XML API Developer-Documentation

Version 2.01

07/23/2015

Content

Introduction

This document contains all functions and specifications of the RBH Secure-Pay (S-PAY) API. The S-PAY API is a so called XML program interface which allows you to fully automate your payment processes. Therefore the following functions are available by using our API:

- Sending of single payments
- Re-posts between internal own held accounts
- Calling of account history
- Calling of single transactions
- Verification of the validity of accounts

Those possibilities are necessary for all providers which want to include S-PAY as an online payment service into their projects and for sure to automate their payment transactions. With our API you are able to in example check deposits which were made via S-PAY regarding validations and therefore pay commissions to many recipients. But that is just one of many more opportunities you are able to create with our API!

Who needs this information?

This documentation is referred to developers who want to include S-PAY as an additional module into their services or scripts. For a successful implementation you should have experience in the following sectors:

- Creation and handling of server requests
- Receiving and evaluation of data which are sent from web servers.
- Knowledge of the XML standard routine

S-PAY Testing Environment

S-PAY offers you a whole testing environment which can be found here: https://paytest.s-pay.me .You are able to test the whole payment system and that's what we refer to do before you implement your code into a live project.

URL to our API

For any API requests you are able to use the following addresses:

Test-API: https://paytest.s-pay.me/api

Live-API: https://www.s-pay.me/pay/api

Preparation

To work with the API you need to work on the following preparations:

1. Create a customer account in our testing system at https://paytest.s-pay.me
2. Create internal accounts of all currencies in need
3. Activate the API for the referred account by visiting “Accounts → Account Overview → Edit”. You need to enter the API – Key which is unique for this account. This will be your very own password for every access to our API.
4. If you want to limit the API access to the account you are also able to enter IP addresses in the “IP - Whitelist” function.

Requirements

In order to work with the S-PAY API you need to use a programming language which allows you to create XML compliant data and to send HTTPS POST and GET requests. Also make sure the language of your choice is able to use the sha512 hashing function. In example this is possible since PHP version 5.1.2. Generally the API can also be accessed by any other programming language. We use PHP5 in our further examples.

API functions in overview

The S-PAY API offers the following functions:

Function	Type attribute	Description
Account balance	getBalance	Retrieval of the account balance of the actual account with its specified currency
Statement	statement	Retrieval of all booking in a dedicated time frame
Single transaction	getTransaction	Retrieval of a single transaction by using the transaction id
Account-check	checkAccount	Request if the account exists
Re-post	repost	Re-post between 2 own held internal accounts
Remittance	internal	Remittance from one of your accounts to another customer account. This also allows mass payments.

Authentication

For every access to the S-PAY API the own account number and the API password is in need. The following authentication information are necessary for every API request:

- Own account number max. 12 signs
- Date in UTC format Y-m-d
- API password
- unique_id String generated by yourself.

Creation of the authentication -HASH

An authentication-token will be generated with the details above the following way:

Account – Number:Date:API - Password

Example for building the HASH

- Account Number= E0123456789
- Date= 2015-12-06 im UTC Format
- API-Password=]d_RS&DIlUYM;PJ#j%DM){wSviiK`\s42,!_2C^&jmgh55%~jJ/'1iD%7
- Unique_id= 999719-4599

With this details we now generate a string where those 3 parameters are separated by colons. Therefore the string would look like this:

´E0123456789:2015-12-06:]d_RS&DIlUYM;PJ#j%DM){wSviiK`\s42,!_2C^&jmgh55%~jJ/'1iD%7: 999719-4599´

This string will now be hashed with the sha512 algorythm and the result following our example data is this one:

2de672b8187a47d23556a3aa83a43378f8dd785c52fad84196f0a4b430af918f08071b901f27046ca4a3eaee78b0fa85b1ef2815cab4380cbb4e3cbe4218fb02

Request / Return Identification

The S-PAY XML API requires a parameter (unique_id) for verifications of requests and returns. Therefore you are able to verify if the returned answer is valid to the request your server just sent. The unique_id parameter needs to be unique and explicit for the chosen account. It have to be included as an mandatory value into your API request and will be returned from the S-PAY server without any changes. S-PAY uses this one to prevent from double-requests. Just if there is no other request with this unique id the actual request will be handled. If this was already in use in past the API will return the error code “double-request”.

User-defined parameters

The S-PAY XML API allows any optional parameters to be used to include it in the data-block. These ones will be returned to your server without any changes.

Examples for XML requests and answers in PHP 5

For our following examples we will use the following variables:

```
$api_key =      ']d_RS&DIlUYM;PJ#j%DM){wSviiK`\s42,!_2C^&jmgh55%~jJ/'1iD%7';      // api-password
$account_no =  'E0123456789';            // own account number (11 signs, 12 for exchanging services!)
$unique_id=     '999719-4599';
$api_path=      'https://paytest.s-pay.me/api/';          // path to our test-api
```

Retrieval of account balance

Return the actual account balance of the specified account.

Parameters to be committed

- unique_id String generated by yourself. For our example: 999719-4589

API-Return

- Balance, decimal places separated by a point (.) . EUR/USD 2, Gold 3 and BTC 8 decimal places.
- Currency : EUR, USD, BTC, XAU in g

Request to the API

```
$request_xml = '
<request type="getBalance">
        <auth>
                <hash>'.hash('sha512', $account_no.':'.gmdate('Y-m-d').':'. $api_key.':'.$unique_id).'</hash>
                <account>'.$account_no.'</account>
        </auth>
        <data>
                <unique_id>'.$unique_id.'</unique_id>
        </data>

</request>
';
```

Returns from the API

```
$response_xml = '
<response type="getBalance">
        <data>
                <unique_id>999719-4599</unique_id>
        </data>
        <result>
                <result>1</result>
                <balance>9543.91</balance>
                <currency>EUR</currency>
        </result>
</response>';
```

Retrieval of bookings

This will return the last 1000 bookings

Parameters to be committed

- datefrom — starting date (optional), Format YYYY-MM-DD
- dateto — ending date (optional) , Format YYYY-MM-DD
- unique_id — String generated by yourself. For our example: 999719-4589

API Return

- account — Account number of recipient / sender
- subject1 — Payment purpose 1
- subject2 — Payment purpose 2
- refid — S-Pay reference id
- reference_id — Reference number out of a SCI payment (if set)
- amount — Amount
- type — 1 = internal re-post
2 = RBH - internal remittance
101 = Fees for internal re-posts
102 = Fees for internal remittance
- rate — Exchanging fee between account currency and base currency
- currency — Base currency
- currencyamount — Amount in base currency
- valuta — Time-frame of valuta date

Request to the API

```
$request_xml = '

<request type="statement">
        <auth>
                <hash>'.hash('sha512', $account_no.':'.gmdate('Y-m-d').':'. $api_key.':'.$unique_id).'</hash>
                <account>'.$account_no.'</account>
        </auth>

        <data>
                <datefrom>2015-06-01</datefrom>
                <dateto>2015-06-13</datefrom>
                <unique_id>'.$unique_id.'</unique_id>
        </data>
</request>';
```

Return of the API

```
$response_xml = '

<response type="statement">
	<data>
		<datefrom>2015-06-01</datefrom>
		<dateto>2015-06-13</dateto>
		<unique_id>999719-4599</unique_id>
	</data>
	<result>
		<bookings>
			<booking>
				<account>E9876543210</account>
				<subject1>Payment purpose 1</subject1>
				<subject2>Payment purpose 2</subject2>
				<refid>1234</refid>
				<amount>123.45</amount>
				<type>2</type>
				<valuta>2015-06-03 10:19:21</valuta>
			</booking>

			<booking>
				<account>E1234567890</account>
				<subject1>Payment purpose 1</subject1>
				<refid>1239</refid>
				<amount>1234.56</amount>
				<type>2</type>
				<valuta>2015-06-09 10:18:58</valuta>
			</booking>

			//  (max. 1000 lines of bookings for each request)


		</bookings>
	</result>

</response>';
```

Internal re-post between accounts of one customer account

Parameters to be committed

- account Account number of recipient
- amount Amount
- currency Currency (EUR, USD, BTC, XAU)
- subject1 Payment purpose 1
- subject2 Payment purpose 2
- unique_id String generated by yourself. For our example: 999719-4599

API Return

- transaction ID of the transaction
- Balance Current Balance decimal places separated by a point (.) . EUR/USD 2, Gold 3 and BTC 8 decimal places.

Request to the API

```
$request_xml = '
        <request type="repost">
                <auth>
                  <hash>'.hash('sha512', $account_no.':'.gmdate('Y-m-d').':'. $api_key.':'.$unique_id).'</hash>
                  <account>'.$account_no.'</account>
                </auth>

                <data>
                        <account>E1234567890</account>
                        <amount>123.45</amount>
                        <currency>EUR</currency>
                        <subject1>Payment purpose 1</subject1>
                        <subject2>Payment purpose 2</subject2>
                        <unique_id>'.$unique_id.'</unique_id>
                </data>
        </request>';
```

Return from the API

```
$response_xml = '

<response type="repost">
        <data>
                <account>E1234567890</account>
                <amount>123.45</amount>
                <currency>EUR</currency>
                <subject1>Payment purpose 1</subject1>
                <subject2>Payment purpose 2</subject2>
                <unique_id>999719-4599</unique_id>
        </data>

        <result>
                <result>1</result>
                <transaction>1234</transaction>
                <balance>9543.91</balance>
        </result>
</response>';
```

S-PAY internal remittance

Parameters to be committed

- account Account number of recipient
- amount Amount
- currency Currency (EUR, USD, BTC, XAU)
- subject1 Payment purpose 1
- subject2 Payment purpose 2
- unique_id String generated by yourself. For our example: 999719-4589

API Return

- transaction ID of the transaction
- balance Current Balance decimal places separated by a point (.) . EUR/USD 2, Gold 3 and BTC 8 decimal places.

Request to the API

```
$request_xml = '
    <request type="internal">
        <auth>
          <hash>'.hash('sha512', $account_no.':'.gmdate('Y-m-d').':'. $api_key.':'.$unique_id).'</hash>
          <account>'.$account_no.'</account>
        </auth>
        <data>
            <account>E1234567890</account>
            <amount>123.45</amount>
            <currency>EUR</currency>
            <subject1>Payment purpose 1</subject1>
            <subject2>Payment purpose 2</subject2>
            <unique_id>'.$unique_id.'</unique_id>
        </data>
    </request>';
```

Antwort von der API

```
$response_xml = '
<response type="internal">
    <data>
        <account>E1234567890</account>
        <amount>123.45</amount>
        <currency>EUR</currency>
        <subject1>Payment purpose 1</subject1>
        <subject2>Payment purpose 2</subject2>
        <unique_id>999719-4589</unique_id>
    </data>
    <result>
        <result>1</result>
        <transaction>1234</transaction>
        <balance>9543.91</balance>
    </result>
    </response>';
```

Retrieve a single transaction

Parameters to be committed

- transaction — Transaction - ID (booking-number)
- unique_id — String generated by yourself. For our example: 999719-4589

API Return

- account — Account number of recipient / sender
- subject1 — Payment purpose 1
- subject2 — Payment purpose 2
- refid — S-Pay reference id
- reference_id — Reference number from SCI payment
- amount — Amount
- type — 1 = internal re-post
2 = interne remittance
101 = fees for internal re-post
102 = fees for internal remittance
- rate — Exchanging values between account currency and basis currency
- currency — Basis currency
- currencyamount — Amount in basis currency
- valuta — Time-frame of valuta date

Request to the API

```
$request_xml = '
    <request type="getTransaction">
        <auth>
          <hash>'.hash('sha512', $account_no.':'.gmdate('Y-m-d').':'. $api_key.':'.$unique_id).'</hash>
          <account>'.$account_no.'</account>
        </auth>
        <data>
            <transaction>1478</transaction>
            <unique_id>'.$unique_id.'</unique_id>
        </data>
    </request>';
```

Return from the API

```
$response_xml = '
<response type="getTransaction">
    <data>
        <transaction>1478</transaction>
    </data>
    <result>
        <bookings>
            <booking>
                <account>XA0004471041</account>
                <subject1>Payment purpose 1</subject1>
                <subject2>Payment purpose 2</subject2>
                <refid>1478</refid>
                <amount>-123.45</amount>
                <type>2</type>
                <valuta>2015-06-11 11:43:29</valuta>
            </booking>
        </bookings>
```

```
        </result>
</response>';
```

Check the existence of an account number

Parameters to be committed

- account Account number
- unique_id String generated by yourself. For our example: 999719-4589

API Return

- status 1 = account number valid
  0 = account number invalid

Request to the API

```
$request_xml = '
        <request type="checkAccount">
                <auth>
                  <hash>'.hash('sha512', $account_no.':'.gmdate('Y-m-d').':'. $api_key.':'.$unique_id).'</hash>
                  <account>'.$account_no.'</account>
                </auth>
                <data>
                        <account>XA0004471041</account>
                        <unique_id>.$unique_id.</unique_id>
                </data>
        </request>';
```

Return of the API

```
$response_xml = '
        <response type="checkAccount">
                <data>
                        <account>XA0004471041</account>
                        <unique_id>999719-4589</unique_id>
                </data>
                <result>
                        <status>1</status>
                </result>
        </response>';
```

API request using the GET-Method

```
$response_xml = file_get_contents($api_path.'?data='.urlencode($request_xml));

echo $response_xml;
```

API request using the Post-Method (using CURL)

```
$ch = curl_init($api_path);

curl_setopt($ch, CURLOPT_POST, 1);

curl_setopt($ch, CURLOPT_POSTFIELDS, 'data='.$request_xml);

curl_setopt($ch, CURLOPT_HEADER, 0);

curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);

$response_xml = curl_exec($ch);

echo $response_xml;
```

Handling Errors

In our example we will use a possible error code while retrieving bookings. The error will be generated because the date of the 31st of February is not valid. In case of an error the API will return an error code anytime including a short description of it.

Inaccurate request to the API

```
$request_xml = '

<request type="statement">
        <auth>
          <hash>'.hash('sha512', $account_no.':'.gmdate('Y-m-d').':'. $api_key.':'.$unique_id).'</hash>
          <account>'.$account_no.'</account>
        </auth>

        <data>
                <datefrom>2015-02-31</datefrom>
                <unique_id>'.$unique_id.'</unique_id>
        </data>
        </request>';
```

Antwort von der API

```
$response_xml = '

<response type="error">
        <data>
                <datefrom>2015-02-31</datefrom>
                <unique_id>999719-4589</unique_id>
        </data>
        <error>
                <code>53</code>
                <text>Invalid date value in field (datefrom)</text>
        </error>
</response>';
```

Table of possible error codes

Error code	Error message
1	Received no POST or GET data
2	There was no XML data
3	No request type
4	Unknown request type
10	Authentication failed: no account and/or hash
11	Authentication failed: wrong account and/or hash
12	API is not active for this account
13	IP XXX.XXX.XXX.XXX is not white-listed for this account
50	Empty mandatory fields (Name of Field with Error)
51	Additional field is not set (Name of Field with Error)
52	Invalid value for field (Name of Field with Error)
53	Invalid date value in field (Name of Field with Error)
60	Unknown currency
100	Not enough money in account
101	Different customers
102	Unknown beneficiary account
9997	unique_id was already used for this account
9998	API is temporarily disabled
9999	Unknown Error

A few hints regarding security

To make sure that every S-PAY XML API access will be done within the most optimal circumstances we refer you toll look at the following hints to get the most security out of your account:

- Never give any login details of your S-PAY customer account to a third party person!
- Make sure you are the only one who knows the API password. This is the ultimate key to your account and the funds which you store in it!
- If you have hired external programmers do let them use the testing environment for developing your applications. If all is working make sure you are the only one who has access to any authentication data used in your live account!
- Never login to your customer accounts in public like an internet cafe or similar. Always make sure you just access your account from private computer which use updated anti-virus and malware scanners as well as the latest operating system updates.
- Never store your access details in not encrypted text files. If you want to securely safe your login information we refer that you use software like “Keypass” or similar.
- The perfect usage of any API would be to use a computer with a static IP. With our white-listing feature you can make sure only this special computer is able to use it without the chance to be hacked by other IP’s.

Do you have any questions or recommendations?

If you have any questions or recommendations related to the S-PAY API you are welcome to contact us using the ticket system anytime. You have urgent problems or questions? You are also welcome to contact our dedicated Skype support service which can be found with id “rbh-service”. We always struggle to develop the last secured products and the best enhancements to meet our customer’s needs. Help us doing that!

Revision notifications

Date	Type	Description
26.05.2015	API System	IP Whitelisting added for each account
08.06.2015	API returns advanced	For booking requests the reference_id is transferred too if it was set while SCI payments
11.06.2015	API request added	Requests of the existence of accounts added
11.06.2015	API request added	Requests of booking using the transaction number
12.06.2015	Request parameter added	Parameter unique_id added to prevent double requests
12.06.2015	API error code update	Error codes 13, 9997 and 9998 added
12.06.2015	API System	Changing date for hash generation to UTC format
20.07.2015	API System	Extended hashing by unique_id and unique_id is now mandatory
23.07.2015	API returns advanced	added current balance to API-Return of repost and internal