API Reference Guide

Use PayPal with Web Services

Last Revision: May 26, 2004

PayPal API Reference Guide

Notice of Liability

Table of Contents

Introduction

An introduction to PayPal Web Services

What are PayPal Web Services?

Most PayPal merchants currently use the PayPal website to manage their PayPal transactions. They can additionally use PayPal merchant tools, such as PayPal Shopping Cart and Instant Payment Notification, for more advanced payment functions.

PayPal now extends this flexibility with the introduction of PayPal Web Services. Using an Application Programming Interface (API), merchants can now use Web services technology to create applications that work directly and automatically with PayPal. PayPal API calls can automate certain PayPal functions that normally would require a person to manually enter information.

For example, the PayPal Refund API call allows merchants to automate refunds to buyers. This is especially useful for large merchants who make hundreds of refunds each month.

Why is PayPal offering Web Services?

PayPal has introduced Web services to allow merchants greater flexibility and control when using PayPal for payment transactions. The API takes advantage of available open standards, such as SOAP and WSDL, so that businesses can easily integrate PayPal services into their own transaction framework.

Who can use PayPal Web Services?

PayPal API calls are accessible by qualified Business and Premier accounts.

What can I do with PayPal Web Services?

The following are some applications of PayPal Web Services:

- refund full or partial payments
- search transactions based on date and a number of other search fields
- get details of a particular transaction
- send mass payments to a list of recipients
- withdraw funds from buyers' accounts with their prior permission (available to select merchants)

More functionality will be available in the coming months.

Are there any usage limits for PayPal Web Services?

Currently there are no usage limits for the PayPal Web Services, though PayPal reserves the right to limit usage in the future.

Where can I access the PayPal Web Services WSDL files?

https://api.sandbox.paypal.com/wsdl/PayPalSvc.wsdl

https://api.sandbox.paypal.com/wsdl/ebayTypes.wsdl

Note: To access the above WSDL files, you must have apply, download, and configure a certificate that PayPal provides. To learn how to convert the issued certificate into a .P12 file, see the the "Convert the PEM certificate" section in this guide.

Audience

This guide is aimed at website developers who are familiar with Web services. Because Web services are open standards, you can use the programming language you are most comfortable with, whether it is C# (.NET), Java, or C++. You should be familiar with SOAP, a form of XML.

Document Conventions

This guide uses the following formatting conventions:

Format	Meaning
`monospace`	Names of elements, messages, and classes, as well as snippets of sample code.
italics	Variables that should be replaced with merchant-specific information.

Before using PayPal Web Services

Before creating applications with PayPal Web Services, PayPal recommends testing out the API calls first. The easiest way to get started is to run the sample clients. Sample clients for both Java and .NET are available. Each sample includes a set of instructions for setting up and running the sample. See the **Quick Start** sections of this guide for more information.

Quick Start

Testing PayPal Web Services

In this Quick Start, the Refund API call is used as an example, since it is the most basic API call among PayPal Web Services. These are the steps that apply to each environment:

- **Set up the PayPal Sandbox and test accounts:** See the *API Sandbox User Guide* for information on getting started on the PayPal Sandbox.
- **Request a certificate for testing:** In order to access PayPal API calls, you must request a certificate from PayPal.
- **Download and install the proper SOAP tools and files:** For information specific to your environment, see the appropriate Quick Start section that follows.
- **Send a test payment from a buyer to a seller:** Log into your buyer test account and send a test payment.
- **Perform the Refund API call:** Using the following variables to perform the refund call:
  - Developer username and password
  - certificate file
  - transaction ID of the buyer's payment
- **Confirm that the refund was made:** Log into your seller test account and confirm that the refund was made.

Quick Start Setup for All Environments

Request a Certificate to Use with API Calls

Follow these steps to request a test certificate needed for the API Refund Call:

1. Log into the PayPal Sandbox with your test **Seller** account.
   For more information about getting set up on the PayPal Sandbox environment, see the *API Sandbox User Guide*.
2. Click on the **Profile** subtab.
3. Click on the **API Access** link.
4. Click on the **API Certificate Request** link.
5. Follow the on-screen instructions to complete your certificate request.

You will be emailed a link to download your certificate. Sandbox-related certificates will be sent to your Sandbox-based email and will also be available immediately from the **Test Certificates** tab of Developer Central. For more information about the Sandbox, see the *PayPal API Sandbox User Guide*.

Convert the PEM Certificate to a .P12 Certificate

1. Rename API certicate file from a .txt extension to a .pem extension. This file contains both your private key and certificate.
2. Use a text editor to separate the private key and certificate into two files with a .pem extension.
3. Using a cryptographic tool, convert the test certificate to a PKCS12 (.P12) certificate file. For example, you can use the following command in OpenSSL (http://www.openssl.org/):

```
openssl pkcs12 -export -inkey userkey.pem -in usercert.pem  -out
my_cert.p12
```

- `userkey.pem`: your private key file (separated from the original certificate file provided by PayPal and set with permissions so that only you can read it)
- `usercert.pem`: your certificate file (separated from the original certificate file provided by PayPal)
- `my_cert.p12`: the name of the exported .P12 file

  Note: When you convert the test certificate, you will be asked to enter a password that will be associated with the .P12 file:
  `Enter Export Password:`

  The password can be any string that they want to use.

Quick Start for C# (Microsoft .NET)

Installation Requirements:

- **Microsoft .NET Framework 1.1:**
  The .NET Framework is required on all environments that are testing the API client. For more information see:
  http://msdn.microsoft.com/netframework/technologyinfo/howtoget/
- **Microsoft Visual Studio .NET 2003:**
  - You should have knowledge of installing and using Microsoft Visual Studio .NET 2003

- The PayPal API has only been tested with Microsoft Visual Studio .NET 2003.

- **Access to the PayPal Sandbox environment.**
  For more information, see the *API Sandbox User Guide*.

- **A converted .P12 certificate file:** This file is created when you convert the PayPal-provided PEM certificate. As explained later, the PEM file is exported as a **.cer** certificate, which is used when making the Refund API call.

- **A PayPal Developer username and password:**
  The PayPal Developer username is provided when you apply for an API certificate. You also create your password at that time.

Convert the .P12 Certificate to a .CER certificate

In order for you to use the PayPal-provided certificate with the .NET environment, it must first be converted into a .CER certificate.

1. Double-click the previously created .P12 file. The Certificate Import Wizard will appear.
2. On the **Welcome to the Certificate Import Wizard** screen, click **Next**.
3. On the **File to Import** screen, keep the default setting and click **Next**.
4. On the **Password** screen, type the password provided when you applied for the certificate and click **Next**.
5. On the **Certificate Store** screen, keep the default setting and click **Next**.
6. On the **Completing the Certificate Import Wizard** screen, click **Finish**.
7. When the confirmation dialog box appears, click **OK**.
8. In Internet Explorer, open the **Tools** menu and click **Internet Options...**
9. On the **Content** tab, click **Certificates**.
10. After selecting the .P12 certificate that you use to access the Sandbox, click **Export**.
11. On the **Welcome to Certificate Export Wizard** screen, click **Next**.
12. On the **Export Private Key** screen, keep the default setting and click **Next**.
13. On the **Export File Format** screen, keep the default setting (DER encoded) and click **Next**.
14. On the **File to Export** screen:

    • Click **Browse** and choose the same folder where you will extract the API test page..

- In the **File name:** field, enter a name for the certificate file (for example "certificate").
- Click **Save** to save the file.

15. Click **Next**.
16. On the **Completing the Certificate Export Wizard** screen, click **Finish**.
17. A dialog box will appear saying "The export was successful." The certificate is now successfully exported. Click **OK** and then Finish.

Download and extract the API test package

1. Download the test client package from the Help Center section of PayPal Developer Central:
   https://developer.paypal.com

   Note: A PayPal Developer Central account is required to access PayPal Developer Central.

2. Use a decompression application (such as WinZip) to extract the files in **APIClient.zip** to a folder on your computer.

   Note: Remember the folder where you extract the files in **APIClient.zip**, since you will export your .CER certificate to the same folder. PayPal recommends creating a folder that is close to the root directory on your hard drive. (Example: `c:\APIClient\`)

Run the API executable to see available options

1. From a command prompt (run **cmd** in Windows XP), navigate to the folder containing the newly extracted files.
2. Type **APIClient** and press **Enter** to see a list of available options. Keep the command prompt open, since you will need it later to perform a test refund.

   Note: The transaction ID used to identify the transaction to the **seller** is different from the transaction ID given to the **buyer**.

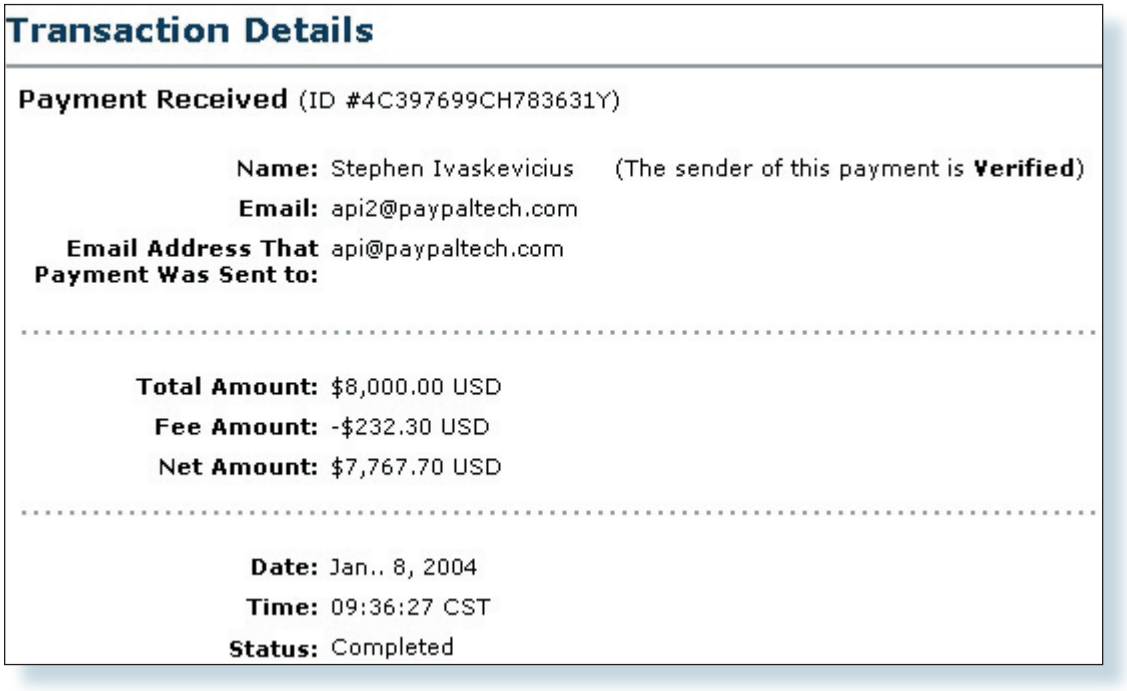
Perform a test payment from a buyer

1. Log into the PayPal Sandbox with your test **buyer** account.
   For more information about getting set up on the PayPal Sandbox environment, see the *API Sandbox User Guide*.

2. Use PayPal to send a test payment to your test **seller** account.
3. Log out of your test buyer account.

Refund the test payment with the API executable

1. To get the Transaction ID number needed for the Refund API, log into the PayPal Sandbox with your test **seller** account.
2. In the **My Account**, click Details next to the newly received test payment.
3. In the **Transaction Details** screen, write down the Transaction ID number, located next to **Payment Received**. In the following example, the Transaction ID number is **#4C397699CH783631Y**:



4. In the command prompt window, type the following command to refund the payment:

```
APIClient RefundTransaction -t [Transaction ID] -c [certificate file] -u [Username] -p [Password]
```

Example:

```
APIClient RefundTransaction -t 4C397699CH783631Y -c certfile.cer -u testusername -p testpassword
```

Important: The username and password you use in this command is different from those used to access the PayPal Sandbox. You must use the username and password provided when you originally requested the certificate.

Tip: You can either pass the username, password and certificate from the command line or directly from the **Class1.cs** file. If you do not pass any information, it will use the information in the **Class1.cs** file by default. The following is a snippet of code from the Class1.cs showing the lines to change the username, password, and certificate name:

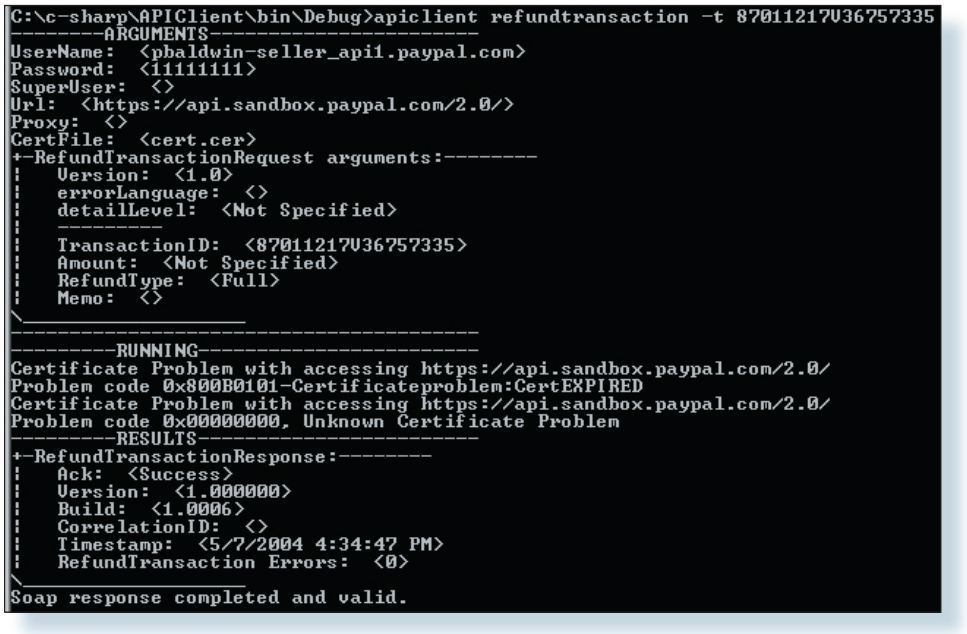
Class1.cs File Snippet

```
// Set up the shared data
ApiArgs.sUserName = "testusername";
ApiArgs.sPassword = "12345678";

ApiArgs.sUrl = "https://api.paypal.com/2.0/";
ApiArgs.sProxy = "";

ApiArgs.sCertFile = "certfile.cer";
```

Notes:

- For testing API calls on the PayPal Sandbox environment, use the following URL in place of the above URL:
  `ApiArgs.sUrl = "https://api.sandbox.paypal.com/2.0/";`
- The Transaction ID number must be entered from the command line, since there is no default available in the Class1.cs file.
- The certificate file has to be in the same directory as the Application file.

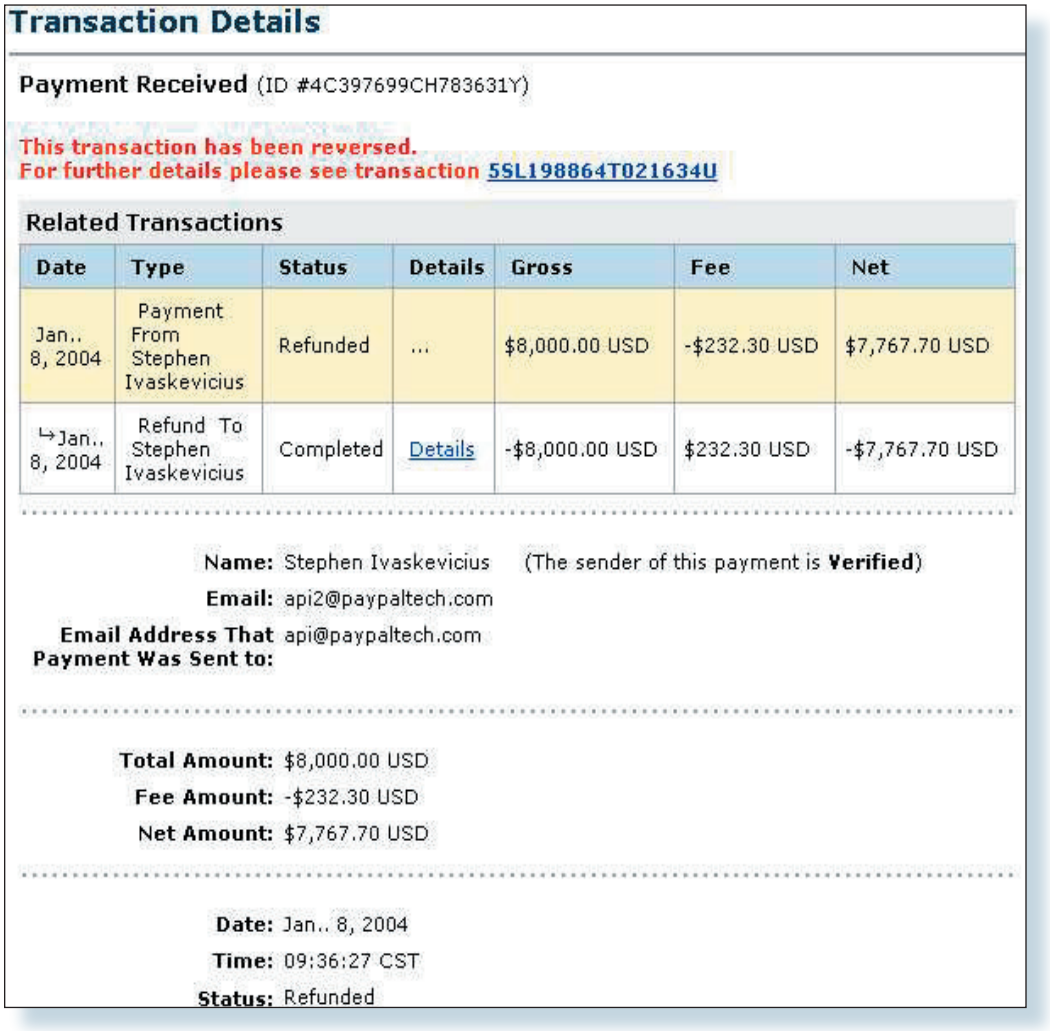
The following is an example of a successful Refund API call:



Confirm the refund

1. To confirm that the refund by the API actually was performed, log into the PayPal Sandbox with your test **seller** account.

2. In the **My Account** tab, click **Details** next to the payment from your test buyer account.
   If the refund is successful, the **Transaction Details** screen should look similar to this:



Transaction Details

Payment Received (ID #4C397699CH783631Y)

This transaction has been reversed.
For further details please see transaction 5SL198864T021634U

Related Transactions

Date	Type	Status	Details	Gross	Fee	Net
Jan.. 8, 2004	Payment From Stephen Ivaskevicius	Refunded	...	$8,000.00 USD	-$232.30 USD	$7,767.70 USD
↳Jan.. 8, 2004	Refund To Stephen Ivaskevicius	Completed	Details	-$8,000.00 USD	$232.30 USD	-$7,767.70 USD

Name: Stephen Ivaskevicius (The sender of this payment is **Verified**)
Email: api2@paypaltech.com
Email Address That Payment Was Sent to: api@paypaltech.com

Total Amount: $8,000.00 USD
Fee Amount: -$232.30 USD
Net Amount: $7,767.70 USD

Date: Jan.. 8, 2004
Time: 09:36:27 CST
Status: Refunded

Additionally, a PayPal email will be sent to your Sandbox-based inbox, and if you have Instant Payment Notification set up, you will receive an IPN notification.

Quick Start for Java

Installation Requirements:

- Java 1.4 or higher
- Axis 1.1
- Access to the PayPal Sandbox environment. For more information, see the *API Sandbox User Guide*.

Download and extract Axis and API test package

1. Download the Axis 1.1 package from the following URL:
   http://apache.mirrors.pair.com/ws/axis/1_1/axis-1_1.tar.gz

   Tip: Use the following command on UNIX compatible systems to download Axis 1.1:
   ```
   $ wget http://apache.mirrors.pair.com/ws/axis/1_1/axis-1_1.tar.gz
   ```

2. Download the test client package at the following URL:

   https://developer.paypal.com

   Note: A PayPal Developer Central account is required to access PayPal Developer Central.

3. Extract both packages to a single directory:

   - In Windows, use a decompression application (such as WinZip) or command to extract the files in **axis-1.1.tar.gz** and **java_client.tar.gz** to a directory on your computer. In this example, the directory is **paypal_api**.
   - In UNIX or Cygwin environments, use the **tar** command . Example:
     ```
     $ tar xzf java_client.tar.gz
     ```

4. Copy the certificate file requested from PayPal to the newly created **java_sample** directory.

   Two files should be present in the directory:
   ```
   axis-1.1.tar.gz
   java_client.tar.gz
   ```

   Two directories should be present in the directory:
   ```
   axis-1_1
   java_sample
   ```

Modify Variables in the Configuration and Script Files

Before making an API call, you edit the configuration and script files so they contain the correct paths and values.

1. Edit the file **client_config.nvp** with the correct values for:

   - `cert_file`: This is the path and name of the certificate file provided by PayPal.
   - `cert_password`: This is the password you created when first exporting the PEM file as a .P12 file. See "Quick Start Setup for All Environments" for more information.
   - `user_name`: This is the username provided by PayPal when you apply for a certificate.
   - `user_passwd`: This is the password you provide when you apply for a certificate.
   - `user_subject`: For third-party access to API access, this is the email address of the account for which you are making API calls.

   Additionally you can specify input parameter values (ie. refund amount, full or partial refund):

   ```
   # testing data for RefundTransaction API
   refund_version = "1.0"
   refund_amount  = "10.00"
   refund_is      = "Partial"    # Full,Partial
   refund_currency_code="USD"
   refund_memo    = ""
   ```

Test the API connection

1. Test API connection with a fake Transaction ID:

   ```
   java RefundTransaction 777
   ```

   Note: If an **invalid transaction id** error occurs, the connection is correctly established. If the connection is not correctly established, errors such as the following may occur:

   ```
   faultString: javax.net.ssl.SSLHandshakeException: Received fatal
   alert: handshake_failure
   ```

Perform a Test Payment from a Buyer

Note: First configure Internet Explorer before logging into the PayPal Sandbox environment. See Quick Start Setup for All Environments for configuration instructions.

1. Log into the PayPal Sandbox with the test **buyer** account.
   For more information about getting set up on the PayPal Sandbox environment, see the *API Sandbox User Guide*.
2. Use PayPal to send a test payment to the test **seller** account.
3. Log out of the test buyer account.

Refund the Test Payment with the Refund API

1. Run the refund API call:

```
java RefundTransaction [transaction-id]
```

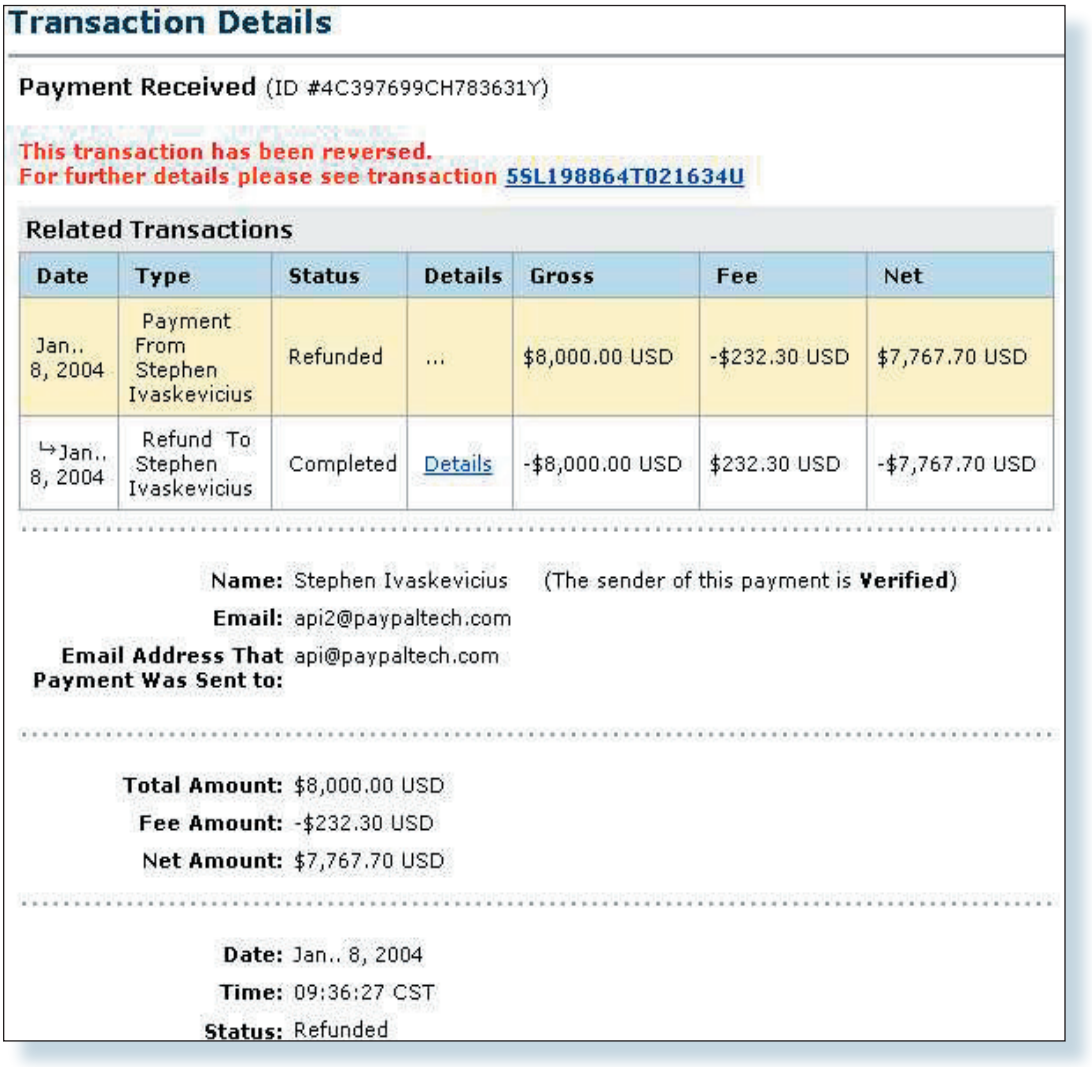
The following is an example of a successful API call:

```
java RefundTransaction 79E856056A569794A
READING "client_config.nvp"
connecting to "https://api.paypal.com/2.0/": testing RefundTransaction("79E856056A569794A")

RefundTransaction Request ---------
transactionID:  79E856056A569794A
refundType:     Full
memo:           auto refund
version:        1.0

RefundTransaction Response ---------
Ack:            Success
TimeStamp:              Mon Apr 26 11:24:32 PDT 2004
Version:                1.000000
Build:          1.0006
```

Confirm the Refund to Ensure Refund API Call Worked

1. To confirm that the refund by the API was actually performed, log into the PayPal Sandbox with the test **seller** account.
2. In the **My Account** tab, click **Details** next to the payment from the test buyer account.

If the refund is successful, the **Transaction Details** screen should look similar to this:



Transaction Details

Payment Received (ID #4C397699CH783631Y)

This transaction has been reversed.
For further details please see transaction 5SL198864T021634U

Related Transactions

Date	Type	Status	Details	Gross	Fee	Net
Jan.. 8, 2004	Payment From Stephen Ivaskevicius	Refunded	...	$8,000.00 USD	-$232.30 USD	$7,767.70 USD
↳Jan.. 8, 2004	Refund To Stephen Ivaskevicius	Completed	Details	-$8,000.00 USD	$232.30 USD	-$7,767.70 USD

Name: Stephen Ivaskevicius (The sender of this payment is **Verified**)
Email: api2@paypaltech.com
Email Address That Payment Was Sent to: api@paypaltech.com

Total Amount: $8,000.00 USD
Fee Amount: -$232.30 USD
Net Amount: $7,767.70 USD

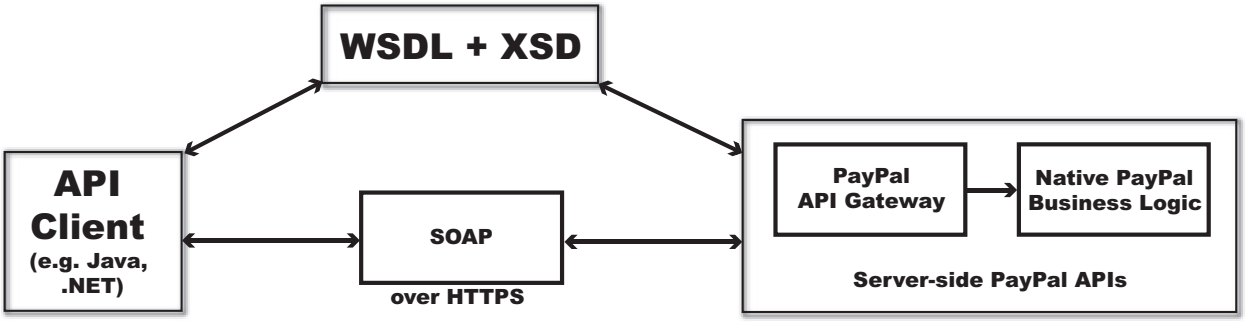
Date: Jan.. 8, 2004
Time: 09:36:27 CST
Status: Refunded

Architecture Overview

The PayPal Web Services Architecture

The PayPal API provides programmatic access to PayPal features via Web services. It enables third-party developers to build custom applications, tools, and services that leverage the PayPal system in new ways. Typical applications include assisting merchants in managing payments on their websites. The power of the PayPal API lies in making these applications possible independently of the PayPal user interface. That is, the Merchant application doesn't need to change every time the PayPal user interface changes, and it doesn't need to present data in the same way PayPal does. A PayPal-enabled application can present data in custom ways that best meet users' needs.

The PayPal API is based on open standards using SOAP and WSDL. These standards are supported by a wide range of development tools on a variety of platforms.



With the PayPal API, the request interface is an object in the application's native programming language. A third-party SOAP client is used to generate business-object interfaces and network stubs from a WSDL document that specifies the PayPal message schema, the service, and other information. The application works with data in the form of object properties, and it sends and receives the data by calling object methods. The SOAP client handles the details of building the SOAP request and sending it to PayPal, and converting the response back to an object that is easy to work with. This frees merchants from the need to build and parse XML documents, so they can focus on managing and presenting the data itself. By simplifying the way merchants access PayPal data, the PayPal API helps them to get applications up and running more quickly and to adapt more easily to changes.

The PayPal API uses the eBay Business Language (eBL) eBay schema model as a foundation, which includes these features:

1. versioning
2. naming standards

3. error handling
4. error codes

The schema design is guided by principles specified in global Web services standards like UBL and ebXML.

In addition, PayPal has added constructs in the new eBay/PayPal schema to help reduce errors due to invalid input. For example:

- When working with the PayPal API, the enumerations are defined directly in the schema.
- With the PayPal API, PayPal returns information about the elements that triggered errors.

Development Environments

PayPal has currently tested the PayPal API using the following client environments:

SOAP Client	Programming Language	Operating Environment
Microsoft .NET 1.1 Framework	C# (Microsoft Visual Studio 2003 .NET)	Microsoft Windows 2000 and Windows XP Professional
Apache Axis 1.1 Final	Java (1.4.X)	Linux, Microsoft Windows 2000 and Windows XP Professional
gSOAP	C++	Unix / Linux

If trying a different SOAP client, please be sure it supports document-style messaging (RPC-style messaging is not used). Also, the PayPal API only supports the UTF-8 encoding scheme.

For information and downloads, please see the following sites.

For Java Development:

J2SE or J2EE 1.4.X
Sun Microsystems, Inc.
http://java.sun.com/downloads/index.html

Apache Axis 1.1 Final
The Apache Software Foundation
http://ws.apache.org/axis/

For .NET Development:

Microsoft .NET Framework Version 1.1 (required for running API clients)
Microsoft Corporation
http://www.microsoft.com/downloads/details.aspx?FamilyID=262d25e3-f589-4842-8157-034d1e7cf3a3&DisplayLang=en

Security

Authentication Errors

PayPal authenticates the requesting user's username and password or authentication token with each message. If the requesting user cannot be authenticated, a SOAP security fault is returned.

Security Keys

The PayPal environment is protected to make sure that only authorized applications use PayPal Web Services. This security scheme uses a set of keys that uniquely identify the application and verify that it is authorized to access PayPal Web Services.

PayPal Web Services uses public key infrastructure, specifically PKCS#7 and Privacy Enhanced Mail (PEM). For more information about PKCS#7 and PEM files, see the following sites:

- **PKCS#7 and PEM:** ftp://ftp.rsasecurity.com/pub/pkcs/ascii/pkcs-7.asc
- **PEM:** http://www.ietf.org/rfc/rfc1421.txt

See the following sites for information about cryptographic tools according to platform:

- **C++:** OpenSSL
  http://www.openssl.org/
- **Java:** J2SE and IAIK-JCE
  http://java.sun.com/j2se/1.4/
  http://jce.iaik.tugraz.at/products/01_jce/
- **.NET:** System.Security.Cryptography
  http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconcryptographicservices.asp

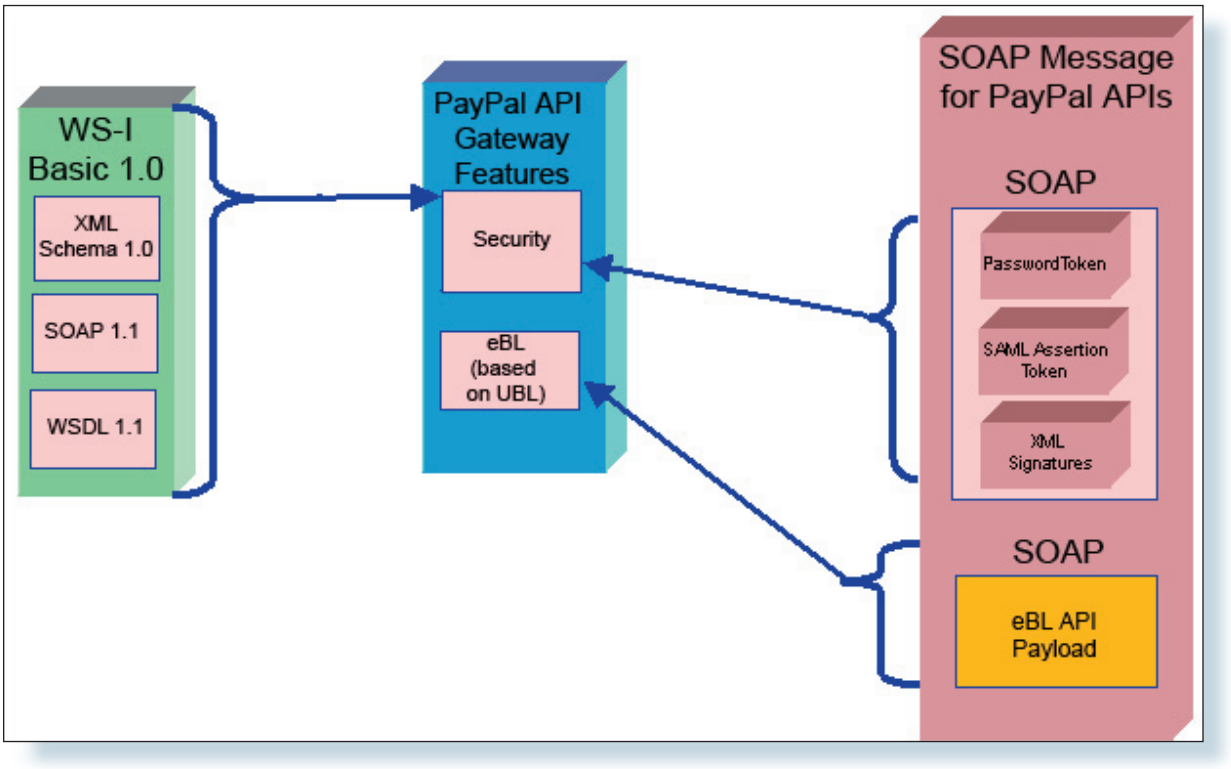
Authentication and Authorization

PayPal needs to verify that merchants are permitted to initiate a transaction before they initiate one. To do this, the merchant must send a username/password in a SOAP envelope. Client certificates are required as part of the SSL handshake. The security token is only returned when authentication succeeds.

The client must use Username and Password to pass the API a username and password combination for authentication. The following is an example of a SOAP request header used with PayPal API calls:

```
<RequesterCredentials xmlns="urn:ebay:api:PayPalAPI"
xsi:type="ebl:CustomSecurityHeaderType">
     <Credentials xmlns="urn:ebay:apis:eBLBaseComponents" xsi:type="ebl:
     UserIdPasswordType">
          <Username xsi:type="xs:string">username</Username>
          <Password xsi:type="xs:string">password</Password>
     </Credentials>
</RequesterCredentials>
```

The following illustrates what needs to go into the SOAP envelope:



The eBL Schema

Components of the eBL Schema

Major Components of the Schema

The eBL schema library leverages existing business component schema models such as Universal Business Language (UBL), ebXML, and EDI to re-use and customize common industry definitions of core business message components to meet specific business needs of eBay applications such as buying, selling, payment, cataloguing, and product search. The PayPal Refund API is built upon the following eBL schema:

- `AbstractRequestType`
- `AbstractResponseType`

To reduce interoperability issues (due to differences in SOAP client implementations for different languages and operating environments), PayPal has made certain design decisions that may affect how merchants design their applications:

- **Polymorphism**: The API limits extension to the request/response type schemas. For example, the RefundTransactionRequestType extends the base AbstractRequestType. In all other cases, polymorphism is avoided. The application will need to handle such business logic (the "or" logic is not exposed in the schema).
- **Null Return Values:** If the value of an optional element is null, The API doesn't return the element in response messages for data-retrieval (Get) calls. If the value is required, PayPal returns the element with a null="true" attribute.
- **Backward Compatibility:** The API uses a versioning system so that the applications will be backward compatible when new elements appear in the server-side schema.
- **Cardinality/Multiplicity**: For base components, the cardinality of major containers is set to minOccurs="0" (but maxOccurs can vary). This will allow us to reuse the same containers across different use cases that might require different combinations of child elements in the future.

Data Types

The API uses the following data types in the schema:

- For numeric data types, the API uses int (32 bit) instead of integer and float instead of decimal (for percentage values).
- The API derives the core monetary amount type from string.
- The API returns time values in GMT/UTC, using the ISO format.

AbstractRequestType

Base type definition of a request payload that can carry any type of payload content with following optional elements:

- The timestamp of the response message.
- An application level acknowledgement.
- Application-level errors and warnings.

Namespace

```
urn:ebay:apis:eBLBaseComponents
```

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

Element	Description	Data Type	Possible Values	Required?
DetailLevel	An integer defining the detail level of the transaction	xs:token	"ReturnAll" = Return all detail levels in response message "ItemReturnDescription" = Return item description "ItemReturnAttributes" = Return attributes as part of the item	No
ErrorLanguage	A string representing the standard RFC 3066 language identification tag, such as en_US. See http://www.faqs.org/rfcs/rfc3066.html for more options for this string	xs:string	"en_US"	No
Version	A string representing the version of the request payload schema	xs:string	1.0	Yes

AbstractRequestType Source

```
<xs:complexType name="AbstractRequestType" abstract="true">
  <xs:annotation>
    <xs:documentation>
      Base type definition of request payload that can carry any type
      of payload content with optional versioning information and
      detail-level requirements.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:any processContents="lax" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="detailLevel" type="xs:token" use="optional"/>
  <xs:attribute name="errorLanguage" type="xs:string" use="optional">
    <xs:annotation>
  <xs:documentation>
        This should be the standard RFC 3066 language identification
        tag, e.g., en_US.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="version" type="xs:string" use="required">
    <xs:annotation>
  <xs:documentation>
        This refers to the version of the request payload schema.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>
```

AbstractResponseType

Base type definition of response payload that can carry any type of payload content with optional versioning information and detail level requirements

Namespace

`urn:ebay:apis:eBLBaseComponents`

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

Element	Description	Possible Values / Data Type
Timestamp	An element of type dateTime that represents the date and time (GMT) the response was generated by a service provider as a result of processing a request	xs:dateTime Transaction-specific
Ack	A token representing the application-level acknowledgement code Contains one the following enumerated values: Success – Request processing succeeded Failure – Request processing failed SuccessWithWarning: Request processing completed successfully, but with some warning information that could be useful for the requesting application to process and/or record. FailureWithWarning: Request processing failed with some error and warning information that the requesting application should process to determine cause(s) of failure. CustomCode: Reserved for internal or future use.	xs:token "Success" "Failure" "SuccessWithWarning" "FailureWithWarning" "CustomCode"
CorrelationId	A string; may be used optionally with an application-level acknowledgement	xs:string Transaction-specific
Errors	Error code of type ErrorType to be used for debugging a response message. It consists of: ShortMessage – string containing the error message LongMessage – string describing the error message ErrorCode – token defining the error code number	xs:string xs:token (ErrorCode) Transaction-specific
Version	A string representing the version of the response payload schema	xs:string "1.0"
Build	A string representing the specific software build used in the deployment for processing the request and generating the response	xs:string

AbstractResponseType Source

```
<xs:complexType name="AbstractResponseType" abstract="true">
  <xs:annotation>
    <xs:documentation>
      Base type definition of a response payload that can carry any
      type of payload content with following optional elements:
        - timestamp of response message,
        - application level acknowledgement, and
        - application-level errors and warnings.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="Timestamp" type="xs:dateTime" minOccurs="0">
      <xs:annotation>
        <xs:documentation>
          This value represents the date and time (GMT) when the
          response was generated by a service provider (as a result of
          processing of a request).
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element ref="ns:Ack">
      <xs:annotation>
        <xs:documentation>
          Application level acknowledgement code.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element ref="ns:CorrelationId" minOccurs="0">
      <xs:annotation>
        <xs:documentation>
          CorrelationId may be used optionally with an application
          level acknowledgement.
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Errors" type="ns:ErrorType" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:any processContents="lax" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="version" type="xs:string" use="required">
    <xs:annotation>
      <xs:documentation>
        This refers to the version of the response payload schema.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="build" type="xs:string" use="required">
    <xs:annotation>
      <xs:documentation>
        This refers to the specific software build that was used in the
        deployment for processing the request and generating the
        response.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>
```

The PayPal Schema

Components of the PayPal Schema

RefundTransactionRequest

Request for a transaction to refund a payment

Namespace

`urn:ebay:api:PayPalAPI`

Extension Of

`ebl:AbstractRequestType`

Used By

`RefundTransactionRequestType`

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

Element	Description	Data Type	Possible Values	Required?
Version	A string representing the version of the response payload schema	xs:string	"1.0"	Yes
TransactionID	Unique identifier for a transaction.	xs:string	Transaction-specific	Yes
RefundType	Type declaration to be used by other schemas. This code identifies the types of refund transactions supported.	xs:token	"Full" "Partial"	No
Amount	Refund amount. Used along with RefundType	xs:string	Transaction-specific	No if RefundType is "Full" Yes if RefundType is "Partial"
Memo	Custom memo regarding the refund.	xs:string	Transaction-specific	No

RefundTransaction .NET Sample Code

```
public void EntireApiCall()

{
// Construct the specific API object
API ApiArgs = new RefundTransactionWrapper(); // Or another API call wrapper

// Set the variables for the call
ApiArgs.sUserName = "username";

…

// Actually make the call
ApiArgs.TryAPICall();
}
```

RefundTransaction Java Sample Code

```
public void runRefundTransaction(String txn_id, String memo, SimpleConfigFile cfg) throws
Exception
{
        PayPalAPIInterface binding = null;
        try {
                binding = new PayPalAPIInterfaceServiceLocator().getPayPalAPI(new URL(url));
                ((PayPalAPISoapBindingStub)binding).setTimeout(60000);
        }
        catch(Exception e) {
        ...
        }
        ...
        RefundTransactionRequestType request = new RefundTransactionRequestType();
        _RefundTransactionReq _req = new _RefundTransactionReq();
        _req.setRefundTransactionRequest(request);

        RefundTransactionResponseType resp = null;
        try {
                resp = binding.refundTransaction(_req);
      }
        catch(Exception e) {
        ...
        }
        ...
}
```

RefundTransactionResponse

Response to the refund transaction request

Namespace

`urn:ebay:api:PayPalAPI`

Extension Of

`ebl:AbstractResponseType`

Used By

`RefundTransactionResponseType`

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

Element	Description	Possible Values / Data Type
Same as AbstractResponseType		

Error Codes

Note: "Long Error" error names are not publicly viewable.

Short Message	Error	Descriptions
User can not refund this transaction	PA_Long_Error_Refund_Permission_Denied	You do not have permission to refund this transaction
Time limit has passed for this refund	PA_Long_Error_Refund_Over_Time_Limit	You are over the time limit to perform a refund on this Transaction
User can not refund this transaction	PA_Long_Error_Refund_Type_Not_Allowed	You can not refund this type of transaction

TransactionSearchRequest

Request to search transaction history.

Namespace

urn:ebay:api:PayPalAPI

Extension Of

ebl:AbstractRequestType

Used By

TransactionSearchType

TransactionSearch .NET Sample Code

```
public void EntireApiCall()
{
// Construct the specific API object
API ApiArgs = new TransactionSearchWrapper(); // Or another API call wrapper

// Set the variables for the call
ApiArgs.sUserName = "username";

…

// Actually make the call
ApiArgs.TryAPICall();
}
```

TransactionSearch Java Sample Code

```
public void runTransactionSearch(SimpleConfigFile cfg) throws Exception
{
        PayPalAPIInterface binding = null;
        try
        {
                binding = new PayPalAPIInterfaceServiceLocator().getPayPalAPI(new URL(url));
                (PayPalAPISoapBindingStub)binding).setTimeout(60000);
        }
        catch(Exception e)
        {
        ...
        }

        TransactionSearchRequestType request = new TransactionSearchRequestType();
        _TransactionSearchReq _req = new _TransactionSearchReq();
        _req.setTransactionSearchRequest(request);

        TransactionSearchResponseType resp = null;
        try
        {
        resp = binding.transactionSearch(_req);
        }
        catch(Exception e)
        {
        ...
        }
        ...
}
```

Note: When using this API call, up to 100 exact matches are returned. No wildcards are allowed, though partial matches are displayed (ie. "Jess" will return results such as "Jessica" and "Jesse"). Also certain search elements (including TransactionID, AuctionItemNumber, and ReceiptID) will ignore other search restrictions.

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

TransactionSearchRequest				
Element	**Description**	**Data Type**	**Possible Values**	**Required?**
StartDate	The earliest transaction date to be returned	xs:dateTime	Transaction-specific	Yes
EndDate	The last transaction date to be returned	xs:dateTime	Transaction-specific	No
Payer	Search by the buyer's email	xs:string	Transaction-specific	No
Receiver	Search by the receiver's email address. If the account has only one email, this is the primary email. Could also be a non-primary email.	xs:string	Transaction-specific	No
ReceiptID	Search by the PayPal Account Optional receipt ID	xs:string	Transaction-specific	No
TransactionID	Search by either the transaction ID of the seller or buyer (returned results will be from seller's perspective	xs:string	Transaction-specific	No

TransactionSearchRequest				
Element	**Description**	**Data Type**	**Possible Values**	**Required?**
PayerName	Search by the buyer's name	xs:string	Transaction-specific	No
	Salutation	xs:string Transaction-specific 20 character limit		
	FirstName	xs:string Transaction-specific 25 character limit		
	MiddleName	xs:string Transaction-specific 25 character limit		
	LastName	xs:string Transaction-specific 25 character limit		
	Suffix	xs:string Transaction-specific 12 character limit		
AuctionItemNumber	Search by auction item number	xs:string	Transaction-specific	No

TransactionSearchRequest				
Element	**Description**	**Data Type**	**Possible Values**	**Required?**
TransactionClass	Search by classifications	xs:token	"All" = search all transaction classes	No
			"Payments" = search all payments	
			"Sent" = search only payments sent	
			"Recieved" = search only payments received	
			"MassPay" = search only mass payments	
			"MoneyRequest" = search only money requests	
			"FundsAdded" = search only funds added to balance	
			"FundsWithdrawn" = search only funds withdrawn from balance	
			"PayPalDebitCard" = search only transactions involving a PayPal debit card	
			"PrimaryDebitCard" = search only transactions involving a primary debit card	
			"SecondaryDebitCard" = search only transactions involving a secondary debit card	
			"Referral" = search only transactions involving a referrals	
			"Fee" = search only transactions involving fees	

TransactionSearchRequest				
Element	**Description**	**Data Type**	**Possible Values**	**Required?**
			"Subscription" = search only transactions involving subscriptions	
			"Dividend" = search only transactions involving dividends	
			"Billpay" = search only transactions involving BillPay Transactions	
			"Refund" = search only transactions involving funds	
			"CurrencyConversions" = search only transactions involving currency conversions	
			"BalanceTransfer" = search only transactions involving balance transfers	
			"Reversal" = search only transactions involving BillPay reversals	
			"Shipping" = search only transactions involving UPS shipping fees	
			"BalanceAffecting" = search only transactions that affect the account balance	
			"ECheck" = search only transactions involving eCheck	
Amount	The transaction amount (already charged to the buyer)	xs:string Numbers and a period decimal separator (".") are allowed.	Transaction-specific	No
CurrencyCode	Search by currency code	xs:token	"USD" "GBP" "EUR" "CAD" "JPY"	No

TransactionSearchRequest				
Element	**Description**	**Data Type**	**Possible Values**	**Required?**
Status	Search payments by status	xs:token	”None” ”Completed” ”Failed” ”Pending” ”Denied” ”Refunded” "Reversed" "Canceled-Reversal" "Processed"	No

TransactionSearchResponse

Response to search transaction history.

Namespace

urn:ebay:api:ppapi:TransactionSearchResponse

Extension Of

ebl:AbstractResponseType

Used By

TransactionSearchType

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

TransactionSearchResponse		
Element	**Description**	**Possible Values / Data Type**
Timestamp	An element of type dateTime that represents the date and time (GMT) the response was generated by a service provider as a result of processing a request	xs:dateTime Transaction-specific
Timezone	The time zone of the transaction	xs:string Transaction-specific
Type	The type of transaction	xs:string Transaction-specific
Payer	The email of the buyer	xs:string Transaction-specific
PayerDisplayName	Display name of the buyer	xs:string Transaction-specific
TransactionID	The transaction ID of the seller	xs:string Transaction-specific

TransactionSearchResponse		
Element	**Description**	**Possible Values / Data Type**
Status	The status of the transaction	xs:string Transaction-specific Example: "completed" "refunded"
GrossAmount	The final amount charged, including any profile shipping and taxes that were appended.	xs:string Transaction-specific
FeeAmount	The PayPal fee that was charged for the transaction	xs:string Transaction-specific
NetAmount	The net amount received by the seller	xs:string Transaction-specific

Error Codes

Note: "Long Error" error names are not publicly viewable.

Short Message	Error	Descriptions
Receipt id is not valid	PA_Long_Error_Receipt_Id_Invalid	The provided Receipt ID is not valid
Auction item id is not valid	PA_Long_Error_Auction_Item_Id_Invalid	The provided Auction Item ID is not valid
End date is invalid	PA_Long_Error_Search_EndDate_Invalid	The provided end date is not valid
Start date is invalid	PA_Long_Error_Search_StartDate_Invalid	The provided start date is not valid
Start date is a required element	PA_Long_Error_Search_StartDate_Missing	The start date has not been provided
You can not search for a transaction id and a receipt id	PA_Long_Error_Search_Multiple_Transaction_Ids	The search cannot contain multiple transaction IDs or a transaction ID and a receipt ID
You do not have permissions to search for this transaction	PA_Long_Error_Search_Permission_Denied	You do not have access to this transaction
Receiver email is invalid	PA_Long_Error_Search_Receiver_Email_Invalid	The provided receiver email is not valid
Receiver can only be specified for payments you've received payments	PA_Long_Error_Search_Receiver_Only_For_Payments_Received	You can use the Receiver element for payments you have received.
The number of results were truncated. Please change your search elements if you wish to see all your results.	PA_Long_Warning_More_Search_Results	Over 100 results were returned. Narrow your search to receive more accurate results.

GetTransactionDetailsRequest

Request for details of a payment (similar to details sent via IPN).

Namespace

urn:ebay:api:ppapi:GetTransactionDetails

Extension Of

ebl:AbstractRequestType

Used By

`GetTransactionDetailsType`

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

Element	Description	Data Type	Possible Values	Required?
TransactionID	Unique identifier for a transaction	xs:string	Transaction-specific	Yes
Version	The version of the request payload schema	xs:string	1.0	Yes

GetTransactionDetails Java Code Sample

```
public void testGetTransactionDetails(String txn_id) throws Exception
{
        PayPalAPIInterface binding = null;
        try {
        binding = new PayPalAPIInterfaceServiceLocator().getPayPalAPI(new URL(url));
        ((PayPalAPISoapBindingStub)binding).setTimeout(60000);
        }
        catch(Exception e)
        {
        ...
        }
        ...

        _GetTransactionDetailsReq _req = new _GetTransactionDetailsReq();
        GetTransactionDetailsRequestType request = new GetTransactionDetailsRequestType();
        _req.setGetTransactionDetailsRequest(request);

        GetTransactionDetailsResponseType resp = null;
        try {
        resp = binding.getTransactionDetails(_req);
        } catch(Exception e) {
        ...
        }
        ...
}
```

GetTransactionDetails .NET Sample Code

```
public void EntireApiCall()

{
// Construct the specific API object
API ApiArgs = new GetTransactionDetailsWrapper(); // Or another API call wrapper

// Set the variables for the call
ApiArgs.sUserName = "username";

…

// Actually make the call
ApiArgs.TryAPICall();
}
```

GetTransactionDetailsResponse

Response of the transaction details request

Namespace

`urn:ebay:api:ppapi:GetTransactionDetails`

Extension Of

ebl:AbstractResponseType

Used By

GetTransactionDetailsType

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

GetTransactionDetailsResponse			
Element	**Child Element**	**Description**	**Possible Values / Data Type**
RecieverInfo	Business	Email address or account ID of the payment recipient (i.e., the merchant). Equivalent to "Receiver" if payment is sent to primary account	xs:string Transaction-specific
	Receiver	Primary email address of the payment recipient (i.e., the merchant). If the payment is sent to a non-primary email address on your PayPal account, the Reciever will still be your primary email	xs:string Transaction-specific
	ReceiverID	Unique account ID of the payment recipient (ie., the merchant). This is the same as the receipient's referral ID	xs:string Transaction-specific
PayerInfo	Payer	Email address or account id of payment sender	Transaction-specific ns:EmailAddressType
	PayerID	Unique customer ID	xs:string Transaction-specific
	PayerStatus	Status of payer's address	xs:token "Verified" "Unverified"

GetTransactionDetailsResponse			
Element	**Child Element**	**Description**	**Possible Values / Data Type**
	Salutation	Payer's salutation	xs:string Transaction-specific 20 character limit
	FirstName	Payer's first name	xs:string Transaction-specific 25 character limit
	MiddleName	Payer's middle name	xs:string Transaction-specific 25 character limit
	LastName	Payer's last name	xs:string Transaction-specific 25 character limit
	Suffix	Payer's suffix	xs:string Transaction-specific 12 character limit
	PayerCountry	Payment sender's country of residence using standard 2-letter ISO 3166 country codes	xs:string Transaction-specific 2 character limit
	PayerBusiness	Payer's business name	xs:string Transaction-specific
	Name	Payer's Address Information	xs:string Transaction-specific
	Street1		xs:string Transaction-specific
	Street2		xs:string Transaction-specific
	CityName		xs:string Transaction-specific

GetTransactionDetailsResponse			
Element	**Child Element**	**Description**	**Possible Values / Data Type**
	StateOrProvince		xs:string Transaction-specific
	Country		xs:token Transaction-specific 2 character limit
	CountryName		xs:string Transaction-specific

GetTransactionDetailsResponse			
Element	**Child Element**	**Description**	**Possible Values / Data Type**
PaymentInfo	TransactionID	The Transaction ID	xs:string Transaction-specific
	ReceiptID	The Receipt ID	xs:string Transaction-specific
	TransactionType	The type of transaction cart: This payment was sent by your customer via the PayPal Shopping Cart feature send_money: This payment was sent by your customer from the PayPal website, using the "Send Money" tab web_accept: The payment was sent by your customer via Buy Now Buttons, Donations, or Auction Smart Logos	xs:token “cart” “send_money” “web_accept”
	PaymentType	The type of payment	"none" "echeck" "instant"
	PaymentDate	Date and time of payment	xs:dateTime Transaction-specific
	GrossAmount	Full amount of the customer's payment, before transaction fee is subtracted.	xs:string Transaction-specific
	FeeAmount	Transaction fee associated with the payment.	xs:string Transaction-specific
	SettleAmount	Amount that is deposited into the account’s primary balance after a currency conversion from automatic conversion through your Payment Receiving Preferences or manual conversion through manually accepting a payment	xs:string Transaction-specific cc:BasicAmountType
	TaxAmount	Amount of tax for transaction	xs:string Transaction-specific
	ExchangeRate	Exchange rate for transaction	xs:string Transaction-specific

GetTransactionDetailsResponse			
Element	**Child Element**	**Description**	**Possible Values / Data Type**
	PaymentStatus	The status of the payment: None: No status Canceled-Reversal: This means a reversal has been canceled (e.g. you, the merchant, won a dispute with the customer and the funds for the transaction that was reversed have been returned to you) Completed: The payment has been completed and the funds have been added successfully to your account balance Denied: You, the merchant, denied the payment. This will only happen if the payment was previously pending due to one of the "pending reasons" below Failed: The payment has failed. This will only happen if the payment was made from your customer's bank account Pending: The payment is pending - see the "pending reason" variable below for more information. Note: You will receive another instant payment notification when the payment becomes "completed", "failed", or "denied" Refunded: You, the merchant, refunded the payment. Reversed: This means that a payment was reversed due to a chargeback or other type of reversal. The funds have been removed from your account balance and returned to the buyer. The reason for the reversal is given by the reason_code variable Processed: This means that a payment was has been accepted, though the funds have not been added successfully to your account.	xs:token "None" "Canceled-Reversal" "Completed" "Denied" "Failed" "Pending" "Refunded" "Reversed" "Processed" Note: This is a non-standard case in which the first letter is capitalized.

GetTransactionDetailsResponse			
Element	**Child Element**	**Description**	**Possible Values / Data Type**
	PendingReason	The reason why the payment is payment, if pending: none: No pending reason address: The payment is pending because your customer did not include a confirmed shipping address and you, the merchant, have your Payment Receiving Preferences set such that you want to manually accept or deny each of these payments. To change your preference, go to the "Preferences" section of your "Profile" echeck: The payment is pending because it was made by an eCheck, which has not yet cleared intl: The payment is pending because you, the merchant, hold a non-U.S. account and do not have a withdrawal mechanism. You must manually accept or deny this payment from your Account Overview multi-currency: You do not have a balance in the currency sent, and you do not have your Payment Receiving Preferences set to automatically convert and accept this payment. You must manually accept or deny this payment unilateral: The payment is pending because it was made to an email address that is not yet registered or confirmed upgrade: The payment is pending because it was made via credit card and you, the merchant, must upgrade your account to Business or Premier status in order to receive the funds. It may also mean that you have reached the monthly limit for transactions on your account verify: The payment is pending because you, the merchant, are not yet verified. You must verify your account before you can accept this payment other: The payment is pending for a reason other than those listed above. For more information, contact customer service.	xs:token "none" "address" "echeck " "intl " "multi-currency" "unilateral " "upgrade" "verify" "other"

GetTransactionDetailsResponse			
Element	**Child Element**	**Description**	**Possible Values / Data Type**
	ReasonCode	The reason for a reversal, if TransactionType is reversal: none: No reason code chargeback: A reversal has occurred on this transaction due to a chargeback by your customer guarantee: A reversal has occurred on this transaction due to your customer triggering a money-back guarantee buyer-complaint: A reversal has occurred on this transaction due to a complaint about the transaction from your customer refund: A reversal has occurred on this transaction because you have given the customer a refund other: A reversal has occurred on this transaction due to a reason not listed above	xs:token "none" "chargeback " "guarantee" "buyer-complaint" "refund" "other"

GetTransactionDetailsResponse			
Element	**Child Element**	**Description**	**Possible Values / Data Type**
PaymentItemInfo	InvoiceID	Invoice number as passed by you, the merchant. Your customer is not able to view or edit this. It must be unique per transaction	xs:string Transaction-specific
	Custom	Custom field as passed by you, the merchant	xs:string Transaction-specific
	Memo	Memo as entered by your customer in PayPal Website Payments note field	xs:string Transaction-specific
	SaleTax	Amount of tax charged on payment	xs:string Transaction-specific
	Name	Item name as passed by you, the merchant. Or, if not passed by you, as entered by your customer. If this is a shopping cart transaction we will append the number of the item (e.g. item_name1, item_name2)	xs:string Transaction-specific
	Number	Item number as passed by you, the merchant. If this is a shopping cart transaction, the number of the item is appended (e.g. item_number1, item_number2)	xs:string Transaction-specific
	Quantity	Quantity as entered by your customer or as passed by you, the merchant. If this is a shopping cart transaction, the number of the item is appended (e.g. quantity1, quantity 2)	xs:string Transaction-specific
	Options	PayPal item options for shopping cart	ns:OptionType xs:string (name) xs:string (value) Transaction-specific
Subscription	SubscriptionID	ID generated by PayPal for the subscriber.	xs:string Transaction-specific

GetTransactionDetailsResponse			
Element	**Child Element**	**Description**	**Possible Values / Data Type**
	SubscriptionDate	Subscription start date	xs:dateTime Transaction-specific
	EffectiveDate	Date when the subscription modification will be effective	xs:dateTime Transaction-specific
	RetryTime	Date PayPal will retry a failed subscription payment	xs:dateTime Transaction-specific
	Username	Username generated by PayPal and given to subscriber to access the subscription.	xs:string Transaction-specific
	Password	Password generated by PayPal and given to subscriber to access the subscription (password will be hashed).	xs:string Transaction-specific
	Recurrences	The number of payment installments that will occur at the regular rate	xs:string Transaction-specific
	Terms (Amount)	The amount subscriber is to be charges in one payment.	xs:string Transaction-specific
	Terms (Period)	The period of time that the subscriber will be charged.	xs:string Transaction-specific
	reattempt	Indicates whether reattempts should occur upon payment failures	xs:string Transaction-specific Example: "1"=Yes
	recurring	Indicates whether regular rate recurs	xs:string Transaction-specific Example: "1"=Yes
Auction	BuyerID	Customer's auction ID	xs:string Transaction-specific
	ClosingDate	Auction's close date.	xs:string Transaction-specific
	multiItem	Counter used for multi-item auction payments	xs:string Transaction-specific

Error Codes

Note: "Long Error" error names are not publicly viewable.

Short Message	Error	Description
Can not get details on this transaction	PA_Long_Error_GetTransaction_Permission_Denied	You do not have permission to get the details of this transaction
Error fetching refund info	PA_Long_Error_GetTransaction_Could_Not_Load	Transaction details could not be loaded

MassPayRequest

Request by a merchant to distribute payments.

Namespace

`urn:ebay:api:PayPalAPI`

Extension Of

`ebl:AbstractRequestType`

Used By

`MassPayRequestType`

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

MassPayRequest				
Element	**Description**	**Data Type**	**Possible Values**	**Required?**
EmailSubject	Subject for the email sent to recipients	xs:string	Transaction-specific	No
ReceiverEmail	Email address of recipients	xs:string	Transaction-specific	Yes
Amount	Amount to be sent to each recipient	xs:string	Transaction-specific	Yes
UniqueID	Unique identifier for each recipient that can be used by merchants for tracking and reconcilation purposes.	xs:string	Transaction-specific	No
Note	Custom note for each recipient	xs:string	Transaction-specific	No

MassPay .NET Sample Code

```
public void EntireApiCall()

{
// Construct the specific API object
API ApiArgs = new MassPayWrapper(); // Or another API call wrapper

// Set the variables for the call
ApiArgs.sUserName = "username";

…

// Actually make the call
ApiArgs.TryAPICall();
}
```

MassPay Java Sample Code

```
public void runMassPay (String num_entries, String file_path, SimpleConfigFile cfg) throws
Exception {

{
        PayPalAPIInterface binding = null;
        try
        {
               binding = new PayPalAPIInterfaceServiceLocator().getPayPalAPI(new URL(url));
               (PayPalAPISoapBindingStub)binding).setTimeout(60000);
        }
        catch(Exception e)
        {
        ...
        }
        MassPayRequestType request = new MassPayRequestType();
             _MassPayReq _req = new _MassPayReq();
              setMassPayReq(num_entries, file_path, cfg, request);
             _req.setMassPayRequest(request);

        PayPalAPI.api.ebay.MassPayResponseType resp = null;
try
        {
               resp = binding.massPay(_req);       }
        catch(Exception e)
        {
        ...
        }
        ...
}
```

MassPayResponse

Response to the request by a merchant to distribute payments.

Namespaces

`urn:ebay:api:PayPalAPI`

Extension Of

`ebl:AbstractResponseType`

Used By

`MassPayResponseType`

Elements

Note: For the most accurate and up-to-date information about API call elements, view the relevant WSDL file.

MassPayResponse		
Element	**Description**	**Possible Values / Data Type**
Status	Status of the transaction	xs:token "Success" = All transactions have been processed Note: "Success" does not mean that any or all the recipients have recieved the payment in their accounts. It only means that all payments were successfully withdrawn from your account. "Failure" = There was an error that prevents processing of the transactions.

Error Codes

Note: "Long Error" error names are not publicly viewable.

Short Message	Error	Description
Invalid argument	PA_Long_Error_MassPay_Receiver_Email_Absent	The receiver's email is missing
Invalid argument	PA_Long_Error_MassPay_Amount_Absent	The amount is missing
Invalid argument	PA_Long_Error_MassPay_Currency_Absent	The currency is missing
Invalid argument	PA_Long_Error_MassPay_Amount_Invalid_Number	The amount is not a valid number
Invalid argument	PA_Long_Error_MassPay_Amount_Max_Limit_Exceeded	The amount exceeds the max limit of a single mass pay item ($10000)
Invalid argument	PA_Long_Error_MassPay_UniqueId_Max_Length_Exceeded	The unique id string length exceeds the maximum limit of 30 characters
Invalid argument	PA_Long_Error_MassPay_UniqueId_Contains_Space	The unique id string contains a space as a character
User not allowed	PA_Long_Error_MassPay_User_Not_Allowed	The user is not allowed to send money through Mass Pay
Account locked	PA_Long_Error_MassPay_Account_Locked	The user account is locked
Unconfirmed email	PA_Long_Error_MassPay_Account_Unconfirmed_Email	The user account has unconfirmed email
Limit Exceeded	PA_Long_Error_MassPay_Account_Limit_Exceeded	The user's account limit has exceeded
Internal Error	PA_Long_Error_MassPay_Account_Limit_Exceeded_Intl	The user's international account limit has exceeded
Receive only account	PA_Long_Error_MassPay_Account_Receive_Only	The user account is receive only and therefore cannot send payments out
Masspay server configuration erro	PA_Long_Error_MassPay_Server_Configuration_Error	There is some configuration error

Masspay server unavailable	PA_Long_Error_MassPay_Server_Unavailable	The mass pay server is unavailable
Unable to create payment	PA_Long_Error_MassPay_Create_Payment	Unable to create payments for masspay
Unable to submit payment	PA_Long_Error_MassPay_Submit_Payment	Unable to submit payments for masspay
Masspay server error	PA_Long_Error_MassPay_Server_Error	The masspay server has reported errors
Insufficient funds	PA_Long_Error_MassPay_Insufficient_Funds	The account does not have sufficient funds to do this masspay
Invalid argument	PA_Long_Error_MassPay_Num_Records_Greater_Than_Max	The number of input records is greater than maximum allowed
Invalid argument	PA_Long_Error_MassPay_Num_Records_Less_Than_Or_Equal_To_Zero	The number of input records is less than or equal to zero
Invalid argument	PA_Long_Error_MassPay_Note_Max_Length_Exceeded	The note string length exceeds the maximum limit of 4000 characters

Error Codes, Messages, and Meanings

Note: "Long Error" error names are not publicly viewable.

The following error codes apply to all API calls:

Number	Description
10001	Unexpected Error
10002	Authentication Error
10003	Missing Required Arguments
10004	Argument Supplied is invalid
10005	Unsupported feature
10006	Version is not supported
10007	Permission denied
10009	Request was refused by the Server
11001	Invalid argument warning
11002	FYI warning.

The following long messages apply to all API calls:

Message	Error	Description
ACH Not Verified	PA_Long_Error_ACH_Not_Verified	You do not have a verified ACH
UserID is not valid	PA_Long_Error_Authentication	Username/Password is incorrect
Not enough permission to make the call	PA_Long_Error_Authentication_Not_Authorized	You do not have permissions to make this API call
Invalid auth token	PA_Long_Error_Authentication_Invalid_Token	Token is not valid
User is locked or Inactive	PA_Long_Error_Counterparty_Locked_Or_Inactive	The account for the counterparty is locked or inactive
Currency not supported	PA_Long_Error_Currency_Invalid	Currency is not valid
Currency not supported	PA_Long_Error_Currency_Not_Supported	Currency is not supported
Invalid characters in the memo	PA_Long_Error_Memo_Invalid	The Memo field contains invalid characters
Soap header is null	PA_Long_Error_Soap_Header_Null	Soap header is NULL
Transaction not found	PA_Long_Error_Transaction_Could_Not_Load	The transaction could not be loaded
Invalid transaction id	PA_Long_Error_Transaction_Id_Invalid	The transaction id is not valid

Message	Error	Description
User is locked or Inactive	PA_Long_Error_User_Locked_Or_Inactive	Account is locked or inactive
User restricted	PA_Long_Error_User_Restricted	Account is restricted
Version not supported	PA_Long_Error_Version_Not_Supported	Version is not supported
Version not valid	PA_Long_Error_Version_Not_Valid	Version is not valid
Could not encrypt the transaction id	PA_Long_Warning_Encryption_Failed	Warning an internal error has occured. The transaction id may not be correct
Memo was truncated due to length	PA_Long_Warning_Memo_Truncated	The memo field was truncated because it was too long
Could not retrieve WS-Security elements	PA_Long_Error_WS_Security	The WS-Security elements cannot be found
Security header is not valid	PA_Long_Error_Security	The security header used is not valid
Invalid account number	PA_Long_Error_Invalid_Account_Number	The account number is invalid

Customer Support Information

Customer support information is available at https://developer.paypal.com. Developer Central offers the following:

- create and view Sandbox accounts
- view test certificates
- read emails tied to test accounts
- view and share information with the PayPal developer community in forums

You can also call your account manager for additional support.

Index