Adyen REST API Manual

(Communication via HTTP Name/Value pairs)

Version 1.05

Adyen B.V.

Table of Contents

Introduction

The purpose of this manual is to enable you to submit requests to the Adyen Payment System using HTTP POST requests (REST Architecture) rather than SOAP requests. In the following chapters we will cover how you can:

- **Authorisations**: Submit payments for authorisation
- **Modifications**: Capture/cancel and refund payments
- **Notifications**: Process notifications of payment state updates and other important events

This document is an addendum to the Adyen Merchant Integration Manual and will reference (without citation) concepts explained there.

The latest version of this document is available by contacting Adyen Support at https://support.adyen.com/. In this Support Suite also various code examples in various programming languages are provided: https://support.adyen.com/links/examples

Audience

This is a technical manual aimed at IT personnel involved in integrating merchants' platforms with those at Adyen.

Changelog

Version	Date	Changes
1.02	2010-07-26	• Added changelog and audience sections • Manual reviewed for English and layout consistency
1.03	2012-09-25	• Added Chapter on Authorisations • Added Chapter on Notification • Added Overview of REST
1.04	2012-11-01	• Various small textual updates and clarifications
1.05	2013-04-23	• Adding information regarding Client Encryption

Why use the Adyen REST API?

While there are various benefits in using SOAP for complex messages instead of the REST communication (HTTP Name/Value pairs), there are some situations in which it makes sense to use HTTP POST to submit requests to Adyen. Some systems are not able to handle SOAP. Or, in some cases, the integration is more convenient with HTTP POST instead of SOAP. This manual describes on how to connect to the Adyen REST API.

What is REST?

The REST methodology has two major components. The first one is that two platforms communicate via the HTTP Name/Value pair protocol. This protocol is very well known, since browsing sites on the world wide web is based on this protocol. E.g. a URL with posibly various parameters.

For example:

https://www.google.com/search?q=adyen

- **https://** indicates the secure HTTP protocol variant
- **www.google.com/search** is the address of the platform (service) with which is communicated
- **?q=adyen** is a variable Name (q) / Value (adyen) pair that lets the service know information about adyen needs to be queried and returned to the requesting service

Next to browsing the web, the HTTP Name/Value pair protocol can also be used for direct server-to-server API communication.

With the first component of REST defined, the second component is a methodology on the state of an object at one of the platforms/services that communicate together. Basically REST assumes that an object can only be in a steady-state (the system in REST). This means that this state is both a potential start-state and a potential end-state. When a communication (request) is received the object can start to transition from one RESTful state to another RESTful state. The possible transitions are limited to a uniform set of actions.

Each transition can be completely understood by the state of the object and the data that is provided in the request.

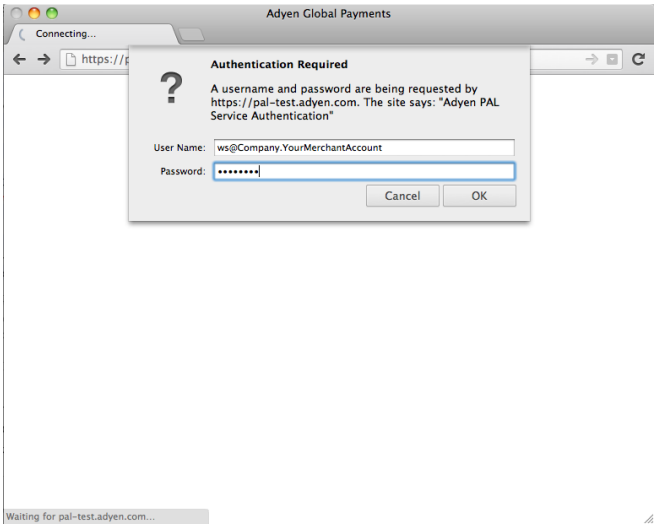
When we look at the Adyen Payment Processing and Modification APIs, Adyen conforms to the REST architecture. There is one simple request which contains all the payment information and the result is an Authorised, Pending or Refused payment.

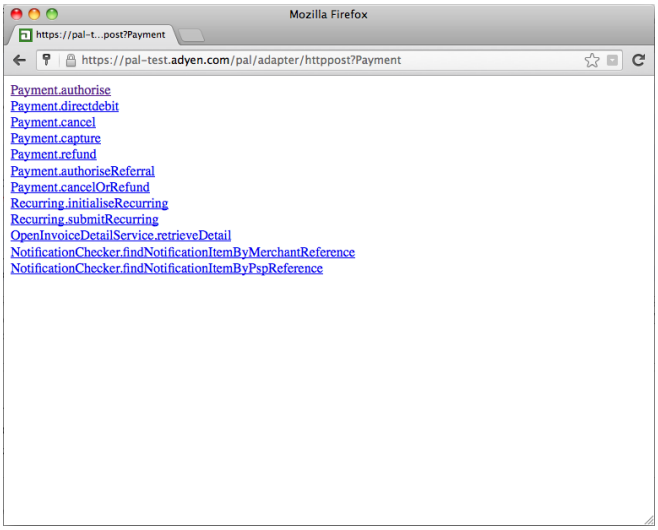
General remarks on HTTP Name/Value pair Communication

The Adyen APIs are exactly the same for SOAP and HTTP. Therefore the REST API can also be referenced as the HTTP Adapter. This manual will for this reason also not re-itterate the definitions and explanations provided in the Adyen Integration Manual.

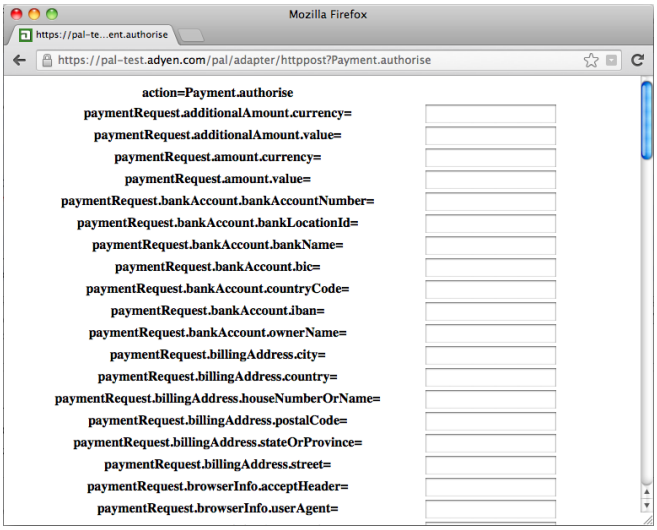
This manual explains the REST concepts, a few REST specifics and provides examples of the various Adyen API requests. Please note that replicationg/copying the exact examples will not work, since your specific account (and transactions) need to be referenced.

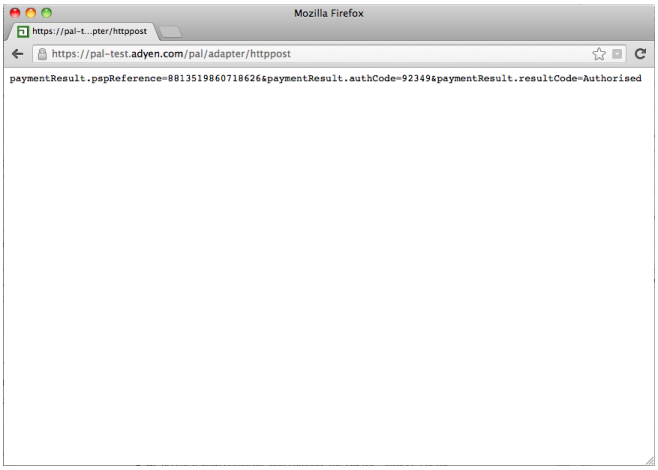
One of the benefits of having an HTTP Adapter for offering a REST system, is that there is an easy testing option via a regular web browser showing a default form for the variables. The URL where the HTTP adapter is located is specified in Appendix A: Production and Test URLs. The following four screens show how it is easy to research the specifications and make test payments.





The first screenshot shows the HTTP Log-in when browsing to the HTTP Adapter url in a browser. (The authentication is discussed in more detail in the next section.) On the second page you can select an Adyen API function you want to test / explore.

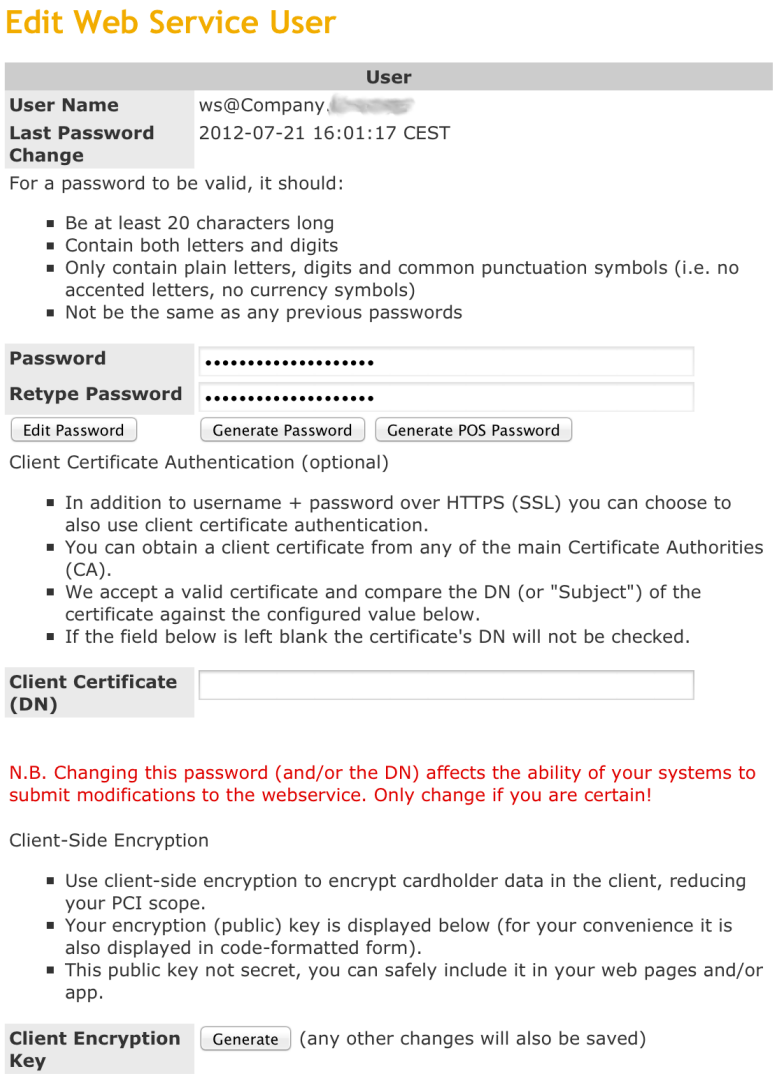




The third page allows the variables to be filled in. Please provide your specific account details and the (mandatory) fields and press submit at the bottom of this page. The browser communicates the values as HTTP Name/Value pairs and the resulting page is shown as fourth and final page. The response on the request is outputted in the browser.

Security

Security settings are configured for the *ws@Company.[YourCompanyAccount]* user account in the Merchant Backoffice (under Settings > Users):



Authentication

To submit authorisation messages you supply HTTP basic authentication credentials (username/password). This will be configured in the library that you use to communicate via the HTTP protocol the server-to-server request (or response) to the Adyen platform. You must set the password for the user or allow the system to generate it for you.

Client-Side Card Encryption (optional)

Merchants that require more stringent security protocols may decide to implement Client-Side Card Encryption. This is particularly important for Mobile payment flows where only cards are being offered, as it may result in faster load times and an overall improvement to the shopper flow.

Clicking on the Generate button will result in Adyen generating a unique Public Key, for

that specific *ws@Company.[YourCompanyAccount]* user, that is to be used to encrypt your authorisation messages.

Please note that while the use of the Client-Side Card Encryption increases your PCI coverage when compared to implementing the API without encryption; it does not provide as much PCI coverage as the use of the Hosted Payment Page.

1 Authorisations

This chapter explains how to submit authorisations messages for your payments as HTTP Post requests. To submit authorisation messages you supply HTTP basic authentication credentials (username/password). The URL for the authorisation call is located is specified in Appendix A: Production and Test URLs.

In order to send an authorisation via direct API, you will need to conform to the PCI-DSS industry regulations. More information about this topic can be found on the following Support page:

https://support.adyen.com/index.php?/support/Knowledgebase/Article/View/1081/0/how-do-i-get-access-to-the-adyen-api

When comparing the SOAP fields and HTTP fields they are exactly the same. Typically it is just one field with the same name, but more complex structures like the amoun will be transferred in two individual fields:

```
<amount xmlns="http://payment.services.adyen.com">
   <currency xmlns="http://common.services.adyen.com">EUR</currency>
   <value xmlns="http://common.services.adyen.com">500</value>
</amount>
```

Example 1: XML representation of an amount

```
paymentRequest.amount.currency=EUR&paymentRequest.amount.value=500
```

Example 2: HTTP Name/Value pair representation of an amount

A complete example request and response for an authorisaion are shown in Examples 3 and 4.

```
action=Payment.authorise
&paymentRequest.amount.currency=EUR&paymentRequest.amount.value=1234&paymentRequest.card.cvc=737&paymentRequest.card.expiryMonth=12&paymentRequest.card.expiryYear=2012&paymentRequest.card.holderName=Test+Tester&paymentRequest.card.number=5555444433331111paymentRequest.merchantAccount=YourMerchantAccount&paymentRequest.reference=test1234
```

Example 3: HTTP Authorisation Request

```
paymentResult.authCode=98356&paymentResult.pspReference=9913642236790892&paymentResult.resultCode=Authorised
```

Example 4: HTTP Authorisation Response

Client-Side Encryption

Transactions that are submitted using Client-Side encrypted card details follow the same process as a regular authorisation request. However, instead of the card object, the encrypted data is sent in the additional data field.

Example 5 below shows a request for a Client-Side encrypted authorisation. Please note, for the encrypted string has been shortened for display purposes.

```
action=Payment.authorise
paymentRequest.amount.currency=EUR
paymentRequest.amount.value=1234
paymentRequest.merchantAccount=YourAccountCode
paymentRequest.reference=Example Order 1
paymentRequest.additionalData.card.encrypted.json=adyenjs_0_1_0$eGc…Q=='
```

Example 5: HTTP Client-Side Encrypted Authorisation Request

2 Modifications

This chapter explains how to submit modifications messages for your payments as HTTP POST requests. It is possible to perform modifications using the Adyen Customer Area (Merchant Backoffice).At the payment detail page are when appriopriate Cancel, Capture and Refund buttons are shown to authorised users on the payment detail page.

It is however generally a desired to automate this if you are processing more than a handful of payment modifications daily.

To submit modification messages you supply HTTP basic authentication credentials (username/password). The URL for the modification calls is located is specified in Appendix A: Production and Test URLs.

Please note that for all messages if you receive either a *captureReceived*, *cancelReceived*, or *refundReceived* response this doesn't mean that the payment was actually modified. Instead it means that the request has been accepted. Once the request has been processed by Adyen you will receive a notification which tells you whether the modification was successful.

Capture

An example request and response for a capture are shown in Examples 5 and 6.

```
action=Payment.capture&modificationRequest.merchantAccount=TestMerchant&modificationReques
t.modificationAmount.currency=EUR&modificationRequest.modificationAmount.value=100&modific
ationRequest.originalReference=9999888877776666
```

Example 6: HTTP Capture Request

If the message was syntactically valid and merchantAccount is correct you receive a HTTP response with the following fields:

```
modificationResult.pspReference=9912493830470197&modificationResult.response=
%5Bcapture-received%5D
```

Example 7: HTTP Capture Response

Our Customer Area (CA) provides the option to configure an automated capture process, capturing payments automatically after a configured number of days, ranging from immediate up to 14 days. More information about Capture configurations can be found in the Adyen Merchant Manual, The Support Knowledgebase or at the tutorial video section.

Refund

An example request and response for a refund are shown in Examples 7 and 8.

```
action=Payment.refund&modificationRequest.merchantAccount=TestMerchant&modificationRequest
.modificationAmount.currency=EUR&modificationRequest.modificationAmount.value=50&modificat
ionRequest.originalReference=9999888877776666
```

Example 8: HTTP Refund Request

```
modificationResult.pspReference=9912493830470197&modificationResult.response=
%5Brefund-received%5D
```

Example 9: HTTP Refund Response

Cancel

An example request and response for a cancel are shown in Examples 9 and 10.

```
action=Payment.cancel&modificationRequest.merchantAccount=TestMerchant&modificationRequest
.originalReference=9999888877776666
```

Example 10: HTTP Cancel Request

```
modificationResult.pspReference=9912493830470197&modificationResult.response=
%5Bcancel-received%5D
```

Example 11: HTTP Cancel Response

Cancel or Refund

In order to cancel or refund a payment (depending on the status of the payment) this method can be used. An example request and response for a *cancelOrRefund* are shown in Examples 11 and 12.

```
action=Payment.cancelOrRefund&modificationRequest.merchantAccount=TestMerchant&modificatio
nRequest.originalRef erence=9999888877776666
```

Example 12: HTTP cancelOrRefund Request

```
modificationResult.pspReference=9912493830470197&modificationResult.response=
%5BcancelOrRefund-received%5D
```

Example 13: HTTP cancelOrRefund Response

3 Notifications

Whenever a payment is made, a modification is processed or when a report is available to be downloaded, the Adyen platform will notify you of the event and whether or not it was performed successfully. Notifications should be used to keep your backoffice systems up to date with the status of each payment and modification.

An example request and response for a *Notification* are shown in Examples 13 and 14:

```
eventDate=2012-09-25T13%3A41%3A33.81Z&reason=2120%3A1111%3A12%2F2012&originalReference=&me
rchantReference=reference_415579&currency=EUR&pspReference=8613485804662747&merchantAccoun
tCode=TestMerchant&eventCode=AUTHORISATION&value=24205&operations=CANCEL%2CCAPTURE
%2CREFUND&success=true&paymentMethod=mc&live=false
```

Example 14: HTTP Notification Request (generated by Adyen)

```
notificationResponse=&sBaccepted%5D
```

Example 15: HTTP Notification Response

There are two tools made available to easily generate test notifications, besides going through the whole new payment cycle. One can be found in the Adyen Customer Area and the second one is located in the Support suite:

- https://ca-live.adyen.com/ca/ca/config/notification.shtml
- https://support.adyen.com/index.php?/support/Knowledgebase/Article/View/1571/0/test-http-post-notifications

Notifications are sent to a server of your choice. Notification settings are configured in the Customer Backoffice. You can set the method (HTTP POST/SOAP), URL to submit to, and if applicable the user name/password for your Notifcation Listener Service.

Please note your system should be able to handle requests/responses which contain additional fields. An (authorisation) notification is also likely to be sent twice due to the architecture of the Adyen platform, which is scalable, redundant and able to handle a high throughput.

Appendix A: Production and Test URLs

Test URLs

HTTP Adapter (Browser)	https://pal-test.adyen.com/pal/adapter/httppost?Payment
Authorisation	https://pal-test.adyen.com/pal/adapter/httppost?Payment.authorise
Test Capture	https://pal-test.adyen.com/pal/adapter/httppost?Payment.capture
Test Refund	https://pal-test.adyen.com/pal/adapter/httppost?Payment.refund
Test Cancel	https://pal-test.adyen.com/pal/adapter/httppost?Payment.cancel

Production URLs

HTTP Adapter (Browser)	https://pal-live.adyen.com/pal/adapter/httppost?Payment
Authorisation	https://pal-live.adyen.com/pal/adapter/httppost?Payment.authorise
Capture	https://pal-live.adyen.com/pal/adapter/httppost?Payment.capture
Refund	https://pal-live.adyen.com/pal/adapter/httppost?Payment.refund
Cancel	https://pal-live.adyen.com/pal/adapter/httppost?Payment.cancel