

Klarna Integration

Simple Order 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 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 Unpublished rights reserved under the copyright laws of the United States.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: 24.01

Contents

Klarna Integration	5
Recent Revisions to This Document.....	6
VISA Platform Connect: Specifications and Conditions for Resellers/Partners.....	7
Introduction to Klarna Integration	8
Merchant Account Types.....	9
Klarna Workflow Types.....	10
Klarna HOP Workflow.....	11
Klarna Inline Workflow.....	12
Installing the Klarna Widget and Inline Workflow.....	14
Calculating the Grand Total.....	16
Klarna Services	19
Creating a HOP Session.....	19
Line Items.....	21
Requesting to Create a HOP Session.....	21
XML Example: Creating a Session.....	24
Creating an Inline Session.....	25
Requesting to Create an Inline Session.....	27
XML Example: Creating a Session.....	29
Updating a Session.....	30
Required Fields for Updating a Session.....	32
Optional Fields for Updating a Session.....	33
XML Example: Updating a Session.....	34
Authorizations.....	35
Required Fields for an Authorization.....	36
XML Example: Authorization.....	37
Follow-on Authorizations.....	39
Required Fields for a Follow-on Authorization.....	39
XML Example: Follow-on Authorization.....	40
Captures.....	41
Required Fields for a Capture.....	42
Optional Fields for Captures.....	42
XML Example: Capture.....	43

Refunds.....	44
Required Fields for a Refund.....	44
Optional Fields for Refunds.....	45
XML Example: Refund.....	45
Authorization Reversals.....	46
Required Fields for Authorization Reversal.....	47
XML Example: Authorization Reversal.....	47
Check Status.....	48
Required Fields for Check Status.....	48
XML Example: Check Status.....	48
Reference Information.....	50
Reason Codes and Klarna Response Codes.....	50
Reporting.....	54
Simple Order API Field Map.....	55

Klarna Integration

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

Audience and Purpose

This guide is written for merchants who want to offer Klarna payments services to customers. It describes tasks that a merchant must complete in order to make a payment, request the status of a payment, or refund a payment. It is intended to help the merchant provide a seamless customer payment experience.

Convention

These statements appear in this document:



Important

An Important statement contains information essential to successfully completing a task or learning a concept.



Warning

A Warning contains information or instructions, which, if not heeded, can result in a security risk, irreversible loss of data, or significant cost in time or revenue or both.

Customer Support

For support information about any service, visit the Support Center:

<http://support.cybersource.com>

Recent Revisions to This Document

24.01

Updated the reason code descriptions. See [Reason Codes and Klarna Response Codes](#) on page 50.

23.03

Added link to the legacy Klarna Services Developer Guide. See [Introduction to Klarna Integration](#) on page 8.

23.02

Added a link to the previous version of the Klarna Developer Guide. See [Introduction to Klarna Integration](#) on page 8.

Added HOP and inline transaction workflows. See [Klarna HOP Workflow](#) on page 11 and [Klarna Inline Workflow](#) on page 12.

Updated widget install workflow. See [Installing the Klarna Widget and Inline Workflow](#) on page 14

23.01

The guide has undergone a major reorganization.

22.02

Updated **billTo_language** field description to include more countries and languages.

22.01

New feature: Follow-On Authorization allows updates to already submitted orders. Bill to, ship to, purchase total, itemization, and item break up can be updated before Capture or Reversal.

Added fields **merchantDefinedData5** to Authorization. See [Authorizations](#) on page 35.

Updated the Supported Countries and Currencies table. See [Supported Countries and Currencies](#) on page 9.

20.02

Indicated that the Klarna.Credit.load function enables the display of the Klarna widget in the workflow. The Klarna SDK documentation refers to the Klarna.Payments.load function, which is not valid for Cybersource integration. See [Installing the Klarna Widget and Inline Workflow](#) on page 14.

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.

Introduction to Klarna Integration

Klarna is a Buy Now Pay Later (BNPL) payment method that you can offer your customers through Cybersource. With Klarna, you can enable your customers to split their payments into multiple installments. You can display the Klarna payment method to your customers during checkout by either presenting a Klarna widget or redirecting your customers to a Klarna-hosted page.



Important

If you integrated with Klarna through Cybersource before September 2023, review this document and reintegrate to the latest fields and values. To reference the previous version of the Klarna integration, see the no-longer-supported [Klarna Services Developer Guide](#).

Supported Services

These are the API services you must integrate with to process a transaction using Klarna:

- Session
- Authorization
- Capture
- Refund
- Authorization-reversal
- Check status

Requirements

You must obtain a Cybersource merchant ID and a Klarna API key for each country in which you process transactions. Contact your Cybersource account manager for further details.

Supported Countries and Currencies

Contact your account manager for the latest supported countries and currencies information.

For information about the country codes, currency codes, and language codes, see the [ISO Standard Country Codes](#), the [ISO Standard Currency Codes](#) and the [ISO Standard Language Codes](#).

Klarna Review Process

Before you can launch Klarna payments, Klarna reviews your integrations. For more information about Klarna's pre-launch review process, contact your Cybersource account manager.

Shipping Policies

Always follow the shipping policies for each country as outlined by Klarna to ensure that Klarna assumes liability for fraudulent transactions. For Klarna's shipping policy, see:

<https://www.klarna.com/international/shipping-policies/>

Disputes and Fraud

Klarna has a standard process for handling risky transactions and disputes between you and your customers. For more information, contact your technical account manager or customer support.

Endpoints

Send your API requests to one of these Cybersource endpoints using version 1.187 or later:

- Production: <https://ics2ws.ic3.com/commerce/1.x/transactionProcessor>
- Test: <https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor>

Use the Klarna test triggers when directly connected to Klarna, which consist of email addresses. For more information, see:

<https://docs.klarna.com/resources/test-environment/>

Merchant Account Types

There are two types of Cybersource merchant accounts, Cybersource settlement services account and processor direct contract account.

Cybersource Settlement Services Account

This merchant account has no direct contract with a payment provider partner. The Cybersource Financial Settlement Partner (FSP) collects funds on your behalf and settles them to your merchant account.



Important

Cybersource requests the export compliance service for every transaction using the Cybersource settlement services account. The export compliance service compares customer information to export control lists maintained by government agencies. If a customer's name appears on any government list, the transaction is declined.

To facilitate compliance checks for Cybersource settlement services accounts, you must include these fields in your authorization service requests.

You must include these fields in the live environment to avoid receiving error messages.

Mandatory Authorization Fields for a Cybersource Settlement Services Account

billTo_city

billTo_country

billTo_firstName

billTo_lastName

billTo_street1

Processor Direct Contract Account

This merchant account must use the payment provider selected by Cybersource. If you have existing direct contracts, you must inform your sales representative.

Klarna Workflow Types

There are two methods for processing a payment that are determined by how customer accesses their Klarna account to complete the checkout:

Hosted Order Page (HOP)

The customer is redirected to a Klarna-hosted page to complete the checkout. For overview information about how to process a payment using the HOP method, see [Klarna HOP Workflow](#) on page 11.

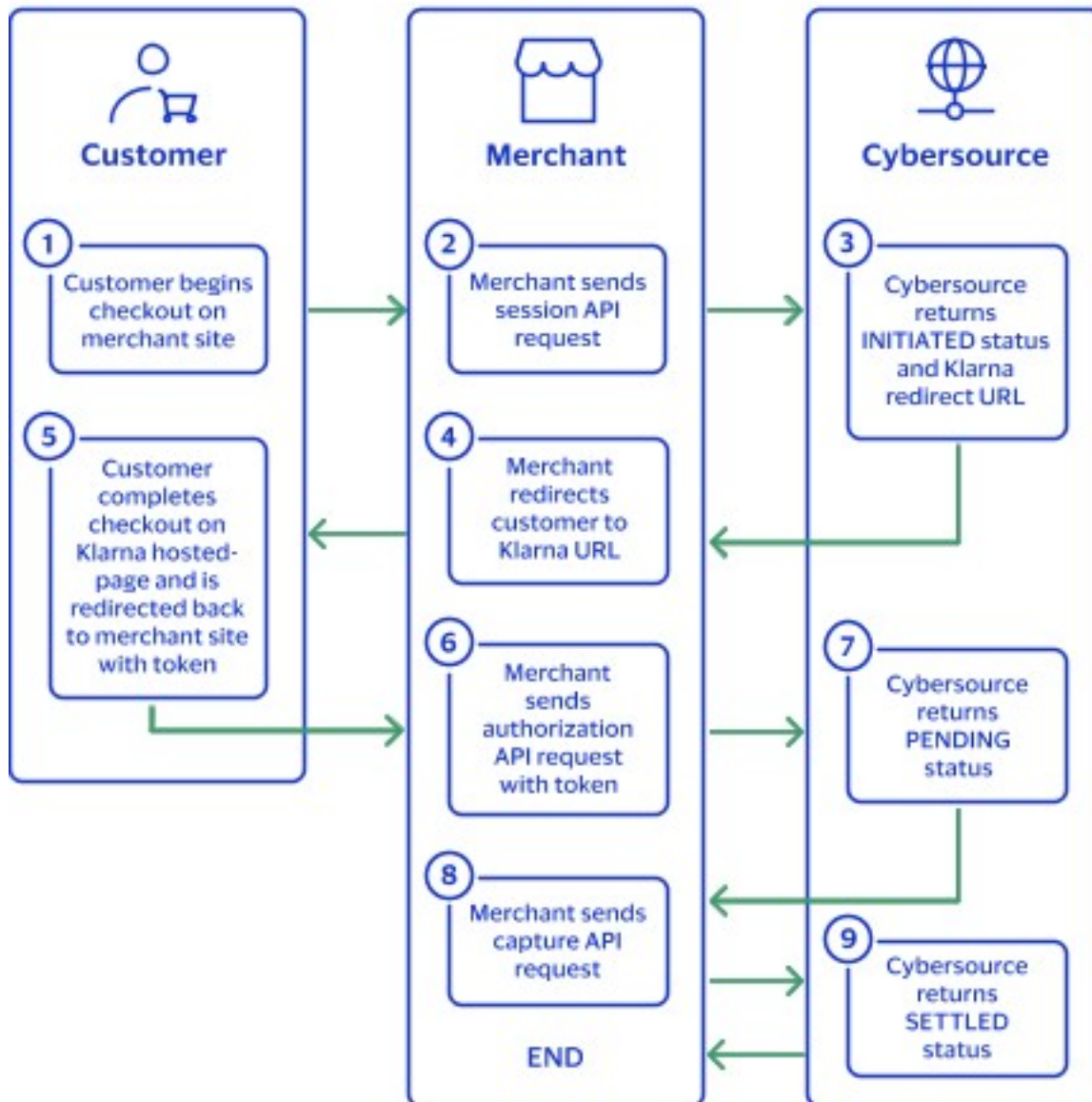
Inline

The customer uses the Klarna widget that is installed into the merchant website to complete the checkout. For overview information about how to process a payment using the inline method, see [Klarna Inline Workflow](#) on page 12.

For information about how to install the Klarna widget into the merchant website, see [Installing the Klarna Widget and Inline Workflow](#) on page 14.

Klarna HOP Workflow

This workflow describes the sequence of events that comprise a successful transaction using a Klarna hosted-page.



Klarna HOP Workflow

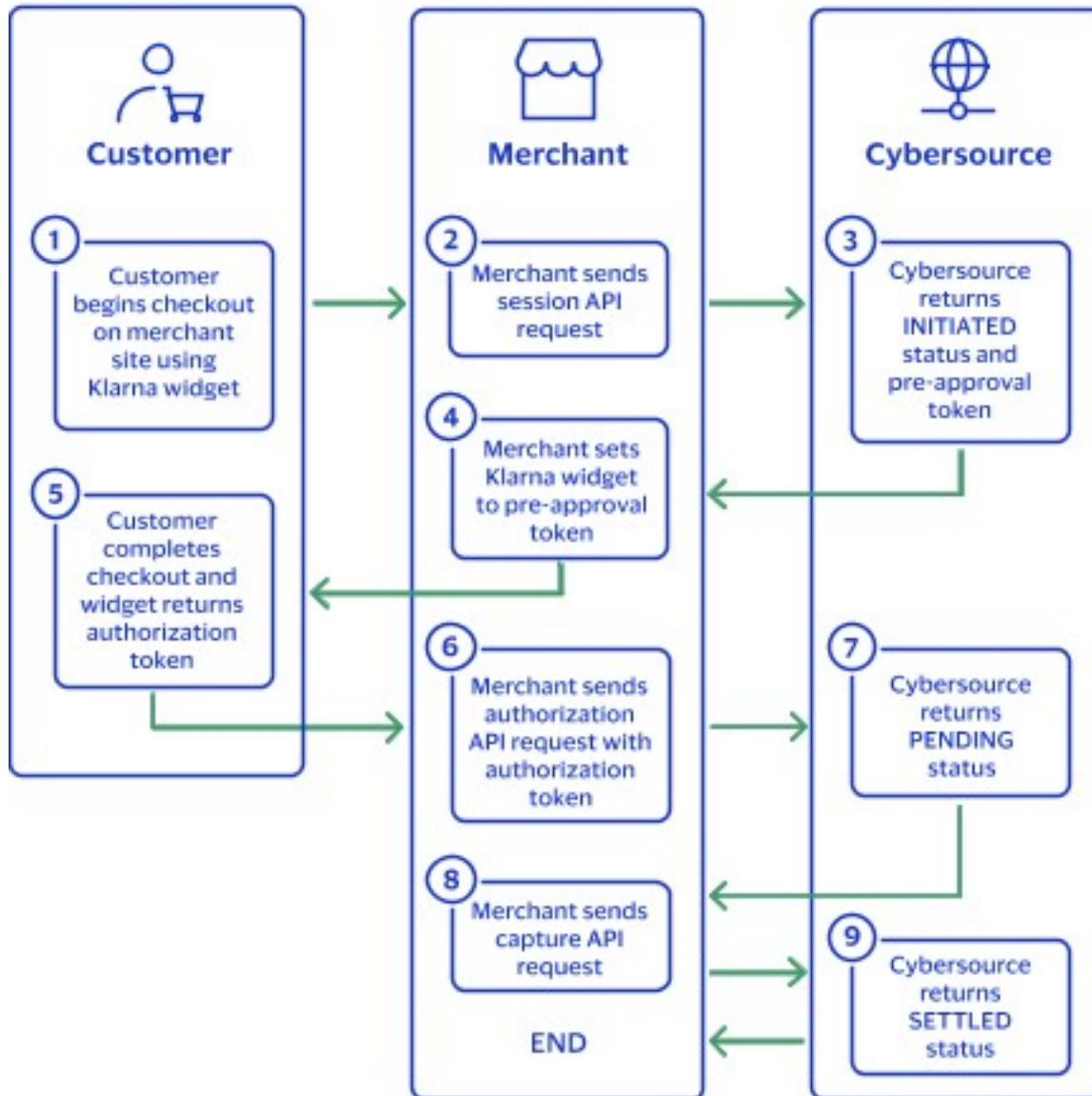
1. The customer begins to checkout on the merchant website and selects the Klarna payment option.
2. The merchant sends a session API request. For more information, see [Requesting to Create a HOP Session](#) on page 21 in [Creating a HOP Session](#) on page 19.

- a. If the customer changes their item selection after the merchant sends the session API request, the merchant must send an update session API request. For more information, see [Updating a Session](#) on page 30.
3. Cybersource sends the merchant with an **INITIATED** status and a Klarna redirect URL.
4. The merchant redirects the customer to the Klarna URL.
5. The customer completes checkout on the Klarna hosted-page and is redirected back to the merchant website. Klarna adds a unique token to the merchant redirect URL.
6. The merchant sends an authorization API request and includes the unique token Klarna added to the merchant URL. See [Authorizations](#) on page 35.
7. Cybersource returns a **PENDING** status.
8. The merchant sends a capture API request. For more information, see [Captures](#) on page 41.
9. Cybersource returns a **SETTLED** status.

Klarna Inline Workflow

This workflow describes the sequence of events that comprise a successful transaction using a Klarna widget.

For information about how to install the Klarna widget, see [Installing the Klarna Widget and Inline Workflow](#) on page 14.



Klarna Inline Workflow

1. The customer begins to checkout on the merchant website and uses the Klarna widget displayed on the merchant checkout page.
2. The merchant sends a session API request. For more information, see [Requesting to Create an Inline Session](#) on page 27 in [Creating an Inline Session](#) on page 25.
 - a. If the customer changes their item selection after the merchant sends the session API request, the merchant must send an update session API request. For more information, see [Updating a Session](#) on page 30.
3. Cybersource sends the merchant an **INITIATED** status and a pre-approval token.
4. The merchant sets the Session Token ID in the Klarna widget to the pre-approval token.

5. The customer completes checkout using the Klarna widget and the Klarna widget returns a unique authorization token.
6. The merchant sends an authorization API request and includes the unique authorization token the Klarna widget displayed. See [Authorizations](#) on page 35.
7. Cybersource returns a **PENDING** status.
8. The merchant sends a capture API request. For more information, see [Captures](#) on page 41.
9. Cybersource returns a **SETTLED** status.

Installing the Klarna Widget and Inline Workflow

This workflow describes how to install the Klarna widget and complete a successful transaction. For a description of a successful transaction with the Klarna widget already installed, see [Klarna Inline Workflow](#) on page 12.

1. Add the container for the Klarna widget to the HTML for your checkout page. This is a one-time operation and provides an iframe into which the Klarna widget will be dynamically loaded when the Klarna widget is initialized.

```
<div id=#klarna_container"></div>
```

2. When the customer displays your checkout page, send a create session request to Cybersource. The sessions service creates a unique customer session and returns a pre-approval token. See [Creating a HOP Session](#) on page 19 and [Updating a Session](#) on page 30.
3. Present the available payment methods to the customer. When the customer chooses the Klarna payment method on your checkout page, install the Klarna software developer kit (SDK) and initialize it by calling `Klarna.Payments.init`. In the request, set the client token field to the value of the pre-approval token returned by Cybersource. Initializing the Klarna SDK can take up to 10 seconds. Cybersource recommends that you try to initialize the SDK every three seconds, up to a maximum of three attempts. For additional information about initializing the Klarna SDK, see:

<https://docs.klarna.com/klarna-payments/in-depth-knowledge/klarna-payments-sdk-reference/>

```
Klarna.Payments.init({client_token: '<%=processorToken%>'})
  if (count < 3)
  {
    setTimeout(initializeKlarna.bind(null, count), 3000);
  }
  else
  {
    showError()
  }
```

4. Load the Klarna widget into the Klarna container by calling `Klarna.Payments.load` and specifying the Klarna container.

```
Klarna.Payments.load({
  container: "#klarna_container",
  (...)
})
```

5. Display the Klarna payment options on your checkout page. The `show_form = true` statement dynamically updates the payment options in the Klarna widget.

```
if (res["show_form"] == true)
{
  logging("Klarna Available Payment Option");

  document.getElementById("auth_button").innerHTML =
  "<br><button type=\"button\" name=\"buy\"
  onclick=\"authorizeKlarnaOrder();\">Pay</button>"
}
else
{
  logging("Klarna Not Available As A Payment Option");
}
```

6. When the customer chooses one of the Klarna payment options:
 - Send an update session request to Cybersource with available customer information. See [Creating a HOP Session](#) on page 19 and [Updating a Session](#) on page 30.
 - Call `Klarna.Payments.authorize` to authorize the order with Klarna. In the call, include an empty JSON object. For additional information about Klarna authorizations, see:

<https://docs.klarna.com/klarna-payments/api-call-descriptions/authorize-the-purchase/>

```
Klarna.Payments.authorize({}, function(res) {
  var auth_token = res["authorization_token"];
  var isApproved = res["approved"];
  var show_form = res["show_form"];
})
```

7. Klarna validates the customer's information and determines whether to authorize the order. When Klarna authorizes the order, Klarna returns an authorization token.
8. Send an authorization request to Cybersource. Set the pre-approval token field to the value of the authorization token returned by Klarna. See [Authorizations](#) on page 35.
9. When Cybersource approves the authorization, send a capture request to complete the purchase. See [Captures](#) on page 41.
When the authorization response indicates that the purchase is pending, send a check status request every hour until the payment status changes. See [Check Status](#) on page 48.

Calculating the Grand Total

Most Klarna services require that the grand total amount of a purchase be included in the service request in the **purchaseTotals_grandTotalAmount** field. The country of the transaction and the use of coupons affect how to calculate the grand total amount.

US Grand Total with Coupons

To calculate the grand total amount for US transactions with coupons, use this formula:

sum of (unit price x quantity) for all items + item-level tax amount – sum of (coupon amount x quantity) for all items – item-level discount amount

This is the same formula with the respective API fields:

sum of (item_#_unitPrice x item_#_quantity) for all items + purchaseTotals_taxAmount – sum of coupons (item_#_unitPrice x item_#_quantity) – purchaseTotals_discountAmount

US Grand Total Amount with Coupons

This example shows the proper syntax and calculation of a grand total amount within an API request.

```
<item id="0">
  <unitPrice>100</unitPrice>
  <quantity>1</quantity>
  <totalAmount>100</totalAmount>
</item>
<item id="1">
  <unitPrice>75</unitPrice>
  <quantity>2</quantity>
  <totalAmount>150</totalAmount>
</item>
<item id="2">
  <unitPrice>30</unitPrice>
  <quantity>1</quantity>
  <productCode>coupon<productCode>
  <productName>first-time customer<productName>
  <productSKU>12345</productSKU>
</item>
<purchaseTotals>
  <taxAmount>40</taxAmount>
  <discountAmount>10</discountAmount>
  <grandTotalAmount>250</grandTotalAmount>
</purchaseTotals>
```

US Grand Total without Coupons

To calculate the grand total amount for US transactions without coupons, use this formula:

sum of (unit price x quantity) for all items + item-level tax amount – item-level discount amount

This is the same formula with the respective API fields:

sum of (item_#_unitPrice x item_#_quantity) for all items + purchaseTotals_taxAmount – purchaseTotals_discountAmount

Grand Total Amount

This example shows the proper syntax and calculation of a grand total amount within an API request.

```
<item id="0">
  <unitPrice>100</unitPrice>
  <quantity>1</quantity>
  <totalAmount>100</totalAmount>
</item>
<item id="1">
  <unitPrice>75</unitPrice>
  <quantity>2</quantity>
  <totalAmount>150</totalAmount>
</item>
<purchaseTotals>
  <discountAmount>10</discountAmount>
  <taxAmount>40</taxAmount>
  <grandTotalAmount>280</grandTotalAmount>
</purchaseTotals>
```

Non-US Countries Grand Total with Coupons

To calculate the grand total amount for non-US transactions with coupons, use this formula:

sum of (unit price x quantity) for all items + sum of (item-level tax amount) for all items – sum of (coupon amount x quantity) for all items – item-level discount amount

This is the same formula with the respective API fields:

sum of (item_#_unitPrice x item_#_quantity) for all items + sum of (item_#_taxAmount) for all items – sum of coupons (item_#_unitPrice x item_#_quantity) for all times – purchaseTotals_discountAmount

Grand Total Amount

This example shows the proper syntax and calculation of a grand total amount within an API request.

```
<item id="0">
  <unitPrice>100</unitPrice>
  <quantity>1</quantity>
  <taxAmount>20</taxAmount>
  <totalAmount>120</totalAmount>
</item>
<item id="1">
  <unitPrice>75</unitPrice>
  <quantity>2</quantity>
  <taxAmount>20</taxAmount>
  <totalAmount>170</totalAmount>
</item>
<item id="2">
  <unitPrice>30</unitPrice>
```

```

<quantity>1</quantity>
<productCode>coupon</productCode>
<productName>first-time customer</productName>
<productSKU>12345</productSKU>
<purchaseTotals>
  <discountAmount>10</discountAmount>
  <grandTotalAmount>290</grandTotalAmount>
</purchaseTotals>

```

Non-US Countries Grand Total without Coupons

To calculate the grand total amount for non-US transactions without coupons, use this formula:

sum of (unit price x quantity) for all items + sum of (item-level tax amount) for all items – item-level discount amount

This is the same formula with the respective API fields:

sum of (item_#_unitPrice x item_#_quantity) for all items + sum of (item_#_taxAmount) for all items – purchaseTotals_discountAmount

Grand Total Amount

This example shows the proper syntax and calculation of a grand total amount within an API request.

```

<item id="0">
  <unitPrice>100</unitPrice>
  <quantity>1</quantity>
  <totalAmount>120</totalAmount>
</item>
<item id="1">
  <unitPrice>75</unitPrice>
  <quantity>2</quantity>
  <taxAmount>20</taxAmount>
  <totalAmount>170</totalAmount>
</item>
<purchaseTotals>
  <discountAmount>10</discountAmount>
  <grandTotalAmount>280</grandTotalAmount>
</purchaseTotals>

```

Klarna Services

These are the available Klarna API services:

- [Creating a HOP Session](#) on page 19
- [Creating an Inline Session](#) on page 25
- [Updating a Session](#) on page 30
- [Authorizations](#) on page 35
- [Follow-on Authorizations](#) on page 39
- [Captures](#) on page 41
- [Refunds](#) on page 44
- [Authorization Reversals](#) on page 46
- [Check Status](#) on page 48

Creating a HOP Session

Create a new session whenever the customer displays your checkout page. The session service is requested either by using the HOP session method or inline session method.

HOP Specifications

When you send a session request, set the **apSessionsService_paymentFlowMode** field to **HOP**. Cybersource responds with a Klarna redirect URL in the **apSessionsReply_merchantURL** field. Send the customer to the redirect URL to complete the checkout. When the customer completes the checkout, Klarna redirects the customer back to the merchant website and includes a token in the merchant URL. The token generated by Klarna can then be used in the authorization request to link it to the session. See [Requesting to Create a HOP Session](#) on page 21.

Klarna Payment Methods

When you request the session service, you must determine the Klarna payment method you want to offer your customers. Set the **paymentMethod_name** field to a value listed in the Payment Method column.

Payment Methods

Payment Method	Description	Klarna Products
pay_later	Customer pays total bill at a set date.	Klarna Invoice Klarna Pay Later Klarna Pay Later by Card (PLBC)
pay_now	Customer pays total bill at the time of checkout.	Klarna Direct Bank Transfer Klarna Direct Debit
pay_over_time	Customer pays bill in equal multiple installments.	Klarna Account Klarna Financing Klarna Fixed Installment Plans Klarna Interest Free

Example: Payment Methods in Session Request

```
<paymentMethod_name>pay_later</paymentMethod_name>
```

Endpoints

Set the **apSessionsService_run** field to **true**, and send the request to one of these endpoints:

Production: <https://ics2ws.ic3.com/commerce/1.x/transactionProcessor>

Test: <https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor>

Response Status

The session service responds with one of these statuses as the **apSessionReply_status** field value:

- **COMPLETED**: The customer completed the checkout process.
- **INITIATED**: The session is initiated and the customer may now checkout.

The session service also responds with a reason code as the `apSessionReply_reasonCode` field value. For more information about reason codes, see the [Reason Codes for the Simple Order API](#).

Line Items

Klarna uses line items when you send a create session and update session requests. *Line items* are used to include information about the goods that your customers purchase, such as product name, quantity, and price.

Line items are represented as the `item_#_` fields, starting with `item_0_`, and increasing in numerical order.

These fields are required for each line item that you use:

`item_#_productCode`

`item_#_productDescription`

`item_#_productName`

`item_#_quantity`

`item_#_unitPrice`

Including Line Items in a Service Request

This example shows three valid line items.

```
<item id="0">
  <productCode>123456</productCode>
  <productDescription>Red</productDescription>
  <productName>Shirt</productName>
  <quantity>1</quantity>
  <unitPrice>30.00</unitPrice>
</item>
<item id="1">
  <productCode>456789</productCode>
  <productDescription>Green</productDescription>
  <productName>Pants</productName>
  <quantity>3</quantity>
  <unitPrice>19.99</unitPrice>
</item>
<item id="2">
  <productCode>987654</productCode>
  <productDescription>Blue</productDescription>
  <productName>Dress</productName>
  <quantity>2</quantity>
  <unitPrice>25.50</unitPrice>
</item>
```

Requesting to Create a HOP Session

Follow these steps to successfully create a HOP session.

1. Send a **POST** request to the <https://ics2ws.ic3.com/commerce/1.x/transactionProcessor> endpoint and include these required fields:

**Important**

Do not send any personally identifiable information (PII) data about the customer in the request to create a session.

apPaymentType	Set to KLI .
apSessionService_paymentFlowMode	Set to HOP to display the Klarna widget on your checkout page.
apSessionsService_cancelURL	Set to the URL to which the customer is redirected after cancelling the Klarna payment.
apSessionsService_failureURL	Set to the URL to which the customer is redirected after the Klarna payment fails.
apSessionsService_paymentMethod_name	Set to one of these values to display the Klarna product you are offering: <ul style="list-style-type: none"> • pay_later: pay later • pay_now: pay now • pay_over_time: pay over time
apSessionsService_run	Set to true .
apSessionsService_sessionsType	Set to N .
apSessionsService_successURL	Set to the URL to which the customer is redirected after successfully completing the Klarna payment. Include ?auth-token={{authorization_token}} to the end of the URL. Example: <pre>https://www.merchant.com?auth-token={{authorization_token}}</pre>
billTo_country	
item_#_productName	Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.
item_#_quantity	Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_totalAmount

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_unitPrice

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

merchantID**merchantReferenceCode****purchaseTotals_currency****purchaseTotals_grandTotalAmount**

Non-US Countries

US

item_#_taxAmount**billTo_state****purchaseTotals_taxAmount**

2. You can include this optional field in the request:

purchaseTotals_discountAmount

3. To include an optional coupon in the request, include these fields:

item_#_productCode

Set to coupon.

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_productName

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_productSKU

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_quantity

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_unitPrice

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

4. Redirect the customer to the returned URL in the **merchantURL** field.

```
<merchantURL>https://pay.playground.klarna.com/na/hpp/payments/1vmXvue</merchantURL>
```

5. When the customer completes the checkout using their Klarna credentials, Klarna redirects the customer to the URL you specified in the **apSessionsService_successURL** field. This URL contains a unique token.

```
https://www.test.com?auth-token=d9b4496b-f4b1-5ad5-bbe0-7f666682126
```

6. When you send the authorization request, set the **apAuthService_preapprovalToken** field to the unique token Klarna returned in the success URL.

```
<preapprovalToken>d9b4496b-f4b1-5ad5-bbe0-7f666682126</preapprovalToken>
```

XML Example: Creating a Session

This example shows a successful create session request.

HOP Request

```
<requestMessage xmlns="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantID>test_merchant</merchantID>
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <billTo>
    <firstName>John</firstName>
    <lastName>Smith</lastName>
    <street1>123 Happy St</street1>
    <city>Sunnyville</city>
    <state>VA</state>
    <postalCode>12345</postalCode>
    <country>US</country>
    <email>test@cybs.com</email>
  </billTo>
  <item>
    <unitPrice>150.00</unitPrice>
    <quantity>1</quantity>
    <productCode>A4890B5023</productCode>
    <productName>Skirt on the sky</productName>
    <productSKU>skirtonsky$bluegreen</productSKU>
    <taxAmount>4.50</taxAmount>
    <totalAmount>154.50</totalAmount>
    <productDescription>Amnesiac Shirt</productDescription>
  </item>
  <purchaseTotals>
    <currency>USD</currency>
    <grandTotalAmount>154.50</grandTotalAmount>
  </purchaseTotals>
  <apPaymentType>kli</apPaymentType>
  <apSessionsService run="true">
    <cancelURL>https://www.merchant.com</cancelURL>
    <successURL>https://www.merchant.com?auth-token={{authorization_token}}</successURL>
    <failureURL>https://www.merchant.com</failureURL>
    <sessionsType>N</sessionsType>
    <paymentMethod_name>pay_now</paymentMethod_name>
    <paymentFlowMode>HOP</paymentFlowMode>
  </apSessionsService>
</requestMessage>
```


Payment Methods

Payment Method	Description	Klarna Products
pay_later	Customer pays total bill at a set date.	Klarna Invoice Klarna Pay Later Klarna Pay Later by Card (PLBC)
pay_now	Customer pays total bill at the time of checkout.	Klarna Direct Bank Transfer Klarna Direct Debit
pay_over_time	Customer pays bill in equal multiple installments.	Klarna Account Klarna Financing Klarna Fixed Installment Plans Klarna Interest Free

Example: Payment Methods in Session Request

```
<paymentMethod_name>pay_later</paymentMethod_name>
```

Endpoints

Set the **apSessionsService_run** field to **true**, and send the request to one of these endpoints:

Production: <https://ics2ws.ic3.com/commerce/1.x/transactionProcessor>

Test: <https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor>

Response Status

The session service responds with one of these statuses as the **apSessionReply_status** field value:

- **COMPLETED**: The customer completed the checkout process.
- **INITIATED**: The session is initiated and the customer may now checkout.

The session service also responds with a reason code as the **apSessionReply_reasonCode** field value. For more information about reason codes, see the [Reason Codes for the Simple Order API](#).

Requesting to Create an Inline Session

Follow these steps to successfully create an inline session.

1. Send a **POST** request to the `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor` endpoint and include these required fields:



Important

Do not send any personally identifiable information (PII) data about the customer in the request to create a session.

apPaymentType	Set to KLI .
apSessionService_paymentFlowMode	Set to inline to display the Klarna widget on your checkout page.
apSessionsService_cancelURL	Set to the URL to which the customer is redirected after cancelling the Klarna payment.
apSessionsService_failureURL	Set to the URL to which the customer is redirected after the Klarna payment fails.
apSessionsService_paymentMethod_name	Set to one of these values to display the Klarna product you are offering: <ul style="list-style-type: none"> • pay_later: pay later • pay_now: pay now • pay_over_time: pay over time
apSessionsService_run	Set to true .
apSessionsService_sessionsType	Set to N .
apSessionsService_successURL	Set to the URL to which the customer is redirected after successfully completing the Klarna payment. Add /Klarna/Reply.jsp to the end of the URL.
	<code>https://www.test.com/Klarna/Reply.jsp</code>
billTo_country	
item_#_productName	Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.
item_#_quantity	Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_totalAmount

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_unitPrice

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

merchantID**merchantReferenceCode****purchaseTotals_currency****purchaseTotals_grandTotalAmount**

Non-US Countries

item_#_taxAmount

US

billTo_state**purchaseTotals_taxAmount**

2. You can include this optional field in the request:

purchaseTotals_discountAmount

3. To include an optional coupon in the request, include these fields:

item_#_productCode

Set to coupon.

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_productName

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_productSKU

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_quantity

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_unitPrice

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

4. Set the Session Token ID in the Klarna widget to the value in the **processorToken** returned in the Session response.

```
IDeyJhbGciOiJSUzI1NiIsImtpZCI6IjgyMzA1ZWJjLWI4MTEtMzYzNy...
```

5. When the customer completes the payment using the Klarna widget, the Klarna widget returns a unique authorization token. Store the token for the authorization request.

```
Authorizaition Token: a6b2229c-665b-5660-a5fe-063bef3d9a1e
```

6. When you begin to authorize the payment, set the **apAuthService_preapprovalToken** field in the authorization request to the authorization token returned in the Klarna widget.

```
<preapprovalToken>a6b2229c-665b-5660-a5fe-063bef3d9a1e</preapprovalToken>
```

XML Example: Creating a Session

This example shows a successful create session request.

Inline Request

```
<requestMessage xmlns="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantID>test_merchant</merchantID>
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <billTo>
    <firstName>John</firstName>
    <lastName>Smith</lastName>
    <street1>123 Happy St</street1>
    <city>Sunnyville</city>
    <state>VA</state>
    <postalCode>12345</postalCode>
    <country>US</country>
    <email>john@email.com</email>
  </billTo>
  <item>
    <unitPrice>9.00</unitPrice>
    <quantity>1</quantity>
    <productCode>A4890B5023</productCode>
    <productName>Skirt on the sky</productName>
    <productSKU>skirtonsky$bluegreen</productSKU>
    <taxAmount>1.00</taxAmount>
    <totalAmount>10.00</totalAmount>
    <productDescription>Amnesiac Shirt</productDescription>
  </item>
  <purchaseTotals>
    <currency>USD</currency>
    <grandTotalAmount>10.00</grandTotalAmount>
  </purchaseTotals>
  <apPaymentType>KLI</apPaymentType>
  <apSessionsService run="true">
    <cancelURL>https://www.merchant.com/Klarna/Reply.jsp</cancelURL>
    <successURL>https://www.merchant.com/Klarna/Reply.jsp</successURL>
    <failureURL>https://www.merchant.com/Klarna/Reply.jsp</failureURL>
    <sessionsType>N</sessionsType>
    <paymentMethod_name>pay_now</paymentMethod_name>
    <paymentFlowMode>inline</paymentFlowMode>
  </apSessionsService>
</requestMessage>
```


Payment Methods

Payment Method	Description	Klarna Products
pay_later	Customer pays total bill at a set date.	Klarna Invoice Klarna Pay Later Klarna Pay Later by Card (PLBC)
pay_now	Customer pays total bill at the time of checkout.	Klarna Direct Bank Transfer Klarna Direct Debit
pay_over_time	Customer pays bill in equal multiple installments.	Klarna Account Klarna Financing Klarna Fixed Installment Plans Klarna Interest Free

Example: Payment Methods in Session Request

```
<paymentMethod_name>pay_later</paymentMethod_name>
```

Endpoints

Set the **apSessionsService_run** field to **true**, and send the request to one of these endpoints:

Production: <https://ics2ws.ic3.com/commerce/1.x/transactionProcessor>

Test: <https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor>

Response Status

The session service responds with one of these statuses as the **apSessionReply_status** field value:

- **COMPLETED**: The customer completed the checkout process.
- **INITIATED**: The session is initiated and the customer may now checkout.

The session service also responds with a reason code as the **apSessionReply_reasonCode** field value. For more information about reason codes, see the [Reason Codes for the Simple Order API](#).

Required Fields for Updating a Session

Include these required fields to update a session.

apPaymentType	Set to KLI .
apSessionsService_cancelURL	Set to the URL the customer is directed to after cancelling the Klarna payment.
apSessionsService_failureURL	Set to the URL the customer is directed to after the Klarna payment fails.
apSessionsService_requestID	
apSessionsService_run	Set to true .
apSessionsService_sessionsType	Set to U .
apSessionsService_successURL	Set to the URL the customer is directed to after successfully completing the Klarna payment.
billTo_country	
item_#_productName	Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.
item_#_quantity	Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.
item_#_totalAmount	Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.
item_#_unitPrice	Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.
merchantID	
merchantReferenceCode	
purchaseTotals_currency	
purchaseTotals_grandTotalAmount	

Country-Specific Fields

Include the country-specific field(s) in addition to the above required fields.

Non-US Countries	item_#_taxAmount
US	billTo_state
	purchaseTotals_taxAmount

Coupon Fields

To include a coupon in your request, include these fields.

item_#_productCode	Set to <code>coupon</code> . Replace the <code>#</code> character with the number <code>0</code> for the first item and consecutive numbers for any additional items.
item_#_productName	Replace the <code>#</code> character with the number <code>0</code> for the first item and consecutive numbers for any additional items.
item_#_productSKU	Replace the <code>#</code> character with the number <code>0</code> for the first item and consecutive numbers for any additional items.
item_#_quantity	Replace the <code>#</code> character with the number <code>0</code> for the first item and consecutive numbers for any additional items.
item_#_unitPrice	Replace the <code>#</code> character with the number <code>0</code> for the first item and consecutive numbers for any additional items.

Optional Fields for Updating a Session

These are the optional fields you can include to update an existing session.

billTo_city	
billTo_district	Set to the same value as the <code>billTo_state</code> field if the field is present.
billTo_email	
billTo_firstName	
billTo_language	For possible values, see the country and language code column in the Supported Countries and Currencies on page 9 table.
billTo_lastName	
billTo_postalCode	
billTo_state	Set to the same value as the <code>billTo_district</code> field.
billTo_street1	
billTo_street2	
purchaseTotals_discountAmount	
shipTo_city	

[shipTo_country](#)[shipTo_district](#)[shipTo_email](#)[shipTo_firstName](#)[shipTo_lastName](#)[shipTo_postalCode](#)[shipTo_state](#)[shipTo_street1](#)[shipTo_street2](#)

XML Example: Updating a Session

This example includes optional fields.

Request

```

<requestMessage
  xmlns="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantID>test_merchant</merchantID>
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <billTo>
    <firstName>Anna</firstName>
    <lastName>Schmidt</lastName>
    <street1>Leopoldstrasse 4</street1>
    <street2>Apt 2</street2>
    <city>Lichtenberg</city>
    <district>Berlin</district>
    <state>Berlin</state>
    <postalCode>10318</postalCode>
    <country>DE</country>
    <phoneNumber>5551234567</phoneNumber>
    <email>test@cybs.com</email>
    <dateOfBirth>19820101</dateOfBirth>
    <language>DE-DE</language>
  </billTo>
  <shipTo>
    <firstName>Anna</firstName>
    <lastName>Schmidt</lastName>
    <street1>Leopoldstrasse 4</street1>
    <street2>Apt 2</street2>
    <city>Lichtenberg</city>
    <state>Berlin</state>
    <district>Berlin</district>
    <postalCode>10318</postalCode>
    <country>DE</country>
    <phoneNumber>5551234567</phoneNumber>
    <email>test@cybs.com</email>
  </shipTo>
  <item id="0">
    <unitPrice>19.99</unitPrice>

```

```

<quantity>1</quantity>
<productName>Green Widget</productName>
<taxAmount>1.00</taxAmount>
<totalAmount>20.99</totalAmount>
</item>
<item id="1">
<unitPrice>10.00</unitPrice>
<quantity>2</quantity>
<productName>Blue Widget</productName>
<taxAmount>2.00</taxAmount>
<totalAmount>22.00</totalAmount>
</item>
<item id="2">
<unitPrice>5.00</unitPrice>
<quantity>1</quantity>
<productName>shipping</productName>
<totalAmount>5.00</totalAmount>
</item>
<purchaseTotals>
<currency>EUR</currency>
<discountAmount>3.00</discountAmount>
<grandTotalAmount>44.99</grandTotalAmount>
</purchaseTotals>
<apPaymentType>KLI</apPaymentType>
<apSessionsService run="true">
<sessionsType>U</sessionsType>
<sessionsRequestID>4848446567036715804007</sessionsRequestID>
</apSessionsService>
</requestMessage>

```

Response

```

<replyMessage xmlns:c="urn:schemas-cybersource-com:transaction-data-1.213">
<merchantReferenceCode>refnum-1234</merchantReferenceCode>
<requestID>4848446567036715804007</requestID>
<decision>ACCEPT</decision>
<reasonCode>100</reasonCode>
<purchaseTotals>
<currency>EUR</currency>
</purchaseTotals>
<apSessionsReply>
<reasonCode>100</reasonCode>
<responseCode>000000</responseCode>
</apSessionsReply>
</replyMessage>

```

Authorizations

The authorization service responds with a Klarna URL to which you direct the customer after the transaction is complete. The Klarna URL is returned in the **apAuthReply_merchantURL** field. You can capture an authorization for up to 28 days after a payment is authorized.

Merchant-Defined Data

Merchant-defined data is included in the **merchantDefinedData_mddField_5** field.



Important

This field has replaced the **merchantDefinedData_field5** field. If you submit a request with the **merchantDefinedData_field5** field, Cybersource automatically replaces the field with **merchantDefinedData_mddField_5**.



Warning

You are prohibited from including personally identifying information (PII) in the **merchantDefinedData** fields for the purposes of capturing, obtaining, and/or transmitting any PII. PII includes, but is not limited to, address, credit card number, social security number, driver's license, state-issued identification number, passport number, and card verification number (CVV, CVC2, CVV2, CID, CVN). In the event Cybersource discovers you are capturing and/or transmitting PII in the **merchantDefinedData** fields, whether or not intentionally, Cybersource immediately suspends your merchant account, which results in a rejection of any and all transaction requests submitted by the merchant account after the point of suspension.

Endpoints

Set the **apAuthService_run** field to **true**, and send the request to one of these endpoints:

Production: <https://ics2ws.ic3.com/commerce/1.x/transactionProcessor>

Test: <https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor>

Response Status

The authorization service responds with one of these statuses as the **apAuthReply_status** field value:

- **FAILED**: The authorization request failed.
- **PENDING**: The authorization request is accepted but is not authorized. Request the check status service to retrieve status updates. For more information, see [Check Status](#).

The authorization service also responds with a reason code as the **apAuthReply_reasonCode** field value. For more information about reason codes, see the [Reason Codes for the Simple Order API](#).

Required Fields for an Authorization

Include these required fields to process an authorization.

apAuthService_preapprovalToken

Set to the token generated by Klarna.

apAuthService_run

Set to **true**.

apPaymentType	Set to KL .
billTo_city	
billTo_country	Set to the same value used in the sessions request.
billTo_email	
billTo_firstName	
billTo_lastName	
billTo_street1	
merchantDefinedData_mddField_5	Set to the same value used in the create session request. Possible values: <ul style="list-style-type: none"> HOP: display the Klarna widget on your checkout page. inline: redirect the customer to the Klarna-hosted page.
merchantID	
merchantReferenceCode	
purchaseTotals_currency	
purchaseTotals_grandTotalAmount	Set to the same value as the purchaseTotals_grandTotalAmount field in the latest sessions update request.

Country-Specific Fields

Include the country-specific field(s) in addition to the above required fields.

Non-US Countries	item_#_taxAmount
US	billTo_state purchaseTotals_taxAmount

XML Example: Authorization

This example shows a successful authorization.

Authorization Request

```
<requestMessage xmlns="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantID>test_merchant</merchantID>
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <billTo>
    <firstName>John</firstName>
    <lastName>Smith</lastName>
    <street1>Happy St</street1>
    <street2>123</street2>
    <city>Austin</city>
```

```

<district>TX</district>
<state>TX</state>
<postalCode>78757</postalCode>
<country>US</country>
<email>test@cybs.com</email>
</billTo>
<purchaseTotals>
<currency>USD</currency>
<grandTotalAmount>154.50</grandTotalAmount>
</purchaseTotals>
<merchantDefinedData>
<field5>inline</field5>
</merchantDefinedData>
<apPaymentType>KLI</apPaymentType>
<apAuthService run="true">
<preapprovalToken>d9b4496b-f4b1-5ad5-bbe0-7f6666821260</preapprovalToken>
</apAuthService>
</requestMessage>

```

Authorization Response

```

<replyMessage xmlns:c="urn:schemas-cybersource-com:transaction-data-1.213">
<merchantReferenceCode>refnum-1234</merchantReferenceCode>
<requestID>6957570615206883603012</requestID>
<decision>ACCEPT</decision>
<reasonCode>100</reasonCode>
<requestToken>AxjnrwSTeSuwS73eT7JE/6IZYjWmdqbZrNGMpoujmjMMSTSB3i0ZiwyasZejFlof+k3krsEu93k
+yRAABS1G</requestToken>
<purchaseTotals>
<currency>USD</currency>
</purchaseTotals>
<exportReply>
<reasonCode>100</reasonCode>
<ipCountryConfidence>-1</ipCountryConfidence>
</exportReply>
<apReply>
<productID>paylaterbycard</productID>
</apReply>
<apAuthReply>
<reasonCode>100</reasonCode>
<status>AUTHORIZED</status>
<processorResponse>00003</processorResponse>
<amount>154.50</amount>
<dateTime>2023-09-26T19:37:43Z</dateTime>
<paymentStatus>authorized</paymentStatus>
<responseCode>00003</responseCode>
<reconciliationID>XFZ3ZMYV41J7</reconciliationID>
<processorTransactionID>a3367116-69e7-4ddf-b1fe-645413b6c63d</processorTransactionID>
</apAuthReply>
</replyMessage>

```

Follow-on Authorizations

Already submitted authorizations can be updated as follow-on authorizations before the capture or reversal. Follow-on authorizations allow you to update the billing, shipping, itemization, and item break up information of already submitted orders. Request a new authorization with the **linkToRequest** field to link the follow-on authorization to the initial authorization. Set the **linkToRequest** field to the **requestID** value from the initial authorization response.

Endpoints

Set the **apAuthService_run** field to `true`, and send the request to one of these endpoints:

Production: `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`

Test: `https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor`

Response Status

The authorization service responds with one of these statuses as the **apAuthReply_status** field value:

- **AUTHORIZED**: The payment is successfully authorized.
- **FAILED**: The authorization request failed.
- **PENDING**: The authorization request is accepted but is not authorized. Request the check status service to retrieve status updates. For more information, see [Check Status](#).

The authorization service also responds with a reason code as the **apAuthReply_reasonCode** field value. For more information about reason codes, see the [Reason Codes for the Simple Order API](#).

Required Fields for a Follow-on Authorization

Include these required fields to process a follow-on authorization.

apAuthService_preapprovalToken	Set to the token generated by Klarna.
apAuthService_run	Set to <code>true</code> .
apPaymentType	Set to <code>KLI</code> .
billTo_city	
billTo_country	Set to the same value used in the sessions request.
billTo_email	
billTo_firstName	
billTo_lastName	
billTo_street1	

linkToRequest

Set to the requestID from the initial authorization response.

merchantDefinedData_mddField_5

Set to the same value used in the create session request. Possible values:

- **HOP**: display the Klarna widget on your checkout page
- **inline**: redirect the customer to the Klarna-hosted page

merchantID**merchantReferenceCode****purchaseTotals_currency****purchaseTotals_grandTotalAmount**

Set to the same value as the **purchaseTotals_grandTotalAmount** field in the latest sessions update request.

Country-Specific Fields

Include the country-specific field in addition to the required fields for a session.

Non-US Countries**item_#_taxAmount****US****billTo_state****purchaseTotals_taxAmount**

XML Example: Follow-on Authorization

This example shows a successful follow-on authorization request.

Follow-on Authorization Request

```
<requestMessage xmlns="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantID>test_merchant</merchantID>
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <billTo>
    <firstName>Jane</firstName>
    <lastName>Smith</lastName>
    <street1>Happy St</street1>
    <street2>456</street2>
    <city>Austin</city>
    <district>TX</district>
    <state>TX</state>
    <postalCode>78797</postalCode>
    <country>US</country>
    <email>test@cybs.com</email>
  </billTo>
  <purchaseTotals>
    <currency>USD</currency>
    <grandTotalAmount>10.00</grandTotalAmount>
  </purchaseTotals>
```

```

<merchantDefinedData>
  <field5>inline</field5>
</merchantDefinedData>
<linkToRequest>6958684668456844304007</linkToRequest>
<apPaymentType>KLI</apPaymentType>
<apAuthService run="true">
  <preapprovalToken>9cb0b304-c901-5a58-afd5-06c87d7bb217</preapprovalToken>
</apAuthService>
</requestMessage>

```

Follow-on Authorization Response

```

<replyMessage xmlns:c="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <requestID>6958688084846249104011</requestID>
  <decision>ACCEPT</decision>
  <reasonCode>100</reasonCode>
  <requestToken>AxjnrwSTeTsyWHVD+dKL/6IZYjWmjCbZrSIzSwojmjst4nSBX0GY0MmkmXoxZaH/
  pN5OzJYdUP50osAA8wHb</requestToken>
  <purchaseTotals>
    <currency>USD</currency>
  </purchaseTotals>
  <exportReply>
    <reasonCode>100</reasonCode>
    <ipCountryConfidence>-1</ipCountryConfidence>
  </exportReply>
  <apReply>
    <productID>paylaterbycard</productID>
  </apReply>
  <apAuthReply>
    <reasonCode>100</reasonCode>
    <status>AUTHORIZED</status>
    <processorResponse>000003</processorResponse>
    <amount>10.00</amount>
    <dateTime>2023-09-28T02:40:10Z</dateTime>
    <paymentStatus>authorized</paymentStatus>
    <responseCode>000003</responseCode>
    <merchantURL>https://js.playground.klarna.com/na/kp/v1/sessions/f2bebdbb-b46c-551e-
    a8a4-048a4a185c10/redirect</merchantURL>
    <reconciliationID>XFZ40MYVHF4K</reconciliationID>
    <processorTransactionID>fc2b177e-f70a-4ff6-8847-3ab7b8db6d9c</processorTransactionID>
  </apAuthReply>
</replyMessage>

```

Captures

The capture service enables you to capture the entire authorized amount or part of the authorized amount. Klarna supports multiple capture requests when the total amount of all captures is less than the authorized amount.

You can capture an authorization for up to 28 days after a payment is authorized.

Endpoints

Set the **apCaptureService_run** field to `true`, and send the request to one of these endpoints:

Production: <https://ics2ws.ic3.com/commerce/1.x/transactionProcessor>

Test: <https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor>

Response Status

The capture service responds with one of these statuses as the **apCaptureReply_status** response field value:

- **FAILED**: The capture request failed.
- **PENDING**: The capture request is accepted but is not captured. Request the check status service to retrieve status updates. For more information, see [Check Status](#).
- **SETTLED**: The capture request is settled for the requested amount.

The capture service also responds with a reason code as the **apCaptureReply_reasonCode** field value. For more information on reason codes, see the [Reason Codes for the Simple Order API](#).

Required Fields for a Capture

Include these required fields to capture an authorized payment.

apCaptureService_authRequestID	Set to the request ID included in the authorization response.
apCaptureService_run	Set to <code>true</code> .
apPaymentType	Set to <code>KLI</code> .
merchantID	
merchantReferenceCode	
purchaseTotals_currency	
purchaseTotals_grandTotalAmount	

Optional Fields for Captures

Choose from these optional fields to include additional information when capturing a payment.

item_#_productName	Replace the # character with the number <code>0</code> for the first item and consecutive numbers for any additional items.
item_#_quantity	Replace the # character with the number <code>0</code> for the first item and consecutive numbers for any additional items.

item_#_totalAmount

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_unitPrice

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

XML Example: Capture

This example shows a successful capture request.

Request

```
<requestMessage xmlns="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantID>test_merchant</merchantID>
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <purchaseTotals>
    <currency>USD</currency>
    <grandTotalAmount>154.50</grandTotalAmount>
  </purchaseTotals>
  <apPaymentType>KLI</apPaymentType>
  <apCaptureService run="true">
    <authRequestID>6957570615206883603012</authRequestID>
  </apCaptureService>
</requestMessage>
```

Response

```
<replyMessage xmlns:c="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <requestID>6957576170986768803011</requestID>
  <decision>ACCEPT</decision>
  <reasonCode>100</reasonCode>
  <requestToken>AxjnrwSTeSvECLGs0aTD/6IZYjWmrWbZoV7NKCojmjMTOvSB3i0ZiwyasZejF1of+k3krsEu93k+yRAAAy04</requestToken>
  <purchaseTotals>
    <currency>USD</currency>
  </purchaseTotals>
  <apCaptureReply>
    <reasonCode>100</reasonCode>
    <status>SETTLED</status>
    <processorResponse>00004</processorResponse>
    <amount>154.50</amount>
    <dateTime>2023-09-26T19:46:58Z</dateTime>
    <reconciliationID>XFZ55MYPWYRA</reconciliationID>
    <paymentStatus>settled</paymentStatus>
    <responseCode>00004</responseCode>
  </apCaptureReply>
</replyMessage>
```

Refunds

The refund service enables you to refund the entire captured amount or part of the captured amount. Klarna supports multiple refund requests when the total amount of all refunds is less than the captured amount.'

Cybersource recommends using the optional fields for a refund to provide the best service for the customer. For more information, see [Optional Fields for Refunds](#) on page 45.

Endpoints

Set the **apRefundService_run** field to `true`, and send the request to one of these endpoints:

Production: `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`

Test: `https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor`

Status Responses

The refund service responds with one of these statuses as the **apRefundReply_status** field value:

- **FAILED**: The refund request failed.
- **PENDING**: The refund request is accepted but is not refunded. Request the check status service to retrieve status updates. For more information, see [Check Status](#).
- **REFUNDED**: The captured payment is successfully refunded.

The refund service also responds with a reason code as the **apRefundReply_reasonCode** field value. For more information about reason codes, see the [Reason Codes for the Simple Order API](#).

Required Fields for a Refund

Include these required fields to

apPaymentType

Set to `KLI`.

apRefundService_refundRequestID

Set to the request ID included in the capture response.

apRefundService_run

Set to `true`.

merchantID

merchantReferenceCode

purchaseTotals_currency

purchaseTotals_grandTotalAmount

Optional Fields for Refunds

Choose from these optional fields to include additional information when refunding a payment.

billTo_email

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

billTo_firstName

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

billTo_lastName

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_productName

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_quantity

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_totalAmount

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

item_#_unitPrice

Replace the # character with the number 0 for the first item and consecutive numbers for any additional items.

XML Example: Refund

This example shows a successful refund.

Request

```
<requestMessage xmlns="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantID>test_merchant</merchantID>
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <purchaseTotals>
    <currency>USD</currency>
    <grandTotalAmount>154.50</grandTotalAmount>
  </purchaseTotals>
  <apPaymentType>KLI</apPaymentType>
  <apRefundService run="true">
    <refundRequestID>6957576170986768803011</refundRequestID>
  </apRefundService>
</requestMessage>
```

Response

```

<replyMessage xmlns:c="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <requestID>6957582387066840103011</requestID>
  <decision>ACCEPT</decision>
  <reasonCode>100</reasonCode>
  <requestToken>AxjnrwSTeSvaHi+ZXThj/6IZYjWmrWbZoV7NKCojnmjMcX7SB3i0ZiwyaSZeJFlof+k3krsEu93k+yRAABS1G</requestToken>
  <purchaseTotals>
    <currency>USD</currency>
  </purchaseTotals>
  <apRefundReply>
    <reasonCode>100</reasonCode>
    <transactionID>b30d7de1-215d-4989-8b82-f190d39457a4</transactionID>
    <status>REFUNDED</status>
    <processorResponse>000006</processorResponse>
    <amount>154.50</amount>
    <dateTime>2023-09-26T19:57:19Z</dateTime>
    <reconciliationID>XFZ55MYPWYRA</reconciliationID>
    <returnRef>XFZ40MYVGPWJ</returnRef>
    <paymentStatus>refunded</paymentStatus>
    <responseCode>000006</responseCode>
  </apRefundReply>
</replyMessage>

```

Authorization Reversals

The authorization reversal service enables you to reverse the amount that was authorized. To reverse an authorization, you must have the authorization request ID, which is the **requestID** field value from the authorization response. To reverse a follow-on authorization, use the request ID from the follow-on authorization response.

Endpoints

Set the **apAuthReversalService_run** field to **true**, and send the request to one of these endpoints:

Production: <https://ics2ws.ic3.com/commerce/1.x/transactionProcessor>

Test: <https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor>

Response Status

The authorization reversal service responds with one of these statuses as the **apAuthReversalReply_status** field value:

- **AUTH_REVERSED**: The authorization is successfully reversed.
- **FAILED**: The authorization reversal failed.

The authorization reversal service also responds with a reason code as the **apAuthReversalreply_reasonCode field value. For more information on reason codes, see the [Reason Codes for the Simple Order API](#).**

Required Fields for Authorization Reversal

Include these required fields to reverse an authorization.

<code>apAuthReversalService_authRequestID</code>	Set to the request ID included in the authorization response. If you sent a follow-on authorization, use the request ID from the follow-on authorization response.
<code>apAuthReversalService_run</code>	Set to <code>true</code> .
<code>apPaymentType</code>	Set to <code>KLI</code> .
<code>merchantID</code>	
<code>merchantReferenceCode</code>	

XML Example: Authorization Reversal

This example shows a successful authorization reversal.

Request

```
<requestMessage xmlns="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantID>test_merchant</merchantID>
  <merchantReferenceCode>ref-123456</merchantReferenceCode>
  <apPaymentType>KLI</apPaymentType>
  <apAuthReversalService run="true">
    <authRequestID>6958688084846249104011</authRequestID>
  </apAuthReversalService>
</requestMessage>
```

Response

```
<replyMessage xmlns:c="urn:schemas-cybersource-com:transaction-data-1.213">
  <merchantReferenceCode>refnum-1234</merchantReferenceCode>
  <requestID>6958693435966100704008</requestID>
  <decision>ACCEPT</decision>
  <reasonCode>100</reasonCode>
  <requestToken>AxjnrwSTeTtFW0Xx4YsI/6IZYjWmjCbZrSIzaQojmjs01PSBX0GY0MmkmXoxZaH/
  pN5OzJYdUP50osAAFAIu</requestToken>
  <purchaseTotals>
    <currency>USD</currency>
  </purchaseTotals>
  <apAuthReversalReply>
    <reasonCode>100</reasonCode>
    <status>AUTH_REVERSED</status>
    <processorResponse>00007</processorResponse>
    <amount>10.00</amount>
    <dateTime>2023-09-28T02:49:04Z</dateTime>
    <paymentStatus>auth_reversed</paymentStatus>
    <responseCode>00007</responseCode>
    <reconciliationID>XFZ40MYVHF6H</reconciliationID>
  </apAuthReversalReply>
</replyMessage>
```

Check Status

Request the check status service when the authorization response status is **PENDING**. A pending status occurs when Klarna reviews an authorization. Cybersource recommends that you request the check status service hourly until the payment status changes.

Endpoints

Set the **apCheckStatusService_run** field to **true**, and send the request to one of these endpoints:

Production: <https://ics2ws.ic3.com/commerce/1.x/transactionProcessor>

Test: <https://ics2wstest.ic3.com/commerce/1.x/transactionProcessor>

Response Status

The check status service responds with one of these statuses in the **apCheckStatusReply_status** field value:

- **ABANDONED**: The customer did not complete the payment using the redirect URL.
- **AUTHORIZED**: The customer's payment is authorized.
- **AUTH-REVERSED**: The authorization is successfully reversed.
- **COMPLETED**: The customer completed the payment transaction.
- **FAILED**: The service request failed. A failed request can be due to either Klarna rejecting the transaction or a technical error.
- **PENDING**: The service request is accepted but is not completed. Request the check status service to retrieve status updates.
- **SETTLED**: The capture request is settled for the requested amount.

Required Fields for Check Status

Include these required fields to check the status of a transaction.

apCheckStatusService_checkStatusRequestID	Set to the request ID included in the service response you are checking.
apCheckStatusService_run	Set to true .
apPaymentType	Set to KLI .
merchantID	
merchantReferenceCode	

XML Example: Check Status

This example shows a successful check status request.

Request

```
<requestMessage
```

```

xmlns="urn:schemas-cybersource-com:transaction-data-1.213">
<merchantID>test_merchant</merchantID>
<merchantReferenceCode>refnum-1234</merchantReferenceCode>
<apPaymentType>KLI</apPaymentType>
<apCheckStatusService run="true">
  <checkStatusRequestID>6957582387066840103011</checkStatusRequestID>
</apCheckStatusService>
</requestMessage>

```

Response

```

<replyMessage xmlns:c="urn:schemas-cybersource-com:transaction-data-1.213">
<merchantReferenceCode>refnum-1234</merchantReferenceCode>
<requestID>6957587006586637103010</requestID>
<decision>ACCEPT</decision>
<reasonCode>100</reasonCode>
<requestToken>AxjnrwSTeSvqh51d5a+i/6IZYjWmjCbZrR6FeUojmjMjRvSB3i0ZiwyasZejFlof
+k3kr2h4vmV04YwAnSiT</requestToken>
<apCheckStatusReply>
  <reasonCode>100</reasonCode>
  <reconciliationID>XFZ40MYVGPWJ</reconciliationID>
  <paymentStatus>refunded</paymentStatus>
</apCheckStatusReply>
</replyMessage>

```

Reference Information

This section contains reference information that is useful when integrating Klarna.

- [Reason Codes and Klarna Response Codes](#) on page 50
- [Reporting](#) on page 54
- [Simple Order API Field Map](#) on page 55

Reason Codes and Klarna Response Codes

Important

Response fields and reason codes can be added at any time. Cybersource recommends these best practices to ensure you keep up-to-date with the latest possible responses:

- Parse the response data according to the field names instead of the field order in the response message. For more information about parsing response fields, see the documentation for your client.
- Your error handler must be able to process new reason codes without problems.
- Your error handler must use the **decision** field to determine the result if it receives a reason code that it does not recognize.

This table describes the possible reason codes that are returned by the Simple Order API in the **reasonCode** field. For a list of all of the possible reason codes and descriptions, see the [Reason Codes for the Simple Order API](#).

Reason Codes

Reason Code	Processor Response Code	Description
100	00000—status: completed. 00001—status: pending 00002—status: abandoned. 000 003—status: authorized. 00004—status: settled. 00006—status: refun ded. 00008—sta tus: reversed. 00 009—status: cancelled. 00010—status: accepted. 00011—status: cha rgeback. 00012— status: settle_initiated. 00013—status: settle_acce pted. 00014—sta tus: refund_initiated. 00015—status: active. 00016—status: revoked. 00017—status: ex pired. 00020—st atus: refund_rejected.	The transaction was successful.
101		The request is missing one or more required fields. Examine the response fields missingField_0 through missingField_N to identify which fields are missing. Resend the request with all the required fields.
102	10000—status: failed.	One or more fields in the request contain invalid data. Examine the response fields invalidField_0 through invalidField_N to identify which fields are invalid. Resend the request with valid data.

Reason Code	Processor Response Code	Description
150	20000—status: failed. 20001—status: failed. 20002—status: failed. 30000—status: failed. 30100—status: failed.	<p>A system error caused the request to fail. You must design your transaction management system to include a way to correctly handle system errors. Depending on which payment processor is handling the transaction, the error might indicate a valid Cybersource system error, or it might indicate a processor rejection because of invalid data. For either reason, Cybersource recommends you to not design your system to keep resending a transaction when a system error occurs. See the documentation for the Cybersource client (SDK) that you are using for important information about how to handle system errors and retries. Other possible reasons for a failed status:</p> <ul style="list-style-type: none"> The signature was not included in the HTTP header. The signature in the HTTP header has expired or it is not a valid signature.
151		<p>The request was received but a server timeout occurred. This error does not include timeouts that occur between the client and the server. To avoid duplicating the transaction, do not resend the request until you have reviewed the transaction status in the Business Center. See the documentation for your Cybersource client for information about handling retries in the case of system errors.</p>

Reason Code	Processor Response Code	Description
152		The request was received, but a service did not finish processing in time. To avoid duplicating the transaction, do not resend the request until you have reviewed the transaction status in the Business Center. See the documentation for your Cybersource client for information about handling retries in the case of system errors.
153		Your account is not enabled for the OCT service. Contact your Cybersource account manager to have your account enabled for this service.
202		The payment method is expired. You might also receive this value if the expiration date that you provided does not match the date that the issuing bank has on file. Request a different form of payment.
203	30000—status: failed. 30100—status: failed. 30200—status: failed. 30400—status: failed. 30500—status: failed.	The payment method was declined. No other information was provided by the issuing bank. Request a different form of payment.
204	30350—status: failed.	The account does not contain sufficient funds. Request a different form of payment.
223	30600—status: failed. 30700—status: failed.	The processor declined the transaction due to tax errors or government compliance errors.
233	30600—status: failed. 30700—status: failed.	The processor declined the payment method. For more information about the decline, search for the transaction in the Business Center and view the transaction details. Request a different form of payment.

Reporting

You can generate various types of reports for your financial and reconciliation data. For more information about how to generate these reports, see the [Reporting Developer Guide](#) and the [Reporting User Guide](#).

The [Reporting User Guide](#) contains these relevant topics:

- How and When Reports Are Generated
- Downloading Available Reports
- Subscribing to Standard Reports

Additional Resources

For additional information about how to use the Business Center, see these helpful resources.

Business Center Overview

For an overview of the various resources available in the Business Center, see this YouTube video: <https://www.youtube.com/watch?v=UDmAWGHPbWs>

Navigating the Business Center

For a step-by-step demonstration of how to navigate in the Business Center, see this YouTube video: https://www.youtube.com/watch?v=2qj_g2DParI

Managing Report Subscriptions

For an overview of how to manage report subscriptions in the Downloadable Reports section in the Business Center, see this YouTube video: <https://www.youtube.com/watch?v=tFlmkXtvxWE>

Downloading Reports

For an overview of how to download available reports in the Reports section in the Business Center, see this YouTube video: <https://www.youtube.com/watch?v=EOslUYjJvmw>

Simple Order API Field Map

Field Map

Klarna API Field Name	Simple Order API Field	Name
authorization_token	apAuthService_preapprovalToken	
billing_address.city	billTo_city	
billing_address.country	billTo_country	
billing_address.email	billTo_email	
billing_address.family_name	billTo_lastName	
billing_address.given_name	billTo_firstName	
billing_address.phone	billTo_phoneNumber	
billing_address.postal_code	billTo_postalCode	
billing_address.region	billTo_district	
billing_address.street_address	billTo_street1	
billing_address.street_address2	billTo_street2	
billing_address.title	billTo_title	
customer.date_of_birth	billTo_dateOfBirth	
customer.gender	billTo_gender	
locale	billTo_language	
merchant_urls.confirmation	apAuthService_successURL	
options.color_border	apUI_colorBorder	
options.color_border_selected	apUI_colorBorderSelected	
options.color_button	apUI_colorButton	
options.color_button_text	apUI_colorButtontext	
options.color_checkbox	apUI_colorCheckbox	
options.color_checkbox_checkmark	apUI_colorCheckboxCheckMark	
options.color_header	apUI_colorHeader	
options.color_link	apUI_colorLink	
options.color_text	apUI_colorText	
options.payment_method_category	paymentMethod_name	

Klarna API Field Name	Simple Order API Field	Name
options.radius_border	apUI_borderRadius	
order_amount	purchaseTotals_grandTotalAmount	
order_lines.name	item_#_productName	
order_lines.reference	item_#_productSKU	
order_lines.tax_rate	item_#_taxRate	
order_lines.total_amount	item_#_totalAmount	
order_lines.unit_price	item_#_unitPrice	
order_lines.quantity	item_#_quantity	
order_lines.tax_amount	item_#_taxAmount	
purchase_country	billTo_country	
purchase_currency	purchaseTotals_currency	
shipping_address.city	shipTo_city	
shipping_address.country	shipTo_country	
shipping_address.email	shipTo_email	
shipping_address.family_name	shipTo_lastName	
shipping_address.given_name	shipTo_firstName	
shipping_address.phone	shipTo_phoneNumber	
shipping_address.postal_code	shipTo_postalCode	
shipping_address.region	shipTo_district	
shipping_address.street_address	shipTo_street1	
shipping_address.street_address2	shipTo_street2	
shipping_address.title	shipTo_title	