called DWS in the following) is one important interface.

This manual explains the functioning and use of DWS, as well as how you can create a Web service client. While you have also received demo projects in C# and Java along with this manual, you are, of course, free to implement your client in your preferred programming language. Prerequisite to this manual is knowledge of SOAP, XML and Java C #.

1.1 Authentication

Authentication takes place with each request by means of a client certificate, or user name and password in the SOAP message. Please contact us to receive your access data.

1.2 Requirements for C#

A .NET SDK must be installed for an implementation with the .NET framework. Use of the latest SDK is recommended (Version 4.0 of the .NET Framework is the most current as of the publication date of this document). If Visual Studio .NET (2005, 2008 or 2010) is used, then all the tools needed for implementation exist on the machine.

1.3 Requirements for Java

If you would like to develop a client in Java, you need a JDK, Version 1.4 or higher. Our example is based on Java SDK 1.6.0_13. Version 1.4 of the Axis Framework was also used while Netbeans 7.0 is used as the IDE.

There are various frameworks under Java that enable you to implement a Web Service Client. You are free to use your preferred framework, as long as it covers the standards.

1.3.1 Useful tools

The following tools may be useful to you:

- soapUI (http://www.soapui.org). This is an open source tool for the testing of service-oriented architecture.
- Netbeans (www.netbeans.com). This is a development environment for Java, or
- Eclipse (http://www.eclipse.org), which is also a development environment for Java.
- Commons CLI for the simplified parsing of command line arguments for Java programs (http://commons.apache.org)
- Axis 1.4 (http://axis.apache.org/axis/java/releases.html), SOAP implementation for Java programs.

1.4 Definition of term: shipment

We use the term 'shipment' to refer to a file that is exchanged between the Paynet system and your client.

1.5 Symbols used

The following symbols are used in this user manual:

Symbol	Definition
	***Interesting additional information*** Indicates the availability of further information on a topic.
	***Important additional information*** Indicates instructions that must be carefully followed.

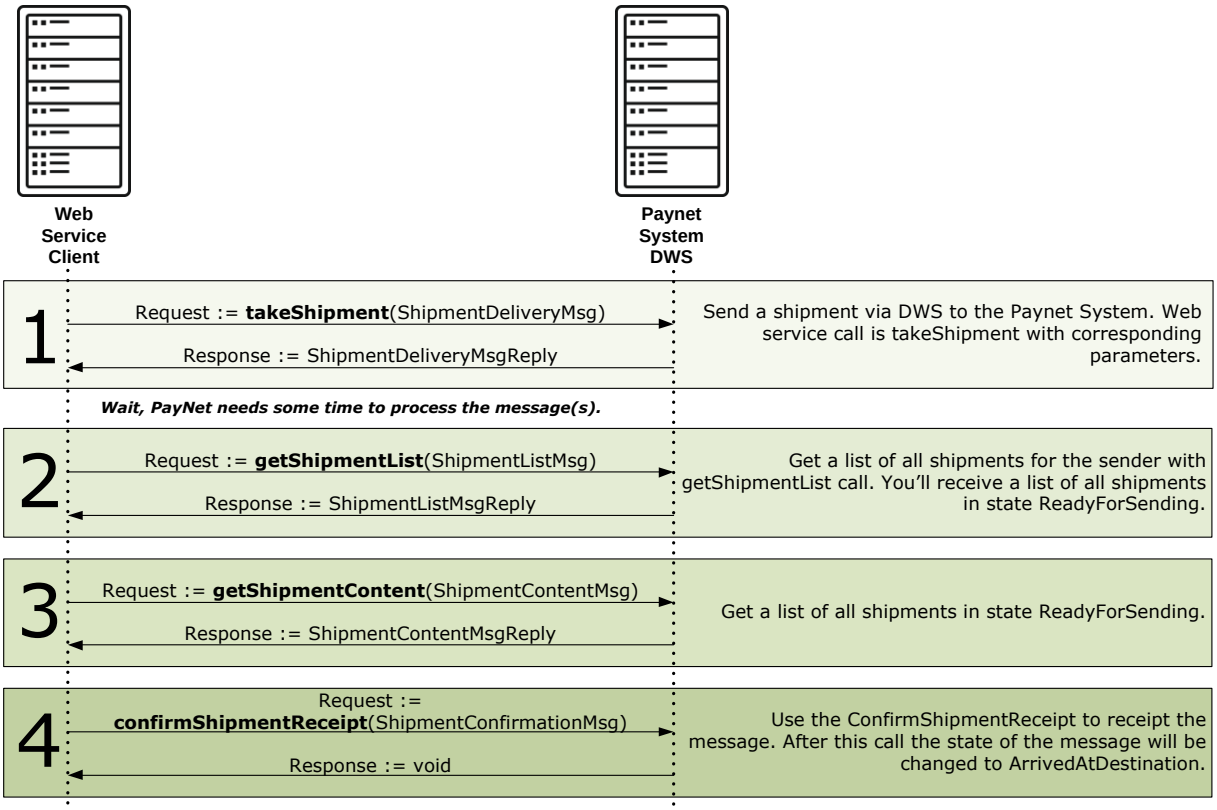
1.6 Sample data

All the sample data used in this user manual (company names, documents, relationships, etc.) are fictitious.

1.7 Disclaimer

While this user manual has been painstakingly created, errors and discrepancies cannot be completely ruled out. SIX Paynet cannot assume legal responsibility or liability for errors in this document, nor for any potential consequences thereof. The information in this user manual is subject to change without prior notifice.

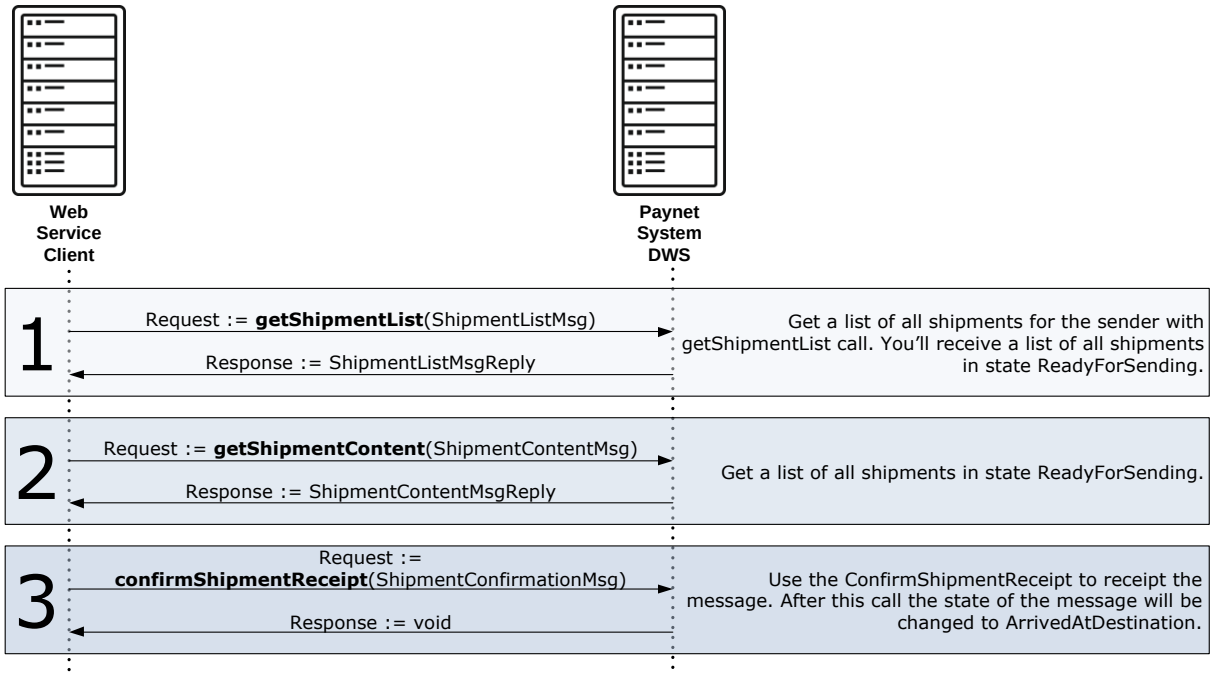
2. Sequential diagram for the sender



As the sender you must implement the four following methods:

- **takeShipment**: Uploads a shipment to the Paynet system. If this method is requested, the shipment is received by the Paynet system and saved for further processing
- **getShipmentList**: Requests a list of shipments that are ready for the sender. After the Paynet system has processed the previously delivered shipment, e.g. receipts are prepared for the sender. All messages that are ready are presented to the sender in this list.
- **getShipmentContent**: The sender goes through the list that he requested previously with getShipmentList; he separately downloads each individual shipment, then saves and processes the contents.
- **confirmShipmentReceipt**: After the shipment has been successfully received and processed, the message recipient confirms the receipt of the message using this method.

3. Sequential diagram for recipients



As the recipient you must implement the three following methods:

- **getShipmentList**: Requests a list of shipments that are ready for the recipient. All messages that are ready are presented in this list.
- **getShipmentContent**: The recipient goes through the list that he requested previously with getShipmentList; he separately downloads each individual shipment, then saves and processes the contents for further processing.
- **confirmShipmentReceipt**: After the shipment has been successfully received and processed, the message recipient confirms the receipt of the message using this method.

4. General information

Each request of a Web service requires an XML message as an entry parameter and provides an XML message as the response.

4.1 Structure of the tables described

The requests and responses are defined in tables in the following sections. The tables contain the following rubrics:

- XML tag
- Format
- Type
- Description

The aggregates, elements and attributes are listed in the XML tag. Aggregates are always marked with a tag and end tag. Elements are marked individually. Attributes are described without tags. In the Type column, the difference is made clearer:

- A = aggregate
- E = element
- Attr. = attribute

The number of tags permitted is also mentioned in the type:

- 1 = only once
- 0..1 = max. one time (XML syntax?)
- 0..n = unlimited (XML syntax: *)
- 1..n = as many as wanted, but at least one must be present (XML syntax: +)

You can find further more detailed information about the formats in the official information from www.w3.org.

5. takeShipment:

A shipment is sent to the Paynet system with this request.

5.1 Request definition

XML tag	Format	Type	Description
<ShipmentDeliveryMsg>	-	A [1]	-
..<Authorization>	-	A [0..1]	Is not needed if you have a client certificate
....<UserName>	String	E [1]	Your user name is issued by SIX Paynet
....<Password>	String	E [1]	Your password is issued by SIX Paynet
..<Client>	-	A [0..1]	**Please do not use this aggregate!**
....<ExternalID>	String	E [1]	Identification
.... <ExternalIDType>	String	E [1]	One of the following values: APPSYSNAME, BC, BCN, BIC, BLZ, BPID, DN, EAN, FTPLUSID, IBAN, MACID, MIME, PATHNAME, PID, SMSNUMBER, STATIONID, UNKNOWN, USERNAME, X400ADDRESS, XISENDERID, XIRECEIVERID
..<ProcessingDate>	Date time	E [0..1]	Preferred processing date; if not provided, the shipment will be processed as quickly as possible.
..<ShipmentPriority>	Integer	E [0..1]	A value between 1 and 9 (default is 5).
..<Content>	Byte[]	E [1]	The content of the file that is being uploaded.
....<encoding>	String	Attr. [0..1]	The encoding information (e.g. UTF-8, ISO-8859-1). Can also be an empty string.
..<Monitoring>	-	A [0..1]	**Please do not use this aggregate!**
....<MessageID>	String	E [1]	Reference to a message
....<MonitorCaseReference>	String	E [1]	Reference to a monitoring object

Example of a minimal SOAP request (without SOAP header and SOAP body):

```
<ShipmentDeliveryMsg>
  <Authorization>
   <UserName>PAYNET USER NAME</UserName>
   <Password>PAYNET PASSWORD</Password>
  </Authorization>
  <Content encoding="">SU5IQUxULURFUi1EQVRFSQ==</Content>
</ShipmentDeliveryMsg>
```

5.2 Response definition

XML tag	Format	Type	Description
<ShipmentDeliveryMsgReply>	-	A [1]	-
..<ShipmentID>	String	E [1]	Shipment identification created by the Paynet system

Example of a minimal response (without SOAP header and SOAP body):

```
<ShipmentDeliveryMsgReply>
  <ShipmentID>SC-99999999999999999999</ShipmentID>
</ShipmentDeliveryMsgReply>
```

6. getShipmentList

The caller requests a list of shipments. A variety of filter criteria can be used for this, such as date range, status and processing priority.

6.1 Request definition

XML tag	Format	Type	Description
<ShipmentListMsg>	-	A [1]	-
..<fromEntry>	Integer unsigned	Attr. [0..1]	Position number as of which the shipments should be retrieved (default is 1).
..<maxEntries>	Integer unsigned	Attr. [0..1]	Maximum number of shipment that should be listed on the list (default is 100).
..<Authorization>	-	A [0..1]	Is not needed if you have a client certificate
....<UserName>	String	E [1]	Your user name is issued by SIX Paynet
....<Password>	String	E [1]	Your password is issued by SIX Paynet
..<FromDate>	Date time	E [0..1]	Start of the date range of the processing. Format: yyyy-mm-ddThh:ssZ
..<ToDate>	Date time	E [0..1]	End of the date range of the processing. Format: yyyy-mm-ddThh:ssZ
..<DocumentIdentifier>	-	A [0..1]	**Please do not use this aggregate!**
....<Name>	String	E [0..1]	Name of the identifier.
....<Type>	String	E [0..1]	Type of identifier. A list of possible values are: • ArchiveMetaData • Bansta • BBXCustomerRegistrationMessage • BBXCustomerRegistrationMessageResponse • Contrl • DocumentSet • DocumentSetReply • EDI Confirmation • Invoice • TransactionJournalReceiver • TransactionJournalSender
....<Format>	String	E [0..1]	The identifier format. A list of possible values are: • XML • MIME • EDIFACT.
....<Version>	String	E [0..1]	The identifier version, e.g. InternalD96A.
....<Encoding>	String	E [0..1]	Encoding (e.g. UTF-8, ISO-8859-1).
....<Extension>	String	E [0..1]	File extension.
....<Category>	String	E [0..1]	Possible values for category are Backpack,

			EdifactStructured, ISF, Unknown, Container, Structured and VerifyLog.
..<ShipmentStates>	-	A [0..1]	-
....<ShipmentState>	String	E [1..3]	The status in which the shipment is currently in. Potential values are ReadyForSending, Submitted and ArrivedAtDestination.
..<FromShipmentPriority>	Integer	E [0..1]	Starting range of the processing priority. Value between 1 and 9.
..<ToShipmentPriority>	Integer	E [0..1]	End range of the processing priority. Value between 1 and 9. Logically larger that the value from FromShipmentPriority.

Example of a minimal SOAP request (without SOAP header and SOAP body):

```
<ShipmentListMsg fromEntry="1" maxEntries="100">
  <Authorization>
   <UserName>PAYNET USER NAME</UserName>
   <Password>PAYNET PASSWORD</Password>
  </Authorization>
</ShipmentListMsg>
```

6.2 Response definition

XML tag	Format	Type	Description
<ShipmentListMsgReply>	-	A [1]	-
..<entriesFound>	Integer unsigned	Attr. [1]	Number of entries in the list. A maximum of 500 entries will be returned. Exist more than 500 Shipments in Paynet system, the Shipments are processed in part to a maximum of 500 Shipments.
..<Shipments>	-	A [0..500]	-
....<ShipmentID>	String	E [1]	Shipment identification created by the Paynet system
....<DocumentIdentifier>	-	A [0..1]	
......<Name>	String	E [0..1]	Name of the identifier.
......<Type>	String	E [0..1]	Type of identifier. Possible values are Bansta, Contrl and DocumentSet.
......<Format>	String	E [0..1]	The identifier format. Possible values are XML, MIME; EDIFACT.
......<Version>	String	E [0..1]	The identifier version, e.g. InternalD96A.
......<Encoding>	String	E [0..1]	Encoding (e.g. UTF-8, ISO-8859-1).
......<Extension>	String	E [0..1]	File extension
......<Category>	String	E [0..1]	Possible values for category are Backpack, EdifactStructured, ISF, Unknown, Container, Structured and VerifyLog.
....<SignatureIdentifier>	-	A [0..1]	-
......<Name>	String	E [0..1]	Name of the identifier.
......<Type>	String	E [0..1]	Type of identifier. Possible values are Bansta, Contrl and DocumentSet.
......<Extension>	String	E [0..1]	File extension
....<CreationDate>	Date time	E [1]	Shipment creation date.
....<ShipmentState>	String	E [1]	The status in which the shipment is currently in. Potential values are ReadyForSending, Submitted and ArrivedAtDestination.
....<SenderID>	String	E [1]	The sender’s identification.
....<ReceiverID>	String	E [1]	The recipient’s identification.
....<ContentSize>	Integer	E [1]	Size of the shipment in bytes.

....<ShipmentPriority>	Integer	E [1]	Processing priority in the Paynet system. A value between 1 and 9.

Example of a minimal response (without SOAP header and SOAP body):

```
<ShipmentListMsgReply entriesFound="1">
  <Shipment>
   <ShipmentID>SC-9999999999999999999</ShipmentID>
   <DocumentIdentifier>
     <Name>MIME.MIME.100</Name>
     <Type>MIME</Type>
     <Format>MIME</Format>
     <Version>1.0</Version>
     <Extension>mime</Extension>
     <Category>Container</Category>
   </DocumentIdentifier>
   <SignatureIdentifier>
     <Name>PKCS7</Name>
     <Type>PKCS7</Type>
     <Extension>p7m</Extension>
   </SignatureIdentifier>
   <CreationDate>2011-01-01T00:00:00Z</CreationDate>
   <ShipmentState>ReadyForSending</ShipmentState>
   <SenderID>41010100000000000</SenderID>
   <ReceiverID>PAYNET-USERNAME</ReceiverID>
   <ContentSize>114461</ContentSize>
   <ShipmentPriority>5</ShipmentPriority>
  </Shipment>
</ShipmentListMsgReply>
```

You can see in the response that a shipment with the status `ReadyForSending` is ready. This is a `MIME` container[1] that was created on `01/01/2001`. The container was signed (`PKCS7` enveloping) and is `114461 Bytes` in size.
The client will next carry out the request getShipmentContent. To do so, the client needs the `ShipmentID` from the list.

[1] See http://www.ietf.org/rfc/rfc2045.txt

7. getShipmentContent

The requester requests a shipment.

7.1 Request definition

XML tag	Format	Type	Description
<ShipmentContentMsg>	-	A [1]	-
..<Authorization>	-	A [0..1]	Is not needed if you have a client certificate
....<UserName>	String	E [1]	Your user name is issued by SIX Paynet
....<Password>	String	E [1]	Your password is issued by SIX Paynet
..<ShipmentID>	String	E [1]	Shipment identification created by the Paynet system

Example of a minimal SOAP request (without SOAP header and SOAP body):

```
<ShipmentContentMsg>
  <Authorization>
   <UserName>PAYNET USER NAME</UserName>
   <Password>PAYNET PASSWORD</Password>
  </Authorization>
  <ShipmentID>SC-9999999999999999999</ShipmentID>
</ShipmentContentMsg>
```

Response definition

XML tag	Format	Type	Description
<ShipmentContentMsgReply>	-	A [1]	-
..<Content>	Byte[]	E [1]	The content of the file that is being uploaded.
....<encoding>	String	Attr. [0..1]	The encoding information (e.g. UTF-8, ISO-8859-1). Can also be an empty string.
..<ShipmentPriority>	Integer	E [0..1]	A value between 1 and 9 (default is 5).

Example of a minimal response (without SOAP header and SOAP body):

```
<ShipmentContentMsgReply>
      <Content>MIKazwYJKoZIhvcNAQcCoIKawDCCmrwCAQE...
      .
      .
      ....qNO6MQ=</Content>
      <ShipmentPriority>5</ShipmentPriority>
</ShipmentContentMsgReply>
```

8. confirmShipment

The requester confirms the receipt of a shipment.

8.1 Request definition

XML tag	Format	Type	Description
<ShipmentConfirmationMsg>	-	A [1]	-
..<Authorization>	-	A [0..1]	Is not needed if you have a client certificate.
....<UserName>	String	E [1]	Your user name is issued by SIX Paynet
....<Password>	String	E [1]	Your password is issued by SIX Paynet
..<ShipmentID>	String	E [1]	Shipment identification created by the Paynet system.

Example of a minimal SOAP request (without SOAP header and SOAP body):

```
< ShipmentConfirmationMsg>
  <Authorization>
   <UserName>PAYNET USER NAME</UserName>
   <Password>PAYNET PASSWORD</Password>
  </Authorization>
  <ShipmentID>SC-9999999999999999999</ShipmentID>
</ ShipmentConfirmationMsg>
```

The shipment status changes after this request from Submitted to ArrivedAtDestination.

8.2 Response definition

The Web Service answers with an empty response. The return value of the function is void.

9. Ping

The caller can request the ping method to check the availability of the service. No authentication is necessary for this method. In the process, the requester sends a series of characters of his choosing to DWS, which then repeats the series of characters in a reply back to the caller.

9.1 Request definition

XML tag	Format	Type	Description
<PingMsg>	-	A [1]	-
..<ClientData>	String	E [0..1]	Any series of characters.

Example of a minimal SOAP request (without SOAP header and SOAP body):

```
< PingMsg>
  <ClientData>Hello PayNet!</ClientData>
</ PingMsg>
```

9.2 Response definition

XML tag	Format	Type	Description
<PingMsgReply>	-	A [1]	-
..<State>	String	E [1]	Status is always "OK"
..<TimeStamp>	Date time	E [1]	The server's current time.
..<Version>	String	E [1]	Current version of the DWS, if released.
..<ClientData>	String	E [0..1]	Series of characters sent.

Example of a minimal response (without SOAP header and SOAP body):

```
< PingMsgReply>
  <State>OK</State>
  <Timestamp>2011-05-05T11:09:11.629Z</Timestamp>
  <Version>Not defined</Version>
  <ClientData>Hello PayNet!</ClientData>
</ PingMsgReply>
```

10. Fault behaviour

If a method cannot be completed successfully, the caller receives a FaultMsg type error message. In such a case, the response is structured as follows:

XML tag	Format	Type	Description
<FaultMsg>	-	A [1]	-
..<ErrorCode>	String	E [1]	4-digit numerical
..<ErrorMessage>	String	E [0..1]	Error text

Example of a minimal response (without SOAP header and SOAP body):

```
<Fault>
  <faultcode> env:Server</faultcode>
  <faultstring dxpp::FaultMsg</faultstring>
  <detail>
   <FaultMsg>
     <ErrorCode>3130</ErrorCode>
     <ErrorMessage>"urn:#getShipmentContent": Shipment [SC-00011110000000000xxxx]
für Request-Sender [PAYNET-USERNAME] was not found!
(LogEventID:999999)</ErrorMessage>
   </FaultMsg>
  </detail>
</Fault>
```

10.1 Error code list

Error	Description
0	Request successfully processed
1000	Unknown Error, perhaps a faulty password was used
1500	Technical sender not identified
1501	Technical sender not authorized
3000	Unknown client
3001	Technical sender not authorized for client
3002	Error inserting intermediate document (only XI)
3100	Shipment ID not unique
3110	Technical sender not authorized to confirm shipment
3130	Unknown Shipment ID
3140	Shipment does not have the required state for confirming (allowed: ReadyForSending, InTransmission, Submitted or ArrivedAtDestination)
3150	Empty shipment content
3200	Unknown SenderID in DocumentSetMsg
3201	Technical sender not authorized for SenderID in DocumentSetMsg

11. Web Service URL

Please consider that the communication between Transfer Client and the Paynet system is taken via a secured channel. Therefore it's important that you trust the publisher (root CA) of the server certificate and the corresponding intermediate CA's. Please make sure that these certificates exist s in the target computer and are placed in the right location.

11.1 Test-System

When using Web Services in combination with with username/password, please use the following URL:

- https://dws-test.paynet.ch/DWS/DWS

When using Web Services in combination with Client Certificate, please use the following URL:

- https://dws-test-cert.paynet.ch/DWS/DWS

11.2 Production-System

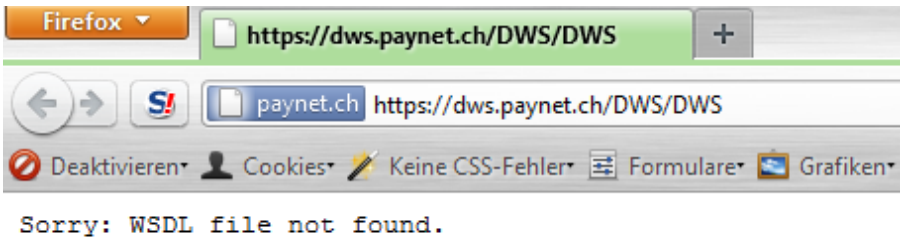
When using Web Services in combination with with username/password, please use the following URL:

- https://dws.paynet.ch/DWS/DWS

When using Web Services in combination with Client Certificate, please use the following URL:

- https://dws-cert.paynet.ch/DWS/DWS

Note: The Web Server is configured so that no WSDL documents can be delivered; should this be attempted, the server will send the following response when you call up the URL:



Web Server certificates

The communication between the client and server is secured using Web server certificates, which means the flow of data between the client and server is encrypted.

The server certificates used are issued by TC TrustCenter GmbH (http://www.trustcenter.de). These are multilevel certificates. The client must trust the entire chain of issuers to be able to successfully establish communication. Otherwise, you will receive a corresponding exception, indicating that the SSL connection could not be established.

The certificate chain that the client must trust (status: May 2011).

TC TrustCenter Class 3 CA II	Root CA	The client must trust this instance…
TC TrustCenter Class 3-II L1 CA IV	Intermediate CA	...and the client must trust this instance…
dws.paynet.ch	Server certificate (production)	…then is server certificate is trustworthy…
dws-test.paynet.ch	Server certificate (test)	…and this server certificate too.

You can also find further information on this topic in the demo projects.

Abbreviations/Glossary

Abbreviation	Description
e-bill	Electronic bill
EDI	Electronic Data Interchange
Paynet system	System from SIX Paynet for the processing of electronic messages (e.g. VAT-compatible electronic bills)
Shipment	A shipment is an object that is transmitted between SIX Paynet Ltd and the Paynet participant.
PKCS#7 enveloping	PKCS#7 enveloping is an internationally standardized signature procedure which wraps around the data to be secure similar to an envelope. This procedure is well-suited for signing any kind of data.
Transport signature	The transport signature ensures the integrity of the data during transmission. It should NOT be confused with the VAT-relevant receipt signature.
SOAP	Simple Object Access Protocol is a network protocol, with which data is exchanged between systems and remote procedure calls can be conducted.
XML	Extensible Mark-up Language is a mark-up language for depicting hierarchal structured data in the form of text data. XML is used for the exchange of data between computer systems, independent of platform and implementation.
Java	Java is an object-orient programming language and registered trademark belonging to Sun Microsystems, which was taken over by Oracle at the end of January 2010. This programming language is a component of Java technology.
C#	C# is a programming language developed by Microsoft within the scope of its .NET strategy. C# is registered as a standard at ECMA and ISO.
DWS	Document Web Service is a Web service provided by SIX Paynet Ltd, which enables communication between the Paynet system and the Paynet participants.

Contact information

SIX Paynet Ltd
Hardturmstrasse 201
P.O. Box
8021 Zurich
Switzerland

www.six-paynet.com

Information and consultation
058 399 9511
paynet-info@six-group.com

Support inquiries
058 399 9577
paynet-support@six-group.com