Recurring Billing

REST API

cybersource
A Visa Solution

Developer Guide

Cybersource Contact Information

For general information about our company, products, and services, go to https://www.cybersource.com.

For sales questions about any Cybersource service, email sales@cybersource.com or call 650-432-7350 or 888-330-2300 (toll free in the United States).

For support information about any Cybersource service, visit the Support Center: https://www.cybersource.com/support

Revision

Version: 25.06.01

Contents

Recurring Billing Developer Guide

This section describes how to use this guide and where to find further information.

Audience and Purpose	This guide is for merchants who use the upgraded or new Cybersource Recurring Billing service. The service is available through the Business Center and the REST API.
Conventions	This statement appears in this document: **Important** An Important statement contains information essential to successfully completing a task or learning a concept.
Related Documentation	Visit the *Cybersource documentation hub* to find additional technical documentation.
Customer Support	For support information about any service, visit the Support Center: *http://support.visaacceptance.com*

Recent Revisions to This Document

25.06.01

Updated information about the amount of time allowed between subscription payments in these sections:

- *Creating a Plan* on page 12
- *Subscription ID* on page 30

25.05.02

Updated this section:

- *System Retry Logic* on page 54

25.05.01

Added these sections:

- *Creating a Follow on Subscription from an Existing Transaction* on page 37
- *Retrieving Details for Follow on Subscription Creation Based on Existing Transaction* on page 39

Updated these sections:

- *Account Validation (Pre-Authorization)* on page 9
- *Subscription Statuses* on page 27
- *Zero-Amount Authorizations* on page 29

Updated these sections:

- *Creating a Subscription with an Existing Plan* on page 30
- *Creating a Subscription with Plan Overrides* on page 32
- *Creating a Fully Customized Subscription with a One-Time Plan* on page 35
- *Amending a Subscription* on page 47

Updated the example in this section:

- *REST Example: Switching a Subscription to a Different Plan* on page 49

Updated the graphics in these sections:

- *Plan Statuses* on page 11
- *Subscription Statuses* on page 27

24.10.01

Added these topics:

- *Creating a Subscription with an Existing Plan* on page 30
- *Creating a Subscription with Plan Overrides* on page 32
- *Creating a Fully Customized Subscription with a One-Time Plan* on page 35
- *Amendable Fields* on page 48

24.09.01

Added Information about system retry logic for recurring billings with payments on a custom frequency. See *System Retry Logic* on page 54.

24.06.01

Updated the introduction and also noted that Recurring Billing is not available for the SCMP API or the Simple Order API. See *Introduction to Recurring Billing* on page 8.

24.04.01

This revision contains only editorial changes and no technical updates.

Introduction to Recurring Billing

This guide explains how to integrate to the Recurring Billing REST API.
The Recurring Billing service enables you to create and manage payment plans and subscriptions for recurring payment schedules. It automates the storage and handling of your customer's payment information and personal data within secure Visa data centers in compliance with credentials-on-file (COF) best practices. Storage risks and the PCI DSS scope are reduced through the use of the Token Management Service (TMS).
Cybersource Recurring Billing consists of these three elements:

- Plan: Stores the billing schedule.
- Subscription: Combines the token and plan and defines the subscription start date, name, and description.
- Token: Stores customer billing, shipping, and payment details.

For information about Recurring Billing in the Business Center, see the *Recurring Billing User Guide*.
Recurring payments can also be handled with the payments API. Merchant-initiated transactions (MITs) are part of the payments API. For more information on recurring payments using MITs, see the Recurring Payments section in the *Payments Developer Guide*.

Important

Do not use this document if you are using the payments API to process recurring payments. When using payments API for MITs, you must capture and store the customer's payment credentials manually. Also, you send the payments API MIT requests to different endpoints than you send the recurring billing requests.

The Recurring Billing service is available for the REST API only. The service is not available in the SCMP API or the Simple Order API.

Supported Processors

For more information, see the *Token Management Service REST API Developer Guide*.

Important

These Latin American processors are not yet supported for Recurring Billing services:

- Comercio Latino
- Prosa

Prerequisites

Your account must be enabled for Recurring Billing and configured for the Token Management Service (TMS). The customer token is the only token type that can be used with Recurring Billing.
See the *Token Management Service REST API Developer Guide*.

Getting Started with the REST API

If you have not already, you must register and obtain authentication credentials for the REST API.
Go to the Cybersource *Hello world sandbox* in the Developer Center.

Account Validation (Pre-Authorization)

When you capture payment card data for future use, Cybersource recommends that you validate the payment card account to ensure that the account number entered is an active, valid account. You can also use the Address Verification Service (AVS), the Card Verification Number (CVN), and expiration date validation. To validate the account, initiate a zero-amount authorization transaction. To verify that your processor supports zero-amount authorization, see the zero-amount authorization information in the *Payment Services Optional Features Supplement*.
If your processor does not support account validation, you can use the minimal supported amount and do a reversal later. You can also enable the Data Enrichment for Card Verification feature, which automatically chooses the minimal auth amount. Contact customer support for more details.
For more information about how to perform a pre-authorization, see *Authorization with Customer Token Creation* in the Token Management Service Developer Guide.

Plans

A plan defines the payment amount and billing schedule for recurring or subscription payments.
After you create a plan, you can assign it to multiple subscriptions and manage changes for multiple subscriptions.

Important

Installment based plans are currently not supported.

Recurring Billing supports two types of plans:

- Standard plans: created and stored within the recurring billing service for re-use. You can assign these plans to multiple subscriptions.
- One-time plans: created specifically for a single subscription and not stored for re-use nor assigned to other subscriptions.

Both plan types have two billing cycle options. Choose one of these billing cycles when defining the number of recurring payments to be made:

- Bill indefinitely
- Fixed number of payments

Bill Indefinitely

This option has a fixed payment amount and a payment schedule with no end date. This option comprises these elements:

- Plan code
- Plan name
- Plan description (optional)
- Billing frequency
- Currency
- Payment amount
- Set-up fee (optional)

Important

One-time plans do not include a plan code, plan name, or plan description.

Fixed Number of Payments

This option has a fixed payment amount, a payment schedule, and a fixed number of payments. This option comprises these elements:

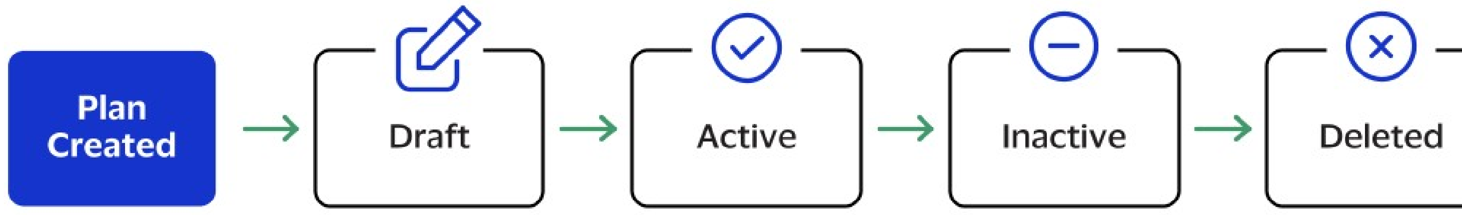
- Plan code
- Plan name
- Plan description (optional)
- Billing frequency
- Currency
- Payment amount
- Number of billing cycles
- Set-up fee (optional)

Important

One-time plans do not include a plan code, plan name, or plan description.

Plan Statuses

A plan has one of these statuses:



Plan Status Flow

- Draft: a draft plan cannot be assigned to new or existing subscriptions. Draft plans can be changed to active or deleted.
- Active: an active plan can be used when you create a new subscription or assigned to an existing subscription. Active plans can be changed to inactive or deleted.
- Inactive: subscriptions that are assigned to an inactive plan are billed until the subscription is completed or canceled. You cannot use inactive plans to create new subscriptions, nor switch subscriptions to an inactive plan. Inactive plans can be changed to active.
- Deleted: a deleted plan is removed from the system.

Important

You cannot amend an inactive plan. To make changes, move the plan to active or draft status.

Important

You can delete a standard plan that has a draft status or has an active or inactive status and has never been assigned to any subscriptions.

Assigning a Plan Code

When you create a standard plan, you can choose to supply a plan code that relates to your business and is used for reference to the plan. This code can be numeric or alphanumeric with dash (-) and dot (.) characters, and up to 10 characters long. If no plan code is assigned, the system automatically assigns a system-generated value.

Creating a Plan

Create a plan to define the billing schedule, which includes the amount, frequency, and billing cycles for a subscription. The interval between subscription payments cannot exceed 12 months.

Important

If a subscription is to use a one-time plan instead of a standard plan, you create the one-time plan when you create the subscription. The plan details are embedded in the subscription request.

When you create a plan, the system assigns an ID to the plan. Use the plan ID to request these actions:

- Creating a subscription
- Retrieving a plan
- Amending a plan
- Activating a plan
- Deactivating a plan
- Deleting a plan

Follow these steps to create a plan:

1. Send the request to the recurring billing endpoint:
    - Production: `POST https://api.cybersource.com/rbs/v1/plans`

- Test: `POST https://apitest.cybersource.test.com/rbs/v1/plans`

2. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

Required Fields for Creating a Plan

Include these fields in your request to create a plan:

orderInformation.amountDetails.billingAmount	
orderInformation.amountDetails.currency	
planInformation.billingCycles.total	**Required for a plan with a defined plan period.**
planInformation.billingPeriod.length	
planInformation.billingPeriod.unit	
planInformation.name	

Related Information

- *API field reference guide for the REST API*

Optional Fields for Creating a Plan

orderInformation.amountDetails.setupFee

planInformation.code

planInformation.description

planInformation.status

Related Information

- *API field reference guide for the REST API*

REST Example: Creating a Plan

Request

```
{
  "planInformation": {
    "billingPeriod": {
      "unit": "w",
      "length": "1"
    },
    "billingCycles":
    {
      "total": "4"
    },
      "code":"1619310018",
      "name": "Test plan",
      "description": "Description",
      "status":"active"
   },
   "orderInformation": {
      "amountDetails": {
         "billingAmount": "7",
         "currency": "USD",
         "setupFee": "0"
      }
   }
}
```

Response to a Successful Request

```
{
   "_links": {
      "self": {
         "href": "/rbs/v1/plans/1619212820",
         "method": "GET"
      },
      "update": {
         "href": "/rbs/v1/plans/1619212820",
         "method": "PATCH"
      },
      "deactivate": {
         "href": "/rbs/v1/plans/1619212820/deactivate",
         "method": "POST"
      }
   },
   "id": "1619212820",
   "status": "COMPLETED",
   "planInformation": {
      "code": "1619310018",
      "status": "ACTIVE"
   }
}
```

Error Response

```
{
  "status": "INVALID_REQUEST",
  "reason": "INVALID_DATA",
  "message": "One or more fields in the request contains invalid data.",
  "details": [
     {
       "field": "planInformation.code",
       "reason": "DUPLICATE"
     }
   ]
 }
```

Retrieving a List of Plans

A list of plans provides these details for each plan:

- Plan ID
- Plan code
- Plan name
- Description
- Status
- Billing period unit
- Billing period length
- Billing cycles total
- Currency
- Billing amount
- Set-up fee

After you retrieve the list of plans, use the plan ID to retrieve, amend, activate, deactivate, or delete a subscription.
Follow these steps to retrieve a list of plans:

1. Filter the list of plans using these query string parameters:
   - `filters`: Use Lucene query syntax. Only keyword-matching and `AND` are supported. Example: `name:"Test plan" AND code:"009" AND status:"ACTIVE"`
   - `offset`: Page offset number.
   - `limit`: Number of items to be returned. Default is `20` and maximum is `100`.
2. Send to one of these endpoints:
   Production: `GET https://api.cybersource.com/rbs/v1/plans`
   Test: `GET https://apitest.cybersource.test.com/rbs/v1/plans`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Example: Retrieving a List of Plans

Response to a Successful Request

```
{
  "_links": {
    "self": {
      "href": "/rbs/v1/plans?limit=2",
      "method": "GET"
    },
    "next": {
      "href": "/rbs/v1/plans?offset=2&limit=2",
      "method": "GET"
    }
    },
    "totalCount": 96,
    "plans": [
      {
        "_links": {
          "self": {
            "href": "/rbs/v1/plans/1619212820",
            "method": "GET"
          },
          "update": {
            "href": "/rbs/v1/plans/1619212820",
            "method": "PATCH"
          },
          "deactivate": {
            "href": "/rbs/v1/plans/1619212820/deactivate",
            "method": "POST"
          }
        },
        "id": "1619212820",
        "planInformation": {
          "code": "1619310018",
          "status": "ACTIVE",
          "name": "Test plan",
          "description": "Description",
          "billingPeriod": {
            "length": "1",
            "unit": "W"
          },
          "billingCycles": {
            "total": "4"
          }
        },
        "orderInformation": {
          "amountDetails": {
            "currency": "USD",
            "billingAmount": "7.00",
            "setupFee": "0.00"
          }
        }
      },
      {
        "_links": {
          "self": {
            "href": "/rbs/v1/plans/6183561970436023701960",
            "method": "GET"
          },
          "update": {
            "href": "/rbs/v1/plans/6183561970436023701960",
            "method": "PATCH"
          },
          "activate": {
            "href": "/rbs/v1/plans/6183561970436023701960/activate",
            "method": "POST"
          }
        },
        "id": "6183561970436023701960",
        "planInformation": {
          "code": "1616024773",
          "status": "DRAFT",
          "name": "Plan Test",
          "description": "12123",
          "billingPeriod": {
            "length": "9999",
            "unit": "Y"
          },
          "billingCycles": {
            "total": "123"
          }
        },
        "orderInformation": {
          "amountDetails": {
            "currency": "USD",
            "billingAmount": "1.00",
            "setupFee": "0.00"
          }
        }
      }
    ]
}
```

Error Response

```
{
  "status": "INVALID_REQUEST",
  "reason": "VALIDATION_ERROR",
  "message": "Field validation errors.",
  "details": [
    {
      "field": "customerInformation.email",
      "reason": "Invalid email"
    }
  ]
}
```

Retrieving a Plan

You can retrieve the details of a specific plan using the plan ID. These plan details are returned in the response:

- Plan ID
- Plan code
- Plan name
- Description
- Status
- Billing period unit
- Billing period length

- Billing cycles total
- Currency
- Billing amount
- Set-up fee

Follow these steps to retrieve a plan:

1. In the endpoint path, include the plan ID that you received when you retrieved a list of plans.
2. Send the request to the recurring billing endpoint:
   Production: `GET https://api.cybersource.com/rbs/v1/plans/{id}`
   Test: `GET https://apitest.cybersource.test.com/rbs/v1/plans/{id}`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Example: Retrieving a Plan

Response to a Successful Request

```
{
  "_links": {
    "self": {
      "href": "/rbs/v1/plans/6183561970436023701960",
      "method": "GET"
    },
    "update": {
      "href": "/rbs/v1/plans/6183561970436023701960",
      "method": "PATCH"
    },
    "activate": {
      "href": "/rbs/v1/plans/6183561970436023701960/activate",
      "method": "POST"
    }
  },
  "id": "6183561970436023701960",
  "planInformation": {
    "code": "1616024773",
    "status": "DRAFT",
    "name": "Plan Test",
    "description": "12123",
    "billingPeriod": {
      "length": "9999",
      "unit": "Y"
    },
    "billingCycles": {
      "total": "123"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "currency": "USD",
      "billingAmount": "1.00",
            "setupFee": "0.00"
        }
    }
}
```

Error Response

```
{
  "status": "NOT_FOUND",
  "reason": "INVALID_DATA"
}
```

Retrieving a Plan Code

When you specify your own plan codes, you can request the next consecutive alphabetical or numeric plan code based on the last plan code that you specified. For example, if the last plan code that you specified was `Plan104`, the system returns the code `Plan105`.

Follow these steps to retrieve a plan code:

1. Send the request to the Recurring Billing endpoint:
   Production: `GET https://api.cybersource.com/rbs/v1/plans/code`
   Test: `GET https://apitest.cybersource.test.com/rbs/v1/plans/code`
2. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Example: Retrieving a Plan Code

Response to a Successful Request

```
{
   "code": "1619310019",
}
```

Error Response

```
{
  "status": "NOT_FOUND",
  "reason": "INVALID_DATA"
}
```

Amending a Plan

Amend a plan by entering one of two values in the processingInformation.subscriptionBillingOptions.applyTo field. Possible values:

- `ALL`: change is applied to all subscriptions (existing and new).
- `NEW` (default): change is applied to new subscriptions only.

You can also make changes to individual subscriptions. See *Amending a Subscription* on page 47.
For a draft plan, you can amend all plan information.
For an active plan, you can amend only the information in these fields:

- **planInformation.billingPeriod**
- **planInformation.billingCycles**
- **orderInformation.amountDetails.currency**

Important

You cannot amend an inactive plan. To make changes, move the plan to active or draft status.

Follow these steps to amend a plan:

1. Include any optional fields.
2. In the endpoint path, include the plan ID that you received when you retrieved a list of plans.
3. Send the request to the Recurring Billing endpoint:
   Production: `PATCH https://api.cybersource.com/rbs/v1/plans/{id}`
   Test: `PATCH https://apitest.cybersource.test.com/rbs/v1/plans/{id}`
4. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

Optional Fields for Amending a Plan

These fields are optional for amending a plan:

orderInformation.amountDetails.billingAmount	
orderInformation.amountDetails.setupFee	
planInformation.billingCycles.total	**Only increases to the number of billing cycles are allowed.**
planInformation.code	
planInformation.description	
planInformation.name	
processingInformation.subscriptionBillingOptions.applyTo	**Default value is `NEW`.**

Related Information

- *API field reference guide for the REST API*

REST Example: Amending a Plan

Request

```
{
   "planInformation": {
      "name": "AmendPlan",
      "description": "Amend Plan 1610394600",
      "billingPeriod":{
         "length": "4",
         "unit": "M"
      },
      "billingCycles":{
         "total": "7"
      }
   },
   "orderInformation": {
      "amountDetails": {
         "billingAmount": "38.00",
         "setupFee": "35"
      }
   },
   "processingInformation": {
      "subscriptionBillingOptions": {
         "applyTo": "all"
      }
   }
}
```

Response to a Successful Request

```
{
   "_links": {
      "self": {
         "href": "/rbs/v1/plans/6183561970436023701960",
         "method": "GET"
      },
      "update": {
         "href": "/rbs/v1/plans/6183561970436023701960",
         "method": "PATCH"
      },
      "activate": {
         "href": "/rbs/v1/plans/6183561970436023701960/activate",
         "method": "POST"
      }
   },
   "id": "6183561970436023701960",
   "submitTimeUtc": "2023-04-13T21:39:34.993Z",
   "status": "COMPLETED",
   "planInformation": {
      "code": "1616024773",
      "status": "DRAFT"
   }
}
```

Error Response

```
{
  "status": "NOT_FOUND",
  "reason": "INVALID_DATA",
  "message": "One or more fields in the request contains invalid data.",
  "details": [
    {
      "field": "subscriptionInformation.planId",
      "reason": "NOT_FOUND"
    }
  ]
}
```

Activating a Plan

You can activate a plan that has a draft or inactive status and that has never been assigned to any subscriptions.
Follow these steps to activate a plan:

1. In the endpoint path, include the plan ID that you received when you retrieved a list of plans.
2. Send the request to the Recurring Billing endpoint:
   Production: `POST https://api.cybersource.com/rbs/v1/plans/{id}/activate`
   Test: `POST https://apitest.cybersource.test.com/rbs/v1/plans/{id}/activate`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Example: Activating a Plan

Response to a Successful Request

```
{
  "_links": {
    "self": {
      "href": "/rbs/v1/plans/1619214189",
      "method": "GET"
    },
    "update": {
      "href": "/rbs/v1/plans/1619214189",
      "method": "PATCH"
    },
    "deactivate": {
      "href": "/rbs/v1/plans/1619214189/deactivate",
      "method": "POST"
    }
  },
  "id": "1619214189",
  "status": "COMPLETED",
  "planInformation": {
    "code": "1619214189",
    "status": "ACTIVE"
  }
}
```

Error Response

```
{
    "status": "NOT_FOUND",
    "reason": "INVALID_DATA",
    "message": "One or more fields in the request contains invalid data.",
    "details": [
      {
        "field": "subscriptionInformation.planId",
        "reason": "NOT_FOUND"
      }
    ]
}
```

Deactivating a Plan

You can deactivate a specific plan that has an active status.
Follow these steps to deactivate a plan:

1. In the endpoint path, include the plan ID that you received when you retrieved a list of plans.
2. Send the request to the Recurring Billing endpoint:
   Production: `POST https://api.cybersource.com/rbs/v1/plans/{id}/deactivate`
   Test: `POST https://apitest.cybersource.test.com/rbs/v1/plans/{id}/deactivate`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Example: Deactivating a Plan

Response to a Successful Request

```
{
    "_links": {
      "self": {
        "href": "/rbs/v1/plans/1619214189",
        "method": "GET"
      },
      "activate": {
        "href": "/rbs/v1/plans/1619214189/activate",
        "method": "POST"
      }
    },
    "id": "1619214189",
    "status": "COMPLETED",
    "planInformation": {
      "code": "1619214189",
      "status": "INACTIVE"
    }
}
```

Error Response

```
{
  "status": "INVALID_REQUEST",
  "reason": "INVALID_DATA",
  "message": "One or more fields in the request contains invalid data.",
  "details": [
    {
      "field": "planInformation.status",
      "reason": "INVALID_DATA"
    }
  ]
}
```

Deleting a Plan

You can delete a standard plan that has a draft status. You can also delete a plan that has an active or inactive status and has never been assigned to any subscriptions.

Follow these steps to delete a plan:

1. In the endpoint path, include the plan ID that you received when you retrieved a list of plans.
2. Send the request to the Recurring Billing endpoint:
   Production: `DELETE https://api.cybersource.com/rbs/v1/plans/{id}`
   Test: `DELETE https://apitest.cybersource.test.com/rbs/v1/plans/{id}`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Example: Deleting a Plan

Response to a Successful Request

```
{
 "status": "COMPLETED"
}
```

Error Response

```
{
  "status": "NOT_FOUND",
  "reason": "NOT_FOUND",
  "message": "One or more fields in the request contains invalid data.",
   "details": [
      {
         "field": "subscriptionInformation.planId",
         "reason": "INVALID_DATA"
      }
   ]
}
```

Subscriptions

Subscriptions store customer details by using a Token Management Service (TMS) token. You must have an assigned payment plan. Subscriptions consist of this information:

- Subscription code
- Subscription name
- Start date
- Token Management Service (TMS) token
- Plan: standard or one-time

Subscriptions can be updated, activated, suspended, or canceled.

Subscription Prerequisites

Before creating a subscription, you must create a customer token.
See *Create a Customer Token with Validated Payment Details* in the Token Management Service Developer Guide.
When you create a subscription, to include the values you received in the response to your request:

- **tokenInformation.customer.id**
- **processorInformation.networkTransactionId**
- **orderInformation.amountDetails.authorizedAmount**

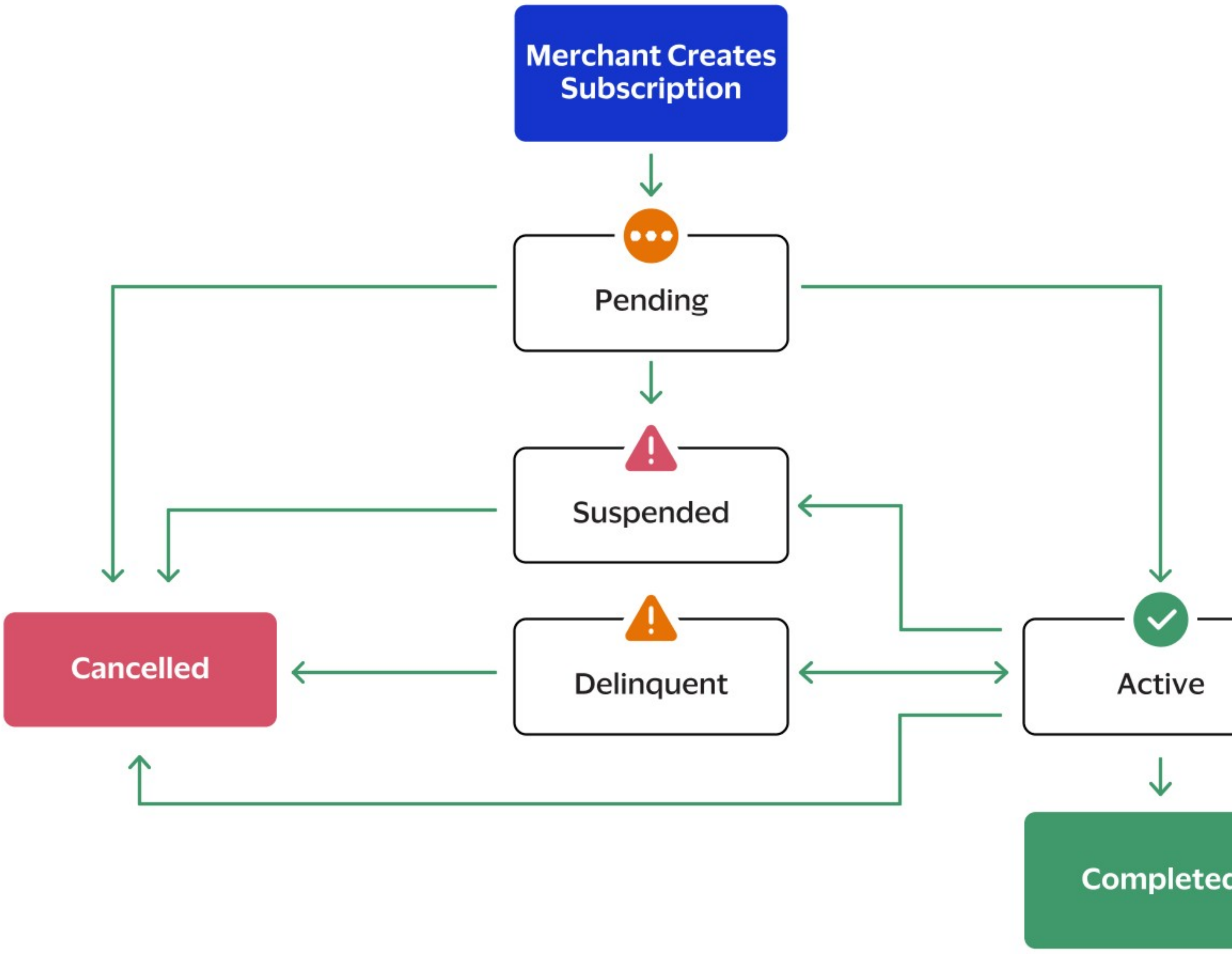
See the table below for the payments to Recurring Billing subscription field mapping:

Payments API Response Field	Recurring Billing Create Subscription Request Field	Required Value Information
tokenInformation.customer.id	paymentInformation.customer.id	Customer token ID

processorInformation.networkTransactionId	subscriptionInformation.originalTransactionId	Network token for the transaction initializing the subscription.
orderInformation.amountDetails.authorizedAmount	subscriptionInformation.originalTransactionAuthorizedAmount	Authorized amount for the transaction initializing the subscription. Required only for Diners or Discover cards.

Subscription Statuses

This diagram illustrates the statuses of a subscription from creation to completion.



Subscription Flow

A subscription has one of these statuses:

Pending	The first payment is scheduled, or the subscription is in transition to another state.
Active	The subscription is currently in use. It is set with a payment instrument, and a payment is scheduled at a pre-determined frequency that you agreed upon with your customer.
Delinquent	A scheduled payment was declined. The system automatically retries the payment a number of times. If the payment succeeds, the system automatically moves the subscription back to the active state. Otherwise, if the automated retry logic fails to obtain a successful payment, the system automatically moves the subscription to the suspended state.
Suspended	The automated retry logic failed to obtain successful payment, or you have explicitly suspended the subscription. In order to resume a suspended subscription for the next billing cycle, choose one of these options: • Collect a different payment method from your customer and then reactivate the subscription. • Cancel the subscription and create a new subscription for your customer.
Cancelled	You have explicitly cancelled the subscription and it cannot be reactivated. You might cancel an active or pending subscription when you and the customer agree to end the subscription. You might choose to cancel a delinquent subscription rather than wait for the automatic retry logic to proceed. You might cancel a suspended subscription if the customer does not have an acceptable alternate payment method. **Important**

	You cannot cancel a subscription within 10 minutes before or after a payment begins processing.
Completed	All scheduled payments were made. This is the state of a subscription that ends with all scheduled payments successfully completed. This state applies to subscriptions set up with a scheduled end date. **Important** You cannot reactivate a completed subscription.

Zero-Amount Authorizations

When you create a subscription before the subscription start date, Cybersource verifies the account with a zero-amount authorization to ensure that the stored card details are valid.
When you create a subscription on the subscription start date, Cybersource does not perform a zero-amount authorization because the first recurring payment is processed immediately.

Important

Deprecated flow: This behavior occurs only when you do not provide the value for **subscriptionInformation.originalTransactionId** and application performs the initial authorization. This flow does not ensure Strong Customer Authentication (SCA) compliance.

Assigning a Subscription Code

When you create a subscription, you can supply a subscription code that relates to your business and that is used for reference to the subscription. This code can be numeric or alphanumeric with dash (-) and dot (.) characters and can be up to 10 characters long.
If you do not supply a subscription code, the recurring billing system automatically assigns a code.

Subscription ID

When you create a subscription, the system assigns an ID to the subscription that you can use for these actions:

- Retrieving a subscription
- Amending a subscription
- Canceling a subscription
- Re-activating a subscription
- Suspending a subscription

Important

The interval between subscription payments cannot exceed 12 months.

Overriding a Plan

When assigning a plan to a subscription or amending a subscription, you can amend standard plan details for individual subscriptions. For example, such changes might include an amendment to the billing amount, plan duration (billing cycles) amendment.

Important

Plan overrides apply only to the individual subscription and do not amend the standard plan details used for other subscriptions.

Creating a Subscription with an Existing Plan

You can shows how to create a subscription with a standard plan.
The start date must be in coordinated universal time (UTC) in this format: YYYY-MM-DDThh:mm:ssZ. The T separates the date and the time. The Z indicates UTC. For example, `2023-08-11T22:47:57Z` indicates August 11, 2023, at 22:47:57 (10:47:57PM). For subscriptions created on the start date, set the time to the current time and day in your time zone.

Basic Steps

Follow these steps to create a subscription:

1. Create the request with the required API fields.
2. Send the request to one of these endpoints:
   - Production: `POST https://api.cybersource.com/rbs/v1/subscriptions`
   - Test: `POST https://apitest.cybersource.test.com/rbs/v1/subscriptions`

3. Verify the response messages to make sure that the request was successful. A 200-level HTTP response code indicates success. See *Transaction Response Codes*.

Required Fields

These fields are required for creating a subscription with an existing plan:

paymentInformation.customer.id	
subscriptionInformation.name	
subscriptionInformation.planId	**Required when you are using a standard plan.**
subscriptionInformation.startDate	

Optional Field

subscriptionInformation.originalTransactionId	Including this field ensures better authorization rates and Strong Customer Authentication (SCA) compliance where necessary.

Related Information

- *API field reference guide for the REST API*

REST Example: Creating a Subscription with an Existing Plan

This example shows how to create a subscription with an existing plan.

This example shows how to create a subscription with an existing plan.

Request

```
{
  "subscriptionInformation": {
    "planId":"1619214515",
    "name": "Daily Gym Subscription",
    "startDate": "2023-04-15T17:01:42Z",
    "originalTransactionId": “016153570198200"
  },
  "paymentInformation": {
    "customer": {
      "id": "C09F227C54F94951E0533F36CF0A3D91"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "self": {
      "href": "/rbs/v1/subscriptions/1619214690",
            "method": "GET"
        },
        "update": {
            "href": "/rbs/v1/subscriptions/1619214690",
            "method": "PATCH"
        },
        "cancel": {
            "href": "/rbs/v1/subscriptions/1619214690/cancel",
            "method": "POST"
        }
    },
    "id": "1619214690",
    "status": "COMPLETED",
    "subscriptionInformation": {
        "code": "AWC-47",
        "status": "PENDING"
    }
}
```

Response to a Failed Request

```
{
    "status": "INVALID_REQUEST",
    "reason": "INVALID_DATA",
    "message": "One or more fields in the request contains invalid data.",
    "details": [
        {
            "field": "subscriptionInformation.startDate",
            "reason": "INVALID_DATA"
        }
    ]
}
```

Creating a Subscription with Plan Overrides

You can create a subscription with a plan overrides.
The start date must be in coordinated universal time (UTC) in this format: YYYY-MM-DDThh:mm:ssZ. The T separates the date and the time. The Z indicates UTC. For example, `2023-08-11T22:47:57Z` indicates August 11, 2023, at 22:47:57 (10:47:57PM). For subscriptions created on the start date, set the time to the current time and day in your time zone.

Fields Specific to This Use Case

These REST API request fields and values are specific to this use case:

- **subscriptionInformation.name**
- **subscriptionInformation.planId**
- **subscriptionInformation.startDate**
- **subscriptionInformation.originalTransactionId**

These REST API request fields are optional:

- **orderInformation.amountDetails.billingAmount**
- **orderInformation.amountDetails.currency**
- **orderInformation.amountDetails.setupFee**
- **planInformation.billingPeriod.length**
- **planInformation.billingPeriod.unit**
- **planInformation.billingCycles.total**

Basic Steps

Follow these steps to create a subscription:

1. Create the request with the required API fields.
2. Send the request to one of these endpoints:
   - Production: `POST https://api.cybersource.com/rbs/v1/subscriptions`
   - Test: `POST https://apitest.cybersource.test.com/rbs/v1/subscriptions`
3. Verify the response messages to make sure that the request was successful. A 200-level HTTP response code indicates success. See *Transaction Response Codes*.

Required Fields

These fields are required for creating a subscription with plan overrides:

paymentInformation.customer.id

subscriptionInformation.name

subscriptionInformation.planId

subscriptionInformation.startDate

Optional Field

subscriptionInformation.originalTransactionId	Including this field ensures better authorization rates and Strong Customer Authentication (SCA) compliance where necessary.

Related Information

- *API field reference guide for the REST API*

REST Example: Creating a Subscription with Plan Overrides

This example show you how to create a subscription with plan overrides.

Request

```
{
   "subscriptionInformation": {
      "planId": "1619214515",
      "name": "SubName With Overrides",
      "startDate": "2023-04-16T17:01:42Z",
      "originalTransactionId": "016153570198200"
   },
   "planInformation": {
      "billingCycles": {
         "total": "3"
      }
   },
   "orderInformation": {
      "amountDetails": {
         "billingAmount": "13.14",
         "setupFee": "1.27"
      }
   },
   "paymentInformation": {
      "customer": {
         "id": "C09F227C54F94951E0533F36CF0A3D91"
      }
   }
}
```

Response to a Successful Request

```
{
   "_links": {
      "self": {
         "href": "/rbs/v1/subscriptions/1619214795",
         "method": "GET"
      },
      "update": {
         "href": "/rbs/v1/subscriptions/1619214795",
         "method": "PATCH"
      },
      "cancel": {
         "href": "/rbs/v1/subscriptions/1619214795/cancel",
         "method": "POST"
      }
   },
   "id": "1619214795",
   "status": "COMPLETED",
   "subscriptionInformation": {
      "code": "AWC-48",
      "status": "PENDING"
   }
}
```

Error Response to a Failed Request

```
{
   "status": "INVALID_REQUEST",
   "reason": "DUPLICATE_REQUEST",
   "message": "Duplicate requests are not supported within 15 minutes.",
   "details": [
      {
         "field": "subscriptionInformation.planId or paymentInformation.customer.id or
 subscriptionInformation.startDate or subscriptionInformation.name",
        "subscriptionId": "1619214795",
        "reason": "INVALID_DATA"
      }
    ]
  }
```

Creating a Fully Customized Subscription with a One-Time Plan

You can create a subscription with a one-time plan.
The start date must be in coordinated universal time (UTC) in this format: `YYYY-MM-DDThh:mm:ssZ`. The T separates the date and the time. The `Z` indicates UTC. For example, `2023-08-11T22:47:57Z` indicates August 11, 2023, at 22:47:57 (10:47:57PM). For subscriptions created on the start date, set the time to the current time and day in your time zone.

Fields Specific to This Use Case

These REST API request fields and values are specific to this use case:

- **orderInformation.amountDetails.billingAmount**
- **orderInformation.amountDetails.currency**
- **orderInformation.amountDetails.setupFee**
- **planInformation.billingCycles.total**
- **planInformation.billingPeriod.length**
- **planInformation.billingPeriod.unit**
- **subscriptionInformation.startDate**

Basic Steps

Follow these steps to create a subscription:

1. Create the request with the required API fields.
2. Send the request to one of these endpoints:
   - Production: `POST https://api.cybersource.com/rbs/v1/subscriptions`
   - Test: `POST https://apitest.cybersource.test.com/rbs/v1/subscriptions`
3. Verify the response messages to make sure that the request was successful. A 200-level HTTP response code indicates success. See *Transaction Response Codes*.

Required Fields

These fields are required for creating a subscription with subscription one-time plan:

orderInformation.amountDetails.billingAmount

orderInformation.amountDetails.currency

orderInformation.amountDetails.setupFee

paymentInformation.customer.id

planInformation.billingPeriod.length

planInformation.billingPeriod.unit

subscriptionInformation.name

subscriptionInformation.startDate

Optional Field

subscriptionInformation.originalTransactionId	Including this field ensures better authorization rates and Strong Customer Authentication (SCA) compliance where necessary.

Related Information

- *API field reference guide for the REST API*

REST Example: Creating a Fully Customized Subscription with a One-Time Plan

This example creates a fully customized subscription with a one-time plan.

Example Request

```
{
   "subscriptionInformation": {
      "name": "SubName Testing",
      "startDate": "2023-04-18T17:01:42Z",
      "originalTransactionId": “016153570198200"
   },
   "planInformation": {
      "billingCycles": {
         "total":"5"
      },
       "billingPeriod": {
         "length": "3",
         "unit":"D"
      }
   },
   "orderInformation": {
      "amountDetails": {
         "billingAmount": "1.21",
         "setupFee": "1.44",
         "currency":"USD"
      }
   },
   "paymentInformation": {
      "customer": {
         "id": "C09F227C54F94951E0533F36CF0A3D91"
      }
   }
}
```

Example Response to a Successful Request

```
{
   "_links": {
      "self": {
         "href": "/rbs/v1/subscriptions/1619214861",
         "method": "GET"
      },
      "update": {
         "href": "/rbs/v1/subscriptions/1619214861",
         "method": "PATCH"
      },
      "cancel": {
         "href": "/rbs/v1/subscriptions/1619214861/cancel",
         "method": "POST"
      }
   },
   "id": "1619214861",
   "status": "COMPLETED",
   "subscriptionInformation": {
      "code": "AWC-49",
      "status": "PENDING"
   }
}
```

Example Error Response to a Failed Request

```
{
   "status": "INVALID_REQUEST",
   "reason": "INVALID_DATA",
   "message": "One or more fields in the request contains invalid data.",
   "details": [
      {
         "field": "planInformation.billingPeriod.length",
         "reason": "MAX_LENGTH"
      }
   ]
}
```

Creating a Follow on Subscription from an Existing Transaction

You can create a subscription using an existing successful transaction by specifying its request ID in the path. This method eliminates the need to provide a customer token. You can use an existing or one-time plan.
The start date must be in coordinated universal time (UTC) in this format: YYYY-MM-DDThh:mm:ssZ. The T separates the date and the time. The Z indicates UTC. For

example, `2023-08-11T22:47:57Z` indicates August 11, 2023, at 22:47:57 (10:47:57 p.m.). For subscriptions created on the start date, set the time to the current time and day in your time zone.
Follow these steps to create a subscription:

1. Create the request with the required API fields.
2. Send the request to one of these endpoints:
   Production: `POST https://api.cybersource.com/rbs/v1/subscriptions/follow-ons/{requestId}`
   Test: `POST https://apitest.cybersource.test.com/rbs/v1/subscriptions/follow-ons/{requestId}`
3. Verify the responses to make sure that the request was successful. A 200-level HTTP response code indicates success. See the *Transaction Response Codes*.

Required Fields

These fields are required for creating a subscription from an existing transaction:

subscriptionInformation.name

subscriptionInformation.startDate

This field is required for using an existing plan:

subscriptionInformation.planId

These fields are required for using a one-time plan:

orderInformation.amountDetails.billingAmount

orderInformation.amountDetails.currency

orderInformation.amountDetails.setupFee

planInformation.billingPeriod.length

planInformation.billingPeriod.unit

Related Information

- *API field reference guide for the REST API*

REST Example: Creating a Follow on Subscription from an Existing Transaction

This example shows how to create a follow on subscription from an existing transaction.

Request

```
{
  "subscriptionInformation": {
    "planId":"1619214515",
    "name": "SubNameFOOTPTesting",
    "startDate": "2023-04-15T17:01:42Z"
```

```
   }
}
```

Example Response to a Successful Request

```
{
   "_links": {
      "self": {
         "href": "/rbs/v1/subscriptions/1619214861",
         "method": "GET"
      },
      "update": {
         "href": "/rbs/v1/subscriptions/1619214861",
         "method": "PATCH"
      },
      "cancel": {
         "href": "/rbs/v1/subscriptions/1619214861/cancel",
         "method": "POST"
      }
   },
   "id": "1619214861",
   "status": "COMPLETED",
   "subscriptionInformation": {
      "code": "AWC-49",
      "status": "PENDING"
   }
}
```

Example Response to a Failed Request

```
{
   "status": "NOT_FOUND",
   "reason": "INVALID_DATA"
}
```

Retrieving Details for Follow on Subscription Creation Based on Existing Transaction

You can retrieve details from the specified transaction request ID that you can include to create a new subscription. You will be able to verify the payment instrument, billing, and shipping address information from an existing transaction before creating a new subscription.
Follow these steps to retrieve details on a subscription:

1. In the endpoint path, include the request ID of an existing transaction.
2. Send the request message to one of these endpoints:
   Production: `POST https://api.cybersource.com/rbs/v1/subscriptions/follow-ons/{requestId}`

Test: `POST https://apitest.cybersource.test.com/rbs/v1/subscriptions/follow-ons/{requestId}`

3. Verify the response messages to make sure that the request was successful. A 200-level HTTP response code indicates success. See the *Transaction Response Codes*.

REST Examples: Retrieving Details for Follow on Subscription Creation based on Existing Transaction

Successful Response

```
{
  "_links": {
    "self":
{
 "href": "rbs/v1/subscriptions/follow-ons/7216512479796378604957",
 "method": "GET"
 },
 "create":
{
 "href": "rbs/v1/subscriptions/follow-ons/7216512479796378604957",
 "method": "POST"
 }
  },
  "buyerInformation": {
    "merchantCustomerId": "1234",
    "email": ""
  },
  "paymentInstrument": {
    "card": {
      "number": "411111XXXXXX1111",
      "expirationMonth": "11",
      "expirationYear": "2037",
      "type": "001"
    },
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "123 Street",
      "locality": "Bellevue",
      "administrativeArea": "AL",
      "postalCode": "12345",
      "email": "test@visa.com",
      "country": "US"
    }
  },
  "shippingAddress": {
    "shipTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "123 Street",
      "locality": "Bellevue",
      "administrativeArea": "AL",
      "postalCode": "12345",
      "country": "US"
    }
  }
}
```

Error Response to a Failed Request

```
{
  "status": "NOT_FOUND",
  "reason": "INVALID_DATA"
}
```

Retrieving a List of Subscriptions

You can retrieve a list of subscriptions with these details for each subscription:

- Subscription ID
- Subscription code
- Subscription status
- Subscription name
- Customer ID
- Plan ID
- Plan code
- Plan name
- Plan description
- Plan status
- Billing period unit
- Billing period length
- Billing cycles total
- Billing cycles current
- Currency
- Billing amount
- Set-up fee

Use the subscription ID to retrieve, amend, activate, suspend, or cancel an individual subscription.
Follow these steps to retrieve a list of subscriptions:

1. Filter the list of subscriptions by these query string parameters:
    - `planName`: Name of the plan.
    - `customerId`: Customer token ID.
    - `status`: Subscription status (e.g., ACTIVE, CANCELLED).
    - `customerFirstName`: Customer's first name.
    - `customerLastName`: Customer's last name.
    - `code`: Specific subscription code.

- `plancode`: Specific plan code.
- `offset`: Page offset number.
- `limit`: Number of items to be returned (default is 20, maximum is 100).

2. Send the request message to one of these endpoints:
   Production: `GET https://api.cybersource.com/rbs/v1/subscriptions`
   Test: `GET https://apitest.cybersource.test.com/rbs/v1/subscriptions`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Examples: Retrieving a List of Subscriptions

Successful Response

```
{
  "_links": {
    "self": {
      "href": "/rbs/v1/subscriptions?status=ACTIVE&limit=2",
      "method": "GET"
    },
    "next": {
      "href": "/rbs/v1/subscriptions?status=ACTIVE&offset=2&limit=2",
      "method": "GET"
    }
  },
  "totalCount": 29,
  "subscriptions": [
    {
      "_links": {
        "self": {
          "href": "/rbs/v1/subscriptions/6192112872526176101960",
          "method": "GET"
        },
        "update": {
          "href": "/rbs/v1/subscriptions/6192112872526176101960",
          "method": "PATCH"
        },
        "cancel": {
          "href": "/rbs/v1/subscriptions/6192112872526176101960/cancel",
          "method": "POST"
        },
        "suspend": {
          "href": "/rbs/v1/subscriptions/6192112872526176101960/suspend",
          "method": "POST"
        }
      },
      "id": "6192112872526176101960",
      "planInformation": {
        "code": "34873819306413101960",
        "name": "RainTree Books Daily Plan",
        "billingPeriod": {
          "length": "1",
          "unit": "D"
        },
        "billingCycles": {
          "total": "2",
          "current": "1"
        }
      },
      "subscriptionInformation": {
        "code": "AWC-44",
        "planId": "6034873819306413101960",
        "name": "Test",
        "startDate": "2023-04-13T17:01:42Z",
        "status": "ACTIVE"
      },
      "paymentInformation": {
        "customer": {
          "id": "C09F227C54F94951E0533F36CF0A3D91"
        }
      },
      "orderInformation": {
        "amountDetails": {
          "currency": "USD",
          "billingAmount": "2.00",
          "setupFee": "1.00"
        },
        "billTo": {
          "firstName": "JENNY",
          "lastName": "AUTO"
        }
      }
    },
    {
      "_links": {
        "self": {
          "href": "/rbs/v1/subscriptions/6192115800926177701960",
          "method": "GET"
        },
        "update": {
          "href": "/rbs/v1/subscriptions/6192115800926177701960",
          "method": "PATCH"
        },
        "cancel": {
          "href": "/rbs/v1/subscriptions/6192115800926177701960/cancel",
          "method": "POST"
        },
        "suspend": {
          "href": "/rbs/v1/subscriptions/6192115800926177701960/suspend",
          "method": "POST"
        }
      },
      "id": "6192115800926177701960",
      "planInformation": {
        "code": "SITPlanCode6",
        "name": "Jan11DeployPlan1",
        "billingPeriod": {
          "length": "1",
          "unit": "W"
          },
          "billingCycles": {
            "total": "6",
            "current": "1"
          }
        },
        "subscriptionInformation": {
          "code": "AWC-45",
          "planId": "6104313186846711501956",
          "name": "Testsub1",
          "startDate": "2023-04-13T17:01:42Z",
          "status": "ACTIVE"
        },
        "paymentInformation": {
          "customer": {
            "id": "C09F227C54F94951E0533F36CF0A3D91"
          }
        },
        "orderInformation": {
          "amountDetails": {
            "currency": "USD",
            "billingAmount": "1.00",
            "setupFee": "5.00"
          },
          "billTo": {
            "firstName": "JENNY",
            "lastName": "AUTO"
          }
        }
      }
    ]
}
```

Error Response

```
{
  "status": "NOT_FOUND",
  "reason": "INVALID_DATA"
}
```

Retrieving a Subscription

You can retrieve the details of a specific subscription. The subscription details returned in the response include:

- Subscription ID
- Subscription code
- Subscription status
- Subscription name
- Customer ID
- Plan ID

- Plan code
- Plan name
- Plan description
- Plan status
- Billing period unit
- Billing period length
- Billing cycles total
- Billing cycles current
- Currency
- Billing amount
- Set-up fee

Follow these steps to retrieve a subscription:

1. In the endpoint path, include the subscription ID that you received from your list of subscriptions.
2. Send the request message to one of these endpoints:
   Production: `GET https://api.cybersource.com/rbs/v1/subscriptions/{id}`
   Test: `GET https://apitest.cybersource.test.com/rbs/v1/subscriptions/{id}`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Examples: Retrieving a Subscription

Response to a Successful Request

```
{
  "_links": {
    "self": {
      "href": "/rbs/v1/subscriptions/6192115800926177701960",
      "method": "GET"
    },
    "update": {
      "href": "/rbs/v1/subscriptions/6192115800926177701960",
      "method": "PATCH"
    },
    "cancel": {
      "href": "/rbs/v1/subscriptions/6192115800926177701960/cancel",
      "method": "POST"
    },
    "suspend": {
      "href": "/rbs/v1/subscriptions/6192115800926177701960/suspend",
      "method": "POST"
    }
  },
  "id": "6192115800926177701960",
  "planInformation": {
    "code": "SITPlanCode6",
    "name": "Jan11DeployPlan1",
    "billingPeriod": {
      "length": "1",
      "unit": "W"
    },
    "billingCycles": {
      "total": "6",
      "current": "1"
    }
  },
  "subscriptionInformation": {
    "code": "AWC-45",
    "planId": "6104313186846711501956",
    "name": "Testsub1",
    "startDate": "2023-04-13T17:01:42Z",
    "status": "ACTIVE"
  },
  "paymentInformation": {
    "customer": {
      "id": "C09F227C54F94951E0533F36CF0A3D91"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "currency": "USD",
      "billingAmount": "1.00",
      "setupFee": "5.00"
    },
    "billTo": {
      "firstName": "JENNY",
      "lastName": "AUTO"
    }
  }
}
```

Error Response

```
{
  "status": "NOT_FOUND",
  "reason": "INVALID_DATA",
  "details": []
}
```

Retrieving the Next Subscription Code

When you specify your subscription code, you can request the next consecutive alphabetical or numeric subscription code based on the last subscription code you specified. For example, if the last subscription code that you specified was `24B`, the system returns the code `24C`.

Follow these steps to retrieve the next subscription code:

1. Send the request to the recurring billing endpoint:
   Production: `GET https://api.cybersource.com/rbs/v1/subscriptions/code`

Test: `GET https://apitest.cybersource.test.com/rbs/v1/subscriptions/code`

2. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Examples: Retrieving the Next Subscription Code

Response to a Successful Request

```
{
   "code": "AWC-50",
}
```

Error Response

```
{
  "status": "NOT_FOUND",
  "reason": "INVALID_DATA"
}
```

Amending a Subscription

Subscriptions store customer details using a Token Management Service (TMS) token and have an assigned payment plan. Subscriptions consist of this information:

Subscription code	Use the **subscriptionInformation.code** REST API field to specify an amended value.
Subscription name	Use the **subscriptionInformation.name** REST API field to specify an amended value.
Start date	Use the **subscriptionInformation.startDate** REST API field to specify an amended value.
Token Management Service (TMS) token	You cannot change this information.
Plan: standard or one-time	A standard plan is created and stored within the recurring billing service for re-use. You can assign these plans to multiple subscriptions. A one-time plan is created specifically for a single subscription, and you create the plan when you create the subscription. A one-time plan does not include a plan code, plan name, or plan description.

Important

The subscription information that you can amend depends on the status of the subscription. Note the limitations described in in the paragraphs that follow.

Important

When you change the plan ID (`subscriptionInformation.planId` field) assigned to a subscription, the first payment is processed immediately. If proration is required after the change, it must be managed outside of the recurring billing service.

Basic Steps

Follow these steps to amend a subscription.

1. Create the request message with the API fields that contain new values.
2. Send the request message to one of these endpoints. In the endpoint path, replace the `{id}` portion of the URL with the subscription ID (**id**) that you received when you retrieved a list of subscriptions:
   - Production: `PATCH https://api.cybersource.com/rbs/v1/subscriptions/{id}`
   - Test: `PATCH https://apitest.cybersource.test.com/rbs/v1/subscriptions/{id}`
3. Verify the response messages to make sure that the request was successful. A 200-level HTTP response code indicates success. See the *Transaction Response Codes*.

Amendable Fields

The status of the determines which fields you can amend.

Subscriptions That Are Delinquent, Suspended, Canceled, or Completed

For a subscription in one of these states, you can modify only these fields:

subscriptionInformation.code

subscriptionInformation.name

Pending Subscriptions

You cannot modify these fields when the subscription is in the pending state:

orderInformation.amountDetails.currency

paymentInformation.customer.id

planInformation.billingPeriod.length

planInformation.billingPeriod.unit

Active Subscriptions

You cannot modify these fields when the subscription is in the active state:

orderInformation.amountDetails.currency

orderInformation.amountDetails.setupFee	
paymentInformation.customer.id	
planInformation.billingPeriod.length	
planInformation.billingPeriod.unit	
subscriptionInformation.startDate	**Payment processing time starts at 2:00 a.m. in your time zone.**

Related Information

- *API field reference guide for the REST API*

REST Example: Switching a Subscription to a Different Plan

This example shows you how to switche a subscription to a different plan.

Request

```
{
  "subscriptionInformation": {
    "planId": 7379850138646475304951,
    "name": "Update Sub Name - Switch Plan 6192115800926177701960",
    "code":"1619215852Code"
  },
  "orderInformation": {
    "amountDetails": {
      "billingAmount": "13.23"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "self": {
      "href": "/rbs/v1/subscriptions/1619215852",
      "method": "GET"
    },
    "update": {
      "href": "/rbs/v1/subscriptions/1619215852",
      "method": "PATCH"
    },
    "cancel": {
      "href": "/rbs/v1/subscriptions/1619215852/cancel",
      "method": "POST"
    }
  },
  "id": "1619215852",
  "status": "COMPLETED",
  "subscriptionInformation": {
    "code": "1619215852Code",
    "status": "PENDING"
```

```
   }
}
```

Error Response to a Failed Request

```
{
   "status": "INVALID_REQUEST",
   "reason": "INVALID_DATA",
   "message": "One or more fields in the request contains invalid data.",
   "details": [
      {
         "field": "subscriptionInformation.code",
         "reason": "DUPLICATE"
      }
   ]
}
```

Reactivating a Suspended Subscription

You can reactivate a suspended subscription for the next billing cycle. The payment processing time starts at 2:00 a.m. in your time zone.
You cannot reactivate a canceled or completed subscription.
Follow these steps to re-activate a subscription:

1. In the endpoint path, include the subscription ID that you received when you retrieved a list of subscriptions.
2. Send the request to the recurring billing endpoint:
   Production: `POST https://api.cybersource.com/rbs/v1/subscriptions/{id}/activate`
   Test: `POST https://apitest.cybersource.test.com/rbs/v1/subscriptions/{id}/activate`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Examples: Reactivating a Suspended Subscription

Response to a Successful Request

```
{
   "_links": {
      "self": {
         "href": "/rbs/v1/subscriptions/6149715492756032001956",
         "method": "GET"
      },
      "update": {
         "href": "/rbs/v1/subscriptions/6149715492756032001956",
         "method": "PATCH"
      },
      "cancel": {
         "href": "/rbs/v1/subscriptions/6149715492756032001956/cancel",
         "method": "POST"
      },
      "suspend": {
        "href": "/rbs/v1/subscriptions/6149715492756032001956/suspend",
        "method": "POST"
      }
    },
    "id": "6149715492756032001956",
    "status": "COMPLETED",
    "subscriptionInformation": {
      "code": "AWC-35",
      "status": "ACTIVE"
    }
}
```

Error Response

```
{
    "status": "INVALID_REQUEST",
    "reason": "INVALID_DATA",
    "message": "The subscription cannot be reactivated at this time.",
    "details": [
      {
        "field": "subscriptionInformation.status",
        "reason": "INVALID_FOR_ACTIVATION"
      }
    ]
}
```

Suspending a Subscription

You can suspend a pending, active, or delinquent subscription. Subscriptions cannot be suspended 10 minutes before or after a payment begins processing.

Follow these steps to suspend a subscription:

1. In the endpoint path, include the subscription ID that you received when you retrieved a list of subscriptions.
2. Send the request to the recurring billing endpoint:
   Production: `POST https://api.cybersource.com/rbs/v1/subscriptions/{id}/suspend`
   Test: `POST https://apitest.cybersource.test.com/rbs/v1/subscriptions/{id}/suspend`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Examples: Suspending a Subscription

Response to a Successful Request

```
{
    "_links": {
     "self": {
       "href": "/rbs/v1/subscriptions/6192112872526176101960",
       "method": "GET"
     },
     "update": {
       "href": "/rbs/v1/subscriptions/6192112872526176101960",
       "method": "PATCH"
     },
     "cancel": {
       "href": "/rbs/v1/subscriptions/6192112872526176101960/cancel",
       "method": "POST"
     },
     "suspend": {
       "href": "/rbs/v1/subscriptions/6192112872526176101960/suspend",
       "method": "POST"
     }
   },
   "id": "6192112872526176101960",
   "status": "ACCEPTED",
   "subscriptionInformation": {
     "code": "AWC-44",
     "status": "ACTIVE"
   }
 }
```

Error Response

```
{
  "status": "NOT_FOUND",
  "reason": "INVALID_DATA",
  "details": []
}
```

Canceling a Subscription

You can cancel a pending, active, suspended, or delinquent subscription. Subscriptions cannot be canceled within 10 minutes before or after a payment begins processing.

Follow these steps to cancel a subscription:

1. In the endpoint path, include the subscription ID that you received when you retrieved a list of subscriptions.
2. Send the request to the recurring billing endpoint:
   Production: `POST https://api.cybersource.com/rbs/v1/subscriptions/{id}/cancel`
   Test: `POST https://apitest.cybersource.test.com/rbs/v1/subscriptions/{id}/cancel`
3. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Examples: Canceling a Subscription

Response to a Successful Request

```
{
  "_links": {
    "self": {
      "href": "/rbs/v1/subscriptions/6192115800926177701960",
      "method": "GET"
    },
    "update": {
      "href": "/rbs/v1/subscriptions/6192115800926177701960",
      "method": "PATCH"
    },
    "cancel": {
      "href": "/rbs/v1/subscriptions/6192115800926177701960/cancel",
      "method": "POST"
    },
    "suspend": {
      "href": "/rbs/v1/subscriptions/6192115800926177701960/suspend",
      "method": "POST"
    }
  },
  "id": "6192115800926177701960",
  "status": "ACCEPTED",
  "subscriptionInformation": {
    "code": "6192177701960Code",
    "status": "ACTIVE"
  }
}
```

Error Response

```
{
  "status": "NOT_FOUND",
  "reason": "INVALID_DATA",
  "details": []
}
```

Additional Features

Recurring Billing includes these additional features.

System Retry Logic

Cybersource automatically retries failed recurring payments based on the type of decline received from the service. The service retries the internal and external payment declines.
If the Recurring Billing service encounters an internal processing error without sending the request out to the banking network, the service retries the payment until the error is resolved.
If the recurring billing service encounters an external processing error when the request is sent out to the banking network, Cybersource retries the payment before changing the subscription status to suspended.
If the issuer provides a reason code like "Do Not Retry", Cybersource stops all retry attempts. Cybersource immediately updates the subscription status to suspended.
The maximum number of retries is five times and is based on the billing frequency. During the retry period, Cybersource changes the subscription status to delinquent.
This example shows the system retry logic based on the billing frequency:

- Daily: retry 1 hour later, 1 time
- Monthly: retry every 2 days, 5 times
- Weekly: retry every 1 day, 3 times
- Yearly: retry every 15 days, 3 times

For a recurring payment that has a custom billing frequency, the Recurring Billing service retries a failed payment based on the billing frequency. As an example, suppose a payment fails for a recurring billing on a 14-day cycle. The recurring billing service uses the Daily retry logic and every 2 weeks uses the Weekly retry logic, even if the duration is the same.

Merchant-Initiated Transactions

For information about merchant-initiated transactions, see *Support for Merchant-Initiated Transactions and Credential-on-File for Visa, Mastercard, and Discover*.

Customer Notifications

The Recurring Billing service sends email notifications to customers using the email address stored on the customer token. The system sends notifications for three defined payment events:

- Prepayment notification: notification of an upcoming recurring payment.
- Successful payment notification: notification of a successful recurring payment.
- Failed payment notification: notification of recurring payment failure.

Cybersource sends email notifications from a Cybersource email address.

Example: Notification of Upcoming Subscription Payment

Hello,
Your recurring subscription will be charged to your payment card on file on `${paymentDate}`.
Subscription ID: `${subscriptionId}`
Subscription Name: `${subscriptionName}`
Billing Amount: `${billingAmount} ${currency}`
Set-up Fee: `${setupFee} ${currency}`
Thank you,
`${merchantName}`

Example: Notification of Successful Subscription Payment

Hello,
Your recurring subscription has been successfully charged to your payment card on file.
Subscription ID: `${subscriptionId}`
Subscription Name: `${subscriptionName}`
Billing Amount: `${billingAmount} ${currency}`
Set-up Fee: `${setupFee} ${currency}`
Transaction ID: `${transactionId}`
Transaction Date: `${paymentDate}`
Thank you,
`${merchantName}`

Example: Notification of Failed Subscription Payment

Hello,
Your recurring subscription has failed to charge to your payment card on file.

Subscription ID: `${subscriptionId}`
Subscription Name: `${subscriptionName}`
Billing Amount: `${billingAmount} ${currency}`
Setup Fee: `${setupFee} ${currency}`
Transaction ID: `${transactionId}`
Transaction Date: `${paymentDate}`
Thank you,
`${merchantName}`

Decision Manager Integration

A recurring payment is a credentials-on-file (COF) transaction in a series of payments that you bill to a customer at a fixed amount at regular intervals that do not exceed one year. Cybersource saves and stores payment credentials for recurring transactions, ensuring compliance with COF best practices.

Recurring transactions are considered low risk compared to unscheduled payments. Therefore, when the Decision Manager fraud detection system is enabled on your account, Cybersource does not submit recurring billing transactions to Decision Manager for fraud screening.

For more information about the Decision Manager, you can access the documentation by logging in to the Business Center.

Account Updater Integration

Account Updater is integrated with the Recurring Billing functionality so that your customer subscriptions can be kept current with credit card data changes. These changes can include a new expiration date, a new credit card number, or a brand change such as a change from Visa to Mastercard.

For more information relating to Account Updater, contact your Cybersource representative.

Related information

- Account Updater User Guide

Testing

Test your REST API integration by sending transactions to the Cybersource test server. See *Go Live*.

VISA Platform Connect: Specifications and Conditions for Resellers/ Partners

The following are specifications and conditions that apply to a Reseller/Partner enabling its merchants through Cybersource for Visa Platform Connect ("VPC") processing. Failure to meet any of the specifications and conditions below is subject to the liability provisions and indemnification obligations under Reseller/Partner's contract with Visa/Cybersource.

1. Before boarding merchants for payment processing on a VPC acquirer's connection, Reseller/Partner and the VPC acquirer must have a contract or other legal agreement that permits Reseller/Partner to enable its merchants to process payments with the acquirer through the dedicated VPC connection and/or traditional connection with such VPC acquirer.
2. Reseller/Partner is responsible for boarding and enabling its merchants in accordance with the terms of the contract or other legal agreement with the relevant VPC acquirer.
3. Reseller/Partner acknowledges and agrees that all considerations and fees associated with chargebacks, interchange downgrades, settlement issues, funding delays, and other processing related activities are strictly between Reseller and the relevant VPC acquirer.
4. Reseller/Partner acknowledges and agrees that the relevant VPC acquirer is responsible for payment processing issues, including but not limited to, transaction declines by network/issuer, decline rates, and interchange qualification, as may be agreed to or outlined in the contract or other legal agreement between Reseller/ Partner and such VPC acquirer.

DISCLAIMER: NEITHER VISA NOR CYBERSOURCE WILL BE RESPONSIBLE OR LIABLE FOR ANY ERRORS OR OMISSIONS BY THE Visa Platform Connect ACQUIRER IN PROCESSING TRANSACTIONS. NEITHER VISA NOR CYBERSOURCE WILL BE RESPONSIBLE OR LIABLE FOR RESELLER/PARTNER BOARDING MERCHANTS OR ENABLING MERCHANT PROCESSING IN VIOLATION OF THE TERMS AND CONDITIONS IMPOSED BY THE RELEVANT Visa Platform Connect ACQUIRER.