Web API

Version 2.5

Revision Date: September 15, 2017

An Important Note to Merchants and System Integrators

Due to Payment Card Industry Data Security Standards (PCI-DSS), certain API calls *must not* be called using HTTPS GET. These methods are any which have "cardnum" in their parameter list. When you include your customer's card number in a URL the card number will be logged in web server log files in unencrypted clear-text which violates PCI-DSS. ()

Any call made to one of these methods using HTTPS GET will return -99 and/or the text "This method accepts requests by POST only."

This only applies to RESTful calls to the web API. SOAP integrators do not need to worry about this.

Web API 2.5

The Convenient Payments Web API is a public interface which allows developers to manage customers and their payments online. The service can be used to integrate payments into software systems. In this document the Convenient Payments Web API may be referred to simply as "webapi."

Developers may connect to the web services using any programming language or environment they desire as long as they (or their IDE) are familiar with SOAP, REST and XML Web Services and/or JSON. The API is completely platform and language neutral.

Due to the sensitive nature of data going back and forth to the payment system only SSL connections (https) to the web services can be used. The server only listens on the Secure Sockets Layer (SSL) port 443.

A CRUDSL (**C**reate, **R**ead, **U**pdate, **D**elete, **S**earch, **L**ist) interface is provided for customers and payments. All cust_ and payment_ and list_ methods required authentication.

Methods

1. version()
2. country_code()
3. country_info()
4. fee()
5. card_valid()
6. card_expired()
7. card_type()
8. card_info()
9. card_decode()
10. merchant_read()
11. bank_exists()
12. bank_name()
13. bank_info()
14. list_bank_returns()
15. list_bank_payments()
16. list_card_payments()
17. list_payments()
18. cust_create()
19. cust_read()
20. cust_update()
21. cust_update_bank()
22. cust_update_card()
23. cust_search()
24. payment_create()
25. payment_read()
26. payment_search()
27. card_payment_restricted()
28. card_authenticate()
29. card_payment()
30. payment_capture()
31. card_swipe()
32. payment_voidable()
33. payment_refundable()
34. payment_void()
35. payment_refund()
36. payment_reverse()
37. bank_payment()
38. batch_create()
39. batch_read()

40. batch_update()
41. schedule_create()
42. schedule_read()
43. schedule_update()
44. schedule_preview()
45. schedule_delete()
46. list_origins()
47. list_customers()
48. email_receipt()
49. get_iframe_url_list()
50. add_iframe_url()
51. remove_iframe_url()
52. verify_bank_account()

Authentication

Most of the methods in the API require the developer to provide authentication credentials. When calling these methods the first two arguments supplied will always be your *Merchant Key* and *API Key*.

- Merchant Key: This is a string of digits assigned to each merchant when their payment processing account was created. It is a permanent assignment. In this document the Merchant Key will be referred to as "merchantkey."
- API Key : This is a key string which is assigned to your merchant setup. In this document the API Key will be referred to as "apikey." Your apikey should be kept private and never shared with anyone. You should protect it just as you would a password.

Together the merchantkey and apikey work as a username/password pair which the system will use to authenticate calls made to it and link to the merchant's CPTeller account.

For security reasons the Convenient Payments Web API is disabled for each merchant by default. It is very easy to have it enabled for your account—all you need do is send an email to support@convenientpayments.com from the email address on record and an apikey will be assigned to your account, the API will be enabled, and a message will be sent to the owner of the account informing them that the programming API for their CPTeller account has been activated.

Tokens

When you create customers and payments "tokens" will be returned which you can store for future reference.

How to Use the Web API

Method 1: Background Processing

You can create payments using the web API for background processing. Some reasons for doing this would be future dating of payments—creating payments to be processed on a particular date, or processing of Bank (ACH) payments. Bank payments take several days to complete and must be processed in the background.

Creating a payment for background processing using the web API involves two steps:

1. Create a customer by calling cust_create(). Successful invocation of cust_create() will return a token in the form of a customer ID called "custid."
2. Create a payment by calling payment_create() and providing the custid returned from cust_create() as a parameter.

Method 2: Real-Time Processing

Many if not most developers will want to make a single call to the web API to authorize a card payment immediately, and have returned to them the results of the transaction. The following methods provide this functionality:

card_payment()
swipe_payment()

Both the card_payment() and swipe_payment() methods process card payments in real-time and return the authorization results, including the customer token (custid) and the payment token (paymentid.)

There is also a single-call method to create ACH payments: bank_payment() which creates the customer record and initiates the ACH payment processing (which is handled over several days in the background.)

SOAP or REST

All of the methods provided in the webapi may be invoked using either SOAP or RESTful methods.

(The only RESTful actions provided are GET and POST. This API does not implement a full complement of RESTful methodologies (PUT, DELETE, etc.) but the 'methods' in the API can be called directly from javascript or a browser. This is what is meant by RESTlets.)

Return Formats

By default the return format for non-SOAP webapi calls is "json."

The default return format of any webapi method may be overridden by providing an additional optional parameter *__bdreturnformat={wddx|plain|json|jsonp}* (Note: There are two underscores at the start of __bdreturnformat.)

Depending on your usage of the webapi you may want different return formats in various situations. Not all formats are supported for all method calls. If you attempt to use a return format which is not supported an exception will be thrown. For example, making a call to the bank_info method and specifying a "plain" return format will throw an exception because the method returns a complex data structure.

Example Usage

To call the version() method from a web browser invoke the following url:

https://secure.cpteller.com/api/25/webapi.cfc?method=version

The browser will display: "2.5.n" *where n is the revision number of the most recent change.*

This is an example of calling the method as a RESTlet. SOAP calls are more complicated and normally the developer has access to a library which will act as a proxy between the client and the server. Microsoft's Visual Studio is an example of this and working with the web API in Visual Studio is very easy.

You have to start somewhere. You can familiarize yourself with the webapi very quickly with your web browser. It is best to do this either with your account in test mode or preferably using the Convenient Payments test server. For testing purposes it is recommended you get set up with a test account in the test environment.

Visual Studio

To use the webapi from visual studio is also quite easy:

1. In the solution explorer right-click on "Service References" and select "Add Service Reference"
2. In the "Add Service Reference" dialog window type in the URL of the webapi WSDL interface: https://secure.cpteller.com/api/25/webapi.cfc?wsdl and click "Go."
3. The web service will be located and the public methods shown in the window. In the "Services" box expand "webapiService" and select "webapi." If you have done this correctly you will see the public methods this service exposes in the "Operations" box on the right.
4. Give the service a name in the "Namespace" box at the bottom of the window and click "OK."
5. The webapi is now a part of your project and you can immediately begin using the public features of it (even if you don't have a CPTeller account some of the generic methods will be accessible.) Declare a webapi object such as:

```
var api = new webapi.webapiClient();
```

That's how easy it is. All of the hard work of dealing with SOAP envelopes etc. is done for you.

Public Methods

Public methods may be invoked without authentication (merchantkey/apikey not required.) The public methods and their return types are listed here:

1. version()
2. bank_exists()
3. bank_name()
4. bank_info()
5. card_valid()
6. card_expired()
7. card_type()
8. card_info()
9. country_code()
10. country_info()

What is a Struct?

Methods that return the *Struct* data type return multiple parameters in a name=value pairing. These are easy to parse if you request the return format to be either JSON or WDDX.

In C# the struct data object takes the form of a StructMap object. Here is an example of how to look up your data elements in a StructMap in C#:

```
api.StructMap info = api.bank_info("011000015");

string name = info.item[getKeyIndex(info, "name")].val.ToString();
string address = info.item[getKeyIndex(info, "address")].val.ToString();
string city = info.item[getKeyIndex(info, "city")].val.ToString();
string state = info.item[getKeyIndex(info, "state")].val.ToString();
string zipcode = info.item[getKeyIndex(info, "zipcode")].val.ToString();
string phone = info.item[getKeyIndex(info, "phone")].val.ToString();

Console.WriteLine(name + "\n"
      + address + "\n"
      + city + ", "
      + state + " "
      + zipcode + "\n"
      + phone);

static int getKeyIndex(api.StructMap map, string keyname)
{
      int retval = -1;
      for (int i = 0; i < map.item.Length; i++)
      {
          if (map.item[i].key.ToString() == keyname)
          {
             retval = i;
             break;
          }
      }
      return retval;
}
```

Looking up a name/value pair in C#

The methods that make up the webapi are described in the following pages.

Method

String version()

Parameters

This public method takes no arguments

Returns

This method returns the current webapi software revision as a string.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=version

Returns: "2.5.n" *where n is a minor revision number.*

Python Example:

```
import requests
url = "https://secure.cpteller.com/api/25/webapi.cfc"
r = requests.get(url, {'method':'version'})
print r.text
```

Output: "2.5.2"

Method

String country_code(code, format)

Parameters

Argument	*Type*	*Required*	*Default*	*Description*
code	String	Yes		A county code in ISO-2, ISO-3, or ISO-Numeric format.
Format	String	No	"ISO-3"	Return format

Returns

A string in the requested format, or an empty string if the country passed in the "code" parameter is not a valid country code.

Notes

This method can be used to convert any country code to the desired format.

Examples

https://secure.cpteller.com/api/25/webapi.cfc?method=country_code&code=840

"USA"

https://secure.cpteller.com/api/25/webapi.cfc?method=country_code&code=840&format=3

"USA"

https://secure.cpteller.com/api/25/webapi.cfc?method=country_code&code=CAN&format=N

"124"

https://secure.cpteller.com/api/25/webapi.cfc?method=country_code&code=USA&format=2

"US"

Method

Struct country_info(code)

Parameters

Argument	*Type*	*Required*	*Description*
code	String	Yes	A county code in ISO-2, ISO-3, or ISO-Numeric format.

Returns

A struct with the following information:

Name	*Type*	Description
iso_2	String	Country code of “code” parameter in ISO-2 format.
iso_3	String	Country code of “code” parameter in ISO-3 format.
iso_numeric	Integer	Country code of “code” parameter in ISO-Numeric format.
name	String	Country name

Notes

This method returns a struct containing information regarding the country code passed to it. If the country code is invalid an empty struct will be returned.

Examples

https://secure.cpteller.com/api/25/webapi.cfc?method=country_info&code=US


```
{"iso_2":"US","iso_3":"USA","iso_numeric":840,"name":"United States of America"}
```


https://secure.cpteller.com/api/25/webapi.cfc?method=country_info&code=BOGUS


```
{"iso_2":"","iso_3":"","iso_numeric":0,"name":""}
```

Method

Boolean bank_exists(string routingnum)

Parameters

Argument	*Type*	*Required*	*Description*
routingnum	String	Yes	The bank routing number

Returns

A Boolean value: true if the bank exists or false if not.

Notes

This is a public method, meaning it requires no authentication arguments.
A bank "exists" if it is a Federal ACH participating depository financial institution, meaning it has a routing/ABA number and is a Federal ACH (Automated Clearing House) participant. Convenient Payments maintains a list of all participating FedACH DFIs which is updated on a weekly basis.

Example

Entering the following into a web browser:

https://secure.cpteller.com/api/25/webapi.cfc?method=bank_exists&routingnum=011000015

returns the following:

true

Method

Boolean bank_name(string routingnum)

Parameters

Argument	*Type*	*Required*	*Description*
routingnum	String	Yes	The bank routing number

Returns

A string: The name of the bank referred to by *routingnum*.

Notes

This is a public method, meaning it requires no authentication arguments.
A bank "exists" if it is a fedach participating depository financial institution, meaning it has a routing/ABA number and is a Federal ACH (Automated Clearing House) participant. Convenient Payments maintains a list of all participating FedACH DFIs which is updated on a weekly basis.

Example

Entering the following into a web browser:

https://secure.cpteller.com/api/25/webapi.cfc?method=bank_name&routingnum=011000015

returns the following:

"FEDERAL RESERVE BANK "

Method

Struct bank_info(string routingnum)

Parameters

Argument	*Type*	*Required*	*Description*
routingnum	String	Yes	The bank routing number

Returns

A structure type with the following elements:

1. Routingnumber
2. Name
3. Address
4. City
5. State
6. Zipcode
7. Phone
8. Newroutingnumber

Notes

The "newroutingnumber" element contains data when a new routing number has been assigned to the bank. Although the old routing number still works, the Fed is telling you to start using the new one.
This is a public method.
The address and phone information provided is the corporate or ACH address of the bank, and not necessarily the address and phone number of your customer's local branch.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=bank_info&routingnum=011000015

{"routingnumber":"011000015","name":"FEDERAL RESERVE BANK","address":"1000 PEACHTREE ST N.E.","city":"ATLANTA","state":"GA","zipcode":"30309","phone":"866-234-5681","newroutingnumber":""}

Method

struct bank_payment(…)

This method creates a customer record with bank account information and then creates a bank (ACH) payment for the customer in the given "amount" all in one method call.

Parameters

Argument	*Type*	*Max Len*	*Required*	*Default*	*Description*
merchantkey	String		Yes		Your merchant key
apikey	String		Yes		Your API key
Account	String	50	No	Empty	Your account number reference for the customer.
firstname	String	50	No	Empty	The customer's first or given name.
lastname	String	50	No	Empty	The customer's last name or surname.
address1	String	50	No	Empty	The customer's address.
address2	String	50	No	Empty	Line 2, if any, of the customer's address.
city	String	50	No	Empty	The city where the customer resides.
state	String	2	No	Empty	The state where the customer resides.
zipcode	String	10	No	Empty	The customer's zip code or postal code.
phone	String	20	No	Empty	The customer's phone number.
email	String	50	No	Empty	The customer's email address.
amount	Double	N/A	Yes		The amount of the payment.
invoice	String	50	No	Empty	Your invoice number if any.
routingnum	String	9	Yes		The bank routing number is always exactly 9-digits.
bankacctnum	String	20	Yes		The bank account number.
Bankaccttype	String	1	No	Checking	C: Checking, S: Savings
bankacctname	String	50	Yes		The name on the bank account.

comment	String	1000	No		Any comment you want to provide.

Returns

A structure type with the following elements:

1. Status Return status for the method:
   0: Success
   -1: Authentication (merchantkey, apikey) failed.
   -2: Failed to create the customer record.
   -3: Failed to create the payment record.
2. custid The customer id token
3. paymentid The payment id token

Notes

If you are making a SOAP call to this method you may need to provide placeholders for the parameters which are not required.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=bank_payment&merchantkey=123&apikey=456&firstname=Jerry&lastname=McGuire&address1=123 Main St.&routingnum=124000054&bankacctnum=999999&bankacctname=Jerry F McGuire&amount=100.00

Method

Numeric card_valid(cardnum)

This is a public method meaning no merchantkey or apikey is required. However since this method has a card number as a parameter you are required to call it with HTTPS POST. An attempt to call this method with a GET will result in a -99 error.

Arguments

Argument	*Type*	*Required*	*Description*
cardnum	String	Yes	The credit card number

Returns

This method returns a numeric value with the following meanings:

1 = VALID
0 = NOT VALID
-1 = ACCESS DENIED: (*This method accepts requests via HTTPS POST only.)

Notes

*This method accepts requests via HTTPS POST only. If you attempt to access it using HTTPS GET the -1 value is returned. This necessary due to the fact that all requests are logged in server log files and GET requests are logged with all of the parameters included in the request. This would log credit card numbers in clear-text in server logs (which is a PCI/DSS violation.)
A card number is typically 16 digits in length. It consists of a single-digit Major Industry Identifier, a six-digit Issuer Identification Number, a variable length individual account identifier, and a single check digit calculated using the Luhn algorithm. An American Express card number is typically 15 digits.
This method will only verify that the cardnum is a valid one according to the Luhn algorithm.

Python Example:

```
import requests
url = "https://secure.cpteller.com/api/25/webapi.cfc"
r = requests.postt(url, {'method':'bank_info', 'routingnum':'124000054'})
print r.text
'{"routingnumber":"124000054","name":"ZB, N.A. DBA ZIONS BANK","address":"2200 SOUTH 3270 WEST","city":"WEST VALLEY CITY","state":"UT","zipcode":"84119","phone":"801-974-8800","newroutingnumber":""}'
```

Method

String card_type(cardnum, [format])

This is another public method. No merchantkey or apikey is required. Since this method has a card number as a parameter it must be called using HTTPS POST.

Parameters

Argument	*Type*	*Required*	*Default*	*Description*
cardnum	String	Yes		The credit card number
format	String	No	"s"	"long" or "l" for a long name. "short" or "s" for a short or abbreviated type. (case-insensitive)

Notes

The cardnum may be any part of the credit card number as long as it includes the first digit. The format argument is optional. If omitted "short" is the default. It is case-insensitive. If this parameter is provided but is invalid "Bad Format" will be returned.

Returns

This method returns a string representing the card type:

1. "American Express", or "Amex" (cards beginning with "3")
2. "Visa Card" or "Visa" (cards beginning with "4")
3. "Mastercard" or "Mast" (cards beginning with "5")
4. "Discover Card" or "Disc" (cards beginning with "6")

Example

Python Example:

```
import requests

url = https://secure.cpteller.com/api/25/webapi.cfc
r = requests.post(url, {'method':'card_type', 'cardnum':'4111111111111111', 'format':'short'})
print r.text

"Visa"
```

Method

Integer card_expired(string expdate)

This is a public method. No merchantkey or apikey is required. Also, since there is not a card number as a parameter you may use HTTPS GET to call this method.

Parameters

Argument	*Type*	*Required*	*Description*
expdate	String	Yes	A numeric-string (a string of all digits) representing the expiration date in the format: MMYY

Returns

1 if the card is expired or 0 if the card is not expired (a valid card.)
-1 if one or more of the argument provided is an integer which cannot be combined with the other argument to create a datetime object. (For example: month=13)

Notes

expdate is a numeric-string--it is passed as a string of 4 characters, all of which must be numeric.
The month (first two digits) must be in the range 01-12. (The leading-zero) is required.
The second two digits are the year.

Example

(The expiration date in this example is December 2015.)

https://secure.cpteller.com/api/25/webapi.cfc?method=card_expired&expdate=1215

1

Method

Struct card_info(string cardnum, string expdate)

This is a public method; no merchantkey or apikey is required to call this method, however due to the fact that a card number parameter exists it must be called with HTTPS POST.

Parameters

Argument	*Type*	*Required*	*Description*
cardnum	String	Yes	The credit card number
expdate	String	Yes	Card expiration date in the format: MMYY

Returns

A structure with the following elements

1. Longtype The result of card_type() in long format
2. Shorttype The result of card_type() in short format
3. Valid The result of card_valid()
4. Expired The result of card_expired()
5. Display Display format of card number: NN..NNNN
6. Endswith The last four digits of the cardnum

Notes

This method is a one-stop way to validate a card and get desired information.

Python Example:

```
import requests
url = "https://secure.cpteller.com/api/25/webapi.cfc"
r = requests.post(url, {'method':'card_info', 'cardnum':'4111111111111111', 'expdate':'1220'})
print r.text
```

{"valid":1,"longtype":"Visa Card","shorttype":"Visa","display":"41..1111","endswith":"1111","expired":0,"bin":{"bin":"411111","brand":"VISA","issuer":"JPMORGAN CHASE BANK, N.A.","type":"CREDIT","subtype":"","country":"UNITED STATES","countrycode":"840","countrycode2":"US","countrycode3":"USA"}}

Method

Integer cust_create(…) Create a customer.

Because one of the parameters in this method is a card number it must be called via HTTPS POST.

Parameters

Argument	*Type*	*Required*	*Size*	*Default*	*Description*
merchantkey	String	Yes			Your assigned merchantkey
apikey	String	Yes			Your API key
account	String	Yes	50		Account reference number or string for your own reference
firstname	String	Yes	50		Customer's first name
lastname	String	Yes	50		Customer's last name
address1	String	No	50	Empty String	First line of customer's address
address2	String	No	50	Empty String	Second line of customer's address
city	String	No	50	Empty String	City in which customer resides
state	String	No	2	Empty String	State in which customer resides
zipcode	String	No	10	Empty String	Customer's zip code or postal code
phone	String	No	20	Empty String	Customer's phone number
email	String	No	50	Empty String	Customer's email address
Country	String	No	3	"USA"	ISO-2, ISO-3, or ISO-Numeric country code.
Cardnum	String	No	16	Empty String	Card number
Cardname	String	No	50	Empty String	Name on card
Expdate	String	No	4	Empty String	Card expiry date: MMYY
Routingnum	String	No	9	Empty String	Bank routing/ABA number
Bankacctnum	String	No	17	Empty String	Bank account number

Bankacctname	String	No	50	Empty String	Name on the bank account
Bankaccttype	String	No	1	"C"	C = checking, S = Savings
Pin	String	No	25	Empty String	For use with invoice search (customer portal) a Personal ID Number (or string)

Returns

If a positive integer is returned then the customer has been successfully created in the system and the number is the customer ID or token (custid) which can be used in subsequent actions dealing with this customer.

-1 Authentication failed: merchantkey/apikey not authenticated.
-2 A system error occurred. Possibly because some arguments are not valid.

Method

Struct Cust_read(string merchantkey, string apikey, integer custid)

Read a customer record.

Parameters

Argument	*Type*	*Required*	*Description*
merchantkey	String	Yes	Your assigned merchantkey
apikey	String	Yes	Your API key
custid	Integer	Yes	The customer id or token returned by cust_create() or cust_search()

Returns

This method returns a struct with the following elements:

1. status — Return status of this method call (see notes)
2. account — Customer's account or reference number (your reference number)
3. firstname — Customer's first name
4. lastname — Customer's last name
5. address1 — Line 1 of customer's address
6. address2 — Line 2 of customer's address
7. city — City
8. state — Two-character state abbreviation
9. zipcode — Customer's postal code
10. country — ISO-3 country code of customer's address
11. phone — Customer's phone number
12. email — Customer's email address
13. routingnum — Bank routing/ABA number
14. bankacctnum — Bank account number
15. bankaccttype — Bank account type (C=Checking, S=Savings)
16. cardtype — Credit card type
17. cardnumdisplay — Credit card number in NN..NNNN format
18. cardending — Last 4 digits of customer's credit card number
19. expmonth — Credit card expiration month
20. expyear — Credit card expiration year

Notes

The *status* element of the return struct may have the following values:

1. -1 Authentication failed (check merchantkey/apikey) combination
2. -2 A system error has occurred. If you see this you should contact customer support
3. *Custid* If the status is a positive integer then the cust_read() call has found an item and the other elements in the struct should contain all applicable data.
4. 0 No record was found

If the status element is non-positive then the status element will be the only one present in the struct.

Example

https://dev.cpteller.com/api/25/webapi.cfc?method=cust_read&merchantkey=209121408&apikey=9s87df9s8d&custid=271773

Returns:

{"status":271773,"account":"Doe1","firstname":"John","lastname":"Doe","address1":"123 Main St.","address2":"None","city":"Bogalusa","state":"LA","zipcode":"12345","phone":"123-456-7890","email":"john.doe@johndoe.com","routingnum":null,"bankacctnum":null,"bankaccttype":null,"cardtype":null,"cardnumdisplay":null,"cardending":null,"expyear":null}

Method

Integer cust_update(…) Update a customer record

This method must be called with HTTPS POST because there is a card number parameter.

Arguments

Argument	*Type*	*Required*	*Size*	*Default*	*Description*
merchantkey	String	Yes			Your assigned merchantkey
apikey	String	Yes			Your API key
custid	Integer	Yes			Customer ID or Token
account	String	Yes	50		Account reference number or string for your own reference
firstname	String	Yes	50		Customer's first name
lastname	String	Yes	50		Customer's last name
address1	String	No	50	Empty String	First line of customer's address
address2	String	No	50	Empty String	Second line of customer's address
city	String	No	50	Empty String	City in which customer resides
state	String	No	2	Empty String	State in which customer resides
zipcode	String	No	10	Empty String	Customer's zip code or postal code
phone	String	No	20	Empty String	Customer's phone number
email	String	No	50	Empty String	Customer's email address
country	String	No	3	"USA"	ISO-2, ISO-3, or ISO-Numeric country code.
cardnum	String	No	16	Empty String	Card number
cardname	String	No	50	Empty String	Name on card
expdate	String	No	4	Empty String	Card expiry date: MMYY
routingnum	String	No	9	Empty String	Bank routing/ABA number

bankacctnum	String	No	17	Empty String	Bank account number
bankacctname	String	No	50	Empty String	Name on the bank account
bankaccttype	String	No	1	"C"	C = checking, S = Savings
pin	String	No	25	Empty String	For use with invoice search (customer portal) a Personal ID Number (or string)

Returns

1. -1 Authentication failed (check merchantkey/apikey) combination
2. -2 A system error has occurred. If you see this you should contact customer support
3. *Custid* If the status is a positive integer then the cust_read() call has found an item and the other elements in the struct should contain all applicable data.
4. 0 No record was found
5. *Custid* If the return value is a positive number then the call to cust_update() succeeded. The return value should be the same as the custid you provided.

Notes

If you are calling this method as a RESTlet then the only arguments that are required are the ones you are changing (other than merchantkey/apikey.)
SOAP requires you provide all arguments, but you can pass them as empty strings and they will be ignored.
cust_update() only updates the customer's personal information. To update a customer's bank account or credit card information see: cust_update_card() and cust_update_bank().

Method

Integer cust_update_card(…) Update customer credit card information

This method must be called with HTTPS POST because it has a card number parameter.

Arguments

Type	Name	Len	Description
String	merchantkey	50	Your assigned merchantkey
String	apikey	50	Your API key
Integer	custid	N/A	The customer ID as returned by cust_reate()
String	cardnum	16	Credit card number
String	cardname	50	Name on credit card
String	expdate	4	Expiration Date (MMYY)

Returns

1. -1 Authentication failed (check merchantkey/apikey) combination
2. -2 A system error has occurred. If you see this you should contact customer support
3. 0 No record was found
4. *Custid* If the return value is a positive number then the call to cust_update_card() succeeded. The return value should be the same as the custid you provided.

Notes

You only need supply those parameters (except merchantkey/apikey) which you are going to change . If you are using the SOAP interface empty strings (even for the expiration dates) will cause those items to be ignored.
If you are calling cust_update_card() as a RESTlet you may also supply empty strings to ignore arguments, or omit them.

Example

This example demonstrates a call to this method using a simple HTTPS POST in C#.

```
using System;
using System.Collections.Specialized;
using System.Text;
using System.Net;

namespace SampleCode
{
    class CustUpdateCard
    {
        static void Main(string[] args)
        {
            using (var wb = new WebClient())
            {
                var url = "https://secure.cpteller.com/api/25/webapi.cfc";
                var data = new NameValueCollection();
                data["method"] = "cust_update_card";
                data["merchantkey"] = "YOURMERCHANTKEY";
                data["apikey"] = "YOURAPIKEY";
                data["custid"] = "1234567";
                data["cardnum"] = "4111111111111111";
                data["cardname"] = "Captain Kangaroo";
                data["expdate"] = "0519";

                var response = wb.UploadValues(url, "POST", data);

                Console.WriteLine("Result: {0}", Encoding.ASCII.GetString(response));
            }
        }
    }
}
```

Method

Integer cust_update_bank(…)

Update customer bank account information

Arguments

String	merchantkey	50	Your assigned merchantkey
String	apikey	50	Your API key
Integer	custid	N/A	The customer ID as returned by cust_create()
String	routingnum	9	Bank routing/ABA number (always exactly 9-digits)
String	bankacctnum	20	Bank account number
String	bankaccttype	1	Bank account type (“C” for checking, “S” for savings)

Returns

1. -1 Authentication failed (check merchantkey/apikey) combination
2. -2 A system error has occurred. If you see this you should contact customer support
3. *Custid* If the status is a positive integer then the cust_update_bank() call has found an item and the other elements in the struct should contain all applicable data.
4. 0 No record was found
5. *Custid* If the return value is a positive number then the call to cust_update_bank() succeeded. The return value should be the same as the custid you provided.

Notes

You only need supply those parameters (except merchantkey/apikey) which you are going to change . If you are using the SOAP interface empty strings (even for the expiration dates) will cause those items to be ignored.
If you are calling cust_update_bank() as a RESTlet you may also supply empty strings to ignore arguments, or omit them.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=cust_update_bank?merchantkey=23423&apikey=92873492834custid=9999999&bankaccttype=S

(For the customer at custid=9999999 change the bank account to a savings account)

Method

Integer cust_search(…)

Search for a customer record

Arguments

Type	Name	Len	Description
String	merchantkey	50	Your assigned merchantkey
String	apikey	50	Your API pass key
String	account	50	Account reference number or string for your reference
String	firstname	50	Customer's first name
String	lastname	50	Customer's last name
String	address1	50	First line of customer's address
String	address2	50	Second line of customer's address
String	city	50	City
String	state	2	Two-character state abbreviation
String	zipcode	10	Postal code
String	country	3	ISO-3 country code of address (default = "USA")
String	phone	20	Customer's phone number
String	email	50	Customer's email address
String	routingnum	9	Bank routing/ABA number
String	bankacctnum	20	Customer's bank account number
String	cardname	50	Name on customer's credit card
String	cardtype	1	Type of credit card (first digit in card number)
String	cardending	4	Last for digits of customer's credit card number

Returns

1. -1 Authentication failed (check merchantkey/apikey) combination
2. *Custid* If the status is a positive integer then the cust_search() call has found a single item matching your search criteria and this is the customer id of that record.
3. 0 No record was found
4. *-N* If the return value is a negative number then the absolute value of the number is the number of items found. For example -5 means your search found 5 items.

Notes

If you are calling this method as a RESTlet then the only arguments that are required are the ones you are using in the search (other than merchantkey/apikey.)
SOAP requires you provide all arguments, but you can pass them as empty strings and they will be ignored.

Method

Integer payment_create(…)

Create a single payment.

Arguments

Type	Name	Len	Description
String	merchantkey	50	Your assigned merchant key
String	apikey	50	Your API pass key
Integer	custid	10	Customer ID of the customer making the payment
Float	amount	10	Amount of payment to be made
String	type	1	Type of payment (“A” for ACH or bank, “C” for credit card)
String	date	10	Date on which the payment is to be made
String	invoice	25	Reference invoice (for your own use)
String	seccode	3	Type of authorization you received (see notes)
String	comment	100	Free form comment (anything you want)

Returns

If a positive integer is returned then the payment has been successfully created in the system and the number is the payment ID or token.

-1	Authentication failed: merchantkey/apikey not authenticated.
-2	A system error occurred. Possibly because some arguments are not valid.
-3	Invalid payment type (only “A” and “C” are allowed)
-4	Invalid routing/ABA number in the customer record
-5	Missing bank account number in the customer record
-6	Invalid bank account type in the customer record
-7	Credit card is expired
-8	Missing credit card number
-9	
-10	Invalid payment amount

Notes

This method creates a payment for batch processing. The return values are intended to check all of the parameters of the payment to make sure the batch processor can handle it with the greatest likelihood of success. The actual payment will be processed on the server back-end. The result of the payment will appear in reports.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=payment_create?merchantkey=123&apikey=12
12312&custid=12314&amount=100.00&type=A&date=12/12/12&invoice=983-
13&seccode=TEL&comment=Superman is making a payment of $100 on December 12 2112

982734

Method

Struct payment_read(string merchantkey, string apikey, int paymentid)

Read a payment record

Arguments

Type	Name	Len	Description
String	merchantkey	50	Your assigned merchantkey
String	apikey	50	Your API pass key
Integer	paymentid	10	Payment ID

Returns

This method returns a struct with the following elements:

1. status — Return status of the call to payment_read()
2. custid — Customer ID of the customer making the payment
3. entrydate — Date on which this payment record entered the system
4. amount — Payment amount
5. invoice — Invoice (provided with your call to payment_create())
6. state — Current state of payment record (see notes)
7. comment — Comment supplied with call to payment_create()
8. type — Payment Type
9. authdate — Authorization date (for credit card payments)
10. authresult — Credit card authorization result (A=approved, D=declined, E=Error)
11. authstatus
12. returncode — ACH payment return code
13. returnreason — ACH payment description of return code
14. returndate — The date on which the payment was returned.
15. seccode — ACH payment SEC code (“PPD”, “TEL”, “WEB”)
16. scheduleid — Schedule ID if the payment has a payment schedule
17. paymentdate — Date on which payment is to be made
18. hold — 1=Payment is on hold and won’t be processed until hold=0
19. void — 1=Payment has been voided or cancelled
20. voiddate — Date on which payment was voided or cancelled
21. feepaymentid — The paymentid of a fee payment if a convenience fee was processed.
22. feeamount — The amount of the fee if the payment is a fee payment.
23. cardname — The name on the card if this is a card payment.
24. Cardnumber — The card number (masked)

25. Refpaymentid The paymentid of a refundpayment if a refund was issued.
26. Ipaddress The ip address of the person making the payment if available.
27. Username The username of the person entering the payment if available.
28. Origin The origin of the payment.
29. Avsdata The address verification system match information if available.
30. Refundamount The amount of this payment which has been refunded.
31. Batchid The batchid of this payment if it is part of a batch.
32. Routingnum The routing number if this is a bank payment.
33. Bankacctnum The bank account number if this is a bank payment.
34. Bankaccttype C = Checking, S = Savings
35. Effectivedate The date on which this payment is effective in the payers bank account.
36. Disbursementid A token representing the disbursement which included this payment.
37. Disbursementdate The date on which the payment disbursed.
38. Nocroutingnum The new routing number to use for future payments to the individual.
39. Nocbankacctnum The new bank account number for future payments to the individual.
40. Nocindividualname The new name of the individual for future payments
41. Scheduleid A token representing the payment schedule the payment belongs to.

Notes

The status member of the return struct will contain a positive number if the record was found. This is the paymentid as provided in the request.
If the payment was not found status will be 0 (zero).
If authentication failed -1 is returned in status.

Method

Integer payment_search(…)

Search for a payment record

Arguments

Type	Name	Len	Description
String	merchantkey	50	Your assigned merchantkey
String	apikey	50	Your API pass key
Integer	custid	10	Customer ID of the customer in the search
Float	amount	10	Amount of the payment to search for
String	invoice	25	Your reference or invoice number
String	returncode	3	Return code of ACH payment being sought
String	authcode	10	Authorization code being sought

Returns

1. -1 Authentication failed (check merchantkey/apikey) combination
2. *Custid* If the status is a positive integer then the cust_search() call has found a single item matching your search criteria and this is the customer id of that record.
3. 0 No record was found
4. *-N* If the return value is a negative number then the absolute value of the number is the number of items found. For example -5 means your search found 5 items.

Notes

If you are calling this method as a RESTlet then the only arguments that are required are the ones you are using in the search (other than merchantkey/apikey.)
SOAP requires you provide all arguments, but you can pass them as empty strings and they will be ignored.

Example

http://secure.cpteller.com/api/25/webapi.cfc?method=payment_search&merchantkey=123123&apikey=982734928&authcode=6AB123

121236

Method

Struct list_payments()

Arguments

Type	Name	Len	Description
String	merchantkey	50	Your assigned merchantkey
String	apikey	50	Your API pass key
Float	datefilter	10	Field to apply fromdate and todate to
String	fromdate	25	Return items starting from this date (default is today)
String	todate	3	Return items up to this date (default is today)

Returns

This method returns a structure containing the following elements:

Status	-1 = Authentication failed (check merchantkey/apikey) combination *n* = A positive integer represents the number of items found. 0 = No records were found. -2 = Invalid fromdate—the date format is: MM/DD/YY -3 = Invalid todate. -4 = Invalid date range—fromdate must be less-than or equal-to todate. -5 = Invalid date filter (see date filter values)	
Message	A description of the error if status is a negative number	
Fromdate	The fromdate provided in the request	
Todate	The todate provided in the request	
Items	A list of structs. If Status is a whole number it represents the number of Items. Each struct in Items consists of the following members:	
	Paymentid	Token or Reference ID or token of this payment item
	Customerid	Token or Reference ID of the customer record for this item
	Scheduleid	Token or Reference ID of payment schedule if any
	Departmentid	Token or Reference ID of Department or Category if any
	Refpaymentid	Token or Reference ID of an associated payment if any

	Feepaymentid	Token or Reference ID of an associated fee payment if any
	Batchid	Token or Reference ID of Batch payment belongs to if any
	Status	Payment Status * (See notes)
	Paymentdate	Date on which payment was processed
	Entrydate	Date on which the payment has effect on your customer
	Invoice	Invoice information for the payment if any
	Amount	Dollar amount of payment
	Feeamount	Dollar amount of fee payment if any
	Refundamount	Dollar amount refunded if any
	Cardnum	Card Number if this is a card payment
	Cardname	Name on Card if this is a card payment
	Authcode	Authorization Code
	Authstatus	Authorization Status
	Avsdata	Address Verification System match information
	Origin	Origin of payment (*See Notes)
	Voided	1=Payment was voided, 0=Payment not voided
	Voiddate	Date on which payment was voided if any
	Declinereason	If payment declined, reason for decline if any was given
	Returndate	Date on which an ACH item returned if any
	Returncode	ACH Returncode if any
	Settlementdate	Date on which item settled if any
	Disbursementid	Token or Reference ID of disbursement if any
	NOC	1=There is a Notification Of Change for this item
	Noc_routingnum	New Routing/ABA number if any

	Noc_bankacctnum	New Bank account number if any
	Noc_Bankaccttype	New Bank account type if any
	Paymenttype	Type of payment: 1=ACH, 2=Card
	Account	Your customer account reference for this customer
	Firstname	Customer first name
	Lastname	Customer last name

Notes

Valid Date Filter Values
"eff" Report by effective date (default)
"prc" Report by payment date or process date

Payment Status values and their meanings

0 = New unprocessed payment
1 = Queued ACH payment
2 = Transmitted ACH payment
3 = Returned ACH payment or Declined Card payment
4 = Settled (but not disbursed ACH payment)
5 = Completed

Origin values and their meanings

The Origin of a payment has to do with the method used to enter the payment into the system:

1 = Virtual Terminal
2 = Online Payment Page
3 = Swipecard Terminal
4 = Batch Upload
5 = Web API
6 = Mobile Application
7 = Recurring Payment Schedule
8 = Payment Plan
9 = Payment Wizard
10 = Customer Portal
11 = Advanced Terminal
12 = One Terminal

Example

https://secure.cpteller.com/api/25/webapi.cfc?merchantkey=209121408&apikey=mykey&method=list_payments&fromdate=1/2/14&todate=9/20/14

Method

Struct list_customers()

This method returns a list of customers which match the parameters you provide.

Parameters

Argument	*Type*	*Required*	*Size*	*Default*	*Description*
merchantkey	String	Yes			Your assigned merchantkey
apikey	String	Yes			Your API key
account	String	Yes	50		Account reference number or string for your own reference
firstname	String	Yes	50		Customer's first name
lastname	String	Yes	50		Customer's last name
address1	String	No	50	Empty String	First line of customer's address
address2	String	No	50	Empty String	Second line of customer's address
city	String	No	50	Empty String	City in which customer resides
state	String	No	2	Empty String	State in which customer resides
zipcode	String	No	10	Empty String	Customer's zip code or postal code
phone	String	No	20	Empty String	Customer's phone number
email	String	No	50	Empty String	Customer's email address
Country	String	No	3	"USA"	ISO-2, ISO-3, or ISO-Numeric country code.
CardEnding	String	No	4	Empty String	The last for digits of the card number
Cardname	String	No	50	Empty String	Name on card
CardType	String	No	1	Empty String	Card expiry date: MMYY
Routingnum	String	No	9	Empty String	Bank routing/ABA number
Bankacctnum	String	No	17	Empty String	Bank account number

Notes:

If you do not provide a value for a parameter or you provide an empty string that parameter will not be searched for. The parameters you provide values for will be combined in an "AND" search.

If you provide no parameters then ALL customer records will be listed. *Be careful with this as many merchants have many thousands of customers.*

Python Example:

```
import requests
url = "https://secure.cpteller.com/api/25/webapi.cfc"
data = {
   'method':'list_customers',
   'merchantkey':'3A1013',
   'apikey':'Lucky13',
   'phone':'123-123-4567',
   'firstname':'John',
   'lastname':'Moss'
}
r = requests.post(url, data)
print r.text
```

{"status":5,"message":"","items":[{"customerid":6312829,"entrydate":"12/28/16
14:04:41","account":"","firstname":"John","lastname":"Moss","address1":"123 Elm
St.","address2":"Suite 1","city":"Bugtussell","state":"UT","zipcode":"84092
","country":"USA","phone":"123-123-4567","email":"moss@roothog.net","routingnum":"
","bankacctnum":"","cardname":"John
Doe","cardtype":"4","cardending":"1111"},{"customerid":6312825,"entrydate":"12/28/16
14:04:09","account":"","firstname":"John","lastname":"Moss","address1":"123 Elm
St.","address2":"Suite 1","city":"Bugtussell","state":"UT","zipcode":"84092
","country":"USA","phone":"123-123-4567","email":"moss@roothog.net","routingnum":"
","bankacctnum":"","cardname":"John
Doe","cardtype":"5","cardending":"5353"},{"customerid":6312832,"entrydate":"12/28/16
14:05:22","account":"","firstname":"John","lastname":"Moss","address1":"123 Elm
St.","address2":"Suite 1","city":"Bugtussell","state":"UT","zipcode":"84092
","country":"USA","phone":"123-123-4567","email":"moss@roothog.net","routingnum":"
","bankacctnum":"","cardname":"John
Doe","cardtype":"4","cardending":"1111"},{"customerid":6315127,"entrydate":"12/28/16
15:22:38","account":"","firstname":"John","lastname":"Moss","address1":"123 Elm
St.","address2":"Suite 1","city":"Bugtussell","state":"UT","zipcode":"84092
","country":"USA","phone":"123-123-4567","email":"moss@roothog.net","routingnum":"
","bankacctnum":"","cardname":"John
Doe","cardtype":"4","cardending":"1111"},{"customerid":7018083,"entrydate":"03/08/17
13:16:12","account":"","firstname":"John","lastname":"Moss","address1":"123 Elm

St.","address2":"Suite 1","city":"Bugtussell","state":"UT","zipcode":"84092
","country":"USA","phone":"123-123-4567","email":"moss@roothog.net","routingnum":"
","bankacctnum":"","cardname":"John Doe","cardtype":"4","cardending":"1111"}]}

Method

Struct list_bank_payments()

Arguments

Type	Name	Len	Description
String	merchantkey	50	Your assigned merchantkey
String	apikey	50	Your API pass key
String	filter	20	Type of payment information to return
Float	datefilter	10	Field to apply fromdate and todate to
String	fromdate	25	Return items starting from this date
String	todate	3	Return items up to this date

Returns

1. -1 Authentication failed (check merchantkey/apikey) combination
2. *N* A positive integer represents the number of items found.
3. 0 No records were found.
4. *-2* Invalid fromdate—the date format is: MM/DD/YY
5. *-3* Invalid todate.
6. *-4* Invalid date range—fromdate must be less-than or equal-to todate.
7. *-5* Invalid filter (see filter values)
8. *-6* Invalid date filter (see date filter values)

Valid Filter Values

1. *all* Return all payments
2. *returns* Return only returned payments
3. *pending* Return only pending payments
4. *settled* Return only settled payments

Valid Date Filter Values

1. *eff* Report by effective date
2. *prc* Report by payment date or process date
3. *ret* Report by return date
4. *dis* Return by disbursement or settlement date

Example

https://test.cpteller.com/api/25/webapi.cfc?merchantkey=209121408&apikey=mykey&method=list_bank_payments&fromdate=1/2/14&todate=9/20/14&__bdreturnformat=json

{"status":2,"message":"","filter":"ALL","datefilter":"EFF","fromdate":"1/2/14","todate":"9/20/14","items":[{"customerid":264420,"batchid":0,"paymentid":265195,"itemid":249061,"routingnum":" ","bankacctnum":" ","bankaccttype":"C","paymentdate":"05/12/14","effectivedate":"05/14/14","returndate":"05/12/14","returncode":"R09","returnreason":"Uncollected
Funds","invoice":"456456","individualid":"P265195","individualname":"John
Doe","amount":1,"disbursementid":null,"noc":0,"noc_routingnum":null,"noc_bankacctnum":null,"noc_bankaccttype":null},{"customerid":264424,"batchid":0,"paymentid":265199,"itemid":249062,"routingnum":"123000054 ","bankacctnum":"John F Doe
","bankaccttype":"C","paymentdate":"05/12/14","effectivedate":"05/14/14","returndate":"05/12/14","returncode":"R09","returnreason":"Uncollected
Funds","invoice":"456456","individualid":"P265199","individualname":"John
Doe","amount":1,"disbursementid":null,"noc":0,"noc_routingnum":null,"noc_bankacctnum":null,"noc_bankaccttype":null}]}

Method

Struct list_bank_returns()

Arguments

Type	Name	Len	Description
String	merchantkey	50	Your assigned merchantkey
String	apikey	50	Your API pass key
String	filter	10	Customer ID of the customer in the search
String	datefilter	10	Amount of the payment to search for
String	fromdate	25	Your reference or invoice number
String	todate	3	Return code of ACH payment being sought

Returns

1. -1 Authentication failed (check merchantkey/apikey) combination
2. *Custid* If the status is a positive integer then the cust_search() call has found a single item matching your search criteria and this is the customer id of that record.
3. 0 No record was found
4. *-N* If the return value is a negative number then the absolute value of the number is the number of items found. For example -5 means your search found 5 items.

Notes

If you are calling this method as a RESTlet then the only arguments that are required are the ones you are using in the search (other than merchantkey/apikey.)
SOAP requires you provide all arguments, but you can pass them as empty strings and they will be ignored.

Example

http://secure.cpteller.com/api/25/webapi.cfc?method=payment_search&merchantkey=123123&apikey=982734928&authcode=6AB123

121236

Method

Struct list_card_payments()

Arguments

Type	Name	Len	Description
String	merchantkey	50	Your assigned merchantkey
String	apikey	50	Your API pass key
Date	fromdate	10	Starting date from which to report payments.
Date	todate	1	Ending date to which to report payments.
Boolean	all	25	true = return all, false = return only authorized payments.

Returns

This method returns a struct with the following members:

Status	-1 = Authentication failed (check merchantkey/apikey) combination *n* = A positive integer represents the number of items found. 0 = No records were found. -2 = Invalid fromdate—the date format is: MM/DD/YY -3 = Invalid todate. -4 = Invalid date range—fromdate must be less-than or equal-to todate.	
Message	A description of the error if status is less-than zero.	
All	true or false depending as provided in the request. (default is false)	
Fromdate	The fromdate provided in the request. (default is today's date)	
Todate	The todate provided in the request. (default is today's date)	
Items	A list of items containing the results of the search with the following members:	
	Paymentid	Token or Reference ID or token of this payment item
	Customerid	Token or Reference ID of the customer record for this item
	Scheduleid	Token or Reference ID of payment schedule if any
	Departmentid	Token or Reference ID of Department or Category if any
	Refpaymentid	Token or Reference ID of an associated payment if any

	Feepaymentid	Token or Reference ID of an associated fee payment if any
	Batchid	Token or Reference ID of Batch payment belongs to if any
	Status	Payment Status
	Paymentdate	Date on which payment was processed
	Entrydate	Date on which the payment has effect on your customer
	Invoice	Invoice information for the payment if any
	Amount	Dollar amount of payment
	Feeamount	Dollar amount of fee payment if any
	Refundamount	Dollar amount refunded if any
	Cardnum	Card Number if this is a card payment
	Cardname	Name on Card if this is a card payment
	Authcode	Authorization Code
	Authstatus	Authorization Status
	Avsdata	Address Verification System match information
	Origin	Origin of payment
	Voided	1=Payment was voided, 0=Payment not voided
	Voiddate	Date on which payment was voided if any
	Authresult	If payment declined, reason for decline if any was given
	Department	Department description if any
	Username	Username of user who entered the payment if known
	IPAddress	IP Address where payment was entered if known
	Comment	User comment if any

Notes

If you are calling this method as a RESTlet then you can supply only the merchantkey/apikey and all authorized payments from the current date will be returned.

Example

https://test.cpteller.com/api/25/webapi.cfc?merchantkey=209121408&apikey=mykey&method=list_card_payments&fromdate=4/20/14&todate=4/20/14&__bdreturnformat=json

{"status":3,"message":"","all":"false","fromdate":"04/20/14","todate":"04/20/14","items":[{"paymentid":265240,"customerid":264471,"scheduleid":0,"departmentid":0,"refpaymentid":0,"feepaymentid":0,"batchid":0,"status":5,"paymentdate":"04/20/14","entrydate":"04/20/14","invoice":"0","amount":1,"feeamount":0,"refundamount":0,"cardnum":null,"cardname":null,"authcode":"637200
","authstatus":null,"authresult":"A","avsdata":"N","origin":0,"voided":0,"voiddate":"","department":"","username":null,"ipaddress":"50.160.0.29","comment":null},{"paymentid":265243,"customerid":264479,"scheduleid":0,"departmentid":0,"refpaymentid":0,"feepaymentid":0,"batchid":0,"status":5,"paymentdate":"04/20/14","entrydate":"04/20/14","invoice":"C264479","amount":1,"feeamount":0,"refundamount":0,"cardnum":null,"cardname":null,"authcode":"435404
","authstatus":null,"authresult":"A","avsdata":"N","origin":0,"voided":0,"voiddate":"","department":"","username":null,"ipaddress":"50.160.0.29","comment":null},{"paymentid":265242,"customerid":264478,"scheduleid":0,"departmentid":0,"refpaymentid":0,"feepaymentid":0,"batchid":0,"status":5,"paymentdate":"04/20/14","entrydate":"04/20/14","invoice":"0","amount":1,"feeamount":0,"refundamount":0.5,"cardnum":null,"cardname":null,"authcode":"048661
","authstatus":null,"authresult":"A","avsdata":"N","origin":0,"voided":0,"voiddate":"","department":"","username":null,"ipaddress":"50.160.0.29","comment":null}]}

Method

Struct card_payment() Process a card payment.

Arguments

Argument	*Type*	*Size*	*Required*	*Default*	*Description*
merchantkey	String	50	Yes		Your assigned merchantkey
apikey	String	50	Yes		Your API key
amount	Money	10	Yes		Payment Amount
cardnum	String	16	No	Obtained from trackdata	Card number (15 or 16 digits)
cardname	String	50	No	Obtained from trackdata	The name on the card
expdate	String	4	No	Obtained from trackdata	The expiration date of the card. (Format: MMYY)
custid	Integer	10	No	A new record will be created	ID of customer record to use
account	String	50	No	Empty	Account reference number or string for your reference
firstname	String	50	No	Obtained from trackdata	Customer's first name
lastname	String	50	No	Obtained from trackdata	Customer's last name
address1	String	50	No	Empty	First line of customer's address
address2	String	50	No	Empty	Second line of customer's address
city	String	50	No	Empty	City
state	String	2	No	Empty	Two-character state abbreviation
zipcode	String	10	No	Empty	Postal code
country	String	3	No	"USA"	ISO-2, ISO-3, or ISO-Numeric country code

phone	String	20	No	Empty	Customer's phone number
email	String	50	No	Empty	Customer's email address
invoice	String	25	No	Empty	Your payment invoice information
cvv	String	4	No	Empty	The card verification code
trackdata	String	N/A	No	Empty	The trackdata from a card reader
comment	String	50	No	Empty	Any comment text
confirm	Boolean	N/A	No	"False"	No payment—confirm only. *See Notes*
authonly	Boolean	N/A	No	"False"	Auth-Only
username	String	25	No	Empty	A username to attach to payment (appears in "by-user" reports)

Returns

This method returns a struct containing status and card authorization information. If the status member is a negative integer then a problem occurred. If a positive integer is in the status then the method completed successfully and the other members will contain more information. The meanings of the status member are:

-1 Authentication failed: merchantkey/apikey not authenticated.
-2 A customer record could not be created from the information given or the custid provided is invalid.
-3 A payment record could not be created with the information given.
-4 The connection to the card payment system failed.
-5 The payment appears to have processed but the system update failed.
0 A Confirmation was requested and the results contain the service fee amount.
N The payment successfully processed. Check return structure variables for results.

Return Structure Variables

Name	*Type*	*Description*
Status	Integer	Status (see above)
custid	Integer	Customer record identifier
paymentid	Integer	Payment record identifier
response	String	A=Approved, D=Declined, E=Error

authcode	String	If response is A this is the authorization code
declinereason	String	If response is not A a reason will be given
avsresult	String	The result of the Address Verification System match.
Fee	Money	The service fee amount charged if payment is made, or the fee amount which would be charged if confirm = "true"

Notes

- The confirm parameter is a Boolean value which if true, will cause the API to return the return structure variables with status = 0, and fee as the amount of a service fee if a payment is made by calling this method with confirm = "false" or with the confirm argument omitted. All other members of the return structure will have empty or zero amounts and can be ignored.
- The cvv parameter is used to process the payment and then discarded.
- If you omit the 'custid' parameter or provide it with a zero value a new customer record will be created. Otherwise the existing customer record will be used.
- If you use an existing customer record and provide additional data in other parameters the existing customer record will be updated with the parameter data.
- If you call this method with authonly=true (or authonly=1) the payment will be authorized but not "captured." This means that you must either call payment_capture() for the payment to be funded or payment_void() to cancel it. If you do nothing it will be automatically voided after a number of days.

Example

This example is of a SOAP call to this method from C#:

```
var merchantKey = "2191214";
var apiKey = "09284509v8n2v45";
var api = new webapi.webapiClient();
var result = api.card_payment
(
    merchantkey: merchantKey,
    apikey: apiKey,
    custid: 0,
    account: "123",
    firstname: "John",
    lastname: "Doe",
    address1: "123 Main St.",
    address2: "Apt 3G",
    city: "Bugtussel",
    state: "WV",
    zipcode: "12345",
    country: "USA",
```

```
    phone: "801-910-0150",
    email: "john.doe@doe.com",
    amount: 1.00,
    invoice: "1",
    cardnum: "4111111111111111",
    cardname: "John Doe",
    expdate: "0515",
    cvv: "123",
    trackdata: "",
    comment: "This is a test."
);
Console.WriteLine("result.status = {0}", result.item[getKeyIndex(result,
"status")].val.ToString());
Console.WriteLine("result.custid = {0}", result.item[getKeyIndex(result,
"custid")].val.ToString());
Console.WriteLine("result.paymentid = {0}", result.item[getKeyIndex(result,
"paymentid")].val.ToString());
Console.WriteLine("result.response = {0}", result.item[getKeyIndex(result,
"response")].val.ToString());
Console.WriteLine("result.authcode = {0}", result.item[getKeyIndex(result,
"authcode")].val.ToString());
Console.WriteLine("result.declinereason = {0}", result.item[getKeyIndex(result,
"declinereason")].val.ToString());
            Console.WriteLine("result.avsresult = {0}", result.item[getKeyIndex(result,
"avsresult")].val.ToString());
```

```
<?php

$url = "https://secure.cpteller.com/api/25/webapi.cfc";
$fields = array
(
   'method'        => urlencode('card_payment'),
   'merchantkey'   => urlencode('Your Merchant Key Here'),
   'apikey'        => urlencode('Your API Key Here'),
   'account'       => urlencode('9999999'),
   'firstname'     => urlencode('John'),
   'lastname'      => urlencode('Doe'),
   'address1'      => urlencode('405 East 12450 South'),
   'address2'      => urlencode('Suite E'),
   'city'          => urlencode('Draper'),
   'state'         => urlencode('UT'),
   'zipcode'       => urlencode('84020'),
   'phone'         => urlencode('801-910-0150'),
   'email'         => urlencode('jdoe@bogus.net'),
   'cardnum'       => urlencode('4111111111111111'),
   'cardname'      => urlencode('John Doe'),
   'expdate'       => urlencode('1221'),
   'cvv'           => urlencode('123'),
   'amount'        => urlencode(1.00),
   'invoice'       => urlencode('1234567'),
   'username'      => urlencode('test')
);

foreach($fields as $key => $value)
{
   $fields_string .= $key.'='.$value.'&';
}
rtrim($fields_string, '&');

$ch = curl_init();
```

```
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_POST, count($fields));
curl_setopt($ch, CURLOPT_POSTFIELDS, $fields_string);
$result = curl_exec($ch);
curl_close($ch);

?>
```

Method

Struct card_payment_restricted(…)

This method is a wrapper to the card_payment() method. It adds some additional card validation. The parameters are the same as card_payment() with one additional parameter added:

Arguments

Argument	*Type*	*Max Length*	*Required*	*Default*	*Description*
restrict	String	1	No	Empty	See “Restrictions” (below)

Restrictions:

“C” Do not allow credit cards. This is most commonly used for merchants that are collecting debts and they don’t want their customers to pay debts by creating more debts. The type of card will be checked and if it turns out to be a “credit” card then it will not be allowed.

“D” Do not allow debit cards

“U” Do not allow unknown types of cards. Some cards don’t specify. If the type of card is unspecified it will be disallowed.

Returns:

See card_payment()

Method

Struct card_authenticate(…)

This method performs a $0.00 auth-only transaction on a card. This is useful to verify if the card is likely to be able to be used for future payments. Not all processors support this feature so if you are going to use card_authenticate() you should check with CP Support first.

Because this method has a card number in its parameter list it must be called with HTTPS POST.

Arguments

Argument	*Type*	*Max Length*	*Required*	*Default*	*Description*
Merchantkey	String		Yes		Your merchant key
Apikey	String		Yes		Your API key
Address	String	50	No	Empty	The billing address for the card
Zipcode	String	10	No	Empty	The billing zipcode
Cardnum	String	16	Yes		The card number
Cardname	String	50	No	Empty	The name on the card
Expdate	String	4	Yes		The expiration date on the card in format MMYY
Cvv	String	4	No	Empty	The CV2 value on the card

Returns:

This method returns a struct with the following fields:

1. Status: If status is a negative number then an error has occurred. Otherwise it will be 0 (zero.)
2. Response code: “A” if approved, “D” if declined, etc.
3. Auth code: The authcode returned from the processor if the card was approved.
4. Decline reason: If the card is declined this may give some additional information on the reason for the decline. This comes directly from the card gateway an is passed along to the caller.
5. AVS data: The address verification system response.

Method

Struct card_swipe()

Process a swiped card payment.

Arguments

Argument	*Type*	*Max Length*	*Required*	*Default*	*Description*
merchantkey	String	50	Yes		Your assigned merchantkey
apikey	String		Yes		Your API key
amount	String	10	Yes		The amount of the payment to be processed
trackdata	String	N/A	Yes		The trackdata from a card reader
account	String	50	No	Empty	Account reference number or string for your reference
firstname	String	50	No	Obtained from trackdata	Customer's first name
lastname	String	50	No	Obtained from trackdata	Customer's last name
address1	String	50	No	Empty	First line of customer's address
address2	String	50	No	Empty	Second line of customer's address
city	String	50	No	Empty	City
state	String	2	No	Empty	Two-character state abbreviation
zipcode	String	10	No	Empty	Postal code
country	String	3	No	"USA"	Iso-2, iso-3, or iso-Numeric country code of address
phone	String	20	No	Empty	Customer's phone number
email	String	50	No	Empty	Customer's email address

invoice	String	25	No	Empty	Your payment invoice information
comment	String	50	No	Empty	Any comment text
confirm	Boolean	N/A	No	"False"	No payment—confirm only. *See Notes*
username	String	25	No	Empty	A username to attach to payment (appears in "by-user" reports)

Returns

This method returns a struct object containing members with result data. Generally if the 'status' member is a positive integer (paymentid) then the method completed normally. The other members can then be checked for results. A negative integer indicates a problem.
Optional parameters may need to be included in SOAP calls. If you are doing RESTful calls you can pass the required parameters and omit the optional ones.

Notes

- The confirm parameter is a Boolean value which, if true will cause the API to return what it would return if a payment would have been processed if a payment were made without actually processing a payment. This would allow a card swipe machine to pass the information to the API, receive back the results, including fees, and then allow the operator to confirm whether they want to process the payment or not. To process a payment then, a second call to card_swipe with confirm set to false, or with the confirm argument omitted. A true value could be 'true', 'yes', or 1 (case insensitive.) A false value could be 'false', 'no', or 0.
- If the firstname and lastname fields are provided as parameters they will be used in the customer record. If they are omitted the software will extract them from the name on the card. The first word in the cardname will be the firstname and the last word will be the lastname. Anything else will be dropped. The full cardname from the card is stored in the cardname field in the customer record.
- The length of the trackdata field is shown as "N/A" because it is determined by the card itself and you should never modify it. Trackdata is used for the payment and is then discarded. It is never stored in the system.
- It is often problematic to send track data using an HTTP GET method. POST should be used if invoking this method directly.

Status Meanings

-1 Authentication failed: merchantkey/apikey not authenticated.
-2 A customer record could not be created from the information given.
-3 A payment record could not be created with the information given.
-4 The connection to the card payment system failed.
-5 The payment appears to have processed but the system update failed.
0 A fee confirmation was requested. The return structure contains the fee.
N The payment successfully processed. Check return structure variables for results.

Return Structure Variables

Name	*Type*	*Description*
Status	Integer	Status (see above)
custid	Integer	Customer record identifier
paymentid	Integer	Payment record identifier
response	String	A=Approved, D=Declined, E=Error
authcode	String	If response is A this is the authorization code
declinereason	String	If response is not A a reason will be given
avsresult	String	The result of the Address Verification System match.
Fee	Money	The service fee amount charged if payment is made, or the fee amount which would be charged if confirm = "true"

Method

struct payment_voidable(string merchantkey, string apikey, int paymentid)

This method determines if a payment is able to be voided.

Parameters

Argument	*Type*	*Required*	*Description*
merchantkey	String	Yes	Your account number reference for the customer.
apikey	String	Yes	The customer's first or given name.
paymentid	Integer	Yes	

Returns

An integer:
1: The payment is voidable
0: The payment is not voidable
-1: Authentication failed (merchantkey, apikey)
-2: The payment was not found.

Notes

Generally a card payment is voidable if it has not settled. Most card payments settle on the day they were authorized at 8:30pm Mountain Time.
Bank payments (ACH) are voidable up until 5pm Mountain Time on the day they were supposed to be transmitted to the Fed.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=payment_voidable&merchantkey=123&apikey=456&paymentid=234534534

Method

struct payment_refundable(string merchantkey, string apikey, int paymentid)

This method determines the amount of a payment which is refundable.

Parameters

Argument	*Type*	*Required*	*Description*
merchantkey	String	Yes	Your account number reference for the customer.
apikey	String	Yes	The customer's first or given name.
paymentid	Integer	Yes	The payment ID token

Returns

A Double (Money) type:
N: An amount zero or more is the amount which can be refunded.
-1: Authentication failed (merchantkey, apikey)
-2: The payment was not found.

Notes

It is ok to refund an amount less than or equal to the amount returned but not greater than.
The system keeps track of the total amount a particular payment has been refunded. It is possible to refund more than one partial refund on a payment. The cumulative amount is tracked.
If the return value is zero then a refund is not possible.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=payment_refundable&merchantkey=123&apikey=456&paymentid=234534534

Method

integer payment_void(merchantkey, apikey, paymentid)

This method voids a payment.

Parameters

Argument	*Type*	*Required*	*Description*
merchantkey	String	Yes	Your account number reference for the customer.
apikey	String	Yes	The customer's first or given name.
paymentid	Integer	Yes	

Returns

An integer:

N: The payment has been voided. N is the amount of the payment. (Amount Voided)
-1: Authentication failed (merchantkey, apikey)
-2: The payment was not found.
-3: The payment is not voidable.
-4: The void operation failed. (System Error)

Notes

Generally a card payment is voidable if it has not settled. Most card payments settle on the day they were authorized at 8:30pm Mountain Time.
Bank payments (ACH) are voidable up until 5pm Mountain Time on the day they were supposed to be transmitted to the Fed.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=payment_void&merchantkey=123&apikey=456&paymentid=234534534

Method

integer payment_capture(merchantkey, apikey, paymentid)

This method captures a payment.

Parameters

Argument	*Type*	*Required*	*Description*
merchantkey	String	Yes	Your account number reference for the customer.
apikey	String	Yes	The customer's first or given name.
paymentid	Integer	Yes	The paymentid of the pre-authorized payment.

Returns

An integer:

0: The payment has been captured.
-1: Authentication failed (merchantkey, apikey)
-2: The payment was not found.
-3: The payment was not able to be captured.

Notes

If an "auth-only" payment (see "card_payment()") is not captured by this method, and if it is not voided (see "void_payment()") the payment will automatically be voided after a number of days. (21 days is standard.)

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=payment_capture&merchantkey=123&apikey=456&paymentid=234534534

Method

struct payment_refund(merchantkey, apikey, paymentid, amount)

This method issues a refund on a payment to the card or bank account where the payment originated.

Parameters

Argument	*Type*	*Required*	*Description*
merchantkey	String	Yes	Your account number reference for the customer.
apikey	String	Yes	The customer's first or given name.
paymentid	Integer	Yes	The payment ID token.
Amount	Double	Yes	The amount to be refunded.

Returns

A struct containing the following items:

1. Status — An integer—see below
2. Paymentid — This paymentid
3. Refundid — Payment ID of the refund payment
4. Amount — Amount refunded

N: A positive integer means the refund was issued. The integer returned represents the paymentid of the refund payment.
-1: Authentication failed (merchantkey, apikey)
-2: The payment was not found.
-3: The refund cannot be issued. (See Notes)

Notes

Refunds can be issued for payments cumulatively totaling the amount of the original payment and not more. Multiple partial-refunds can be issued on a single payment as long as the amount of the original payment is not exceeded.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=payment_refund&merchantkey=123&apikey=456&paymentid=23453453&amount=10.00

Method

double payment_reverse(merchantkey, apikey, paymentid)

This method determines if a payment can be voided and if so, voids it. If not it determines the maximum amount that can be refunded for a payment and refunds that amount.

Parameters

Argument	*Type*	*Required*	*Description*
merchantkey	String	Yes	Your account number reference for the customer.
apikey	String	Yes	The customer's first or given name.
paymentid	Integer	Yes	The payment ID token.

Returns

A Numeric Double amount:

N: A positive amount is the amount which was refunded if a refund was performed.
0: The payment was voided.
-1: Authentication failed (merchantkey, apikey)
-2: The payment was not found.
-3: The payment is not voidable and no refundable amount is available.

Notes

If a specific amount needs to be refunded you should use payment_refund() instead. This method will reverse the entire payment amount if it can.
If a payment is voidable but for some reason you want to issue a refund on it then payment_void() should be called. This method will make the decision for you and will always perform a void if it can do so.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=payment_reverse&merchantkey=123&apikey=456&paymentid=234534534

Returns:

100.82 ($100.82 was refunded)

Method

struct schedule_create(string merchantkey, string apikey, int custid, …)

This method creates a payment schedule.

Parameters

Argument	*Type*	*Required*	*Default*	*Description*
merchantkey	String	Yes		Your account number reference for the customer.
Apikey	String	Yes		The customer's first or given name.
Custid	Integer	Yes		The customer ID or Token of the person making the payments.
Type	String	Yes		"ACH" or "CARD" If left blank then derive method from the customer record.
Amount	Money	No	$0.00	The amount of the payments to be made. ♪
Count	Integer	No	0	The number of payments to be made. ♪
Startdate	String	No	Today	The start date of this payment schedule in format: MM/DD/YY or MM/DD/YYYY
Enddate	String	No	Empty	The number of items in the batch.
Balance	String	No	$0.00	Balance due for a custom payment schedule. Zero for a recurring payment schedule.
frequency		No	12	Number of payments per year.
Baseday		No	Day of month of startdate	Day of month on which to base the payment schedule.
Invoice		No	Empty	Any invoice number or text you provide.
Seccode		No	Merchant Default	The SEC Code is the type of authorization you received from the payer. If not provided your merchant default SEC Code will be used.
comment	String	No	Empty	Any comment or note to be added to schedule.

♪ Either amount or count is required. They cannot both be omitted, neither can they both be included—they are mutually exclusive. If you provide an amount then payments will be made in that amount. If you provide a count then the amount will be calculated based on the number of payments. The count parameter can only be used in custom payment schedules (ones with a balance.)

♫ Valid date formats are:

1. MM/DD/YY
2. MM/DD/YYYY
3. YYYY/MM/DD
4. MM-DD-YY
5. MM-DD-YYYY
6. YYYY-MM-DD
7. MM.DD.YY
8. MM.DD.YYYY
9. YYYY.MM.DD

Returns

Upon success this method returns structure containing a positive integer representing the scheduleid. The scheduleid is a token which can be used to refer to this payment schedule in subsequent operations. If an error occurs, the the scheduleid will be 0 (zero) and the errmsg member of the return structure will contain a comma-delimited list of any errors.

Error Messages

- Access Denied: Your merchantkey/apipasskey is not authenticated.
- Customer Not Found: The custid provided does not match one of your customers.
- Amount or (Count + Balance) Required: You can provide a payment amount and if balance is zero a recurring payment schedule will be created (see Notes below.) If you provide a count then the amount will be calculated based on the balance. If you omit amount and count or you provide count but no balance this error will occur.
- Payment Type Missing or Invalid: the “Type” parameter must be either “ACH” or “CARD.”
- Invalid Start Date: You provided a start date but it is not in date format (see Notes.)
- Invalid End Date: You provided an end date but it is not in date format (see Notes.)
- Invalid Frequency: You provided a frequency which is not supported (see Notes.)
- Payment method Not Supported in Customer Record: You specified ACH but the customer record has insufficient bank information to make an ACH payment, or you specified CARD but there is insufficient card information in the customer record to make a card payment.
- Payment Schedule Not Created: This may be due to a system error of some sort. You should never see this message. If you do, please contact support.

Notes

There are two types of payment schedule:

1. Recurring: A recurring payment schedule has no balance. When you create the payment schedule, payments will continue until cancelled or until the (optional) enddate is reached.
2. Custom: A custom payment schedule begins with a balance due and payments continue until the balance is zero. The final payment may be an amount less than the regular amount depending on the remaining balance. If you specify a balance and count, the payment amount will be calculated as balance/count (balance divided by count.)

Frequency

The value of the frequency parameter is the number of payments which will be made in a year. For example, a monthly payment schedule has a frequency of 12. Possible frequencies are:

1. Annual payments (one payment each year)
2. Semi-Annual payments (twice per year, every 6 months)
3. Three-times per year (every 4 months)
4. Quarterly payments (four payments per year, every 3 months)
6. A payment every two months (6 payments per year)
12. Monthly Payments (12 payments each year*)
24. Twice-monthly payments (24 payments per year, on the same 2 days of each month ♪)
26. A payment every two weeks (26 payments each year on the same day of each week)
52. Weekly payments (52 payments each year on the same day each week ♫)

♪ Because some months have only 30 days (or less), if you schedule payments for a day of the month which does not exist the scheduler will process the payment on the last day of that month. If you create a payment schedule which includes payments on the 31st of each month, then these payments will always occur on the last day of each month for which they are scheduled.

♫ ACH payments take several days to complete. Weekly ACH payment schedules are not recommended due to the fact that it is possible for payments to overlap. For example, if an ACH payment returns due to non-sufficient funds you could be processing the next payment in the payment schedule even though the previous one has not been resolved. Also, ACH payments may be re-presented up to 2 times and your customer could be incurring overdraft fees for each presentment. For these reasons we don't recommend weekly ACH payment schedules be used.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=schedule_create&merchantkey=20908&apikey=928374&custid=272010&balance=100&amount=27&type=card&comment=this%20is%20a%20test&seccode=TEL

Returns:

{"scheduleid":7585,"errmsg":"","custid":"272010","type":"CARD","startdate":"7/25/2014","enddate":"",
"baseday":25,"amount":27,"count":"0","balance":100,"frequency":"12","invoice":"","seccode":"TEL","c
omment":"this is a test"}

Method

struct schedule_read(merchantkey, apikey, scheduleid)

This method returns a structure containing the payment schedule information of *scheduleid.* The scheduleid is returned when you create a payment schedule using schedule_create().

Parameters

Argument	*Type*	*Required*	*Default*	*Description*
merchantkey	String	Yes		Your account number reference for the customer.
apikey	String	Yes		The customer's first or given name.
scheduleid	Integer	Yes		The scheduleid returned when you called schedule_create()

Returns

This method returns a structure of items relevant to the payment schedule.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=schedule_read&merchantkey=209121408&apikey=mykey&scheduleid=7586

returns:

{"scheduleid":7586,"customerid":272010,"amount":27,"method":0,"balance":100,"startdate":"07/25/14","enddate":null,"previouspaymentdate":"","nextpaymentdate":"","status":0,"frequency":12,"baseday":25,"paymenttype":2,"paymentcount":0,"beginningbalance":100,"apr":0,"compound":0,"invoice":"","seccode":"TEL","comment":"this is a test"}

Method

struct schedule_delete(merchantkey, apikey, scheduleid)

This method deletes a payment schedule. All future payments for the payment schedule will be removed.

Parameters

Argument	*Type*	*Required*	*Default*	*Description*
merchantkey	String	Yes		Your Merchant Key
apikey	String	Yes		Your API Key
scheduleid	Integer	Yes		The scheduleid is a token which was returned when you called schedule_create()

Returns

This method returns -1 if authentication fails, 0 if the payment schedule was not found, or the scheduleid if the payment schedule was deleted.

Example

https://secure.cpteller.com/api/25/webapi.cfc?method=schedule_delete&merchantkey=209121408&apikey=mykey&scheduleid=7586

returns:

7586

Method

struct schedule_update(merchantkey, apikey, scheduleid, …)

This method updates an existing schedule with new information. All future payments for the payment schedule will be affected. All parameters will be written to the payment schedule or their default values will be used, so it is best to make a call to schedule_read() to get any existing values which you need to preserve.

Parameters

Argument	*Type*	*Required*	*Default*	*Description*
merchantkey	String	Yes		Your Merchant Key
apikey	String	Yes		Your API Key
scheduleid	Integer	Yes		This is the token returned when you created the payment schedule.
type	String	Yes		"ACH" or "CARD" If left blank then derive method from the customer record.
amount	Money	No	$0.00	The amount of the payments to be made. ♪
count	Integer	No	0	The number of payments to be made. ♪
startdate	String	No	Today	The start date of this payment schedule in format: MM/DD/YY or MM/DD/YYYY
enddate	String	No	Empty	The number of items in the batch.
balance	String	No	$0.00	Balance due for a custom payment schedule. Zero for a recurring payment schedule.
frequency		No	12	Number of payments per year.
invoice		No	Empty	Any invoice number or text you provide.
seccode		No	Merchant Default	The SEC Code is the type of authorization you received from the payer. If not provided your merchant default SEC Code will be used.
comment	String	No	Empty	Any comment or note to be added to schedule.

Returns

This method returns the same structure information as schedule_create() including any error messages.

Note

You can change any of the parameters of a payment schedule shown above but sometimes it may make more sense to delete an existing payment schedule using schedule_delete() and then create a new one. It might be confusing, for example to change the balance, but it may make perfect sense to change the payment type or update a recurring payment schedule with an end date.

Error Messages

- Access Denied: Your merchantkey/apipasskey is not authenticated.
- Customer Not Found: The customer referenced in the payment schedule does not match one of your customers.
- Amount or (Count + Balance) Required: You can provide a payment amount and if balance is zero a recurring payment schedule will be created (see Notes below.) If you provide a count then the amount will be calculated based on the balance. If you omit amount and count or you provide count but no balance this error will occur.
- Payment Type Missing or Invalid: the “Type” parameter must be either “ACH” or “CARD.”
- Invalid Start Date: You provided a start date but it is not in date format (see Notes.)
- Invalid End Date: You provided an end date but it is not in date format (see Notes.)
- Invalid Frequency: You provided a frequency which is not supported (see Notes.)
- Payment method Not Supported in Customer Record: You specified ACH but the customer record has insufficient bank information to make an ACH payment, or you specified CARD but there is insufficient card information in the customer record to make a card payment.
- Payment Schedule Not Updated: This may be due to a system error of some sort. You should contact support if you see this message.
- Schedule Not Found: The scheduleid provided does not match one of your payment schedules.

Method

struct merchant_read(merchantkey, apikey, scheduleid, …)

This method returns information about your merchant setup in the cpteller system.

Parameters

Argument	*Type*	*Required*	*Description*
merchantkey	String	Yes	Your Merchant Key
apikey	String	Yes	Your API Key

Returns

Argument	*Type*	*Description*
Status	Integer	Merchant Identifier (-1 = Authentication Failed)
Entrydate	Date	Date on which merchant's account was created
Companyname	String	Company Name
Address1	String	First line of merchant's address
Address2	String	Second line of merchant's address
City	String	Merchant's city
State	String	Merchant's state
Zipcode	String	Merchant's zip code
Phone	String	Merchant's main phone number
Email	String	Merchant's main email address
url	String	Merchant's url
Contactname	String	Main point of contact for this merchant
Contactphone	String	Main contact's phone number
Testmode	Boolean	Merchant is in test mode Yes or No

Routingnum	String	Routing number of disbursement/billing bank
Bankacctnum	String	Bank account for disbursement of ACH funds/billing
Achenabled	Boolean	Merchant ACH processing enabled
Cardenabled	Boolean	Merchant card processing enabled
Fundingsource	String	Type of funding source cards to accept (optional feature must be enabled)
Cashflowid	Integer	Token identifying merchant to the SBPC system.
Supportemail	String	Email address to send support requests to
Billingemail	String	Email address to send billing statements to
Receiptemail	String	Email address to send payment receipts to

Example

http://secure.cpteller.com/api/25/webapi.cfc?merchantkey=999jjj&apikey=mykey&method=merchant_read

Python Example:

```
from requests import get, post
url = "https://secure.cpteller.com/api/25/webapi.cfc"
r = get(url, {"method":"merchant_read", "merchantkey":"3A1013", "apikey":"Lucky13"})
print r.text
```

{"status":13,"entrydate":"12/14/09","companyname":"Wallyworld, Inc","address1":"405 East 12450 South","address2":"Suite E","city":"Draper","state":"UT","zipcode":"84020-7997","phone":"801-999-4323
x102","email":"john.moss@convenientpayments.com","url":"","contactname":"","contactphone":"3852101078","testmode":"Yes","routingnum":"124000054","bankacctnum":"15E2B0D3C3","achenabled":"Yes","cardenabled":"Yes","cardsaccepted":"AVMD","fundingsource":"ACDPHU","cashflowid":0,"supportemail":"john.moss@convenientpayments.com","billingemail":"","receiptemail":"john.moss@convenientpayments.com","feemethod":0,"cplogo":"https://secure.cpteller.com/img/logos/CPTellerLogo.png","customertitle":"Customer","paymenttitle":"Payment"}

Method

struct merchant_read(merchantkey, apikey, scheduleid, …)
This method returns information about your merchant setup in the cpteller system.

Parameters

Argument	*Type*	*Required*	*Description*
merchantkey	String	Yes	Your Merchant Key
apikey	String	Yes	Your API Key

Returns

Argument	*Type*	*Description*
Status	Integer	Merchant Identifier (-1 = Authentication Failed)
Entrydate	Date	Date on which merchant's account was created
Companyname	String	Company Name
Address1	String	First line of merchant's address
Address2	String	Second line of merchant's address
City	String	Merchant's city
State	String	Merchant's state
Zipcode	String	Merchant's zip code
Phone	String	Merchant's main phone number
Email	String	Merchant's main email address
url	String	Merchant's url
Contactname	String	Main point of contact for this merchant
Contactphone	String	Main contact's phone number
Testmode	Boolean	Merchant is in test mode Yes or No
Routingnum	String	Routing number of disbursement/billing bank

Bankacctnum	String	Bank account for disbursement of ACH funds/billing
Achenabled	Boolean	Merchant ACH processing enabled
Cardenabled	Boolean	Merchant card processing enabled
Fundingsource	String	Type of funding source cards to accept (optional feature must be enabled)
Cashflowid	Integer	Token identifying merchant to the SBPC system.
Supportemail	String	Email address to send support requests to
Billingemail	String	Email address to send billing statements to
Receiptemail	String	Email address to send payment receipts to

Example

http://secure.cpteller.com/api/25/webapi.cfc?merchantkey=999jjj&apikey=mykey&method=merchant_read

Python Example:

```
from requests import get, post
url = "https://secure.cpteller.com/api/25/webapi.cfc"
r = get(url, {"method":"merchant_read", "merchantkey":"3A1013", "apikey":"Lucky13"})
print r.text
```

{"status":13,"entrydate":"12/14/09","companyname":"Wallyworld, Inc","address1":"405 East 12450 South","address2":"Suite E","city":"Draper","state":"UT","zipcode":"84020-7997","phone":"801-999-4323 x102","email":"john.moss@convenientpayments.com","url":"","contactname":"","contactphone":"3852101078","testmode":"Yes","routingnum":"124000054","bankacctnum":"15E2B0D3C3","achenabled":"Yes","cardenabled":"Yes","cardsaccepted":"AVMD","fundingsource":"ACDPHU","cashflowid":0,"supportemail":"john.moss@convenientpayments.com","billingemail":"","receiptemail":"john.moss@convenientpayments.com","feemethod":0,"cplogo":"https://secure.cpteller.com/img/logos/CPTellerLogo.png","customertitle":"Customer","paymenttitle":"Payment"}