Easypay Interface Manual

Version	3.8.3
Status	published
Replaces version	3.8.2
Issue date	20/10/2020
Author	Alexander Schamne

Change list:

Updated on	Updated by	Version	Description
06/24/2013	A. Schamne	3.0	- change list inserted - changed to version **3.0** (version adopted to Easypay **3.0**, to avoid misunderstandings)
11/11/2013	A. Schamne	3.1	- additional error codes: - TRANSACTION_ALREADY_REFUNDED - TRANSACTION_REFUND_TO_HIGH - TRANSACTION_TO_OLD_FOR_REFUND - added Notes to AuthSubscription
11/19/2013	R. Schwarz	3.1	- updated Date header format
12/13/2013	R. Schwarz	3.1	- added detail information to AuthSubscription
01/31/2014	R. Schwarz	3.2	- added PHP example for signature
10/15/2014	K. Jordan	3.3	- added refunds for subscriptions
11/17/2014	R. Schwarz	3.4	- small fix and added new business error codes
05/13/2015	R. Schwarz	3.5	- added 24h commit limitation
11/30/2015	R. Schwarz	3.6	- added notes in chapter 6.3.1: optional amount for refund/renew
03/02/2016	R. Schwarz	3.7	- added chap 6.3.2: Alternative Authorize Subscription renew flow (refund individual subscription payment ID's, aka Version 1_1) - added more REST API signing example code on GitHub (chap 7.3) - added chap 7: Error Code Mapping
01/30/2017	R. Schwarz	3.8	- added new error code CE_NOT_AUTHORIZED_NOT_ALLOWED_DATA_ABO
12/12/2017	K. Jordan	3.8.1	- corrected example in for Direct Payment
08/13/2018	R. Schwarz	3.8.2	Added new non-retriable error code in section 7
20/10/2018	K. Jordan	3.8.3	removed depricated error codes: - CE_NOT_AUTHORIZED_NOT_SUPPORTED_ROAMING - CE_NOT_AUTHORIZED_BLOCKED_PREPAID_ACCOUNT_CLOSED - CE_NOT_AUTHORIZED_BLOCKED_PREPAID_ACCOUNT_MOVED

Table of Contents

1 Abstract

The document describes the Easypay REST interface.

2 Request Headers

Header	Description	Required	Example
Accept	media-types which are accepted by client	YES	application/json
Content-Length	length of message body in bytes	YES	123
Content-Type	the media-type of content send withhin the message body	YES	application/json
Host	the client host name	YES	easypay.ch
X-Request-Id	The request id of the http-request. This Id should be unique for each request.	**Strongly recommended** **The correlation between client and server requests can be done using this id.**	1234-asdf-5678-ghjk
X-CE-Client-Specification-Version	the charging engine client version	Recommended The newest specification implementation will be used as default.	1.0
X-Merchant-Id	the merchant id	YES	ABC01

3 Response Headers

Header	Description	Required	Example
Content-Length	length of message body in bytes	YES	123
Content-Type	the media-type of content send withhin the message body	YES	application/json
Location	URI of the created resource	NO Will only be returned for create requests	https://easypay.swisscom.ch/ce-rest-service/payments/1

4 HTTP-Status codes

Status	Description
200	OK The request was successfully completed
201	CREATED A request that created a new resource was completed, and no response body containing a representation of the new resource is being returned. A Location header containing the canonical URI for the newly created resource will be returned.
400	BAD REQUEST The request could not be processed because it contains missing or invalid information (such as validation error on an input field, a missing required value, and so on). The exact ERROR will be returned within the message JSON Object.
403	FORBIDDEN The server recognized your credentials, but you do not possess authorization to perform this request
404	NOT FOUND The request specified a URI of a resource that does not exist.
405	NOT ALLOWED The HTTP verb specified in the request (DELETE, GET, HEAD, POST, PUT) is not supported for this request URI.
406	NOT ACCEPTABLE The resource identified by this request is not capable of generating a representation corresponding to one of the media types in the Accept header of the request.
415	UNSUPPORTED MEDIA TYPE The resource identified by this request is not capable for the given Content-Type.
500	INTERNAL SERVER ERROR The server encountered an unexpected condition which prevented it from fulfilling the request.

5 Resource Models

This section specifies the model objects, which are used for in the resource communication. Some model objects nest other model objects, which will be commonly represented as an URI. Also list representations of the object will contain a short information and an URI to the object. For more detailed information, the GET call to the URI will represent the details of requested object. The occurs field describes how many objects can be expected in case of read request, or must (can) be sent in case of write requests.

5.1 Message List [application/vnd.ch.swisscom.easypay.message.list+json]

The response message list object represents the server messages, which can be extracted and interpreted by the client.

5.1.1 Message-List data model

Field Name	Type	Occurs in GET	Description
messages	Message	1..n	One or more messages for each individual message

5.1.2 Message data model

Field Name	Type	Occurs in GET	Description
message	String	0..1	Localized message describing the nature of the problem reported by this message
field	String	0..1	Name of the field from the request data model that this message is associated with
code	String	1	Symbolic error code identifying the type of error reported by this message
requestId	String	0..1	The request id, sent within the X-Request-Id http header.

5.1.3 Example

```
{
    "messages": [
        {
            "message":     "A mandatory parameter is missing in a HTTP-Header.
                           missing parameter is specified in the field-element
                           of the response message.",
            "field":       "X-Merchant-Id",
            "code":        "MISSING_MANDATORY_HEADER_PARAMETER",
            "requestId":   "12wde233"
        },
        {
            "message":     "A mandatory field is missing in a JSON request object.
                           The missing parameter is specified in the field-element of
                           the response message.",
            "field":       "amount",
            "code":        "MISSING_MANDATORY_FIELD",
            "requestId":    "12wde233"
        }
    ]
}
```

5.2 Direct Payment [application/vnd.ch.swisscom.easypay.direct.payment+json]

5.2.1 Direct Payment data model

The direct payment is a kind of one-charge-event-payment. It does not need any management on the charging-engine side.

Field Name	Type	Occurs in GET (retrieve)	Occurs in PUT (refund, commit, reject)	Description
paymentInfo	String	1	0	The payment info of the service (also known as billing text). NOTE: The paymentInfo will be placed on the end user invoice bill.
isAdultContent	Boolean	1	0	Flag which marks the service as adult (true) or non adult (false). This means that the customer will be checked, if he/she is allowed to consume adult content.
amount	String in Format: DDD.dd (decimals)	1	0..1	The amount of the service. Optional for Refund (if not set full charged amount fill be refunded).
userAgentOrigin	String	0..1	0	The UserAgent of the end customer e.g. Mozilla/5.0 (Linux; U; Android 2.2.1; en-us; Nexus One Build/FRG83) AppleWebKit/533.1 (KHTML, like Gecko) Version/4.0 Mobile

				Safari/533.1
userSourceIP	String	0..1	0	The end user source IP
operation	String allowed values: • REJECT • COMMIT • REFUND	0	1	Allows to commit/ reject the payment or to refund the amount. The amount to be refund must be the same or less as the initial payment.
orderId	String	1	0	The order id which is printed on end user invoice bill.

5.2.2 Example

```
{
    "amount": "10.00",
    "paymentInfo": "Muper Sario",
    "isAdultContent": true
}
```

5.3 AuthSubscription [application/vnd.ch.swisscom.easypay.authsubscription+json]

The Authsubscriptions are always created for services which are managed on content partner side. The services are NOT stored and managed on the Easypay environment. The Authsubscription allows the content partner to authorize the end user for subscription. After the subscription authorization is created, the content partner is allowed to renew the subscription without user interaction. The end user can cancel the recurrent subscription on Swisscom Self-care Online services (Kundencenter) or content partner side at any time.

5.3.1 Authsubscription data model

Field Name	Type	Occurs in GET	Occurs in PUT	Description
operation	String allowed values: • REJECT • COMMIT • RENEW • CANCEL • REFUND	0	1	
isActive	Boolean	1	0	Only in GET response: active AuthSubscription: true/false
msisdn	String	0	0	The customer msisdn

duration	Integer	1	0	The duration amount of the subscription
durationUnit	String allowed values: • MINUTE • DAY • WEEK • MONTH	1	0	The duration unit of the subscription, which is combined with the duration amount.
createdOn	String in ISO8601 format: yyyy-MM-dd'T'HH:mm:ss.SSSZZ	1	0	The valid from of the recurrent subscription
URI	String	1	0	A unique URI that reference these particular authsubscription.
nextPayment	See createdOn	1	0	The next payment of the recurrent subscription
startRefund	See createdOn	1	0..1	The refund start date for payments of the recurrent subscription
merchantId	String	0	0	The external unique merchant id
isAdultContent	Boolean	0	0	Flag which marks the service as adult (true) or non adult (false). This means that the customer will be checked, if he/she is allowed to consume adult content.
amount	String in Format: DDD.dd (decimals) Example: 12.45	1	0..1	The amount for the starting or next period. In case of renewal or refund the amount is optional but if present it must be between zero and AuthSubscription creation amount. See additional notes in chapter 6.3.1
cpServiceId	String	0..1	0	The service Id on content partner side. This is an optional parameter. It is recommended to send it anyway, as it can be very helpful for debugging.
cpUserId	String	0..1	0	The unique user Id on content partner side. This is an optional parameter. It is recommended to send it anyway, as it can be very helpful for debugging.
cpSubscriptionId	String	0..1	0	The unique subscription Id on content partner side. This is an optional parameter. It is recommended to send it anyway, as it can be very helpful for debugging.
orderId	String	1	0	The order id which is printed on end user invoice bill.

6 Operations

In this section all REST operations of Easypay will be described. All URL contain a constant $BASE_URL.

The *$BASE_URL* is one of:

STAGING: https://easypay-staging.swisscom.ch/ce-rest-service

PRODUCTION: https://easypay.swisscom.ch/ce-rest-service

The variables in URL, which must be set by the content partner, are marked with braces e.g. {id}.

All REST operations **MUST be signed**. Signing REST operations is described in 'Signing REST requests' chapter.

6.1 Overview

	Operation	URL	Short Description
Direct Payment Resource	PUT	$BASE_URL/payments/{paymentId}	Commit/Reject/Refund direct payment
	GET	$BASE_URL/payments/{paymentId}	Retrieve direct payment
Authorize Subscription Resource	PUT	$BASE_URL/authsubscriptions/{authSubscriptionId}	Commit/Reject/Renew/Refund/Cancel Authorize Subscription
	GET	$BASE_URL/authsubscriptions/{authSubscriptionId}	Retrieve Authorize Subscription
Authorize Subscription Resource (Version 1_1)	PUT	$BASE_URL/authsubscriptions/1_1/{authSubscriptionId}	Renew/Cancel Authorize Subscription **version 1_1**
	GET	$BASE_URL/authsubscriptions/1_1/{authSubscriptionId}/firstPayment	Retrieve Authorize Subscription first payment **version 1_1**
Request Resource	GET	$BASE_URL/requests/{requesttId}	Retrieve the status of the request with specified Id

6.2 Direct Payment Resource

6.2.1 Commit / Reject / Refund Direct Payment

Description: Commit/Reject or Refund payment.

Commit: The commit on a reserved paymentId must be done within 24h after creation, otherwise the commit will fail and the payment will turn into status REJECTED and hence, can't be charged anymore.

Refund: By default, (no amount in request body) the full charged amount will be refunded.
It is also possible to partly refund a committed payment, a refund is unique per paymentId and requestId (http-header: X-Request-Id). The total refund amount CANNOT be higher than the original amount.

URI: $BASE_URL/payments/{paymentId}

HTTP-Method: PUT

Request Headers: X-Request-Id, X-CE-Client-Specification-Version

Request Body: application/vnd.ch.swisscom.easypay.direct.payment+json

(*operation: mandatory, amount: optional*)

Query Parameters: N/A

Response-Headers: Content-Length, Content-Type

Response Body: application/vnd.ch.swisscom.easypay.message.list+json

HTTP Response-Codes:

- 200, OK: The payment is committed.
- 400, Bad Request: Some Error occurred, the error message is given within the response.
    - Possible Errors (code on message object):
        - MISSING_MANDATORY_FIELD
        - OPERATION_INVALID
        - TRANSACTION_ALREADY_REJECTED
        - TRANSACTION_ALREADY_COMMITTED
        - TRANSACTION_ALREADY_REFUNDED
        - PARTIAL_REFUND_REQUESTID_ALREADY_REFUNDED
        - TRANSACTION_REFUND_TO_HIGH
        - TRANSACTION_TO_OLD_FOR_REFUND
- 403, Forbidden
- 404, Not Found: Occurs, if the payment was not found
- 415, Unsupported Media Type

- 500, Internal Server Error : Occurs because of some internal server error.

Example:

Request:

```
PUT https://easypay-staging.swisscom.ch/ce-rest-service/payments/4657e27c-bc70-4a1a-8517-8cdfcc66e0e2


PUT data:
{ "operation":"COMMIT" }


Request Headers:
Content-Length: 24
X-Request-Id: req1234
Connection: keep-alive
Content-Type: application/vnd.ch.swisscom.easypay.direct.payment+json
Accept: application/vnd.ch.swisscom.easypay.message.list+json
X-Merchant-Id: CH
X-CE-Client-Specification-Version: 1.1
```

Response:

```
Response headers:
HTTP/1.1 200 OK
Content-Length: 0
Date: Wed, 04 Apr 2012 15:54:05 GMT
```

6.2.2 Retrieve Direct Payment

Description: Retrieve the direct payment

URI: $BASE_URL/payments/{paymentId}?fields=status

HTTP-Method: GET

Request Headers: X-Request-Id, X-CE-Client-Specification-Version

Request Body: N/A

Query Parameters: fields

Response-Headers: Content-Length, Content-Type

Response Body: application/vnd.ch.swisscom.easypay.direct.payment+json

HTTP Response-Codes:

- 200, OK: The subscription status is returned in HTTP response body
- 404, Not Found: Occurs, if the direct payment was not found
- 500, Internal Server Error: Occurs because of some internal server error.

Example:

Request:

```
GET https://easypay-staging.swisscom.ch/ce-rest-service/payments/7f4811eb-82ef-4568-80e4-7cd5899ce89e?fields=status


Request Headers:

Connection: keep-alive
```

Response:

```
Response Headers:

HTTP/1.1 200 OK

Content-Type: application/vnd.ch.swisscom.easypay.direct.payment+json

Date: Wed, 04 Apr 2012 15:31:21 GMT


Response Body:

{
    "status": "RESERVED"
}
```

6.3 Authorize Subscription Resource

6.3.1 Commit / Reject / Cancel / Renew /Refund Authorization Subscription

Description: Update an Authorization Subscription

Note:

1. The authsubscriptionId must be committed within 24h after creation through Checkoutpage, otherwise it will be set to REJECTED and hence cannot be charged anymore and the Authorize Subscription will not be activated.
2. The AuthSubscription can be renewed at the earliest 24h before subscription expiry, otherwise RENEW_TO_EARLY Exception will occur.
3. All AuthSubscriptions will be cancelled in case the user changes his msisdn.
4. The AuthSubscription status will remain ACTIVE in case of non-successful renewal due to business exception (i.e. bad debt).
5. In case of non-successful renewal (i.e. bad debt) the renewal request might be repeated in 24h cycles for max. 40 days, after 40 days non-successful renewal cycles the AuthSubscription will be cancelled.
6. Refunds need a start Date maximum 6 months back.
   Put data example: {"operation":"REFUND", "startRefund":"2014-07-27T01:00:00"}
7. Refund or Renew: The parameter amount might optionally be present in the request body, if so it will be taken for the transaction if it is between 0 and AuthSubscription creation amount.
   Example: {“operation”:”RENEW”,”amount”:0}
   If the optional parameter amount is not present in the request body, the AuthSubscription creation amount will be taken for renew or refund.

URI: $BASE_URL/authsubscriptions/{authsubscriptionId}

HTTP-Method: PUT

Request Headers: X-Merchant-Id, X-Request-Id, X-CE-Client-Specification-Version

Request Body: application/vnd.ch.swisscom.easypay.authsubscription+json

Query Parameters: N/A

Response-Headers: Content-Length, Content-Type, Location

Response Body: application/vnd.ch.swisscom.easypay.message.list+json

HTTP Response-Codes:

- `200, OK`
- 400, Bad Request: Some Error occurred, the error message is given within the response.
  - Possible Errors (code on message object):
    - `OPERATION_INVALID`
    - `INVALID_AMOUNT_FORMAT`
    - `MISSING_MANDATORY_FIELD`
    - `SUBSCRIPTION_ALREADY_CANCELED`
    - `RENEWAL_TO_EARLY`
    - `TRANSACTION_ALREADY_REJECTED`
    - `TRANSACTION_ALREADY_COMMITTED`
    - `TRANSACTION_NOT_COMMITTED`
    - See more Error Codes in Section 7
- 403, Forbidden
- 404. Not Found
- 415, Unsupported Media Type
- 500, Internal Server Error : Occurs because of some internal server error.

Example:

Request:

```
PUT https://easypay-staging.swisscom.ch/ce-rest-service/authsubscriptions/4657e27c-bc70-4a1a-8517-8cdfcc66e0e2


PUT data: { "operation":"RENEW" }


Request Headers:
Content-Length: 24
X-Request-Id: req1234
Connection: keep-alive
Content-Type: application/vnd.ch.swisscom.easypay.authsubscription+json
Accept: application/vnd.ch.swisscom.easypay.message.list+json
X-Merchant-Id: CH
X-CE-Client-Specification-Version: 1.1
```

```
Response:
Response headers:
HTTP/1.1 200 OK
Content-Length: 0
Date: Wed, 15 Apr 2013 13:54:05 GMT
```

6.3.2 Alternative Renew / Refund / Cancel Authorization Subscription (Version 1_1)

Additionally, to cumulated subscription refund described in 6.3.1 it is also possible to refund each single payment during or after subscription lifetime individually. Therefore, the AuthSubscription update resource API has been extended with a 1_1 version.
In order to allow individual authSubscription payment refunds, the following API request sequence is required to retrieve individual payment ID's to be refunded.

- retrieve authSubscription first payment ID (start Abo), extract payment ID ("`extTransactionId`") from response body
- renew authSubscription using API version 1_1 in order to get back the renewal payment ID in HTTP response header 'Location'.
- refund each payment ID individually using the **Direct Payment resource refund API** (6.2.1).

6.3.2.1 Retrieve authSubscription first payment ID (Version 1_1)

URI: $BASE_URL/authsubscriptions/1_1/{authSubscriptionId}/firstPayment

HTTP-Method: GET

Request Headers: X-Request-Id, X-CE-Client-Specification-Version

Request Body: N/A

Query Parameters: N/A

Response-Headers: Content-Length, Content-Type

Response Body: application/vnd.ch.swisscom.easypay.direct.payment+json

HTTP Response-Codes:

- 200, OK: The first payment for requested authSubscriptionID in HTTP response body
- 404, Not Found: Occurs, if the requested authSubscriptionId was not found
- 500, Internal Server Error : Occurs because of some internal server error.

Example:

Request:

```
GET https://easypay-staging.swisscom.ch/ce-rest-
service/authsubscriptions/1_1/7f4811eb-82ef-4568-80e4-7cd5899ce89e/firstPayment


Request Headers:
Connection: keep-alive
```

Response:

```
Response Headers:
HTTP/1.1 200 OK
Content-Type: application/vnd.ch.swisscom.easypay.direct.payment+json
Date: Wed, 04 Apr 2012 15:31:21 GMT


Response Body:
{
   "extTransactionId":"CA9B2627-0B6A-45B2-A270-9C7C6D85ABDE",
   "isSilentAuthenticated":false,
   "amount":"1.1",
   "status":"COMPLETED",
   "formattedMsisdn":null
}
```

6.3.2.2 Renew authSubscription (Version 1_1)

URI: $BASE_URL/authsubscriptions/1_1/{authsubscriptionId}

HTTP-Method: PUT

Request Headers: X-Merchant-Id, X-Request-Id, X-CE-Client-Specification-Version

Request Body: application/vnd.ch.swisscom.easypay.authsubscription+json

Query Parameters: N/A

Response-Headers: Content-Length, Content-Type, Location

Response Body: application/vnd.ch.swisscom.easypay.message.list+json

HTTP Response-Codes:

- 200, OK
- 400, Bad Request: Some Error occurred, the error message is given within the response.
    - Possible Errors (code on message object):

- OPERATION_INVALID
- INVALID_AMOUNT_FORMAT
- MISSING_MANDATORY_FIELD
- SUBSCRIPTION_ALREADY_CANCELED
- RENEWAL_TO_EARLY
- TRANSACTION_ALREADY_REJECTED
- TRANSACTION_ALREADY_COMMITTED
- TRANSACTION_NOT_COMMITTED
- See more Error Codes in Section 7

- 403, Forbidden
- 404. Not Found
- 415, Unsupported Media Type
- 500, Internal Server Error : Occurs because of some internal server error.

Example:

Request:

```
PUT https://easypay-staging.swisscom.ch/ce-rest-
service/authsubscriptions/1_1/4657e27c-bc70-4a1a-8517-8cdfcc66e0e2


PUT data:
{"operation":"RENEW"}


Request Headers:
Content-Length: 24
X-Request-Id: req1234
Connection: keep-alive
Content-Type: application/vnd.ch.swisscom.easypay.authsubscription+json
Accept: application/vnd.ch.swisscom.easypay.message.list+json
X-Merchant-Id: CH
X-CE-Client-Specification-Version: 1.1
```

Response:

```
Response headers:
HTTP/1.1 200 OK
Location: https://easypay-staging.swisscom.ch/ce-rest-service/payments/DEFCEA64-
```

```
B237-46D9-88FA-2C9EE0073A2D
Content-Length: 0
Date: Wed, 15 Apr 2013 13:54:05 GMT
```

6.3.3 Retrieve Authorize Subscription

Description: Retrieve the Authorize Subscription

URI: $BASE_URL/authsubscriptions/{authSubscriptionId}

HTTP-Method: GET

Request Headers: X-Request-Id, X-CE-Client-Specification-Version

Request Body: N/A

Query Parameters: fields

Response-Headers: Content-Length, Content-Type

Response Body: application/vnd.ch.swisscom.easypay.authsubscription +json

HTTP Response-Codes:

- 200, OK: The subscription status is returned in HTTP response body
- 404, Not Found: Occurs, if the direct payment was not found
- 500, Internal Server Error : Occurs because of some internal server error.

Example:

```
Request:
GET https://easypay-staging.swisscom.ch/ce-rest-service/authsubscriptions/7f4811eb-
82ef-4568-80e4-7cd5899ce89e?fields=isActive,duration,durationUnit


Request Headers:
Connection: keep-alive
```

```
Response:
Response Headers:
HTTP/1.1 200 OK
Content-Type: application/vnd.ch.swisscom.easypay.authsubscription+json
Date: Wed, 04 Apr 2012 15:31:21 GMT
```

```
Response Body:
{
       "duration":1,
       "durationUnit":"WEEK",
       "isActive":true
}
```

6.4 Request Resource

The Request resource provides operations within the requested. The requested Id is supported in all operations and SHOULD be unique per every merchant request. The request Id is set by the merchants.

6.4.1 Retrieve Request Status

Description: Retrieve the status of request

URI: $BASE_URL/requests/{requestId}

HTTP-Method: GET

Request Headers: X-Merchant-Id, X-Request-Id, X-CE-Client-Specification-Version

Request Body: N/A

Query Parameters: fields

Response-Headers: Content-Length, Content-Type

Response Body: application/vnd.ch.swisscom.easypay.request+json

HTTP Response-Codes:

- 200, OK: The subscription status is returned in HTTP response body
- 400, Bad Request: Some Error occurred, the error message is given within the response.
    - Possible Errors (code on message object):
        - `MISSING_MANDATORY_HEADER_PARAMETER`
        - `NON_UNIQUE_RESULT`
- 404, Not Found: Occurs, if the request was not found
- 500, Internal Server Error : Occurs because of some internal server error.

Example Best Case:

```
Request:
GET https://easypay-staging.swisscom.ch/ce-rest-
service/requests/req123ab56?fields=status
```

```
Request Headers:
Connection: keep-alive
X-Merchant-Id: CH
Accept: application/vnd.ch.swisscom.easypay.request+json
```

Response:

```
Response headers:
HTTP/1.1 200 OK
Content-Type: application/vnd.ch.swisscom.easypay.request+json
Date: Wed, 04 Apr 2012 16:17:17 GMT

Response body:
{
    "status": "COMPLETED"
}
```

Example Error Case:

Request:

```
GET https://easypay-staging.swisscom.ch/ce-rest-
service/requests/req1234?fields=status

Request Headers:
Connection: keep-alive
X-Merchant-Id: CH
Accept: application/vnd.ch.swisscom.easypay.request+json
```

Response:

```
Response headers:
HTTP/1.1 400 Bad Request
Content-Type: application/vnd.ch.swisscom.easypay.request+json
Date: Wed, 04 Apr 2012 16:12:35 GMT
Connection: close

Response Body:
{
    "messages": [
        {
```

```
            "message": "The result is not unique, The invalid parameter is specified
in the field-element.",
            "field": "requestId",
            "code": "NON_UNIQUE_RESULT"
        }
    ]
}
```

7 Error Code Mapping

Error Code	Description	Retriable
CE_NOT_AUTHORIZED	General system error.	Yes
CE_NOT_AUTHORIZED_BLOCKED_TOP_STOP_REACHED	This is a spending limit over all services to prevent bill shocks.	Yes
CE_NOT_AUTHORIZED_BARRING_STATE_REACHED	Checks if the customer is not barred (e.g. due to credit under-run, missed payment)	Yes
CE_NOT_AUTHORIZED_BLOCKED_SPENDING_LIMIT_REACHED	Customer has reached spending limits.	Yes
CE_NOT_AUTHORIZED_BLOCKED_CHILD_PROTECTION_SPENDING_LIMIT_REACHED	Under aged customer has reached child protection limit.	Yes
CE_NOT_AUTHORIZED_BLOCKED_NOT_ENOUGH_PREPAID_CREDIT	Checks if customer has enough credit (only for prepaid customers).	Yes
CE_NOT_AUTHORIZED_BLOCKED_ADULT_CHECK_FAILED	Checks if the customer's age (according to contract with Swisscom) is over a certain limit (today > 16).	No
CE_NOT_AUTHORIZED_BLOCKED_CUSTOMER_FOR_ADULT	Customer has the possibility of barring all adult services on his MSISDN and Adult check enabled (isAdultContent=true).	No
CE_NOT_AUTHORIZED_BLOCKED_VAS_NOT_ALLOWED	Swisscom has the possibility of barring all premium services on his MSISDN.	No
CE_NOT_AUTHORIZED_BLOCKED_VAS_USER_NOT_ALLOWED	Customer has the possibility of barring all premium services on his MSISDN.	No
CE_NOT_AUTHORIZED_NOT_ALLOWED_DATA_ABO	The customer MSISDN subscription type is not allowed for premium services.	No
CE_NOT_AUTHORIZED_BLOCKED_PREPAID	The prepaid account is not allowed for this merchant.	No
CE_NOT_AUTHORIZED_BLOCKED_CHILD_PROTECTION	Under aged customer is not allowed to purchase via Easypay for given merchant ID.	No
CE_NOT_AUTHORIZED_BLOCKED_COMPANY	Corporate user of a company that is barring Easypay services.	No

CE_NOT_AUTHORIZED_NOT_SUPPORTED_TELCO	Checks if the MSISDN belongs to an active Swisscom customer (including MVNOs). (Will only be returned to merchant through REST interface, not in case of Checkout page)	No
CE_NOT_AUTHORIZED_SIM_NOT_ACTIVE	Error code if a de-activated SIM is accessed for e.g. subscription renewal or refund. (Will only be returned to merchant through REST interface, not in case of Checkout page)	No
CE_NOT_AUTHORIZED_NOT_FOUND_ONE_CLICK_AUTH	Only in case of oneClick requests through REST interface for authorized merchants.	No
CE_NOT_AUTHORIZED_TOKEN_EXPIRED	Only in case of Checkout page requests and expired SMS token.	No
CE_MISSING_OR_INVALID_MANDATORY_CHECKOUT_PARAMETER	Only in case of Checkout page requests and missing or invalid request parameter(s).	No

8 Signing REST requests

The REST interface has to be protected by a standard security mechanism. The aim of the measure is that the originator of the request can be identified and that no change can be made to the request in transit without detection [non-repudiation, Integrity].

The common understanding today is that the security mechanism should not be mixed with the payload. That means it should be transported in the header parameters.

8.1 Request Headers

The following table shows all header parameters:

Header	Description	Required	Example
Content-Type	the media-type of content send within the message body	YES : in case of available http body, mostly in PUT or POST methods NO: otherwise	application/json
Date	The date of the request for proper format. use: 0000 for UTC+0 0100 for UTC+1	YES: either Date or X-SCS-Date must be set. The X-SCS-Date http header will be taken over the Date header	Mon, 09 Jul 2012 16:50:16 +0000
Content-MD5	The MD5 of payload, http method etc.	YES : in case of available http body, mostly in PUT or POST methods NO: otherwise	qzZDt5/CsNKxoIGdPyGzMQ==
X-SCS-Signature	The signature of the request	YES	iZM87yKKOrDa9f2DvBf4JmmEA14=
X-SCS-Date	The x-scs-date of the request for proper format use: 0000 for UTC+0 0100 for UTC+1	YES: either Date or X-SCS-Date must be set The X-SCS-Date http header will be taken over the Date header	Mon, 09 Jul 2012 16:50:16 +0000
X-Merchant-Id	The Merchant Identification	YES	ABC01

- X-SCS-date: contains the timestamp to protect against reply attacks. For some reasons it's not possible to set the date header in some clients. In this case The date can be set using the X-SCS-date http header field. The Date must be within 20 minutes of the charging-engine server.
- Content-MD5: The base64 encoded MD5 hash value of the http input stream.
- The X-Merchant-Id: is used to identify the secret Key of the merchant
- X-SCS-Signature: The Signature is the Base64 encoded HMAC-SHA1 (as defined in RFC2104) generated HashString. In Pseudo-Code:

  Signature s = base64Encode(hmacSha1(secretKey, utf8Encode(HashString)));

8.2 Creating Signature

A sender using the REST API composes the request and computes a hash of the request using the algorithm for securing REST messages. The signature is computed by applying HMAC-SHA1 on the hash and using the shared secret that maps to the merchant-ID.

Key to signing the message is building the HashString. The HashString has to be built on sender and on resource side identical.

The HashString is computed as follows:

HTTPRequestMethod + '\n' +

Content-MD5 + '\n' +

Content-Type + '\n' +

Date + '\n' +

CanonicalizedResource

Fields	Description
HTTPRequestMethod	One of the five http method types, in uppercase : GET, POST, PUT, DELETE, HEAD
Content-MD5	Only the value is used, not the header name. If a request does not include an HTTP body, this is an empty string.
Content-Type	Only the value is used, not the header name. If a request does not include an HTTP body, this is an empty string.
Date	Only the date value is used, not the header name. X-SCS-Date will be used over Date header (RFC-822). Java Pattern: "`EEE, dd MMM yyyy HH:mm:ss Z`"
CanonicalizedResource	Path and Query portions of the HTTP request URI. The URL has the following Format: **$URL e.g. https://easypay-staging.swisscom.ch/ce-rest-**

	service/payments/8FE316C3-124A-4AF6-A0BB-EB112CEE9929 $BASE_URL: https://easypay-staging.swisscom.ch/ce-rest-service $RESOURCE _PATH: /payments/8FE316C3-124A-4AF6-A0BB-EB112CEE9929 Only the $RESOURCE_PATH is relevant for the CanonicalizedResource sign information. **Note:** do **not** include filter options in $RESOURCE_PATH, i.e. ?fields=status for CanonicalizedResource!

8.2.1 Response Errors

In case of error the Server will response with HTTP 400 Bad Request. The detailed message will be included in the messages response body (see chapter: message list data model) as JSON object.

Following errors can occur:

- MISSING_MANDATORY_HEADER_PARAMETER
- MD5_CHECKSUM_NOT_MATCH
- REQUEST_TIME_ODD
- AUTHORIZATION_ERROR

8.2.2 Signing Examples:

Request:

```
PUT https://easypay-staging.swisscom.ch/ce-rest-service/payments/8FE316C3-124A-4AF6-A0BB-EB112CEE9929
PUT data: {"operation":"COMMIT"}


Request Headers:
Content-Type: application/vnd.ch.swisscom.easypay.direct.payment+json
x-merchant-id: CH
x-scs-signature: hLLpRHf9qWutr90rUahfP8W/JYM=
x-scs-date: Mon, 27 Aug 2012 13:09:46 +0000
content-md5: 6KEzivnrMza/LaW7bg5n5A==
```

Signature Detail:

```
contentMD5b64: 6KEzivnrMza/LaW7bg5n5A==

PUT
6KEzivnrMza/LaW7bg5n5A==
application/vnd.ch.swisscom.easypay.direct.payment+json
```

```
Mon, 27 Aug 2012 13:09:46 +0000
/payments/c868671e-c8e7-4c62-b350-ce18eb1314f6

Secure key:1234
Signature Expected: hLLpRHf9qWutr90rUahfP8W/JYM=
```

8.2.3 PHP Example

```
Content-md5:

$md5array = array(
  "operation" => "COMMIT"
);
$_params = json_encode($md5array);
$vContentMD5 = md5($_params,true);
$vContentMD5 = base64_encode($vContentMD5);
echo $vContentMD5;

Signature:

$vRfcDate  = gmdate('D, d M Y H:i:s +0000', time());
$vContentMD5 = '6KEzivnrMza/LaW7bg5n5A==';
$vSignatureS =
"PUT"."\n".$vContentMD5."\n"."application/vnd.ch.swisscom.easypay.direct.payment+json"."\n".$vRfcDate."\n"."/payments/8FE31
6C3-124A-4AF6-A0BB-EB112CEE9929";
$vXSCSSignature = base64_encode(hash_hmac('sha1',$vSignatureS, "1234",true));
echo $vXSCSSignature;
```

8.2.4 Java Example

```
String signature = new String(Base64.encodeBase64(sign(jsonString, secretKey, Algorithm.HmacSHA1)));

public byte[] sign(String data, String key, Algorithm algorithm)
        throws SignatureException {
            try {
                        return sign(data.getBytes(Constants.DEFAULT_ENCODING),
                                    key.getBytes(Constants.DEFAULT_ENCODING), algorithm);
            } catch (UnsupportedEncodingException e) {
                        throw new SignatureException("Unable to calculate a request signature: " + e.getMessage(), e);
            }
   }

   private byte[] sign(byte[] data, byte[] key, Algorithm algorithm) throws SignatureException {
      try {
         Mac mac = Mac.getInstance(algorithm.toString());
         mac.init(new SecretKeySpec(key, algorithm.toString()));
         return mac.doFinal(data);
      } catch (Exception e) {
         throw new SignatureException("Unable to calculate a request signature: " + e.getMessage(), e);
      }
   }
```

8.3 Additional Online examples on GitHub

More exhaustive REST API signing examples can be found on GitHub repository:

https://github.com/rolandschwarz/easypay-signature-code/