**Klarna Payments**

Version 17.4.0



Table of Contents

[1. Summary 1-3](#_Toc280189026)

[2. Component Overview 2-4](#_Toc280189027)

[2.1 Functional Overview 2-4](#_Toc280189028)

[2.2 Use Cases 2-4](#_Toc280189029)

[2.3 Limitations, Constraints 2-4](#_Toc280189030)

[2.4 Compatibility 2-4](#_Toc280189031)

[2.5 Privacy, Payment 2-4](#_Toc280189032)

[3. Implementation Guide 3-5](#_Toc280189033)

[3.1 Setup 3-5](#_Toc280189034)

[3.2 Configuration 3-5](#_Toc280189035)

[3.3 Custom Code 3-5](#_Toc280189036)

[3.4 External Interfaces 3-5](#_Toc280189037)

[3.5 Testing 3-5](#_Toc280189038)

[4. Operations, Maintenance 4-6](#_Toc280189039)

[4.1 Data Storage 4-6](#_Toc280189040)

[4.2 Availability 4-6](#_Toc280189041)

[4.3 Support 4-6](#_Toc280189042)

[5. User Guide 5-7](#_Toc280189043)

[5.1 Roles, Responsibilities 5-7](#_Toc280189044)

[5.2 Business Manager 5-7](#_Toc280189045)

[5.3 Storefront Functionality 5-7](#_Toc280189046)

[6. Known Issues 6-8](#_Toc280189047)

[7. Release History 7-9](#_Toc280189048)

# Summary

This cartridge enables a Commerce Cloud store to use the Klarna Payments service.

The developer has to install the cartridge and integrate it into online store following the instructions from this document. It is mandatory that the merchant has to open an account with Klarna and configure some items in Commerce Cloud Business Manager to make the integration working properly.

Klarna offers a test environment, so the integration can be tested before switching to the Klarna production environment.

The integration consists of an archive, which contains the following contents:

* Cartridge called ‘int\_klarna\_payments’, and optionally either "int\_klarna\_payments\_pipelines" or "int\_klarna\_payments\_controllers" depending on which architecture is to be employed.
* A site-template archive containing new attributes and settings
* This document

The integration is based on the Site Genesis demo store provided by Commerce Cloud.

# Component Overview

## Functional Overview

Short description of the Klarna Payments system

Klarna Payments can receive payments in via the Klarna Payments API using JSON REST API and a JavaScript SDK. A result is returned right away. Through Klarna Payments you enable your consumers to try before they buy, finance purchases on your store via Klarna Credit, or make use of other payment methods made available by Klarna. Klarna offers these payment options via a widget, called Klarna Payments widget or just "the widget", which you add inline on your checkout page.

Klarna Payments uses the ‘redirect model’, where the customer is sent to the redirect\_url as a last step of the process before showing the confirmation page. The reason for this redirect is to allow Klarna to recognize the customer's device in future interactions. Klarna needs to interact with the customer's browser as first party (not in an iframe or similar). This is achieved by ‘bouncing’ the browser on a klarna.com page before presenting the confirmation page.

Payment results are sent asynchronously to a 'notification URL' at the Merchant (SCC) site. Payments can change status over the time (Accepted, Rejected and Pending) on Klarna Payments side, and these statuses are updated automatically on the Commerce Cloud side through sending these notifications.

In case order is ACCEPTED order creation in SCC proceeds as usual, in case of REJECTED status, order is failed and in case of PENDING status the 'two phase ordering' approach is used, where the order is first created but only 'placed' after notification for ACCEPTED status is asynchronously received.

<BUSINESS/FUNCTIONAL OVERVIEW OF COMPONENT, INCLUDING FLOW CHARTS, DIAGRAMS WHERE APPLICABLE>

## Use Cases

<USE CASES COVERED BY THE COMPONENT>

## Limitations, Constraints

The merchant will need a configured Klarna Payments account

The merchant can style the Klarna Payments widget (skin), to make it matching the look and feel of their store. The list with the graphic elements that can be customized, through site preferences is as follows:

* "color\_details" (site preference kpColorDetails): "#C0FFEE"
* "color\_button" (site preference kpColorButton): "#C0FFEE"
* "color\_button\_text" (site preference kpColorButtonText): "#C0FFEE"
* "color\_checkbox" (site preference kpColorCheckbox): "#C0FFEE"
* "color\_checkbox\_checkmark" (site preference kpCheckboxCheckmark): "#C0FFEE"
* "color\_header"(site preference kpColorHeader): "#C0FFEE"
* "color\_link"(site preference kpColorLink): "#C0FFEE"
* "color\_border"(site preference kpColorBorder): "#C0FFEE"
* "color\_border\_selected"(site preference kpBorderSelected): "#C0FFEE"
* "color\_text"(site preference kpColorText): "#C0FFEE"
* "color\_text\_secondary"(site preference kpColorTextSecondary): "#C0FFEE"
* "radius\_border"(site preference kpRadiusBorder): "0px"



Successful payment authorizations (Klarna Payments status APPROVED) lead to a 'Paid' order payment status and ‘Ready for Export’ export status

Refused and pending payment authorizations ( Klarna Payments statuses REJECTED and PENDING) lead to a 'Not Paid' order payment status, and ‘Not Exported’ export status

Cancels and refunds lead to a 'Not Paid' order payment status (even if the status was 'Paid' before), and ‘Not Exported’ export status

The cartridge makes use of the Klarna Payments JSON REST API and a JavaScript SDK.

<WHICH PARTS OF THE 3RD PARTY OFFERING ARE NOT SUPPORTED>

<WHICH DEMANDWARE FEATURES OR CONFIGURATIONS ARE NOT SUPPORTED, E.G. MULTIPLE LOCALES, NON-US SHIPPING ADDRESSES, ORDERS WITH MORE THAN ONE SHIPPING ADDRESS, CATALOG SIZE LARGER THAN 500K>

<IF YOUR INTEGRATION REQUIRES A CUSTOM CATALOG FEED, DESCRIBE ANY LIMITATIONS ON PRODUCT ATTRIBUTES, SUCH AS NON-SUPPORTED HTML ATTRIBUTES, 0.00$ PRICES, ONLY ORDERABLE PRODUCTS EXCLUDING BACKORDER>

## Compatibility

Based on Commerce Cloud 17.1.0 and SiteGenesis 103.1.3

<AVAILABLE SINCE DEMANDWARE X.Y.Z>

<DEMANDWARE VERSION REFERENCED FOR INTEGRATION INTO STOREFRONT (REFERENCE APPLICATION) AND USED IN SCREENSHOTS>

## Privacy, Payment

<INDICATE IF CUSTOMER PROFILE DATA IS BEING ACCESSED, IF CREDIT CARD DATA IS BEING PROCESSED OR STORED WITHIN DEMANDWARE>

# Implementation Guide

To setup and implement Klarna Payments Salesforce Commerce Cloud Cartridge, the below steps must be followed:

## Setup

<LIST CARTRIDGES THAT ARE PART OF THE COMPONENT AND ENTRY POINTS/PARAMETERS (API) IN DETAIL>

<HIGHLIGHT WHAT CARTRIDGES NEED TO BE DEPLOYED TO PRODUCTION, WHAT ARE FOR TESTING, WHAT SHOW CUSTOM CODE EXAMPLES (E.G. FOR STOREFRONT INTEGRATION)>

Klarna Payments integration has three cartridges:

* int\_klarna\_payments which implements the core storefront functionality;
* int\_klarna\_payments\_pipelines which implements the storefront functionality with pipeline based approach;
* int\_klarna\_payments\_controllers which implements the storefront functionality with controllers based approach;

### int\_klarna\_payments

The int\_klarna\_payments cartridge has the following components:

**Cartridge name**

* *int\_klarna\_payments*

**Forms**

**Pipelines**

**Scripts**

**Static**

**Templates**

**Resources**

**Webreferences**

**Metadata**

### int\_klarna\_payments\_pipelines

The *int\_klarna\_payments\_pipelines* cartridge has the following components:

**Cartridge name**

* *int\_klarna\_payments\_pipelines*

**Forms**

**Pipelines**

**Scripts**

**Static**

**Templates**

**Resources**

**Webreferences**

**Metadata**

### int\_klarna\_payments\_controllers

The *int\_klarna\_payments\_controllers* cartridge has the following components:

**Cartridge name**

* *int\_klarna\_payments\_controllers*

**Forms**

**Pipelines**

**Scripts**

**Static**

**Templates**

**Resources**

**Webreferences**

**Metadata**

## Configuration

*<CONFIGURATION STEPS INCLUDING ASSIGNMENT OF CARTRIDGES TO SITE, CUSTOM PREFERENCES, SET UP OF JOB SCHEDULES, CONFIGURATION DIFFERENCES FOR DIFFERENT INSTANCE TYPES, IMPORT OF METADATA>*

### Cartridge Path

If using the controller based integration, cartridge path should be setup as such under Sites / Manage Sites /

app\_storefront\_controllers:app\_storefront\_core:int\_klarna\_payments\_controllers:int\_klarna\_payments



If using pipelines, the cartridge path should be similar to:

app\_storefront\_pipelines:app\_storefront\_core:int\_klarna\_payments\_pipelines:int\_klarna\_payments



### Klarna Payments Configurations

Also, there is a page in Business Manager where you have to configure Klarna Payments cartridge, you can find it in Site Preferences > Custom Preferences > Klarna\_Payments:



Here is the description of the Site Preferences fields which have to be configured:

|  |  |  |
| --- | --- | --- |
| **Parameter Name** | **Attribute ID** | **Description** |
| Send product\_url and image\_url | sendProductAndImageURLs | If set to true product\_url and image\_url fields in Klarna Payments create session call will be included in API call, otherwise product\_url and image\_url fields will not be populated. |
| merchant\_reference2 mapping | merchant\_reference2 mapping | The field from SCC order (basket) object that is mapped to merchant\_reference2 field from klarna API request.  Has to be a string corresponding to the names of SCC LineItemCtnr class attributes. |
| Pre-Assesement is on for: | kpPreAssessment | Pre-assessment flag. Comma separated string values with country codes, which should have the pre-assessment flag ON. For example the following flue “UK, CN, AU” would mean the pre-assessment is on for United Kingdom, China and Australia |
| Border Color Preference | kpColorBorder | CSS hex color to be used in Klarna Payments iFrame |
| Border Selected Color Preference | kpColorBorderSelected | CSS hex color to be used in Klarna Payments iFrame |
| Button Color Preference | kpColorButton | CSS hex color to be used in Klarna Payments iFrame |
| Button Text Color Preference | kpColorButtonText | CSS hex color to be used in Klarna Payments iFrame |
| Checkbox Color Preference | kpColorCheckbox | CSS hex color to be used in Klarna Payments iFrame |
| Checkbox checkmark color preference | kpColorCheckboxCheckmark | CSS hex color to be used in Klarna Payments iFrame |
| Details Color Preference | kpColorDetails | CSS hex color to be used in Klarna Payments iFrame |
| Header Color Preference | kpColorHeader | CSS hex color to be used in Klarna Payments iFrame |
| Link Color Preference | kpColorLink | CSS hex color to be used in Klarna Payments iFrame |
| Text Color Preference | kpColorText | CSS hex color to be used in Klarna Payments iFrame |
| Secondary Text Color Preference | kpColorTextSecondary | CSS hex color to be used in Klarna Payments iFrame |
| Border Radius Preference | kpRadiusBorder | Size (in pixels) of the border radius to be used in Klarna Payments iFrame |

### Services Configuration

### Custom attributes

ORDER system object has been extended with custom attribute kpOrderID of type string, which is holding the Klarna Payments Order ID, if Klarna Payments is used for the specific order

## Custom Code

Integration may vary based on the customers' storefront. Site Genesis version 2 is used to demonstrate Klarna Payments integration.

### Integration efforts

The following steps are needed to complete the integration:

* Install the cartridges
* Change and import ‘site-template.zip’ (system objects and custom objects)
* Do the required code changes
* Configure your merchant's Klarna Payments Account in the Klarna back office
* Configure Klarna Payments parameters in Commerce Cloud
* Test
* Create styling for Klarna Widget (optional)

These steps should not require more than a few hours.

Note that the name of your storefront site ID should match the name in the ‘site-template.zip’, otherwise you will get import errors.

Also, note that the payment methods ‘Klarna’ and the payment processors 'KLARNA\_PAYMENTS' will be created or updated from the ‘site-template.zip’.

### Templates modifications - The following changes should be made regardless of whether a controller or a pipeline integration approach are being used.

To integrate Klarna Payments the following storefront cartridge templates should be updated:

* + - default/checkout/billing/billing.isml
    - default/checkout/billing/paymentmethods.isml

**default/checkout/billing/billing.isml**

Add Code:

<script><isinclude template="/resources/klarnapaymentsresources.isml"/></script>

<script type="text/javascript" src="${URLUtils.staticURL('/js/klarna-payments.js')}"></script>

<script src="https://credit.klarnacdn.net/lib/v1/api.js" async></script>

Before</isdecorate> closing tag as shown below:



Add Code:

<input type="hidden" name="klarna\_payments\_authorization\_token" value=""/>

Before</form> closing tag as shown below:



**default/checkout/billing/paymentmethods.isml**

Add Code:

<iscomment>Klarna Payments Processor</iscomment>

<div class="payment-method <isif condition="${!empty(pdict.selectedPaymentID) && pdict.selectedPaymentID=='PayPal'}">payment-method-expanded</isif>" data-method="Klarna">

<div id="klarna\_payments\_container">

</div>

Before</fieldset> closing tag as shown below:



### Pipeline modifications - If using a pipeline based SiteGenesis integration, additionally follow the instructions in this chapter. If integrating via the controller based model see next chapter.

To use Klarna Payments cartridge you will need to make the following changes in pipelines.

### COBilling pipeline. Go to COBilling pipeline, Start node and add KLARNA\_PAYMENTS-CreateSession call node entry point before checkout/billing/billing -> Billing interaction continue node (see screen shot below)



### COSummary pipeline. Go to COSummary pipeline, Submit node and add KLARNA\_PAYMENTS-Redirect call node entry point before COSummary-ShowConfirmation jump node (see screen shot below)



### COPlaceOrder pipeline. Go to COPlaceOrder pipeline, PlaceOrder node and add KLARNA\_PAYMENTS-PendingOrder call node entry point before PlaceOrder script node node (see screen shot below). Create ‘pending’ and ‘error’ transitions and end nodes as shown on the screenshot



### Controller modifications - If using a controller based SiteGenesis integration, additionally follow the instructions in this chapter. If integrating via the pipeline method please see Pipeline modifications chapter above.

### COBilling controller.

### Go to COBilling controller, returnToForm method and add the following code block:

**try**

{

require('int\_klarna\_payments\_controllers/cartridge/controllers/KLARNA\_PAYMENTS.js').CreateSession();

} **catch**( e )

{

require( 'dw/system/Logger' ).getLogger( 'COBilling.js' ).error( 'Klarna Create Session Error: {0}', e );

}

*after pageMeta.update (see screen shot below)*



### COSummary controller.

### Go to COSummary controller, submit() method and add the following code block:

**try**

{

require('int\_klarna\_payments\_controllers/cartridge/controllers/KLARNA\_PAYMENTS.js').Redirect();

} **catch**( e )

{

require( 'dw/system/Logger' ).getLogger( 'COSummary.js' ).error( 'Klarna Redirect Error: {0}', e );

}

***before*** showConfirmation(placeOrderResult.Order)(see screen shot below)



### OrderModel.js

### Go to OrderModel.js, placeOrder method and add the following code block:

**if**( session.custom.KlarnaPaymentsFraudStatus === 'PENDING' )

{

**try**

{

require('int\_klarna\_payments\_controllers/cartridge/controllers/KLARNA\_PAYMENTS.js').PendingOrder();

} **catch**( e )

{

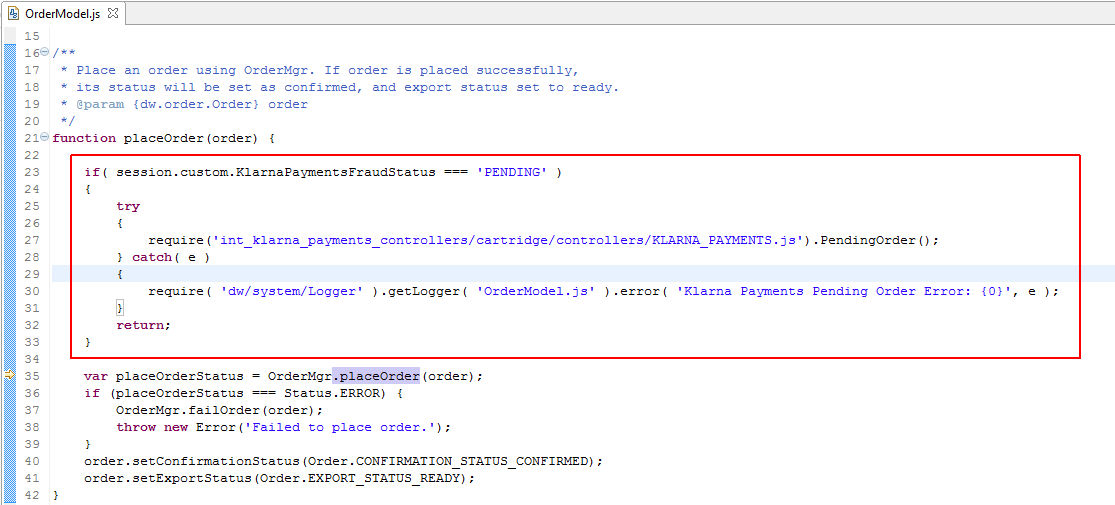
require( 'dw/system/Logger' ).getLogger( 'OrderModel.js' ).error( 'Klarna Payments Pending Order Error: {0}', e );

}

**return**;

}

**before**var placeOrderStatus = OrderMgr.placeOrder(order); (see screen shot below)



### Other modifications

<IF STOREFRONT APPLICATION NEEDS TO BE MODIFIED, EXPLAIN STEPS RELATIVE TO REFERENCE APPLICATION>

<IF ADDITIONAL CUSTOM CODE NEEDS TO BE WRITTEN, EXPLAIN IN DETAIL AND PROVIDE EXAMPLE.>

*<ESTIMATED INTEGRATION EFFORTS>*

*Checkout/billing*

## External Interfaces

<INTERFACES TO EXTERNAL SERVICES, E.G HTTP CLIENT, WEBSERVICE CALLS, DESCRIPTION OF REQUESTS AND RESPONSES>

## Testing

*<SAMPLE DATA, TEST ACCOUNT, TEST CASES, REFERENCE TO CERTIFICATION REQUIREMENTS>*

# Operations, Maintenance

## Data Storage

*<DESCRIBE ANY DATA THAT WILL BE STORED WITHIN DEMANDWARE, E.G. CUSTOM OBJECTS, INCLUDING DURATION AND CLEANUP JOBS IF APPLICABLE>*

<IS THERE A SEPARATE DATA STORAGE OUTSIDE OF DEMANDWARE, SPECIFY LOCATION AND DURATION IF APPLICABLE>

## Availability

<EXPECTED AVAILABILITY /UPTIME OF ANY EXTERNAL SERVICE, INTERFACES>

<FALLBACK SOLUTION, BEHAVIOR IF EXTERNAL SERVICES ARE NOT AVAILABLE, IMPACT ON CUSTOMER STOREFRONT>

<ANY EXISTING UTILITIES THAT HELP TO DETECT AVAILABILITY/UPTIME OF EXTERNAL SERVICE, E.G. WEBSERVICE CALL, GOMEZ PING>

<ESTIMATED PERFORMANCE METRICS FOR PEAK BUSINESS HOURS IF AVAILABLE>

<NOTIFICATION PROCESS IF EXTERNAL SERVICES, INTERFACES ARE NOT RESPONDING, E.G. HOTLINE /SUPPORT PHONE NUMBER>

## Support

<CONTACT PERSON IN CASE DEFECT FIXES OR IMPROVEMENTS FOR COMPONENT ARE REQUIRED>

# User Guide

## Roles, Responsibilities

*<LIST RECURRING TASKS THAT NEED TO BE FULFILLED BY CUSTOMER, MERCHANT TO RUN THE INTEGRATION, E.G. MANUAL FEED OF CATALOG DATA INTO 3RD PARTY SERVICE, IF APPLICABLE>*

## Business Manager

<UI SCREENSHOTS AND DESCRIBTION OF FUNCTIONALITY>

<DESCRIBE NEW BUSINESS MANAGER MODULES AND CONFIGURATION OPTIONS IF APPLICABLE>

## Storefront Functionality

*<DESCRIBE NEW STOREFRONT FUNCTIONALITY>*

# Known Issues

*<LIST KNOWN ISSUES AND WORKAROUNDS>*

# Release History

<RELEASE HISTORY OF THE COMPONENT; THIS DOCUMENT IS PART OF THE COMPONENT AND DOES NOT HAVE ITS OWN RELEASE/VERSION NUMBER.>

*<PLEASE USE THE FOLLOWING NAMING SCHEMA: CHANGES TO THE 1ST DIGIT CONTAIN INCOMPATIBILITIES, CHANGES TO THE 2ND DIGIT PROVIDE NEW MINOR FEATURES WITHOUT CAUSING INCOMPATIBILITY, A CHANGE TO THE 3RD DIGIT PROVIDES BUG FIXES WITHOUT INTRODUCING NEW FEATURES>*

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 1.0.0 | <DATE> | Initial release |