

# Recurring Billing

REST API



## 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](mailto: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>

## Copyright

© 2020. Cybersource Corporation. All rights reserved. Cybersource Corporation ("Cybersource") furnishes this document and the software described in this document under the applicable agreement between the reader of this document ("You") and Cybersource ("Agreement"). You may use this document and/or software only in accordance with the terms of the Agreement. Except as expressly set forth in the Agreement, the information contained in this document is subject to change without notice and therefore should not be interpreted in any way as a guarantee or warranty by Cybersource. Cybersource assumes no responsibility or liability for any errors that may appear in this document. The copyrighted software that accompanies this document is licensed to You for use only in strict accordance with the Agreement. You should read the Agreement carefully before using the software. Except as permitted by the Agreement, You may not reproduce any part of this document, store this document in a retrieval system, or transmit this document, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written consent of Cybersource.

## Restricted Rights Legends

**For Government or defense agencies:** Use, duplication, or disclosure by the Government or defense agencies is subject to restrictions as set forth in the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 and in similar clauses in the FAR and NASA FAR Supplement.

**For civilian agencies:** Use, reproduction, or disclosure is subject to restrictions set forth in subparagraphs (a) through (d) of the Commercial Computer Software Restricted Rights clause at 52.227-19 and the limitations set forth in Cybersource Corporation's standard commercial agreement for this software. Unpublished rights reserved under the copyright laws of the United States.

## Trademarks

Authorize.Net, eCheck.Net, and The Power of Payment are registered trademarks of Cybersource Corporation. Cybersource, Cybersource Payment Manager, Cybersource Risk Manager, Cybersource Decision Manager, and Cybersource Connect are trademarks and/or service marks of Cybersource Corporation. Visa, Visa International, Cybersource, the Visa logo, and the Cybersource logo are the registered trademarks of Visa International in the United States and other countries. All other trademarks, service marks, registered marks, or registered service marks are the property of their respective owners.

## Confidentiality Notice

This document is furnished to you solely in your capacity as a client of Cybersource and as a participant in the Visa payments system.

By accepting this document, you acknowledge that the information contained herein (the "Information") is confidential and subject to the confidentiality restrictions contained in Visa's operating regulations and/or other confidentiality agreements, which limit our use of the Information. You agree to keep the Information confidential and not to use the Information for any purpose other than its intended purpose and in your capacity as a customer of Cybersource or as a participant in the Visa payments system. The Information may only be disseminated within your organization on a need-to-know basis to enable your participation in the Visa payments system. Please be advised that the Information may constitute material non-public information under U.S. federal securities laws and that purchasing or selling securities of Visa Inc. while being aware of material non-public information would constitute a violation of applicable U.S. federal securities laws.

## Revision

Version: 24.06

# Contents

<b>Recurring Billing Developer Guide</b> .....	<b>5</b>
Recent Revisions to This Document.....	5
<b>Introduction to Recurring Billing</b> .....	<b>7</b>
Prerequisites.....	8
Account Validation (Pre-Authorization).....	8
Supported Processors.....	8
REST API.....	8
<b>Plans</b> .....	<b>9</b>
Plan Statuses.....	10
Assigning a Plan Code.....	11
Creating a Plan.....	11
Required Fields for Creating a Plan.....	12
Optional Fields for Creating a Plan.....	12
REST Examples: Creating a Plan.....	12
Retrieving a List of Plans.....	14
REST Examples: Retrieving a List of Plans.....	15
Retrieving a Plan.....	17
REST Examples: Retrieving a Plan.....	17
Retrieving a Plan Code.....	18
REST Examples: Retrieving a Plan Code.....	19
Amending a Plan.....	19
Optional Fields for Amending a Plan.....	20
REST Examples: Amending a Plan.....	20
Activating a Plan.....	21
REST Examples: Activating a Plan.....	22
Deactivating a Plan.....	23
REST Examples: Deactivating a Plan.....	23
Deleting a Plan.....	24
REST Examples: Deleting a Plan.....	24
<b>Subscriptions</b> .....	<b>26</b>
Subscription Prerequisites.....	26
Subscription Statuses.....	26

Zero Amount Authorizations.....	29
Assigning a Subscription Code.....	29
Subscription ID.....	29
Overriding a Plan.....	30
Creating a Subscription.....	30
Required Fields for Creating a Subscription.....	30
Optional Fields for Creating a Subscription.....	31
REST Examples: Creating a Subscription.....	31
Retrieving a List of Subscriptions.....	35
REST Examples: Retrieving a List of Subscriptions.....	36
Retrieving a Subscription.....	38
REST Examples: Retrieving a Subscription.....	39
Retrieving the Next Subscription Code.....	40
REST Examples: Retrieving the Next Subscription Code.....	41
Amending a Subscription.....	41
REST Examples: Amending a Subscription.....	42
Re-activating a Suspended Subscription.....	43
REST Examples: Reactivating a Suspended Subscription.....	44
Suspending a Subscription.....	45
REST Examples: Suspending a Subscription.....	45
Cancelling a Subscription.....	46
REST Examples: Canceling a Subscription.....	46
<b>Additional Features.....</b>	<b>48</b>
System Retry Logic.....	48
Merchant-Initiated Transactions.....	48
Customer Notifications.....	49
Decision Manager Integration.....	50
Account Updater Integration.....	50
<b>Testing.....</b>	<b>51</b>
<b>VISA Platform Connect: Specifications and Conditions for Resellers/Partners.....</b>	<b>52</b>

# 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

### 24.06

Added Information about system retry logic for recurring billings with payments on a custom frequency. See [System Retry Logic](#) on page 48.

## 24.05

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 7.

## 24.04

This revision contains only editorial changes and no technical updates.

## 24.03

Updated the descriptions of subscription statuses. See [Subscription Statuses](#) on page 26.

## 24.02

Updated the description of recurring billing transactions when Decision Manager is enabled on your account. See [Decision Manager Integration](#) on page 50.

## 24.01

This update contains only editorial changes and no technical updates.

# Introduction to Recurring Billing

The Recurring Billing service enables you to 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:

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

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.

## 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](#).

## Account Validation (Pre-Authorization)

When you capture payment card data for future use, Cybersource recommends that you validate the payment card account. Account validation ensures 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 authorizations, see the zero amount authorizations information in the [Credit Card Services Optional Features Supplement](#).

## Supported Processors

For more information, see the [Token Management Service REST API Developer Guide](#).

### Important

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

- Comercio Latino
- Prosa

## REST API

Register and obtain authentication credentials for the REST API.

See the Cybersource [Hello world sandbox](#) in the Developer Center.



# 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:

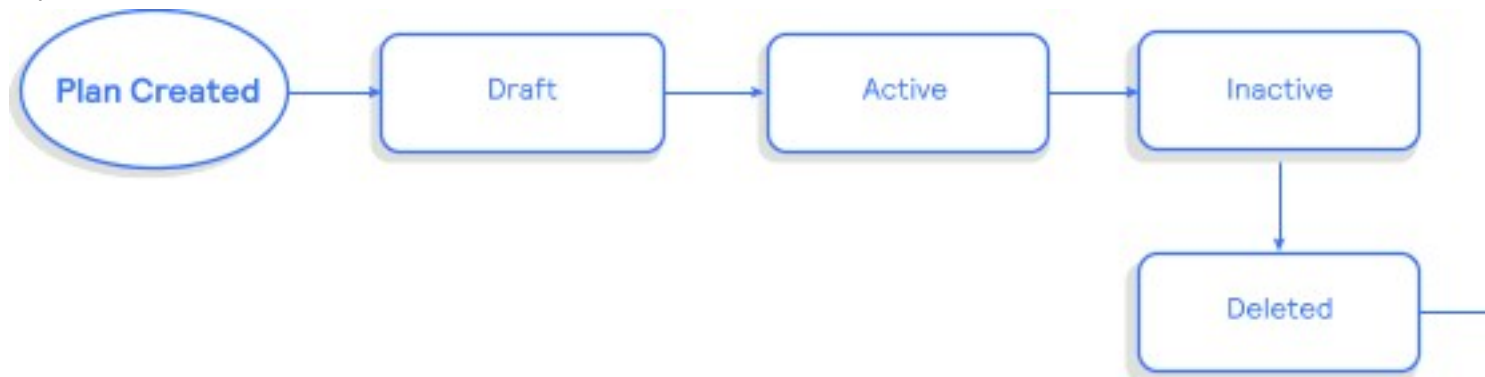
- Plan code
- Plan name
- Plan description (optional)
- Billing frequency
- Currency
- Payment amount
- Number of billing cycles
- Set-up fee (optional)

### 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 creating a new subscription or assigned to an existing subscription. Active plans can be changed to inactive or deleted.
- **Inactive:** subscriptions assigned to an inactive plan continue to be billed until the subscription is completed or cancelled. Inactive plans cannot be used to create new

subscriptions and subscriptions are not allowed to switch 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. A subscription's plan duration (number of billing cycles) cannot exceed 12 months. 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:

POST `https://<url_prefix>/rbs/v1/plans`

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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 for the REST API](#)

## Optional Fields for Creating a Plan

`orderInformation.amountDetails.setupFee`

`planInformation.code`

`planInformation.description`

`planInformation.status`

Related information

- [API Field Reference for the REST API](#)

## REST Examples: 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"
  }
}
}

```

#### Successful Response

```

{
  "_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 by 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. Format the endpoint as follows:
 

```
GET https://<url_prefix>/rbs/v1/plans
```

 Use one of these URL prefixes:
  - Test: `apitest.cybersource.com`
  - Production: `api.cybersource.com`
  - Production in India: `api.in.cybersource.com`
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 Plans

### Successful Response

```
{
  "_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. The plan details returned in the response include:

- 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:

```
GET https://<url_prefix>/rbs/v1/plans/{id}
```

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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 Plan

Successful Response

```
{
  "_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:

```
GET https://<url_prefix>/rbs/v1/plans/code
```

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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 a Plan Code

Successful Response

```
{
  "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 41.

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:

PATCH `https://<url_prefix>/rbs/v1/plans/{id}`

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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.subscriptionBillingOptDefaultType` is `NEW`.

### Related information

- [API Field Reference for the REST API](#)

## REST Examples: 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"
  }
}
}
}

```

#### Successful Response

```

{
  "_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:

```
POST https://<url_prefix>/rbs/v1/plans{id}/activate
```

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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: Activating a Plan

Successful Response

```
{
  "_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:

```
POST https://<url_prefix>/rbs/v1/plans{id}/deactivate
```

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
  - Production: `api.cybersource.com`
  - Production in India: `api.in.cybersource.com`
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: Deactivating a Plan

Successful Response

```

{
  "_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 or 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:

```
DELETE https://<url_prefix>/rbs/v1/plans/{id}
```

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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: Deleting a Plan

Success Response

```

{
  "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 using a Token Management Service (TMS) token and 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

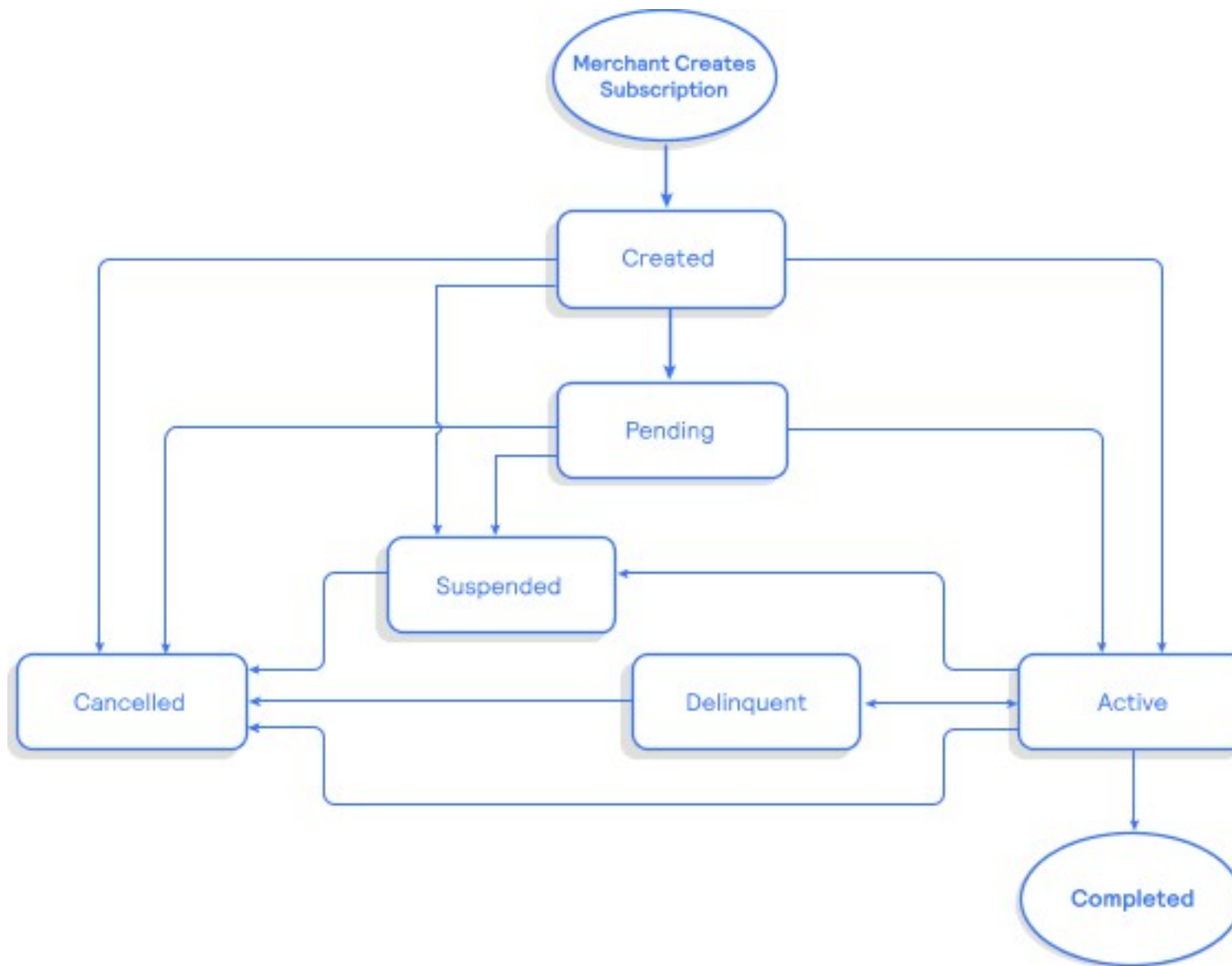
Subscriptions can be updated, activated, suspended, or cancelled.

## Subscription Prerequisites

Before creating a subscription, you must create a customer token. See the [Token Management Service REST API Developer Guide](#).

## Subscription Statuses

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



### Subscription Flow

A subscription has one of these statuses:

Created

The subscription has been created but the first payment has not been processed.

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 agreed upon with your customer.

Delinquent

A scheduled payment was declined.

## Suspended

The system automatically retries the payment for a number of times. If the payment succeeds, the system automatically moves the subscription back to active state. Otherwise, if the automated retry logic fails to obtain a successful payment, the system automatically moves the subscription to the suspended state.

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.



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.

## 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



A subscription's plan duration (number of billing cycles) cannot exceed 12 months. 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, or removal of the set-up fee.



## Important

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

# Creating a Subscription

**You can create a subscription with a standard plan, with plan overrides, or with a fully customized, 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 with a future start date, the payment processing time defaults to start at 2:00a.m. in your time zone. 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. Send the request to the recurring billing endpoint:

POST `https://<url_prefix>/rbs/v1/subscriptions`

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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 Subscription

A request for creating a subscription requires these fields:

`paymentInformation.customer.id`

`subscriptionInformation.name`

`subscriptionInformation.planId`

**Required when you are using a standard plan.**

**subscriptionInformation.startDate**

Related information

- [API Field Reference for the REST API](#)

**Optional Fields for Creating a Subscription**

These fields are optional for creating a subscription:

**clientReferenceInformation.code****clientReferenceInformation.partner.developerId****clientReferenceInformation.partner.solutionId****clientReferenceInformation.transactionId****orderInformation.amountDetails.billingAmount****orderInformation.amountDetails.currency****orderInformation.amountDetails.setupFee****planInformation.billingPeriod.billingCycles** Required for a plan with a defined plan period.**planInformation.billingPeriod.length****planInformation.billingPeriod.unit****planInformation.description****planInformation.name****planInformation.status****subscriptionInformation.code**

Related information

- [API Field Reference for the REST API](#)

**REST Examples: Creating a Subscription**

Request to Create a Subscription with a Pre-Existing Plan

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

```
}

```

#### Response from a Request to Create a Subscription with a Pre-Existing Plan

```
{
  "_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"
  }
}
```

#### Error Response to a Request to Create a Subscription with a Pre-Existing Plan

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

#### Request to Create a Subscription with Subscription Overrides

```
{
  "subscriptionInformation": {
    "planId": "1619214515",
    "name": "SubName With Overrides",
    "startDate": "2023-04-16T17:01:42Z"
  },
  "planInformation": {
    "billingCycles": {
      "total": "3"
    }
  },
  "orderInformation": {
    "amountDetails": {

```



```

    "billingAmount": "13.14",
    "setupFee": "1.27"
  }
},
"paymentInformation": {
  "customer": {
    "id": "C09F227C54F94951E0533F36CF0A3D91"
  }
}
}
}

```

#### Successful Response to a Request to Create a Subscription with Subscription Overrides

```

{
  "_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 Request to Create a Subscription with Subscription Overrides

```

{
  "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"
    }
  ]
}

```

#### Request to Create a Fully Customized Subscription with a One-Time Payment

```

{
  "subscriptionInformation": {

```

```

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

```

#### Response to a Successful Request to Create a Fully Customized Subscription with a One-Time Payment

```

{
  "_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"
  }
}

```

#### Error Response to a Request to Create a Fully Customized Subscription with a One-Time Payment

```

{
  "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"
  }
]
}

```

## 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:
  - **filters**: Use Lucene query syntax. Only keyword-matching and **AND** are supported. Example: `status:"ACTIVE" AND code:"01" AND id:"21345"`
  - **offset**: Page offset number.
  - **limit**: Number of items to be returned. Default is **20** and maximum is **100**.
2. Format the endpoint as follows:

`GET https://<url_prefix>/rbs/v1/subscriptions`

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
  - Production: `api.cybersource.com`
  - Production in India: `api.in.cybersource.com`
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 to the recurring billing endpoint:

```
GET https://<url_prefix>/rbs/v1/subscriptions{id}
```

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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:



GET `https://<url_prefix>/rbs/v1/subscriptions/code`

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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

### Important

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

For a pending subscription, you cannot change this information:

- Customer ID
- Currency
- Billing period unit and length

For an active subscription, you cannot change this information:

- Customer ID
- Currency
- Start date (Payment processing time starts at 2:00a.m. in your time zone.)
- Billing period unit and length
- Set-up fee

For a delinquent, suspended, cancelled, or completed subscription, you can update only the subscription name and code.



### Important

When you change the plan ID 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.

Follow these steps to amend a subscription:

1. Include any optional amendable fields in the request.
2. In the endpoint path, include the subscription ID that you received when you retrieved a list of subscriptions.
3. Send the request to the recurring billing endpoint:

**PATCH** `https://<url_prefix>/rbs/v1/subscriptions/{id}`

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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](#).

## REST Examples: Amending a Subscription

Request to Switch a Subscription to Different Plan

```
{
  "note": "Switch Plan Note",
  "reason": "Switch Plan Reason",
  "subscriptionInformation": {
    "name": "Update Sub Name - Switch Plan 6192115800926177701960",
    "code": "1619215852Code",
    "startDate": "2023-04-15T17:01:42Z"
  },
  "orderInformation": {
    "amountDetails": {
      "billingAmount": "13.23",
      "currency": "USD",
      "setupFee": "8.43"
    }
  }
},
```

```
"reasonCode": "Switch Plan Reason Code"
}
```

#### 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

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

## Re-activating a Suspended Subscription

**You can reactivate a suspended subscription for the next billing cycle. The payment processing time starts at 2:00a.m. in your time zone.**

You cannot reactivate a cancelled 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:

```
POST https://<url_prefix>/rbs/v1/subscriptions{id}/activate
```

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

## 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:

```
POST https://<url_prefix>/rbs/v1/subscriptions/{id}/suspend
```

Use one of these URL prefixes:

- Test: `apitest.cybersource.com`
- Production: `api.cybersource.com`
- Production in India: `api.in.cybersource.com`

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": []
}

```

## Cancelling 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:  
**POST** `https://<url_prefix>/rbs/v1/subscriptions/{id}/cancel`  
 Use one of these URL prefixes:
  - Test: `apitest.cybersource.com`
  - Production: `api.cybersource.com`
  - Production in India: `api.in.cybersource.com`
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. The maximum 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:



<https://support.visaacceptance.com/s/article/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 as 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.**