Getting Started with CitiConnect API

CitiConnect API allows you, a Partner Developer, to integrate your payment applications with Citibank’s CitiConnect API. Citi’s APIs are HTTP-based RESTful APIs that use OAuth 2.0 for authorization. You format payment request and response bodies in JSON, a lightweight data-exchange

**Important:** You cannot run the sample requests in this guide as-is. You will need to replace call-specific parameters such as tokens and IDs with your own values. While this introduction provides some source code examples, please see CitiConnect’s Code Samples for more information on how to integrate Citi’s API with your own code.

See International Developer Questions if you are a developer who works and resides outside of the United States.

For definitions of common REST API terms, see the Glossary.

## About this API guide

This site will show you how to:

* Authenticate your application to CitiConnect servers
* Make and verify a payment request
* Modify or update a payment request
* How to obtain a payment
* Cancel a payment request
* How to return invoice data
* Use optional and required parameters you use in POSTs and GETs

## Authentication and authorization

The Citi REST API uses the OAuth 2.0 protocol to authorize API calls. OAuth is an open standard that Citi uses to allow Partner Developers to access CitiBank’s protected resources. You use this protocol when making API requests to logon to CitiConnect and other Citi resources.

When you create a payment application, Citi generates a set of OAuth client IDs and secret credentials for your app. You use these credentials in your source code to access both the sandbox and production environments. These credentials are stored in Citi’s API Authorization Manager database. You pass these credentials in the **Authorization** header in a get access token request.

Once you provide these credentials, the Citi Authorization Manager issues access tokens. You use these tokens for authorization during REST API requests. A bearer token enables you to complete actions on behalf of, and with the approval of, the resource owner.

The **access\_token** field in the get access token response contains a bearer token, indicated by the **token\_type** of **Bearer**:

{

"scope": "scope",

"nonce": "nonce",

"access\_token": "Access-Token",

"token\_type": "Bearer",

"app\_id": "APP-8987654UY56",

"expires\_in": 32398

}

Include this bearer token in API requests in the **Authorization** header with the **Bearer**authentication scheme.

This following is a sample request that uses a bearer token to list invoices for a merchant:

curl -v -X GET https://api.sandbox.Citi .com/v1/invoicing/invoices?page=3&page\_size=4&total\_count\_required=true \

-H "Content-Type:application/json" \

-H "Authorization: Bearer Access-Token"

Please not that access tokens expire. The **expires\_in** field in the get access token response indicates the lifetime, in seconds, of the access token. For example, a value of **3600**  means that the access token expires one hour from the time the response was generated.

You can detect when an access token expires, by writing code to:

* Track of the **expires\_in** value in the token response. This value is always expressed in seconds.
* Handle the HTTP **401 Unauthorized** status code. The API endpoint issues this status code when it detects an expired token. You will need to generate a new token. See your Citi support representative regarding this issue.

Before you create another token, re-use the access token until it expires. See the rate limiting guidelines.

## Citi API requests

To construct a REST API request, combine these components:

| **Component** | **Description** |
| --- | --- |
| The HTTP method | * **GET**. Requests resource data. * **POST**. Submits data to a resource for processing. * **PUT**. Updates or modifies a resource. * **PATCH**. Changes a resource. * **DELETE**. Deletes a resource. |
| Service URLs | * Sandbox (testing environment). **https://api.sandbox.Citi .com** * Production. **https://api.Citi.com** |
| The URI to the resource | The CitiConnect resource to query, submit data to, update, or delete. For example, **v1/invoicing/invoices**. |
| Query parameters | Optional. Controls which data appears in the response. Use to filter, limit the size of, and sort the data in an API response. |
| HTTP request headers | Includes the **Authorization** header with the access token. |
| A JSON request body | Required for most **GET**, **POST**, **PUT**, and **PATCH** calls. |

This sample request cancels a billing agreement:

curl -v -X POST https://api.sandbox.Citi.com/ver1/payments/billing-agreements/I-1TJ4PAGJ82I9/cancel \

-H "Content-Type:application/json" \

-H "Authorization: Bearer Access-Token" \

-d '{

"note": "Canceling the profile."

}'

## Query parameters

For most REST **GET** calls, you can specify one or more optional query parameters on the request URI to filter, to limit the size of, and to sort data in an API response. For filter parameters, see the individual **GET** calls.

To limit, *page*, and sort the data that Citi returned in some API responses, use the following query parameters:

**Note:** Not all pagination parameters are available for all APIs.

| **Parameter** | **Type** | **Description** |
| --- | --- | --- |
| **count** | integer | The number of items to list in the response. |
| **end\_time** | integer | The end date and time for the range to show in the response, in Internet date and time format. For example, **end\_time=2017-02-06T10:00:00Z**. |
| **page** | integer | Start index of the entire list of items returned in the response. Thus, the combination of **page=0** and **page\_size=20** returns the first 20 items. The combination of **page=20** and **page\_size=20** returns the next 20 items. |
| **page\_size** | integer | The number of items that the response returns. |
| **total\_count\_required** | boolean | Indicates whether to show the total count in the response. |
| **sort\_by** | string | Sorts the payments in the response by a specified value, such as the create time or update time. |
| **sort\_order** | string | Sorts the items in the response in ascending or descending order. |
| **start\_id** | string | The ID of the starting resource in the response. When results are paged, you can use the **next\_id** value as the **start\_id** to continue with the next set of results. |
| **start\_index** | integer | The start index of the payments to list. Typically, you use the **start\_index** to jump to a specific position in the a Citi resource. For example, to start at the second item in a list of results, specify **?start\_index=2**. |
| **start\_time** | string | The start date and time for the range to show in the response, in Internet date and time format. For example, **start\_time=2016-03-09T17:00:00Z**. |

For example, the Invoicing API returns details for four invoices beginning with the third invoice and includes the total count of invoices in the response:

curl -v -X GET https://api.sandbox.Citi .com/v1/invoicing/invoices?page=3&page\_size=4&total\_count\_required=true \

-H "Content-Type:application/json" \

-H "Authorization: Bearer Access-Token"

HTTP request headers

Common HTTP request headers are:

| **Header** | **Description** |
| --- | --- |
| Accept | Required for operations with a response body.  Specifies the response format. The syntax for the Accept header is:  Accept: application/format  Where ***format*** is **json**. |
| Authorization | Required to obtain access tokens or make API calls.  To get an access token, set this header to your **client\_id** and **secret** credentials.  Note: If you use cURL, specify **-u "client\_id:secret"**.  To make REST API calls, include the bearer token in the **Authorization** header with the **Bearer** authentication scheme:  Authorization: Bearer Access-Token |
| Content-Type | Required for operations with a request body.  Specifies the request format. The syntax is:  Content-Type: application/format  Where ***format*** is **json**. |
| Citi -Auth-Assertion | Specifies an API client-provided JSON Web Token (JWT) assertion that identifies your merchant.  At times, your API client may process requests from multiple merchants simultaneously. This becomes difficult and resource expensive because you must generate and manage multiple access tokens. As a solution, Citi API clients can use this header to provide a JWT assertion that identifies the merchant when the API is called.  **Prerequisite:** To use this header, you must get consent to act on behalf of a merchant.  In the header, specify one of these JSON Web Token sub-forms:   * JSON Web Encryption (JWE), in JWS Compact Serialization format. * JSON Web Signature (JWS), in JWE Compact Serialization format. Supports both secure and unsecured JWT. An unsecured JWT specifies an **alg** of **none** in the **JOSE** header and an empty string for the signature.   An unsecured JWT example:   * JOSE header:   {"alg": "none"}   * Payload can contain **email**, **client\_id**, and **payer\_id**. Example payload:   {"iss": "client\_id","email":"my-email@example.com"}   * Signature. Use **""** for the unsigned case. * Resulting unsecured JWT after Base64 and simple concatenation:   eyJhbGciOiJub25lIn0.eyJlbWFpbCI6Im15QGVtYWlsLmNvbSJ9  All API endpoints support this header. |
|  |  |
| Citi -Client-Metadata-Id | Optional.  Verifies that the payment originates from a valid, device and application. Reduces fraud. Transactions that do not include a client metadata ID are not eligible for Citi Seller Protection. To initiate a pre-consented payment from a mobile device, see future payments. |
| Citi -Partner-Attribution-Id | Optional.  Indicates that you are a Citi partner. To receive revenue attribution, specify a unique build notation (BN) code. BN codes provide tracking on all transactions that originate or are associated with a particular partner. To learn more or to request a BN code, contact your partner manager or visit the Citi Partner Portal. |
| Citi -Request-Id | Optional.  Contains a unique user-generated ID that you can use to allow your application to make repeated calls and produce the same response.  **Notes:**   * If you omit this header, the risk of duplicate transactions increases. * Not all APIs support this header. To determine whether your API supports it, see the API reference for your API. |

API responses

Citi API calls return HTTP status codes. Some API calls also return JSON response bodies that include information about the resource including one or more contextual HATEOAS links. Use these links to request more information about and construct an API flow that is relative to a specific request.

HTTP status codes

Each REST API request returns a success or error HTTP status code.

Success

In the responses, Citi returns these HTTP status codes for successful requests:

| **Status code** | **Description** |
| --- | --- |
| **200 OK** | The request succeeded. |
| **201 Created** | A **POST** method successfully created a resource. If the resource was already created by a previous execution of the same method, for example, the server returns the HTTP **200 OK** status code. |
| **202 Accepted** | The server accepted the request and will execute it later. |
| **204 No Content** | The server successfully executed the method but returns no response body. |

## Error

In the responses for failed requests, Citi returns HTTP **4XX** or **5XX** status codes.

For all errors except Identity errors, Citi returns an error response body that includes additional error details in this format:

{

"name": "ERROR\_NAME",

"message": "ERROR\_DESCRIPTION",

"information\_link": "ERROR\_DOCUMENTATION\_LINK",

// Some types of errors also include a details array:

"details": [

{

"field": "field\_name",

"issue": "problem\_with\_field"

}

]

}

The response body for Identity errors includes additional error details in this format:

{

"error": "ERROR\_NAME",

"error\_description": "ERROR\_DESCRIPTION"

}

In the responses, Citi returns these HTTP status codes for failed requests:

| **HTTP status code** | **Typical error code and error message** | **Cause** |
| --- | --- | --- |
| **400 Bad Request** | **INVALID\_REQUEST**. Request is not well-formed, syntactically incorrect, or violates schema. | See Validation errors. The server could not understand the request. Indicates one of these conditions:   * The API cannot convert the payload data to the underlying data type. * The data is not in the expected data format. * A required field is not available. * A simple data validation error occurred. |
| **401 Unauthorized** | **AUTHENTICATION\_FAILURE**. Authentication failed due to invalid authentication credentials. | See Authentication errors. The request requires authentication and the caller did not provide valid credentials. |
| **403 Forbidden** | **NOT\_AUTHORIZED**. Authorization failed due to insufficient permissions. | The client is not authorized to access this resource although it might have valid credentials. For example, the client does not have the correct OAuth 2 scope. Additionally, a business-level authorization error might have occurred. For example, the account holder does not have sufficient funds. |
| **404 Not Found** | **RESOURCE\_NOT\_FOUND**. The specified resource does not exist. | The server did not find anything that matches the request URI. Either the URI is incorrect or the resource is not available. For example, no data exists in the database at that key. |
| **405 Method Not Allowed** | **METHOD\_NOT\_SUPPORTED**. The server does not implement the requested HTTP method. | The service does not support the requested HTTP method. For example, **PATCH**. |
| **406 Not Acceptable** | **MEDIA\_TYPE\_NOT\_ACCEPTABLE**. The server does not implement the media type that would be acceptable to the client. | The server cannot use the client-request media type to return the response payload. You will get this error if the client sends an **Accept: application/xml**request header but the API can generate only an **application/json** response. |
| **415** **Unsupported** **Media** **Type** | **UNSUPPORTED\_MEDIA\_TYPE**. The server does not support the request payload’s media type. | The API cannot process the media type of the request payload. For example, this error occurs if the client sends a **Content-Type: application/xml**request header but the API can only accept **application/json** request payloads. |
| **422 Unprocessable Entity** | **UNPROCCESSABLE\_ENTITY**. The API cannot complete the requested action, or the request action is semantically incorrect or fails business validation. | The API cannot complete the requested action and might require interaction with APIs or processes outside of the current request. No systemic problems limit the API from completing the request. For example, this error occurs for any business validation errors, including errors that are not usually of the **400** type. |
| **429 Unprocessable Entity** | **RATE\_LIMIT\_REACHED**. Too many requests. Blocked due to rate limiting. | The rate limit for the user, application, or token exceeds a predefined value. See RFC 6585. |
| **500 Internal Server Error** | **INTERNAL\_SERVER\_ERROR**. An internal server error has occurred. | A system or application error occurred. Even if the client provides a correct request, something unexpected occurred on the server. |

|  |  |  |
| --- | --- | --- |
| **503 Service Unavailable** | **SERVICE\_UNAVAILABLE**. Service Unavailable. | The server cannot process the request for a service because of temporary maintenance. |