CPU ONLINE PAYMENT INTERFACE

API version 3.0.0 (16 Jan 2019)

History

16. Jan. 2019 version 3.0.0 Tarmo Suominen	New parameters to responses sent by the point-of-sale system. Cancellation of a payment sent to point-of-sale system.
1 Mar. 2017 version 2.1.2 Marko Behm	Language versions for the web shop interface
25 May 2016 version 2.1.1 Marko Behm	*Programmatic payment confirmation* section added to the customer service checkout interface
17 Sept. 2015 version 2.1 Marko Behm	Action parameter added to the message sent to the web shop interface. Payment removal function added to the web shop interface.
16 June 2015 version 2.0 Marko Behm	Online payment interface for the web shop.
27 Feb. 2015 version 1.0 Tarmo Suominen	Online payment interface for the point-of-sale system.

1. Introduction

This document describes the CPU online payment interface. The interface can be used to send product sales events for payment from browser-based source systems to both web shops and the customer service checkout points of the Ceepos system. The difference between these two is the following. Payments made at customer service checkout points are made outside the source system/browser (and possibly the workstation), and its duration can vary. Payments in web shops are made in the same browser window as the source system and by the same user.

The product sales of sent payments have to be paid or cancelled in full. Partial payments are not possible.

2. Customer service checkout points

2.1 The operating principle of the interface

1. The source system forms a connection by sending the parameters of the payment to the checkout server. The parameters and a secret key are used to form a SHA-256 checksum, which is used to check the correctness of the message in the point-of-sale system.

2. The point-of-sale system creates the payment out of the parameters. Depending on the operational mode of the system, it responds to the source system either immediately or after the payment has been handled at the checkout point.

    2.1 If the operational mode of the system is asynchronic point-of-sale system interface and the payment has been created successfully, the response is returned immediately with status 2 (in progress). For more information on payment statuses, see 2.3.1.4.

    2.2 If the operational mode of the system is synchronic point-of-sale system interface, the response is only returned after the payment has been made (status 1) or cancelled (status 0) at the checkout point. For more information on payment statuses, see 2.3.1.4.

3. When a payment has been made or cancelled at a checkout point, the point-of-sale system sends a confirmation of this to the *NotificationAddress* given in the request for creating the payment.

4. The source system checks the correctness of the information sent by the point-of-sale system and marks the status of the invoice as paid or cancelled.

2.2 Sending a payment

Using the parameters in 2.2.1, the source system creates a new request for creating a payment. It forms a UTF 8-coded JSON message and sends it to the point-of-sale server using the HTTP POST method. The header must include “Content-Type: application/json”.

The JSON message is sent to the address /maksu.html. The default port number in the point-of-sale system is 8000.

E.g. http://www.server.fi:8000/maksu.html

The interface also supports the use of the HTTPS protocol. On the response of a sent payment and checking its correctness, see 2.3.1.

2.2.1 Parameters

The parameters cannot include semicolons (;).

Parameter	Description	Limitations	Required
ApiVersion	Version number of the interface (e.g. "3.0.0").		*Required*
Source	Alphanumeric identifier for the source system. This is source-system-specific information created by CPU Oy.		*Required*
Id	Alphanumeric identifier for the payment. An identifier for the payment inside the source system (e.g. order number).	max. length 40 characters	*Required*
Mode	The operational mode of the interface. 1 = asynchronic point-of-sale system interface 2 = synchronic point-of-sale system interface	integer	*Required*
Action	Action to be performed.	value always "new payment"	*Required*
Office	The branch code of the point-of-sale system. A branch code that can be used to limit the visibility of sent payments in the point-of-sale system. If this is empty or not transmitted, the payments are visible at all checkout points.		
Description	Payment-specific free-form description. E.g. customer name + library card number. The description is printed on the receipt as a header before the information on product sales.	max. length 100 characters, no HTML code	
Products	List of product sales events. Information on separate product sales divided by the following product-sales-specific parameters:		*Required*
Code	Alphanumeric product code in the point-of-sale system. Ties the payment to an existing product in the point-of-sale system. Among other things, the product name, tax rate and posting used in bookkeeping are determined automatically in the point-of-sale system by this code.	max. length 25 characters	*Required*
Amount	Number of products (default 1). For refunds, a negative value must be used.	integer	

	Price	Unit price in cents including tax. If this is not entered, the point-of-sale system uses the default price of the product.	integer, always positive	
	Description	Free-form description of product sale. This description is printed on the receipt in connection with product name and price. It can also be reported in the product sales reports of the point-of-sale system.	max. length 100 characters, no HTML code	
	Taxcode	The tax rate code of the point-of-sale system. Among other things, this determines the VAT rate and the tax account used in bookkeeping. Must be entered in situations where the tax rate for the sales is different than the point-of-sale system's default tax rate for the product.	max. length 3 characters	
NotificationAddress		The address of the source system for programme contacts. Using the HTTP POST method, the point-of-sale system sends a response in JSON format to this address. Its parameters are described under 2.3.1.1.	length 1,000 characters	
Hash		Checksum. A SHA-256 checksum calculated from a string including the parameter values and the secret key, used by the point-of-sale system to verify the received payment request.		*Required*

The number of product sales is not limited. The product code of products is required information. The interface will only work when the code finds a product in the product registry of the point-of-sale system. If a tax rate code is entered, it must correspondingly find a tax rate in the point-of-sale system. The use of faulty product and tax codes is checked at the checkout point in connection with the payment. If any are found, the payment status is set to cancelled (status 0). For a particular product sale, the tax code overrides the tax rate code in the product registry of the point-of-sale system. If a product price is not entered, the default sale price defined for the product in the point-of-sale system is used.

2.2.2 Example

The JSON message is edited to improve legibility. Its format does not necessarily correspond to the formatting of the messages returned by the interface.


```
{
	“ApiVersion”: “3.0.0”,
	“Source”: “examplecom”,
	“Id”: “12345”,
	“Mode”: 1,
	“Action”: “new payment”,
	“Office”: “2”,
	“Description”: “Charlie Customer”,
	“Products”: [
		{
			“Code”: “1111”,
			“Amount”: 2,
			“Price”: 100,
			“Description”: “Product-specific info”
		},
		{
			“Code”: “1212”,
			“Price”: 150,
			“Taxcode”: “10”
		}
	],
	“NotificationAddress”: “https://www.example.com/notification-path”,
	“Hash”: “fb7507077cf40ed7d1bd75507cc59d1edccd123944f6ca2607b0f36a2f395a4f”
}
```


In the calculation of the checksum (*Hash* parameter), the value “123” was used as a secret key. For more information on calculating the checksum, see 2.2.3.

2.2.3 Calculating the checksum

The parameters of the message sending the payment are used to calculate a checksum that enables the correctness of the information in the message to be checked in the point-of-sale system. The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated at 2.2.1. The & sign is used to separate the values. The string must also include the parameters included in the message without a value.

Finally, a secret key is added to the string of the parameter values (the example used the value “123”). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

```
3.0.0&examplecom&12345&1&new payment&2&Charlie Customer&
	1111&2&100&Product-specific info&
	1212&150&10&
	https://www.example.com/notification-path&
	123
```

A SHA-256 checksum is calculated from the string:

```
fb7507077cf40ed7d1bd75507cc59d1edccd123944f6ca2607b0f36a2f395a4f
```

2.3 Responses sent by the point-of-sale system

2.3.1 Response of sent payment

When the source system requested the point-of-sale system to create a payment, the point-of-sale system will respond using the information below.

The Status parameter of the response is determined by the mode indicated in the request:

1. If the mode is 1 (*asynchronic*), the status for a successful payment is 2 (*in progress*).
2. If the mode is 2 (*synchronic*), the status for a successful payment is 1 (*payment successful*).

For more information on payment statuses, see 2.3.1.4.

The point-of-sale system adds a *Hash* parameter to the response. The source system must use this to check that the response has been received intact. In some cases of error during the request for creating a payment (status 98 or 99), the *Hash* parameter is not returned. This is the case when the point-of-sale system does not recognise the *Source* identifier sent by the source system in connection with the request for creating a payment.

The response is UTF 8-coded and in JSON format.

2.3.1.1 Parameters

Id	Alphanumeric identifier for the payment. The identifier for the payment internal to the source system and sent in the request for creating a payment.
Status	The status of the payment. For more information, see 2.3.1.4.
Reference	The receipt number for a payment at a checkout point. Not transmitted for payments that are in progress, unsuccessful or cancelled.
Action	Performed action.
Payments	List of payment transactions used to pay the payment. Provided for statistical purposes and if the receipt needs to be printed from the source system. *NB The total sum of the payments may differ from the sum of sent product sales in case there are other transactions registered within the same POS receipt.* Not transmitted for payments that are in progress, unsuccessful or cancelled.
PaymentMethod	Payment method code. For more information, see 2.5.
PaymentSum	Payment sum in cents.
Timestamp	Timestamp of the payment in format of YYYYMMDDHHNN.

PaymentDescription	Additional payment description as preformatted text. May include multiple lines. In case of card payments, this will include receipt lines returned by the card payment terminal.
PaymentPOS	POS number of the customer service checkout point in which the payment was registered.
LoyaltyCard	Loyalty card number of the customer, if registered during the payment process. Not transmitted for payments that are in progress, unsuccessful or cancelled.
Hash	Checksum A SHA-256 checksum calculated from the string including the transmitted parameter values and a secret key. Used to check the correctness of the response message. For more information, see 2.3.1.3.

2.3.1.2 Example

An example of a response from the point-of-sale system to the source system after the former has received information on a new payment. The JSON message is edited to improve legibility. Its format does not necessarily correspond to the formatting of the messages returned by the interface.

Response of asynchronic point-of-sale system:

```
{
        “Id”: “12345”,
        “Status”: 2,
        “Action”: “new payment”,
        “Hash”: “7366aeed4c311b62a777bbfb2645e1be6af3b76d1e7a14981e984861b3669c82”
}
```

Response of synchronic point-of-sale system:

```
{
        “Id”: “12345”,
        “Status”: 1,
        “Reference”: “10456”,
        “Action”: “new payment”,
        “Payments”: [
                {
                        “PaymentMethod”: 4,
                        “PaymentSum”: 250,
                        “Timestamp”: “20190101120000”,
                        “PaymentDescription”: “Card payment details”,
                        “PaymentPOS”: 1
                }
        ],
        “LoyaltyCard”: “”,
        “Hash”: “32c191a8a2e7436886489b3a8ffbc3a3218d25ed2fdb964d6d1164b9f93bea02”
}
```

In the calculation of the checksum (hash), the value “123” was used as a secret key.

2.3.1.3 Calculating the checksum

The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated at 2.3.1.1. The values are separated by the & sign.

Finally, a secret key is added to the string of the parameter values (the example used the value “123”). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

The string for an asynchronic point-of-sale system interface and the SHA-256 checksum calculated:

String: 12345&2&new payment&123
Checksum: 7366aeed4c311b62a777bbfb2645e1be6af3b76d1e7a14981e984861b3669c82

The string for a synchronic point-of-sale system interface and the SHA-256 checksum calculated:

String: 12345&1&10456&new payment&4&250&20190101120000&Card payment details&1&&123

Checksum: 32c191a8a2e7436886489b3a8ffbc3a3218d25ed2fdb964d6d1164b9f93bea02

If the checksum calculated is the same as the *Hash* parameter in the response, the message has been received by the source system intact and directly from the point-of-sale system.

2.3.2 Delayed payment notification

Programmatically sent payment notification after order is successfully paid. Message gets sent to *NotificationAddress* received in the original payment creation message. Contents of delayed notification is similar to contents of *response of sent payment* (2.3.1.).

Delayed payment notification will not be sent if the Status parameter of response of the sent payment message was something other than 1 or 2.

2.3.3 Payment made or cancelled

When a payment has been made or cancelled at a checkout point, the point-of-sale system sends a confirmation of this to the source system. The content and protection of the response are the same as in the *response of sent payment* above. The response is sent using the HTTP POST method to the *NotificationAddress* given as a parameter when the payment was created.

The source system must respond to this with normal HTTP headers:

```
HTTP/1.1 200 OK
Connection: close
```

2.4 Cancelling a payment

Using the parameters in 2.4.1, the source system creates a request for cancelling a payment that was previously sent. It forms a UTF 8-coded JSON message and sends it to the point-of-sale server using the HTTP POST method. The header must include “Content-Type: application/json”.

The JSON message is sent to the address **/maksu.html**. The default port number in the point-of-sale system is 8000.

For example: http://www.server.fi:8000/maksu.html

The interface also supports the use of the HTTPS protocol. The point-of-sale server immediately sends a response to indicate successful cancellation of the payment. The source system must check the correctness of the message sent by the point-of-sale server. *On the response of payment cancellation* and checking its correctness, see 2.4.4.

Cancellation is only possible for payments that have not yet been processed at a checkout point.

2.4.1 Parameters

Parameter	Description	Limitations	Required
ApiVersion	Version number of the interface (e.g. “3.0.0”).		*Required*
Source	Alphanumeric identifier for the source system. This is source-system-specific information created by CPU		*Required*

	Oy.		
Id	Alphanumeric identifier for the payment (that was previously sent). An identifier for the payment inside the source system (e.g. order number).	max. length 40 characters	*Required*
Mode	The operational mode of the interface.	value always 2	*Required*
Action	Action to be performed	value always ”delete payment”	*Required*
Hash	Checksum. A SHA-256 checksum calculated from a string including the parameter values and the secret key used by the point-of-sale system to verify the received payment request.		*Required*

2.4.2 Example

The JSON message is edited to improve legibility. Its format does not necessarily correspond to the formatting of the messages returned by the interface.

```
{
        “ApiVersion”: “3.0.0”,
        “Source”: “examplecom”,
        “Id”: “12345”,
        “Mode”: 2,
        “Action”: “delete payment”,
        “Hash”: “3b0c09271bd66753611d67217d000acb8115d97d7707c7afc7770ebd92bd3f62”
}
```

In the calculation of the checksum (*Hash* parameter), the value “123” was used as a secret key. For more information on calculating the checksum, see 2.4.3.

2.4.3 Calculating the checksum

The parameters of the payment cancellation message are used to calculate a checksum that enables the correctness of the information in the message to be checked in the point-of-sale system. The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated under 2.4.1. The & sign is used to separate the values. The string must also include the parameters included in the message without a value.

Finally, a secret key is added to the string of the parameter values (the example used the value “123”). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

3.0.0**&**examplecom**&**12345**&2&**delete payment**&**123

A SHA-256 checksum is calculated from the string:

3b0c09271bd66753611d67217d000acb8115d97d7707c7afc7770ebd92bd3f62

If the checksum calculated is the same as the Hash parameter in the response, the message has been received by the source system intact and directly from the point-of-sale server.

2.4.4 Response to payment cancellation

Once the source system has sent a payment cancellation request to the point-of-sale server, the server sends an immediate response consisting of the parameters listed under 2.4.4.1. *Status* parameter 1 in the response indicates successful cancellation of the payment.

The point-of-sale system adds a *Hash* parameter to the response. The source system must use this to check that the response has been received intact. In some cases of error during the request for creating a payment (status 98 or 99), the *Hash* parameter is not returned. This is the case when the point-of-sale system does not recognise the *Source* identifier sent by the source system in connection with the request for cancelling a payment.

The response is UTF 8-coded and in JSON format.

2.4.4.1 Parameters

Id	Alphanumeric identifier for the payment. The identifier for the payment internal to the source system and sent in the payment creation request.
Status	The status of the payment. For more information on payment statuses, see 2.3.1.4.
Action	Performed action.
Hash	Checksum A SHA-256 checksum calculated from the string including the transmitted parameter values and a secret key. Used to check the correctness of the response message. For more information, see 2.4.4.3.

2.4.4.2 Example

An example of a response from the point-of-sale system to the source system after the former has received information on a cancellation of a payment. The JSON message is edited to improve legibility. Its format does not necessarily correspond to the formatting of the messages returned by the interface.

```
{
        “Id”: “12345”,
        “Status”: 1,
        “Action”: “delete payment”,
        “Hash”: “87e4b1bb81f59d67955775cdb54a740082485419ddbaf51d10f6783dc4bc50fd”
}
```

In the calculation of the checksum (hash), the value “123” was used as a secret key.

2.4.4.3 Calculating the checksum

The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated at 2.4.4.1. The & sign is used to separate the values. The string must also include the parameters included in the message without a value.

Finally, a secret key is added to the string of the parameter values (the example used the value “123”). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

12345**&**1**&**delete payment**&**123

A SHA-256 checksum is calculated from the string:

87e4b1bb81f59d67955775cdb54a740082485419ddbaf51d10f6783dc4bc50fd

If the checksum calculated is the same as the Hash parameter in the response, the message has been received by the source system intact and directly from the point-of-sale system.

2.5 Payment method codes

Payment method codes returned by the point-of-sale system.

Code	Payment method
3	Cash
4	Card payment
7	Internal sale
8	External invoicing
9	Internal invoicing
10	Smart card
11	Payment from salary
13	Voucher
14	Room billing
16	Other

2.6 Payment statuses

The payment statuses used in the responses of the point-of-sale system:

1 = Payment or deletion made successfully
0 = Payment or deletion was unsuccessful or cancelled
2 = Handling of payment in progress (asynchronic point-of-sale system interface only)
3 = Payment already completed, cannot delete
4 = Payment already deleted
97 = Double Id (*
98 = System error
99 = Faulty payment request

*) If an event has already been sent earlier with the same payment ID, but the content of the message is different (the hash has changed), error code 97 will be returned. If the ID and Hash match an event sent earlier (e.g. a user reloads a page in the browser, sending the same message again), the status of the original event is returned.

2.7 Testing the interface

The customer service checkout interface can be tested with an interface simulator supplied separately by CPU. If you need a simulator, please contact CPU's customer support (asiakastuki@cpu.fi).

3. Web shop

The same web shop environment can be used from several source environments. CPU Oy creates a separate *Source* identifier and secret key for each source environment.

Since personal data and payment information flow through the interface, SSL-encrypted addresses (https) should be used as the return addresses.

3.1 The operating principle of the interface

1. The source system forms a connection by sending the payment parameters to the web shop. The parameters and a secret key are used to form an SHA-256 checksum, which is used to check the correctness of the message in the web shop.

2. The web shop creates an order based on the parameters and proceeds to wait for the customer to complete the payment, for which the customer has 24 hours. Unpaid orders that are older than 24 hours are cleared from the web shop automatically.

3. The web shop immediately includes notification of a new payment in the response message. The status of the payment is normally *in progress (status 2).* The response message contains the URL (payment address), which the customer can use to make the payment.

4. The source system directs the customer to the payment address, after which the web shop may request the customer to provide additional information related to the payment. Next, the customer is directed to the payment page of the payment service provider. The customer then makes the payment.

5. The web shop directs the customer back to the source system based on the return address included as a parameter in the payment creation request.

6. The web shop programmatically notifies the source system about the successful payment using the *NotificationAddress* in the payment creation request.

7. The source system checks the correctness of the parameters in the return addresses and marks the invoice as paid.

3.2 Sending a payment

Using the parameters in 2.2.1, the source system creates a new request for creating a payment. It forms a UTF 8-coded JSON message and sends it to the web shop using the HTTP POST method. The header must include “Content-Type: application/json”.

The JSON message is sent to the address **/maksu.html**.

For example: https://www.palvelin.fi/maksu.html

The web shop returns a response immediately upon notification of a new payment. The source system must check the correctness of the message sent by the web shop and direct the user to the payment address in the return message (PaymentAddress). For more on *the response of a sent payment* and checking its correctness, see 3.2.4.

3.2.1 Parameters

The parameters cannot include semicolons (;).

Parameter	Description	Limitations	Required
ApiVersion	Version number of the API. Use the version number in the title of this document (e.g. “2.1.2”). The interface is compatible downwards inside the main version number (e.g. 2.x).		*Required*
Source	Alphanumeric identifier for the source system. This is source-system-specific information created by CPU Oy.		*Required*
Id	Alphanumeric identifier for the payment. An identifier for the payment inside the source system (e.g. order number).	max. length 40 characters	*Required*
Mode	The operational mode of the interface.	value always 3	*Required*
Action	Action to be performed. * To ensure compatibility with version 2.0, this can be left out of the message. This means that the interface processes the message as the creation of a new payment. If Action is not sent, it will not be returned by the web shop and should not be used in the calculation of the response message checksum, see 3.2.4.1.	value always “new payment”	*Required* *
Description	Payment-specific free-form description. E.g. customer name + library card number. The description is included as the heading preceding the product sale information in the email confirmation sent to the customer following payment.	max. length 100 characters, no HTML code	

Products	List of product sales events. Information on separate product sales divided by the following product-sales-specific parameters:		*Required*
Code	Alphanumeric product code in the web shop. Ties the payment to an existing product in the web shop. Among other things, the product name, tax rate and posting used in bookkeeping are determined automatically in the point-of-sale system by this code.	max. length 25 characters	*Required*
Amount	Number of products (default 1).	positive integer	
Price	Unit price in cents including tax.	positive integer	
Description	Free-form description of product sale. Included for each product line in the email confirmation sent to the customer following payment. It can also be included in the point-of-sale system's product sales reports.	max. length 100 characters, no HTML code	
Taxcode	Tax rate code of the point-of-sale system in the web shop. Among other things, this determines the VAT rate and the tax account used in bookkeeping. Must be entered in situations where the tax rate for the sales is different to the web shop's default tax rate for the product.	max. length 3 characters	
Email	Customer email address. The web shop sends confirmations of payments made to this address. If an email address is not provided as an interface parameter, the web shop will prompt the customer to provide one upon navigation to the payment section.	max. length 100 characters	
FirstName	The customer's first name. If a name is not provided as an interface parameter, the web shop will prompt the customer to provide one upon navigation to the payment section.	max. length 100 characters	
LastName	The customer's last name. If a name is not provided as an interface parameter, the web shop will prompt the customer to provide one upon navigation to the payment section.	max. length 100 characters	
Language	The desired language version of the online payment interface. The available language versions depend on the implementation of the online payment interface.	length 2 characters	

ReturnAddress	The return address back to the source system. The web shop directs the customer to this address after a payment has been made or cancelled and includes the order parameters as normal GET parameters. For more on the parameters, see 3.3.1.	max. length 1000 characters	*Required*
NotificationAddress	The address of the source system for programme contacts. Using the HTTP POST method, the web shop sends a response in JSON format to this address. Its parameters are described under 3.4.1.	max. length 1000 characters	*Required*
Hash	Checksum. A SHA-256 checksum calculated from a string including the parameter values and the secret key used by the web shop to verify the received payment request.		*Required*

The number of product sales is not limited. The product code of products is required information. The interface will only work when the code finds a product in the product registry of the web shop. If a tax rate code is entered, it must correspondingly find a tax rate in the web shop. For a particular product sale, the tax code overrides the tax rate code in the product registry of the web shop. The interface will interrupt order creation under payment status 99 if the product or the possible tax code cannot be found in the web shop. If a product price is not entered, the default sale price defined for the product in the web shop is used.

3.2.2 Example

The JSON message is edited to improve legibility. Its format does not necessarily correspond to the formatting of the messages returned by the interface.

```
{
    “ApiVersion”: “2.1.2”,
    “Source”: “examplecom”,
    “Id”: “12345”,
    “Mode”: 3,
    “Action”: “new payment”,
    “Description”: “Charlie Customer”,
    “Products”: [
        {
            “Code”: “1111”,
            “Amount”: 1,
            “Price”: 100,
            “Description”: “Product-specific info”,
        },
        {
            “Code”: “1212”,
            “Price”: 150,
            “Taxcode”: “10”
        }
    ],
    “Email”: “charlie.customer@example.com”,
    “FirstName”: “Charlie”,
    “LastName”: “Customer”,
    “ReturnAddress”: “https://www.example.com/return-path”,
```

"NotificationAddress": "https://www.example.com/notification-path",
"Hash": "734a651b873a5410d4894ece8261ccd34901942b49871c7c05c68a2a3a6c3561"
}

In the calculation of the checksum (*Hash* parameter), the value "123" was used as a secret key. For more information on calculating the checksum, see 3.2.3.

3.2.3 Calculating the checksum

The parameters of the message sending the payment are used to calculate a checksum that enables the correctness of the information in the message to be checked in the web shop. The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated under 3.2.1. The & sign is used to separate the values. The string must also include the parameters included in the message without a value.

Finally, a secret key is added to the string of the parameter values (the example used the value "123"). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

2.1.2&examplecom&12345&3&new payment&Charlie Customer&
1111&1&100&Product-specific info&
1212&150&10&
charlie.customer@example.com&Charlie&Customer&
https://www.example.com/return-path&https://www.example.com/notification-path&
123

A SHA-256 checksum is calculated from the string:

734a651b873a5410d4894ece8261ccd34901942b49871c7c05c68a2a3a6c3561

3.2.4 Response of sent payment

Once the source system has sent a payment creation request to the web shop, the shop sends an immediate response consisting of the parameters listed under 3.2.4.1. At this stage, the *Status* parameter of a successful payment will be 2, as the web shop will wait for the customer to complete the payment.

The web shop adds a *Hash* parameter to the response. The source system must use this to check that the response has been received intact. In some cases of error during the request for creating a payment (status 98 or 99), the *Hash* parameter is not returned. This is the case when the web shop does not recognise the *Source* identifier sent by the source system in connection with the request for creating a payment.

The response is UTF 8-coded and in JSON format.

3.2.4.1 Parameters

Id	Alphanumeric identifier for the payment. The identifier for the payment internal to the source system and sent in the payment creation request.
Status	The status of the payment. For more information on payment statuses, see 3.6.
Reference	Web shop order number. Payment identifier used internally by the web shop.
Action	Performed action.
PaymentAddress	The payment address which is used to direct the customer to completing the payment. It includes the order identifier (*reference*) and the payment-specific encryption key (*token*) for internal use by the web shop.
Hash	Checksum A SHA-256 checksum calculated from the string including the transmitted parameter values and a secret key. Used to check the correctness of the response message. For more information, see 3.2.4.3.

3.2.4.2 Example

An example of a response from the web shop to the source system after the former has received information on a new payment. The JSON message is edited to improve legibility. Its format does not necessarily correspond to the formatting of the messages returned by the interface.

```
{
        “Id”: “12345”,
        “Status”: 2,
        “Reference”: “10456”,
        “Action”: “new payment”,
        “PaymentAddress”:
          “https://www.example.com/checkout?reference=10456&token=
          3b6fd320a01a672c3a3600d1bcfed5462011de5cc8a9a9c63f987886bc622ece”,
        “Hash”: “2c54b34e2a523fad406b735fa616f72a74b50990bf98d30d94d0afdfe8aa86c3”
}
```

In the calculation of the checksum (hash), the value “123” was used as a secret key.

3.2.4.3 Calculating the checksum

The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated at 3.2.4.1. The & sign is used to separate the values. The string must also include the parameters included in the message without a value.

Finally, a secret key is added to the string of the parameter values (the example used the value “123”). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

> 12345&2&10456&new payment&https://www.example.com/checkout?reference=10456&token=3b6fd320a01a672c3a3600d1bcfed5462011de5cc8a9a9c63f987886bc622ece&123

A SHA-256 checksum is calculated from the string:

> 2c54b34e2a523fad406b735fa616f72a74b50990bf98d30d94d0afdfe8aa86c3

If the checksum calculated is the same as the Hash parameter in the response, the message has been received by the source system intact and directly from the web shop.

3.3 Payment complete

Once the customer has made the payment and returned to the web shop from the payment service provider's payment page, the web shop directs the customer back to the source system. The source system receives a *payment complete* confirmation of this progress.

The response is sent using the HTTP GET method, i.e. conventional linking or redirection.

The *payment complete* response is sent to the ReturnAddress provided as a parameter upon payment creation, and the parameters and protections listed below are added to the address.

3.3.1 Parameters

Id	Alphanumeric identifier for the payment. The identifier for the payment internal to the source system and sent in the request for creating a payment.
Status	The status of the payment. For more information on payment statuses, see 3.6.
Reference	Web shop order number. Payment identifier used internally by the web shop.
Hash	Checksum A SHA-256 checksum calculated from the string including the transmitted parameter values and a secret key. Used to check the correctness of the response message. For more information, see 3.3.3.

3.3.2 Example

An example of navigating back to the source system after a successful payment:

```
https://www.example.com/return-path?
Id=12345&
Status=1&
Reference=10456&
Hash=cf4868d68e5e9ef1b00d7c18e65819027189d1b611a3f7bae90fe5036a195517
```

In the calculation of the checksum (hash), the value “123” was used as a secret key.

3.3.3 Calculating the checksum

The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated under 3.3.1. The & sign is used to separate the values. The string must also include the parameters included in the message without a value.

Finally, a secret key is added to the string of the parameter values (the example used the value “123”). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

```
12345&1&10456&123
```

A SHA-256 checksum is calculated from the string:

```
cf4868d68e5e9ef1b00d7c18e65819027189d1b611a3f7bae90fe5036a195517
```

If the checksum calculated is the same as the Hash parameter in the response, the message has been received by the source system intact and directly from the web shop.

3.4 Programmatic payment confirmation

The payment confirmation is sent to the *NotificationAddress* which was provided upon payment creation and to which the parameters presented under 3.4.1 are added. In terms of its parameters and content, the *programmatic payment confirmation* matches the *payment complete* confirmation. The programmatic confirmation is sent for the first time when the web shop is notified of successful payment. In some cases, this can occur before the customer returns to the source system using the *ReturnAddress*.

The programmatic payment confirmation is not sent for payments which receive a response of sent payment in which the *Status* parameter is other than 2 (see 3.2.4).

The programmatic payment confirmation is sent using the HTTP POST method, in the JSON format and with UTF-8 coding.

The source system must respond to the confirmation with a normal http header 200 status code, which informs the web shop that the payment confirmation has been received. Several attempts will be made to send the payment confirmation until the source system sends a response confirming receipt.

Example of source system headers when the system receives a programmatic payment confirmation:

```
HTTP/1.1 200 OK
Connection: close
```

3.4.1 Parameters

Id	Alphanumeric identifier for the payment. The identifier for the payment internal to the source system and sent in the payment creation request.
Status	The status of the payment. For more information on payment statuses, see 3.6.
Reference	Web shop order number. Payment identifier used internally by the web shop.
Hash	Checksum A SHA-256 checksum calculated from the string including the transmitted parameter values and a secret key. Used to check the correctness of the response message. For more information, see 3.4.3.

3.4.2 Example

Example of the content of a programmatic payment confirmation:

```
{
        “Id”: “12345”,
        “Status”: 1,
        “Reference”: “10456”,
        “Hash”: “cf4868d68e5e9ef1b00d7c18e65819027189d1b611a3f7bae90fe5036a195517”
}
```

In the calculation of the checksum (hash), the value “123” was used as a secret key.

3.4.3 Calculating the checksum

The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated under 3.4.1. The & sign is used to separate the values. The string must also include the parameters included in the message without a value.

Finally, a secret key is added to the string of the parameter values (the example used the value “123”). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

```
12345&1&10456&123
```

A SHA-256 checksum is calculated from the string:

```
cf4868d68e5e9ef1b00d7c18e65819027189d1b611a3f7bae90fe5036a195517
```

If the checksum calculated is the same as the Hash parameter in the response, the message has been received by the source system intact and directly from the web shop.

3.5 Cancelling a payment

Using the parameters in 3.5.1, the source system creates a request for cancelling a payment that was previously sent. It forms a UTF 8-coded JSON message and sends it to the web shop using the HTTP POST method. The header must include “Content-Type: application/json”.

The JSON message is sent to the address **/maksu.html**.

For example: https://www.palvelin.fi/maksu.html

The web shop immediately sends a response to indicate successful cancellation of the payment. The source system must check the correctness of the message sent by the web shop. *On the response of payment cancellation* and checking its correctness, see 3.5.4.

3.5.1 Parameters

Parameter	Description	Limitations	Required
ApiVersion	Version number of the interface (e.g. “2.1.2”). The interface is compatible downwards inside the main version number (e.g. 2.x).		*Required*
Source	Alphanumeric identifier for the source system. This is source-system-specific information created by CPU Oy.		*Required*
Id	Alphanumeric identifier for the payment. An identifier for the payment inside the source system (e.g. order number).	max. length 40 characters	*Required*
Mode	The operational mode of the interface.	value always 3	*Required*
Action	Action to be performed	value always ”delete payment”	*Required*
Hash	Checksum. A SHA-256 checksum calculated from a string including the parameter values and the secret key used by the web shop to verify the received payment request.		*Required*

3.5.2 Example

The JSON message is edited to improve legibility. Its format does not necessarily correspond to the formatting of the messages returned by the interface.

```
{
        “ApiVersion”: “2.1.2”,
        “Source”: “examplecom”,
        “Id”: “12345”,
        “Mode”: 3,
        “Action”: “delete payment”,
        “Hash”: “1c6f688cb117995a7c824066e070884dd8c6555df63be7635a5e7e15ce918fe6”
}
```

In the calculation of the checksum (*Hash* parameter), the value “123” was used as a secret key. For more information on calculating the checksum, see 3.5.3.

3.5.3 Calculating the checksum

The parameters of the payment cancellation message are used to calculate a checksum that enables the correctness of the information in the message to be checked in the web shop. The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated under 3.5.1. The & sign is used to separate the values. The string must also include the parameters included in the message without a value.

Finally, a secret key is added to the string of the parameter values (the example used the value “123”). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

```
2.1.2&examplecom&12345&3&delete payment&123
```

A SHA-256 checksum is calculated from the string:

```
1c6f688cb117995a7c824066e070884dd8c6555df63be7635a5e7e15ce918fe6
```

If the checksum calculated is the same as the Hash parameter in the response, the message has been received by the source system intact and directly from the web shop.

3.5.4 Response to payment cancellation

Once the source system has sent a payment cancellation request to the web shop, the shop sends an immediate response consisting of the parameters listed under 3.5.4.1. *Status* parameter 1 in the response indicates successful cancellation of the payment.

The web shop adds a *Hash* parameter to the response. The source system must use this to check that the response has been received intact. In some cases of error during the request for creating a payment (status 98 or 99), the *Hash* parameter is not returned. This is the case when the web shop does not recognise the *Source* identifier sent by the source system in connection with the request for creating a payment.

The response is UTF 8-coded and in JSON format.

3.5.4.1 Parameters

Id	Alphanumeric identifier for the payment. The identifier for the payment internal to the source system and sent in the payment creation request.
Status	The status of the payment. For more information on payment statuses, see 3.6.
Reference	Web shop order number. Payment identifier used internally by the web shop.
Action	Performed action.
Hash	Checksum A SHA-256 checksum calculated from the string including the transmitted parameter values and a secret key. Used to check the correctness of the response message. For more information, see 3.5.4.3.

3.5.4.2 Example

An example of a response from the web shop to the source system after the former has received information on a cancellation of a payment. The JSON message is edited to improve legibility. Its format does not necessarily correspond to the formatting of the messages returned by the interface.

```
{
        “Id”: “12345”,
        “Status”: 1,
        “Reference”: “10456”,
        “Action”: “delete payment”,
        “Hash”: “bcdcca7335f30a86595fd9edbccdaae12d964bf5493dc8cb20073f08ab2174a5”
}
```

In the calculation of the checksum (hash), the value “123” was used as a secret key.

3.5.4.3 Calculating the checksum

The checksum is calculated from a string that consists of the values of the parameters in the message and a source-system-specific secret key. The values of the parameters have to be added in the order indicated at 3.5.4.1. The & sign is used to separate the values. The string must also include the parameters included in the message without a value.

Finally, a secret key is added to the string of the parameter values (the example used the value “123”). After this, a SHA-256 checksum is formed from the string.

The string has been edited to improve legibility. Real strings cannot include line breaks or formatting:

```
12345&1&10456&delete payment&123
```

A SHA-256 checksum is calculated from the string:

bcdcca7335f30a86595fd9edbccdaae12d964bf5493dc8cb20073f08ab2174a5

If the checksum calculated is the same as the Hash parameter in the response, the message has been received by the source system intact and directly from the web shop.

3.6 Payment statuses

The payment statuses used in the responses of the web shop:

1 = Payment successful/action complete
0 = Payment creation failed or cancelled
2 = Processing of payment in progress
3 = Payment already completed, cannot delete
4 = Payment already deleted
97 = Double Id (*
98 = System error
99 = Faulty payment request

*) If an event has already been sent earlier with the same payment ID, but the content of the message is different (the hash has changed), error code 97 will be returned. If the ID and Hash match an event sent earlier (e.g. a user reloads a page in the browser, sending the same message again), the status of the original event is returned.

3.7 Testing the interface

The web shop interface can be tested using the source system identifier and secret key provided by CPU. Please contact CPU's customer support (asiakastuki@cpu.fi) to receive the credentials.

The test environment is available at http://213.214.168.160/maksu.html

The following resources have been created in the test environment:

Products

Product code (code)	**Name**	**Default price**	**Default tax**
demo_001	Late fee	€10	24%
demo_002	Rent	€10	24%
demo_003	Work performance	€10	24%
demo_004	Invoice	€ 0	24%
demo_005	Donation	€ 0	0 %

VAT rates

Tax rate code (taxcode)	**Name**
24	VAT 24%
14	VAT 14%
10	VAT 10%
0	VAT 0%