**Afterpay Integration**

Version 1.0.0



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 implemetation guide describes how to integrate Afterpay Payment Processor (int\_afterpay) in to Demandware’s reference application SiteGenesis v2.

This cartridge provides the ability to use Afterpay as a Payment Method for during checkout of the Demandware storefront.

To integrate you will need to modify SiteGenesis storefront cartridge.

# Component Overview

## Functional Overview

Afterpay provides a payment processing gateway. Afterpay’s payment gateway connects your storefront to process transactions that allow customers to purchase products

## Use Cases

Allows checkout using Afterpay payment method

## Limitations, Constraints

The following Afterpay API’s are not supported

1. Void Payment
2. Capture Payment (Authorise or Direct Capture is used)
3. Update Shipping Courier
4. List Payments
5. Create Refund

Currently only AUD is supported by Afterpay

This cartridge supports the Afterpay API to return a PENDING order status. The cartridge contains a job schedule that queries Afterpay orders that are in PENDING status. This functionality is not yet enabled in Afterpay however may be in the future.

## Compatibility

Available since Demandware 16.4

SiteGenesis reference application integration examples and references version v2

## Privacy, Payment

This integration requires access to the following customer data elements: Billing Address, Shipping Address, Order Details and Customer Profile.

No credit card details are stored within Demandware using this integration.

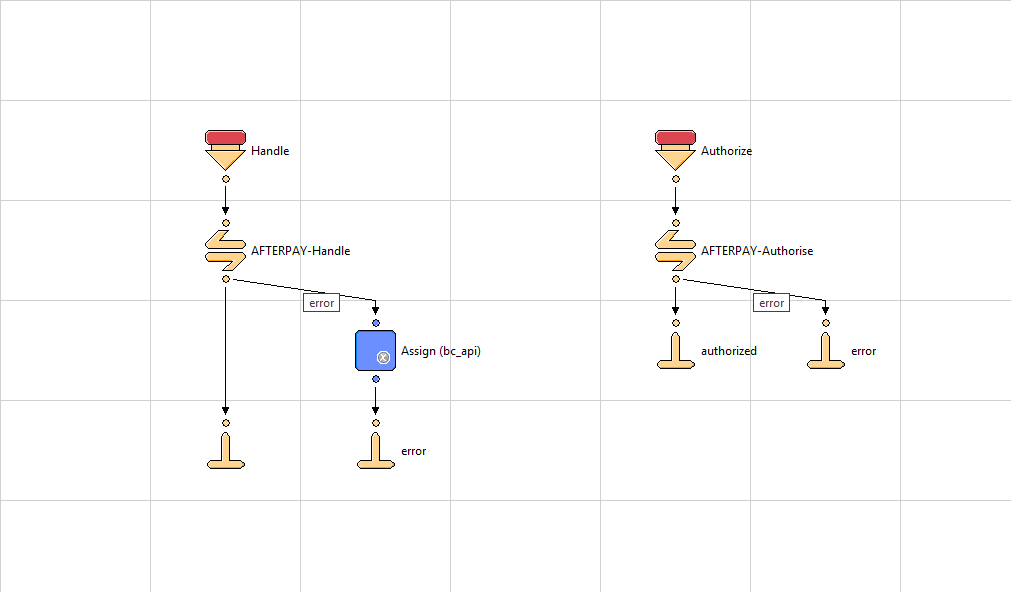
# Implementation Guide

## Setup

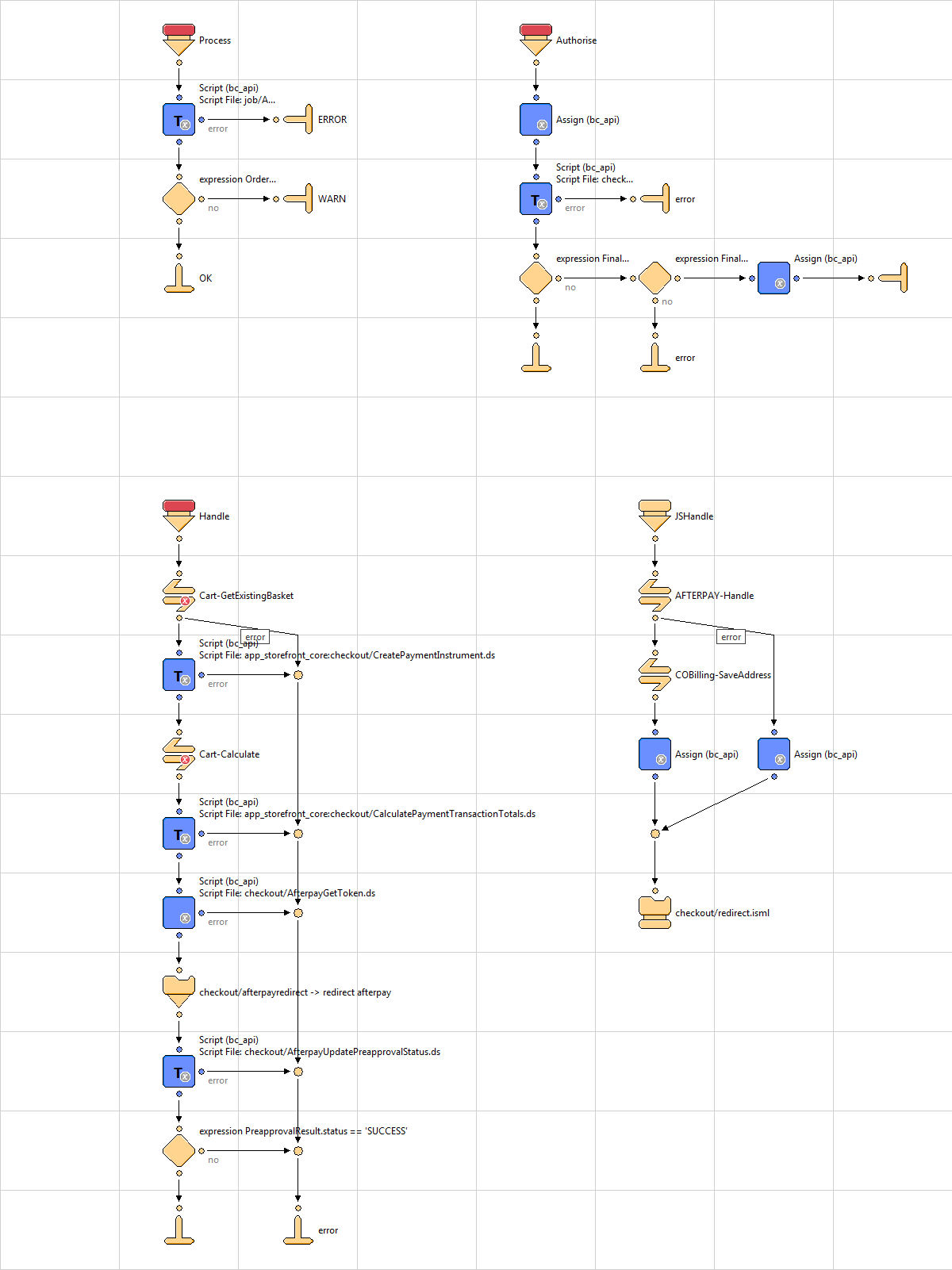
This section describes the pipeline/controller structure and name of files in Afterpay cartridges.

**CARTRIDGE STRUCTURE**

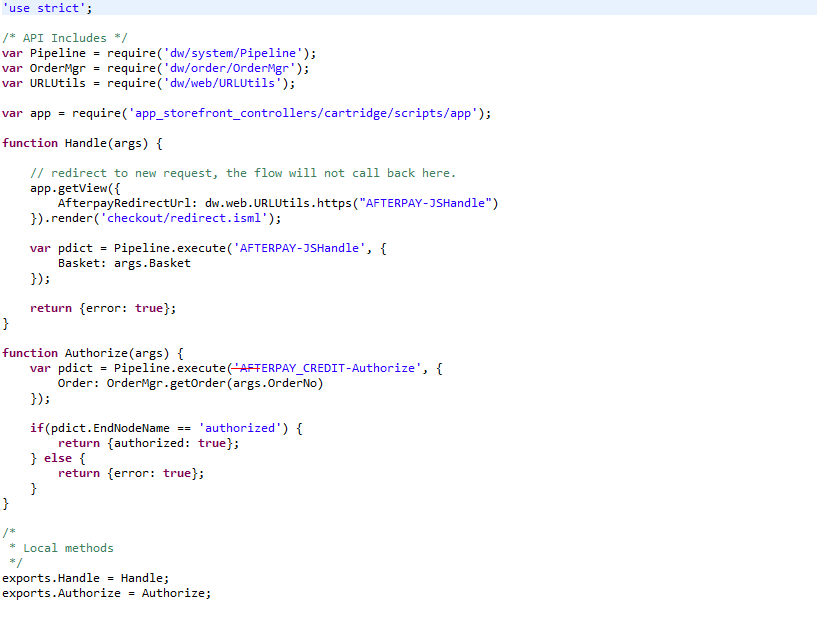
Visualizations of the pipelines and controllers:



**Visualization of the pipeline AFTERPAY\_CREDIT**

****

**Visualization of the pipeline AFTERPAY**

****

**Visualization of the pipeline AFTERPAY\_CREDIT controller**

Pipelines/Controllers:

|  |  |
| --- | --- |
| **PIPELINE/CONTROLLER NAME** | **DESCRIPTION** |
| AFTERPAY\_CREDIT | Main pipeline/controller that acts as the external interface. |
| AFTERPAY | Helper pipeline that supports and performs additional tasks required for implementation |

## INSTALLATION

### Adding the Cartridges in Demandware Studio

To upload the cartridges into the Demandware server you first need to add the cartridges into Demandware studio. In order to do this, follow these instructions:

1. In Demandware UXStudio select in the menu *File 🡪 Import*.
2. In the import dialog select *General 🡪 Existing projects in the workspace* and click next
3. Ensure *Select archive file* is selected and select the compressed cartridge file by clicking on the *Browse* button.
4. Click *Finish* to import the cartridge.
5. Studio will now ask you if you want to link the cartridge to your active Demandware server connection. Click on y*es* or manually link the cartridge to your server by checking the project under project references in the server connection properties.

### Activating the Cartridges in Business Manager

Before the Afterpay functionality can become available to SiteGenesis, the cartridges have to be added to the cartridge path of the Site in question. In order to do this, follow the following instructions:

1. Log into Business Manager
2. Navigate to *Administration 🡪 Sites 🡪 Manage Sites*.
3. Click on the site name and on the next page go to the *Settings* tab.
4. In the textbox *Cartridges* add “**:int\_afterpay**” in front of sitegenesis cartridge.
5. Click *Apply*.
6. To activate the cartridge for the Sandbox/Development/Production instances repeat steps **4** and **5** after selecting the appropriate instance from the *Instance Type* dropdown menu.
7. Repeat steps **3** to **6** for each site that is to use Afterpay.
8. To run the Job in Afterpay cartridge,Navigate to *Administration* 🡪 *Sites* 🡪 *Manage Sites*.
9. Go to “Manage the **Business Manager** site”
10. In the textbox *Cartridges* add “**:int\_afterpay**” in front of sitegenesis cartridge.

## Configuration

This chapter will describe how to configure the cartridge in Business Manager for SiteGenesis.

### Importing Meta Data

All import files can be found in the metadata folder within cartridge installation.

There are some site data files which are located in the metadata/sites/SiteGenesis folder. This folder needs to be renamed to the site id of the site you are importing into.

Create a zip file of the meatadata folder and import into site using Business Manager Site Import/Export.

To import all necessary Afterpay settings, log in to the Business Manager and navigate to *Administration* 🡪 *Site Development* 🡪 *Import & Export*. Now upload the zip file you created using the upload button and, finally go back and use the import button to import the file. After a successful import, you will be able to configure the custom site preferences according to your Afterpay account data.

### Setting Afterpay Custom Site Preferences

In Business Manager, navigate to the *SiteGenesis Site* 🡪 *Site Preferences* 🡪 *Custom Preferences*. A custom site preference group with the ID Integration\_Afterpay is available. Please select it and edit the attributes according to your Afterpay account data and the data shown below.

|  |  |  |
| --- | --- | --- |
| **Site Preference** | **Description** | **Default Values** |
| **Afterpay Display Product Details Page Information** | The field determines if the Afterpay messaging is displayed on the Product Details Page | true |
| **Afterpay Display Product List Page Information** | The field determines if the Afterpay messaging is displayed on the Product List Page (category pages) | false |
| **Afterpay Javascript URL** | The endpoint where the javascript library is obtained from – Use <https://www-sandbox.secure-afterpay.com.au/afterpay.js> for sandbox | www.secure-afterpay.com.au/afterpay.js |
| **Afterpay Learn More URL** | The URL that links to more information about Afterpay that users can click to view | https://www.afterpay.com.au/terms/ |
| **Afterpay Payment Mode** | This field determines if the payment should be Authorised only or if a Direct Capture is to be used | Merchant Specific - Select the ones to be supported. |
| **Redirect Cancel Url** | The Demandware endpoint called if the user cancels out of the Afterpay website | AFTERPAY-Cancel |
| **Redirect Confirm Url** | The Demandware endpoint called by Afterpay when completing transaction | AFTERPAY-Confirm |
| **Afterpay Service Name** | This field is used to identify the Demandware service used to connect with Afterpay. This only needs to be changed if you are using multiple sites where those sites are using additional Afterpay accounts as the accounts are stored in the service | afterpay.http.defaultendpoint |

### Setting Demandware Service

*Administration* 🡪 *Operations* 🡪 *Services*. A service with the name Integration\_Afterpay is available that was imported as part of site import. Select the *Credentials* tab and select *afterpay.http.defaultcredentials* and enter the account details.

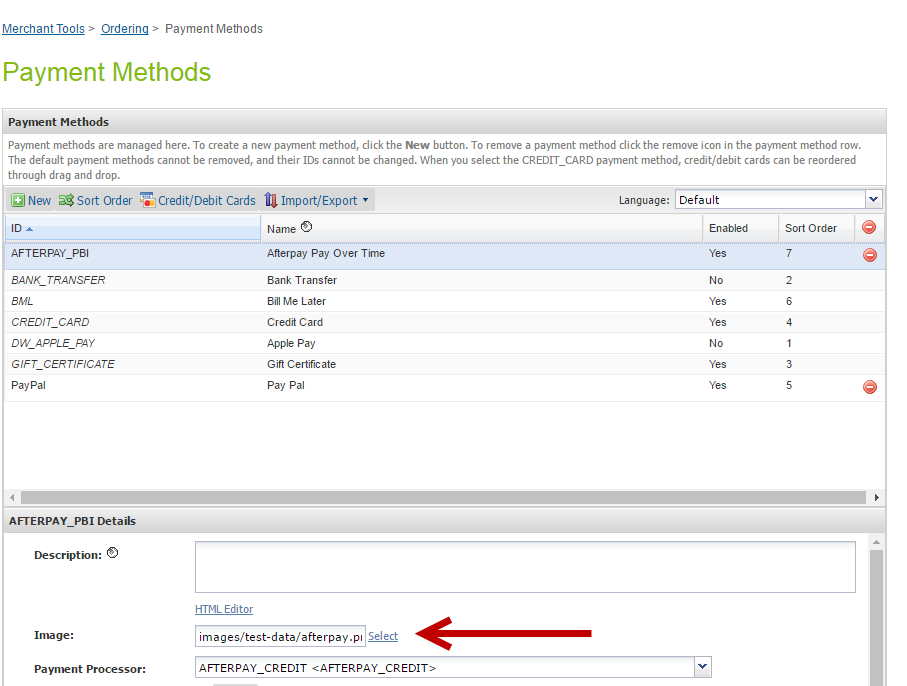
|  |  |
| --- | --- |
| **Attribute** | **Description** |
| **URL** | This is the Afterpay API endpoint  <https://api-sandbox.secure-afterpay.com.au/v1/>  <https://api.secure-afterpay.com.au/v1/> (production) |
| **User** | Enter the Account ID provided by Afterpay |
| **Password** | Enter the Key provided by Afterpay |

### Adding the Afterpay image to Payment method

In order to display the Afterpay image on the site the image needs to be added to your site.

*SiteGenesis Site* 🡪 *Ordering* 🡪 *Payment Methods*

Select the payment method with ID AFTERPAY\_PBI and locate the image attribute and upload the image included in the cartridge (int\_afterpay\cartridge\static\default\images\ afterpay-logo.png)



### Modifying Content

There is a content slot (afterpay-checkout-pbi) and content asset (afterpay-checkout-pbi) added during Site Import/Export that are used in the checkout and are displayed when selecting Afterpay payment method. The content asset (afterpay-checkout-pbi) can be modified are required to suit your site.

There are a few items added to the resource file in this cartridge.

|  |  |
| --- | --- |
| Property | Description |
| afterpay.pbi.text | The text used on product and tiles to show Afterpay payments information (if enabled) |
| afterpay.api.pending | The message displayed if payment is pending (currently not supported by Afterpay) |
| afterpay.api.error | The message displayed if there is an error |

### Setting up Job Schedule

This cartridge supports an Afterpay order status of PENDING by running a Job Schedule against all Afterpay Orders that are in a PENDING state. This is not required at this point as Aferpay will only respond with APPROVED or DECLINED and if the order is DECLINED the checkout cannot complete.

### Setting up payment Min/Max values

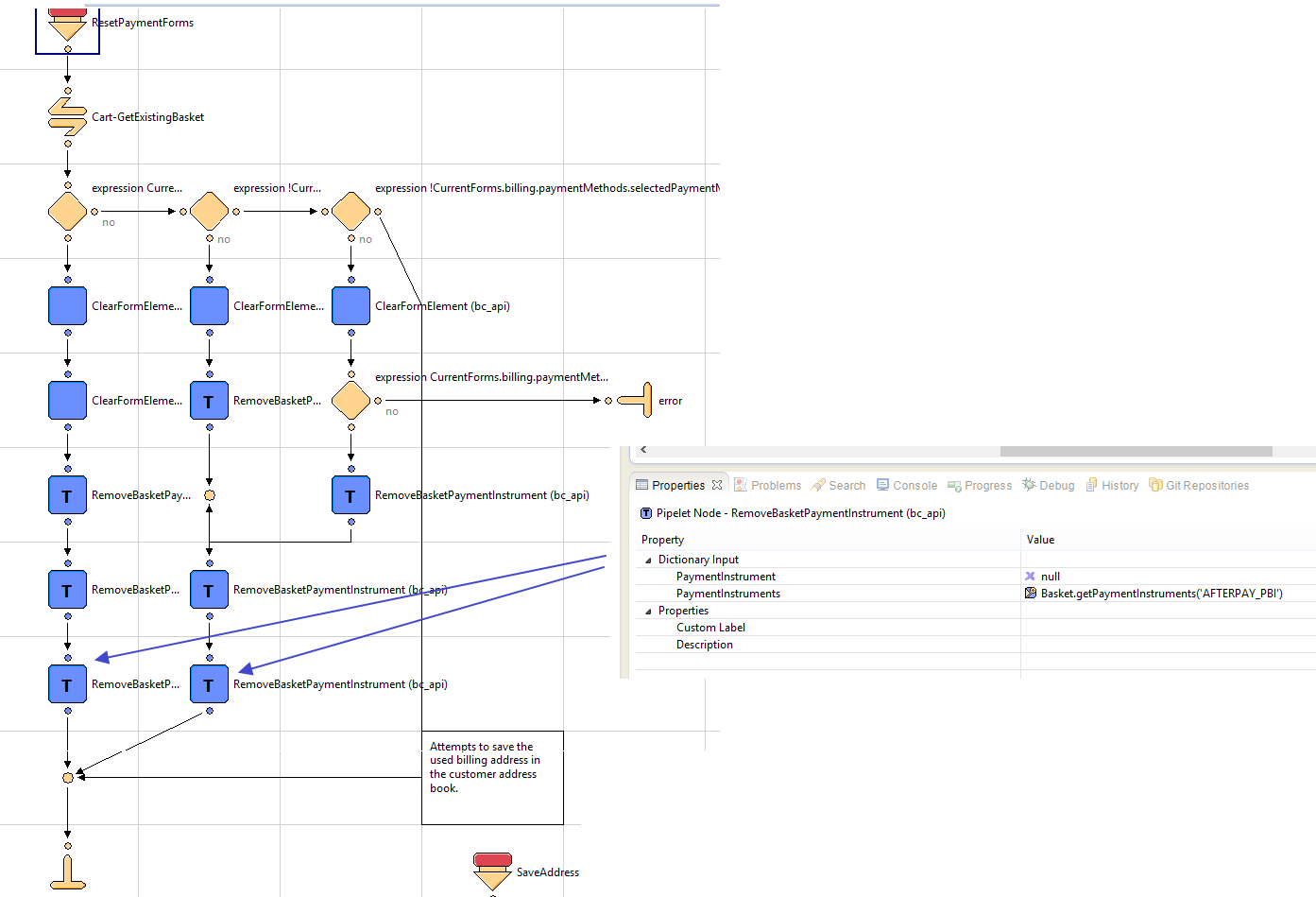
Afterpay have maximum limits that can be set up with Afterpay. To set up these values in Demandware go to *SiteGenesis Site* 🡪 *Ordering* 🡪 *Payment Methods* and select the payment method with ID AFTERPAY\_PBI. Locate Min/Max Payment Ranges and set according to your Afterpay settings.

## Custom Code

In order to make this Afterpay cartridge work, a few modifications to SiteGenesis are required.

### COBilling Pipeline

In pipeline **COBilling-ResetPaymentForms**, add two pipelets 'com.demandware.pipelet.basket.RemoveBasketPaymentInstrument' at the end of the pipeline:



In Dictionary Input 'PaymentInstruments', set it with value: Basket.getPaymentInstruments('AFTERPAY\_PBI')

The cartridge contains a modified version of COBilling.xml. Removed this from the cartridge if you have modified your version using the instructions above

### COBilling.js

If using the controller version the following controller needs to be modified:

In the method resetPaymentForms(), add the command for the three conditions: cart.removePaymentInstruments(cart.getPaymentInstruments('AFTERPAY\_PBI'));

***function*** resetPaymentForms**()** **{**

***var*** cart **=** app**.**getModel**(**'Cart'**).**get**();**

***var*** status **=** Transaction**.**wrap**(*function*** **()** **{**

***if*** **(**app**.**getForm**(**'billing'**).**object**.**paymentMethods**.**selectedPaymentMethodID**.**value**.**equals**(**'PayPal'**))** **{**

app**.**getForm**(**'billing'**).**object**.**paymentMethods**.**creditCard**.**clearFormElement**();**

app**.**getForm**(**'billing'**).**object**.**paymentMethods**.**bml**.**clearFormElement**();**

cart**.**removePaymentInstruments**(**cart**.**getPaymentInstruments**(**PaymentInstrument**.**METHOD\_CREDIT\_CARD**));**

cart**.**removePaymentInstruments**(**cart**.**getPaymentInstruments**(**PaymentInstrument**.**METHOD\_BML**));**

// remove Afterpay PaymentInstrument

cart**.**removePaymentInstruments**(**cart**.**getPaymentInstruments**(**'AFTERPAY\_PBI'**));**

**}** ***else*** ***if*** **(**app**.**getForm**(**'billing'**).**object**.**paymentMethods**.**selectedPaymentMethodID**.**value**.**equals**(**PaymentInstrument**.**METHOD\_CREDIT\_CARD**))** **{**

app**.**getForm**(**'billing'**).**object**.**paymentMethods**.**bml**.**clearFormElement**();**

cart**.**removePaymentInstruments**(**cart**.**getPaymentInstruments**(**PaymentInstrument**.**METHOD\_BML**));**

cart**.**removePaymentInstruments**(**cart**.**getPaymentInstruments**(**'PayPal'**));**

// remove Afterpay PaymentInstrument

cart**.**removePaymentInstruments**(**cart**.**getPaymentInstruments**(**'AFTERPAY\_PBI'**));**

**}** ***else*** ***if*** **(**app**.**getForm**(**'billing'**).**object**.**paymentMethods**.**selectedPaymentMethodID**.**value**.**equals**(**PaymentInstrument**.**METHOD\_BML**))** **{**

app**.**getForm**(**'billing'**).**object**.**paymentMethods**.**creditCard**.**clearFormElement**();**

***if*** **(!**app**.**getForm**(**'billing'**).**object**.**paymentMethods**.**bml**.**ssn**.**valid**)** **{**

***return*** ***false*;**

**}**

cart**.**removePaymentInstruments**(**cart**.**getPaymentInstruments**(**PaymentInstrument**.**METHOD\_CREDIT\_CARD**));**

cart**.**removePaymentInstruments**(**cart**.**getPaymentInstruments**(**'PayPal'**));**

// remove Afterpay PaymentInstrument

cart**.**removePaymentInstruments**(**cart**.**getPaymentInstruments**(**'AFTERPAY\_PBI'**));**

**}**

***return*** ***true*;**

**});**

***return*** status**;**

**}**

### Templates

Numerous templates have been modified and are contained as modified versions of SiteGenesis with this cartridge. The following lists these changes.

**Checkout/billing/billing.isml**

Locate the iscomment “billing address” (line 30) and enter the following:

<isif condition="${(!empty(pdict.AfterpayApiError) && (pdict.AfterpayApiError)) || (!empty(request.httpParameterMap.get('afterpay').stringValue) && (request.httpParameterMap.get('afterpay').stringValue == 'error'))}">

<div class=**"error-form"**>

<isprint value="${Resource.msg('afterpay.api.error','afterpay',null)}" />

</div>

</isif>

**Checkout/billing/paymentmethods.isml**

Locate the following on line 23

<label for=**"is-${radioID}"**><isprint value="${Resource.msg(paymentMethodType.label,'forms',null)}"/></label>

And replace with:

<label for=**"is-${radioID}"**>

<isset name="paymentObject" value="${paymentMethodType.object}" scope="page"/>

<isif condition="${empty(paymentObject.image)}">

<isprint value="${Resource.msg(paymentMethodType.label,'forms',null)}"/>

<iselse>

<span class=**"payment-image ${radioID.toLowerCase()}-image"** style=**"background-image:url(${paymentObject.image.URL})"**></span>

</isif>

</label>

Locate the following on line 138

<iscomment>

**Custom processor**

**--------------------------------------------------------------**

</iscomment>

Add this after:

<iscomment>

**Afterpay Pay Over Time**

**--------------------------------------------------------------**

</iscomment>

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

<isslot id="afterpay-checkout-pbi" description="Afterpay checkout PBI content" context="global" />

</div>

**Checkout/confirmation/confirmation.isml**

Locate the following on line 15

<div class=**"confirmation-message"**>

<h1>**${Resource.msg('confirmation.thankyou','checkout',null)}**</h1>

<iscontentasset aid="confirmation-message" />

</div>

And replace with:

<div class=**"confirmation-message"**>

<h1>**${Resource.msg('confirmation.thankyou','checkout',null)}**</h1>

<isif condition="${!empty(pdict.AfterpayPendingOrderMessage)}">

<isprint value="${pdict.AfterpayPendingOrderMessage}" encoding="off"/>

</isif>

<iscontentasset aid="confirmation-message" />

</div>

**Checkout/summary/summary.isml**

Locate the following on line 225

<input type=**"hidden"** name=**"${dw.web.CSRFProtection.getTokenName()}"** value=**"${dw.web.CSRFProtection.generateToken()}"**/>

Add this after:

<isif condition="${!empty(pdict.apToken)}">

<input type=**"hidden"** name=**"apToken"** value=**"${pdict.apToken}"**/>

</isif>

**Checkout/afterpayredirect.isml**

New template that is required

**Checkout/redirect.isml**

New template that is required

**components/countryselector.isml**

Only Required to make sure cartridge works on SiteGenesis v103+ as it does not support AU

**flow/http\_404.isml**

New template that is required

**mail/** **orderconfirmationjobschedure.isml**

New template that is required \*Only if using job which is currently not required.

**product/components/afterpaymessage.isml**

New template that is required

**product/components/pricing.isml**

Locate first Demandware script block and add before closing tag:

**//init script AfterpayDisplayProductMessage**

**var apMessageService = require("int\_afterpay/cartridge/scripts/product/AfterpayDisplayProductMessage.js");**

Locate this line of code on line 30: <isinclude template="product/components/standardprice"/>

Add this after:

<isinclude template="util/modules">

Locate this code on line 106

<iselse/>

<isif condition="${ShowStandardPrice}">

<span class=**"price-standard"**><isif condition="${StandardPrice.valueOrNull != null && StandardPrice.valueOrNull > **0}">**<isprint value="${StandardPrice}"/><iselse/>**${Resource.msg('product.pricing.noprice','product',null)}**</isif></span>

</isif>

<iscomment>**Let promotional price of 0 display as is, in this case isPromoPrice is true**</iscomment>

<span class=**"price-sales"**><isif condition="${(SalesPrice.valueOrNull != null && SalesPrice.valueOrNull > **0) || isPromoPrice}">**<isprint value="${SalesPrice}"/><iselse/>**${Resource.msg('product.pricing.noprice','product',null)}**</isif></span>

</isif>

And replace with:

<iselse/>

<isif condition="${ShowStandardPrice}">

<span class=**"price-standard"**>

<isif condition="${StandardPrice.valueOrNull != null && StandardPrice.valueOrNull > **0}">**

<isprint value="${StandardPrice}"/>

<isset name="message" value="${apMessageService.getPDPMessage(StandardPrice)}" scope="page" />

<iselse/>

**${Resource.msg('product.pricing.noprice','product',null)}**

</isif>

</span>

</isif>

<iscomment>**Let promotional price of 0 display as is, in this case isPromoPrice is true**</iscomment>

<span class=**"price-sales"**>

<isif condition="${(SalesPrice.valueOrNull != null && SalesPrice.valueOrNull > **0) || isPromoPrice}">**

<isprint value="${SalesPrice}"/>

<isset name="message" value="${apMessageService.getPDPMessage(SalesPrice)}" scope="page" />

<iselse/>

**${Resource.msg('product.pricing.noprice','product',null)}**

</isif>

</span>

</isif>

Locate this code on line 160 at very bottom of file:

<isif condition="${!empty(pdict.OrgProduct)}">

<iscomment>**Restore current product instance**</iscomment>

<isset name="Product" value="${pdict.OrgProduct}" scope="pdict"/>

<isset name="OrgProduct" value="${null}" scope="pdict"/>

</isif>

Add this before:

<isafterpaymessage message="${message}" classname="pdp-afterpay-message"/>

**product/producttile.isml**

Locate this code at the top of file:

<iscontent type="text/html" charset=**"UTF-8"** compact=**"true"**/>

<isif condition="${(pdict.cache != null) ? pdict.cache : true}">

<iscache type="relative" hour="24" varyby="price\_promotion"/>

</isif>

Add this after:

<isinclude template="util/modules">

Locate this code on line 239:

**prices.push(price);**

**if (extraPrice) {prices.push(extraPrice);}**

And replace with:

**if (price** && **price.value) {prices.push(price);}**

**if (extraPrice** && **extraPrice.value) {prices.push(extraPrice);}**

Locate this comment on line 248: <iscomment>Promotion</iscomment>

Add this before:

<iscomment>**Afterpay message**</iscomment>

<iscomment>**++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++**</iscomment>

<isscript>

**var apMessageService = require("int\_afterpay/cartridge/scripts/product/AfterpayDisplayProductMessage.js");**

**var message = apMessageService.getPLPMessage(prices[prices.length-1].value);**

</isscript>

<isafterpaymessage message="${message}" classname="plp-afterpay-message"/>

**util/modules.isml**

Add this:

<iscomment>

**Render the Afterpay message for PLP and PDP**

</iscomment>

<ismodule template="product/components/afterpaymessage"

name="afterpaymessage"

attribute="message"

attribute="classname"

/>

### Javascript

Add all of the javascript files in this cartridge into your solution.

### CSS

This cartridge contains Sass files located in a folder named scss. These need to be compiled into css into your site

### Additional Files Included in Cartridge

There are a few files included in cartridge to make Afterpay work with as only Australia is supported and Australia in not added into SiteGenesis v103+

These are not necessarily required in your solution and can be removed based on your existing code base.

* forms/en\_AU/billingaddress.xml
* forms/en\_AU/shippingaddress.xml
* forms/en\_AU/states.xml
* countries.json

## Testing

Afterpay has a sandbox that can be used for testing. In Business Manager, navigate to the *SiteGenesis Site* 🡪 *Site Preferences* 🡪 *Custom Preferences*. A custom site preference group with the ID Integration\_Afterpay is available. Please select it and locate the Afterpay Javascript URL preference and edit the attributes to use the sandbox URL (<https://www-sandbox.secure-afterpay.com.au/afterpay.js>). The service credentials also need to be updated with an Account ID and Key provided by Afterpay. The Service URL needs to be set to <https://api-sandbox.secure-afterpay.com.au/v1/> See 3.3.3

# Operations, Maintenance

## Data Storage

Some additional custom attributes are stored against system objects. The following table lists these attributes and their uses.

|  |  |  |  |
| --- | --- | --- | --- |
| Object | Attribute | Type | Description |
| Order | Afterpay Order ID  (apPaymentD) | Boolean | Flags the order as an Afterpay order |
| PaymentTransaction | Afterpay Token  (apToken) | String | The token used to place order in Afterpay |
| PaymentTransaction | Afterpay Initial Status  (apInitialStatus) | String | The pre-approval status first returned |
| PaymentTransaction | Afterpay Payment Mode  (apPaymentMode) | String | The payment mode used |
| PaymentTransaction | Afterpay Direct Payment Status  (apDirectPaymentStatus) | String | The status of Direct Payment, if this payment mode is used |
| PaymentTransaction | Afterpay Authorise Status  (apAuthoriseStatus) | String | The status of Authorise, if this payment mode is used |
| PaymentTransaction | Afterpay Order ID  (apPaymentD) | String | The Payment ID generated by Afterpay |

## Availability

If the Afterpay service is unavailable the user will not be able to checkout using this payment method.

The service availability can be tracked in Demandware using the Service Status.

## Support

For support please contact support@afterpay.com.au

# User Guide

## Roles, Responsibilities

Integration of this cartridge will typically be done by a Demndware developer.

Afterpay will provide access keys for be used with the API.

## Business Manager

No additional Business Manager functionality is supported by this cartridge that has not already been covered.

## Storefront Functionality

This cartridge provides an additional payment method during checkout

# Known Issues

No known issues at the time

# Release History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 1.0.0 | 13 September 2016 | Initial release |