int\_Adyen

Version 16.1.0



**Table of Contents**

[1. General Information 5](#_Toc448825628)

[2. Component Overview 6](#_Toc448825629)

[2.1. Functional Overview 6](#_Toc448825630)

[Short description of the Adyen payment system 6](#_Toc448825631)

[2.2. Use Cases 6](#_Toc448825632)

[2.2.1 Use Case: Redirect to Adyen HPP 7](#_Toc448825633)

[2.2.2 Use Case: Cancel Payment on Adyen HPP 14](#_Toc448825634)

[2.2.3 Use Case: Credit Card 15](#_Toc448825635)

[2.2.4 Use Case: 3-D Secure Credit Card 20](#_Toc448825636)

[2.2.5 Use Case: AVS 23](#_Toc448825637)

[2.3. Limitations, Constraints 28](#_Toc448825638)

[2.4. Compatibility 28](#_Toc448825639)

[2.5. Privacy, Payment 28](#_Toc448825640)

[3. Implementation Guide 29](#_Toc448825641)

[3.1. Setup 29](#_Toc448825642)

[3.1.1 Installation 29](#_Toc448825643)

[3.1.2 Metadata Import 29](#_Toc448825644)

[3.2. Business Manager Configuration 30](#_Toc448825645)

[3.3. Adyen Account Configuration 32](#_Toc448825646)

[3.4. Integration Efforts 33](#_Toc448825647)

[3.5. Changes to pipelines, templates and others 35](#_Toc448825648)

[3.5.1 COSummary pipeline 35](#_Toc448825649)

[3.5.2 COPlaceOrder pipeline 35](#_Toc448825650)

[3.5.3 COBilling pipeline 37](#_Toc448825651)

[3.5.4 Form billingaddress.xml 37](#_Toc448825652)

[3.5.5 Script cart/CartUtils.ds 37](#_Toc448825653)

[3.5.6 Script checkout/AddAddressToAddressBook.ds 38](#_Toc448825654)

[3.5.7 Script checkout/UpdateBillingAddress.ds 38](#_Toc448825655)

[3.5.8 Template checkout/billing/billing.isml 39](#_Toc448825656)

[3.5.9 Template checkout/billing/paymentmethods.isml 39](#_Toc448825657)

[3.5.10 Template checkout/summary/summary.isml 40](#_Toc448825658)

[3.5.11 Template components/header/htmlhead.isml 40](#_Toc448825659)

[3.5.13 Script billing.js 40](#_Toc448825660)

[3.5.14 File resources/forms.properties 43](#_Toc448825661)

[3.6. Additional configuration 44](#_Toc448825662)

[3.7. Pipelines in the Adyen Cartridge 44](#_Toc448825663)

[3.7.1 Adyen 44](#_Toc448825664)

[3.7.2 ADYEN\_CREDIT 46](#_Toc448825665)

[3.8. External Interfaces 46](#_Toc448825666)

[4. Testing 47](#_Toc448825667)

[4.1. Test Credit Cards 47](#_Toc448825668)

[4.2. Business Manager Configuration 47](#_Toc448825669)

[4.3. Testing API methods 47](#_Toc448825670)

[4.3.1 Capture & Cancellation Payment 47](#_Toc448825671)

[5. Operations, Maintenance 49](#_Toc448825672)

[5.1. Data Storage 49](#_Toc448825673)

[5.2. Availability 49](#_Toc448825674)

[5.3. Support 49](#_Toc448825675)

[6. User Guide 50](#_Toc448825676)

[6.1. Roles, Responsibilities 50](#_Toc448825677)

[6.2. Business Manager 50](#_Toc448825678)

[6.3. Storefront Functionality 51](#_Toc448825679)

[7. Known Issues 52](#_Toc448825680)

[8. Release History 53](#_Toc448825681)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| **OSF Global Services** | 15.01.2015 | Adding new functionality to existing cartridge. | 2.0 |
| **OSF Global Service** | 16.04.2016 | Corrections | 2.1 |

# General Information

This cartridge enables a Demandware store to use the Adyen payment 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 Adyen and configure some items in Demandware Business Manager as well as in the Adyen back office to make the integration working properly.

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

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

* cartridge called ‘int\_adyen’
* an extension cartridge to the Business Manager ‘bm\_adyen’, which enables the display of Adyen specific information in order details
* a site-template archive containing new attributes and settings
* an example 'skin', which is used for styling the Adyen hosted payment page (HPP)
* an archive with standard storefront cartridges with Adyen integrated. They can be used as an example for the integration and testing the cartridge
* this document

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

# Component Overview

## Functional Overview

### Short description of the Adyen payment system

Adyen can receive payments in 2 ways:

1. via the Adyen API using SOAP or POST. A result is returned right away. This can only be used for Credit Cards. The cartridge uses POST via the Adyen REST API
2. via the 'redirect model' where a form containing certain fields is posted to Adyen and the shopper is redirected to Adyen's Hosted Payment Page (HPP) . After completing the payment, the shopper is returned to the shop's resultURL. This resultURL can be configured and contains the result of the payment (success or failed)

Compared to the API, the redirect model offers much more payments methods as well as some other functionality.

However, if you decide to handle credit card payments yourself, you can use API payments, which are implemented in the cartridge. The stored credit card data could be used for returning customers, saving them the trouble of entering their details again.

Payment results are sent asynchronously to a 'notification URL' at the Merchant (DW) site. Payments can change status over the time (Authorised, Refunded, Cancelled and others) on Adyen side, and these statuses are updated automatically on Demandware side through sending these notifications.

In both cases the 'two phase ordering' approach is used, where the order is first created but only 'placed' after a payment has been authorized (which is now standard behavior in Site Genesis). In case if the payment cannot be authorized, the order will fail.

## Use Cases

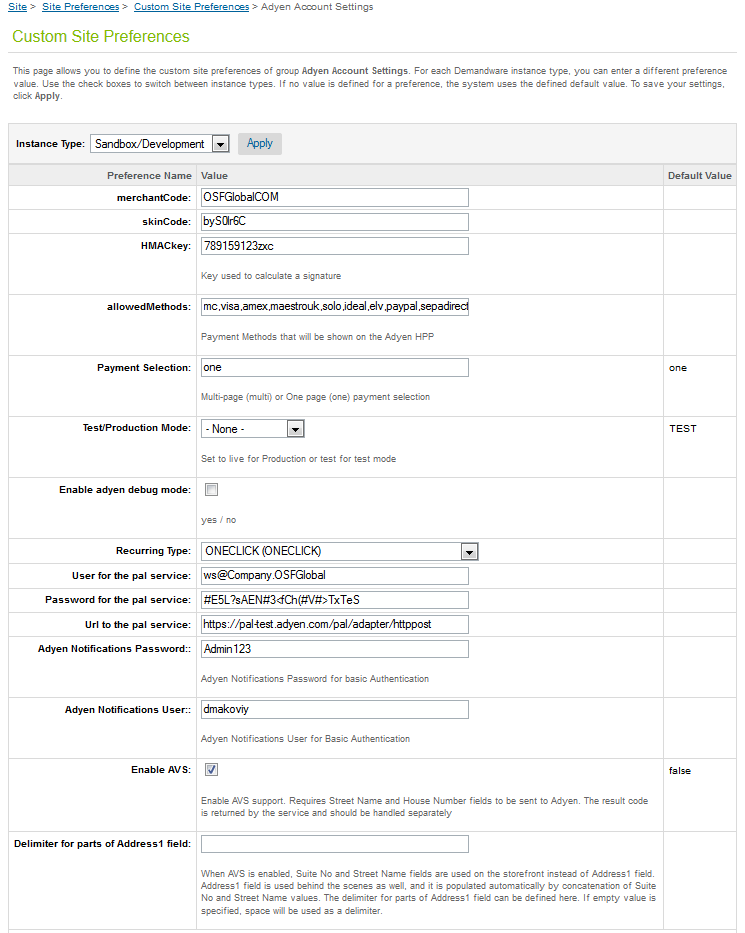
Registered and unregistered shoppers will be able to follow the standard Site Genesis flow with the following changes:

* Adyen redirect, Adyen API and Gift Certificates (GC) will be the only payment options
* if Adyen redirect is enabled:
  + no entering of credit card details on the Billing page is possible when the user selects this payment method (as this is done on the Adyen HPP) and Directory Lookup feature is disabled
  + the possibility to choose one of the predefined payment methods on the Billing page with further redirection to Adyen HPP, if Directory Lookup is enabled and the user selected Adyen payment method
  + a redirect to Adyen HPP after clicking the ‘SUBMIT ORDER’ button on the Order Confirmation page (only if no GCs are used and/or they don't cover the whole amount)
  + a return to the DW Order Summary page after successful payment authorization
  + a return to the DW Order Confirmation page, if the user cancelled the payment on Adyen HPP
  + a return to the DW Order Confirmation page with an error message displayed, if the payment was refused
* if Adyen API payments are enabled:
  + credit card details are entered and stored in your DW shop
  + for the other payment methods you can still redirect to the Adyen HPP, but can disable credit cards via the configuration of your skin
  + if the AVS feature is enabled, the user will see 2 new fields ‘Suite’ and ‘Street’ (or ‘Building number’ and ‘Street name’) on all address forms of the site, instead of field ‘Address 1’

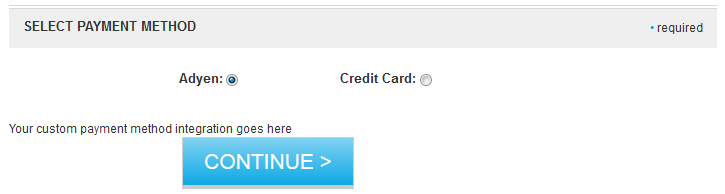
The next pages describe all use cases in details.

### 2.2.1 Use Case: Redirect to Adyen HPP

This use case describes a redirect to the Adyen Payment page (HPP). Make sure the settings are correct (see screenshot below). Usually Payment Selection is set to 'one' to select the one-page flow on the Adyen HPP. If the debug flag is checked, you will see a button before the redirect, which gives you a possibility to inspect the contents of the form being sent to Adyen.

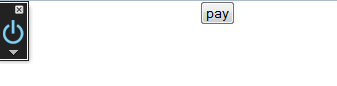


Select 'Adyen' as the payment method on Billing page of checkout process and click the 'CONTINUE' button:

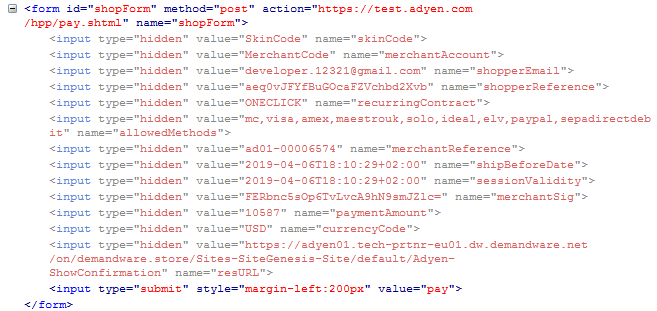


You will see the Order Confirmation page and can press the 'SUBMIT ORDER' button.

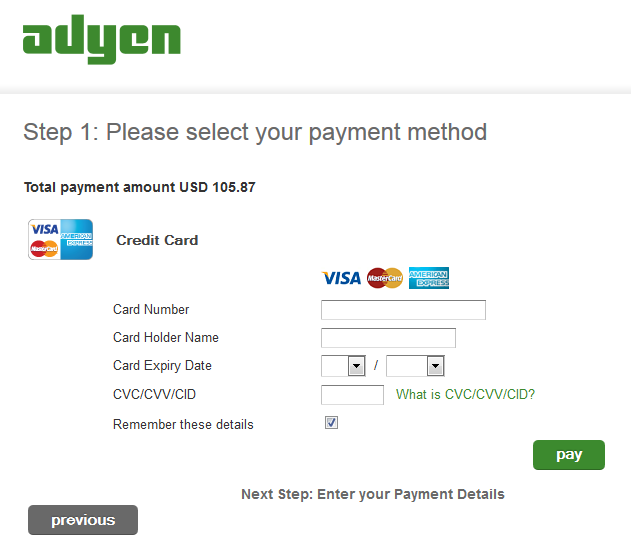
If debug is enabled, you will see an empty page with a small button

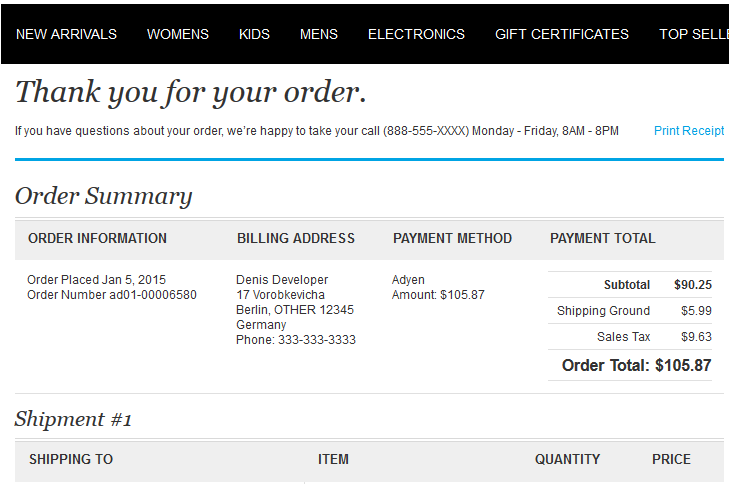


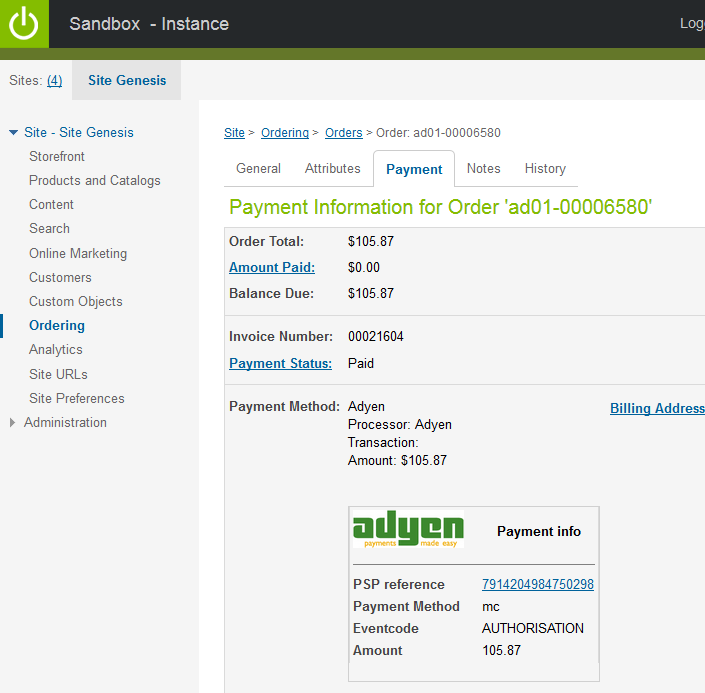
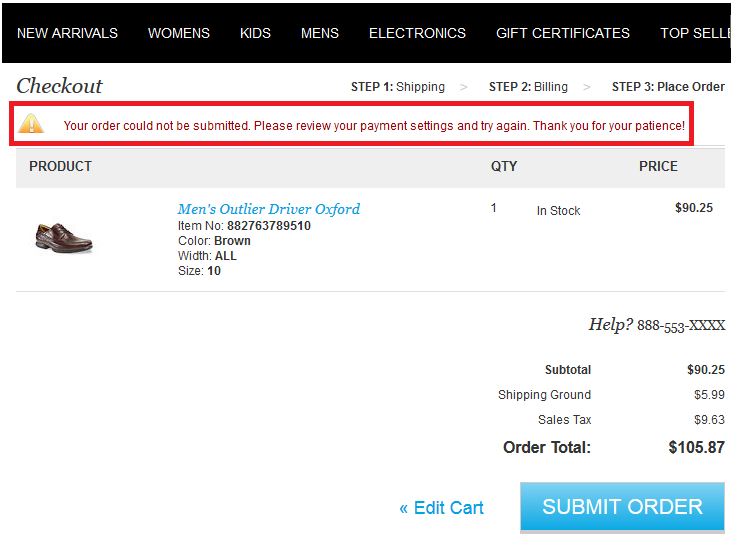
Additionally you will be able to check the form that will be submitted to Adyen via Firebug or an equivalent tool, or just by checking the source code of the page



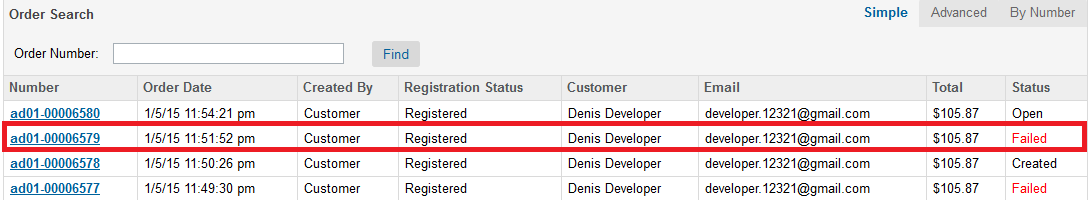
If debug is not enabled, you will be redirected to Adyen HPP immediately, and will see a list of payment methods (the list depends on the chosen countries and the settings you applied)

  
  
After the successful payment authorization, you will be redirected back to the Demandware Order Summary page (should be configured in the Adyen back office):

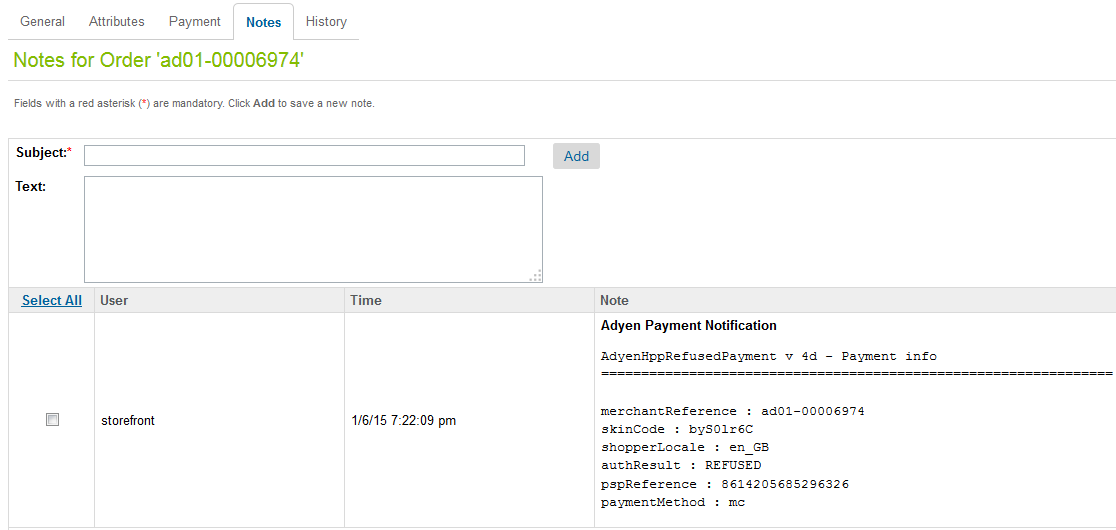


You can check the result of the payment authorization in Business Manager. On the payment tab of the order detail screen, the payment status should be Paid and the Adyen Eventcode should be AUTHORISATION (see the screenshot below):  
  
If the payment authorization was refused, you will be redirected back to the Demandware Order Confirmation page and an error message will be shown:  


In Business Manager, the order will be marked as failed in this case:

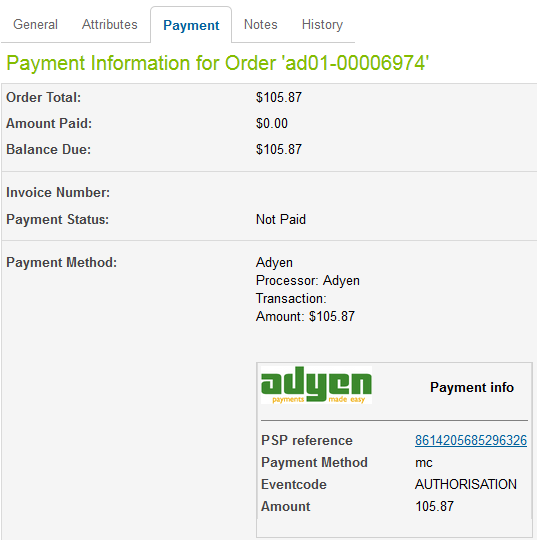


Orders can fail for many reasons, and in order to check that the payment was refused, you should check the notifications tab of the order details page. If the Adyen payment method was used, the notification should be similar to the one provided below:



There may be a few notifications, you should check the newest one.

Refused payments always have the Eventcode equals to AUTHORISATION, regardless of the payment method used:

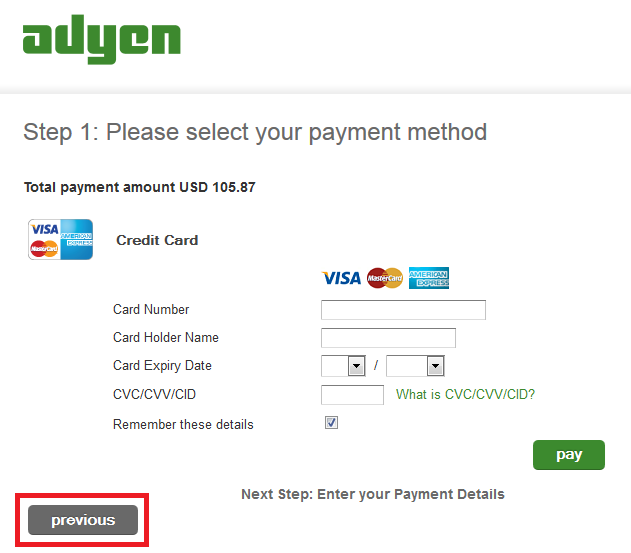


In the Adyen back office, refused payments always have the appropriate status:



### 2.2.2 Use Case: Cancel Payment on Adyen HPP

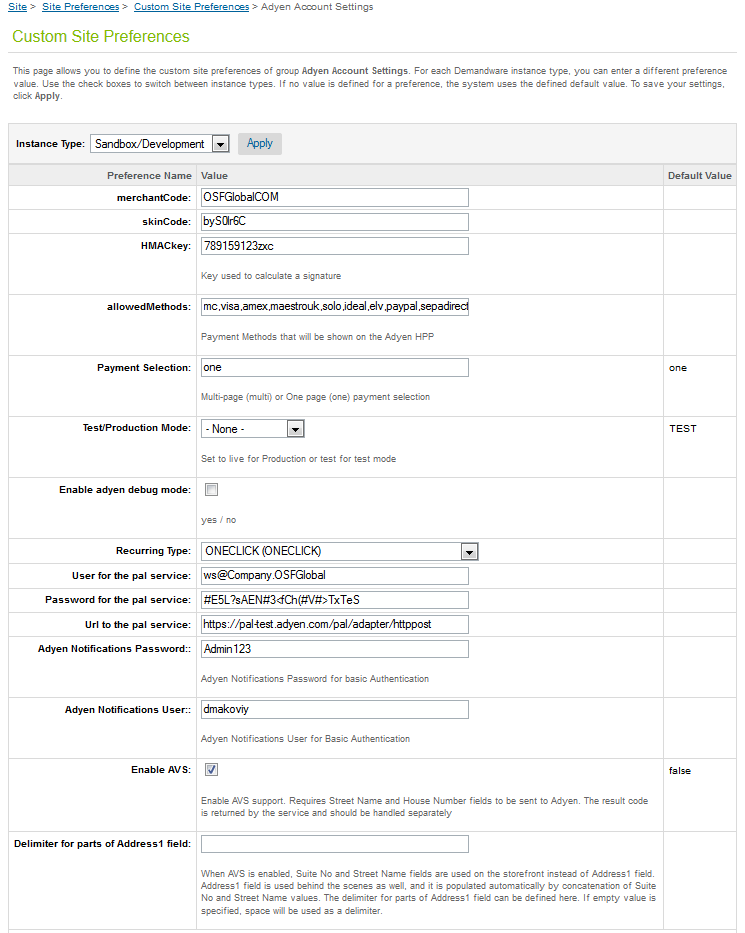
If you click on the button ‘previous’ on the Adyen HPP, you will be redirected back to the Demandware order confirmation page. It is a standard order confirmation page, you can to continue work with the site as usually you may return to any step of checkout process or go back to Adyen HPP by clicking ‘SUBMIT ORDER’ button.



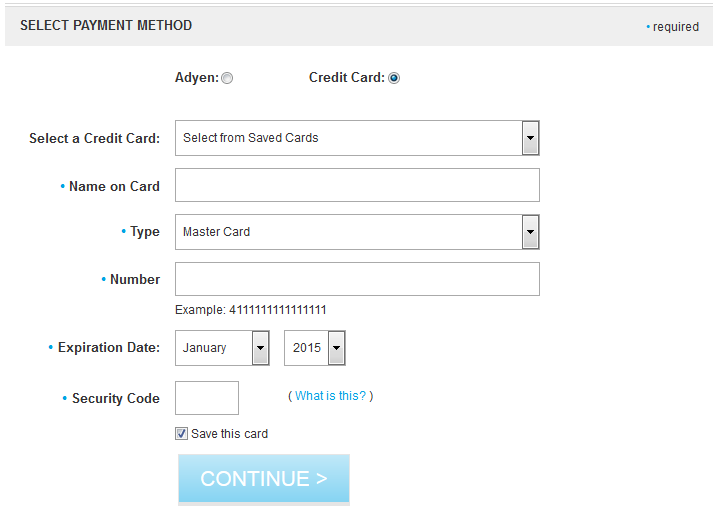
Every time when you go to the Adyen HPP, a new order is created in BM, and when you return back to Demandware, the order status will be failed. There is no any additional info in BM about the fact that the payment was cancelled. The payment does not appear in the Adyen back office if it was cancelled in this way.

### 2.2.3 Use Case: Credit Card

This use case describes the credit card payments, when the card details are entered on the Demandware side and are sent to Adyen via the Adyen API and PAL adapter. Please make sure the settings for Adyen PAL adapter are correct.

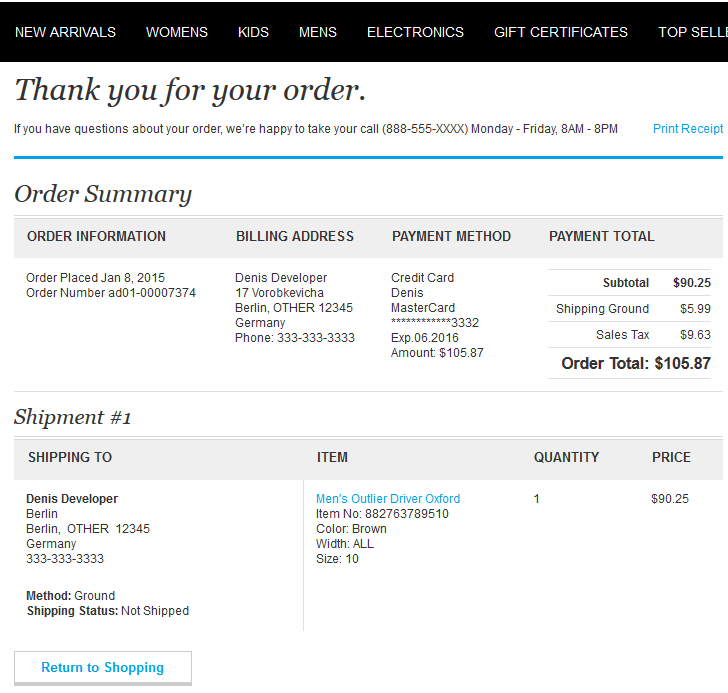


Enter the credit card details on the billing page of the checkout process and click the 'CONTINUE' button:

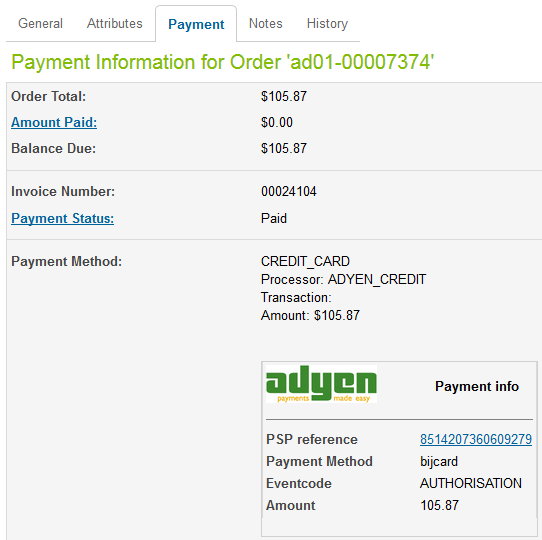


You will see the order confirmation page. The credit card details will be shown on the lower right of the page. You can press the 'SUBMIT ORDER' button in order to complete the purchase.

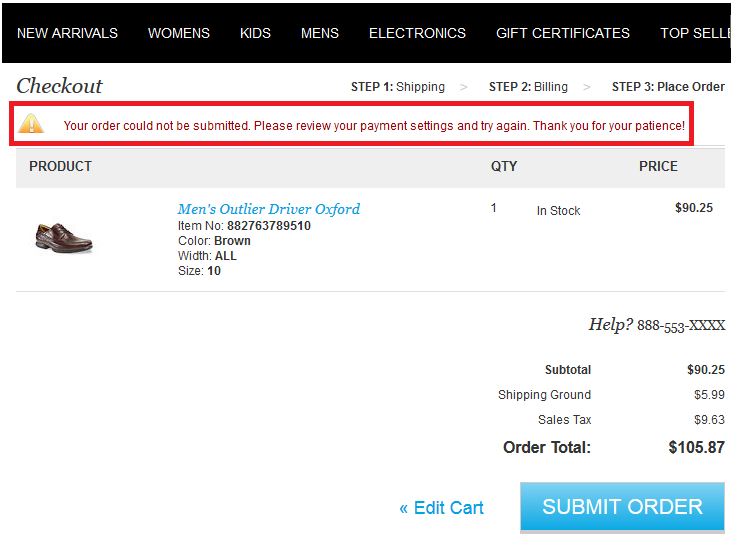
If the payment authorization was successful, you will see the 'Order Summary' page with information about the payment:



You can check the result of the payment authorization in Business Manager. On the payment tab of the order details screen, the payment status should be Paid and the Adyen Eventcode should be AUTHORISATION:



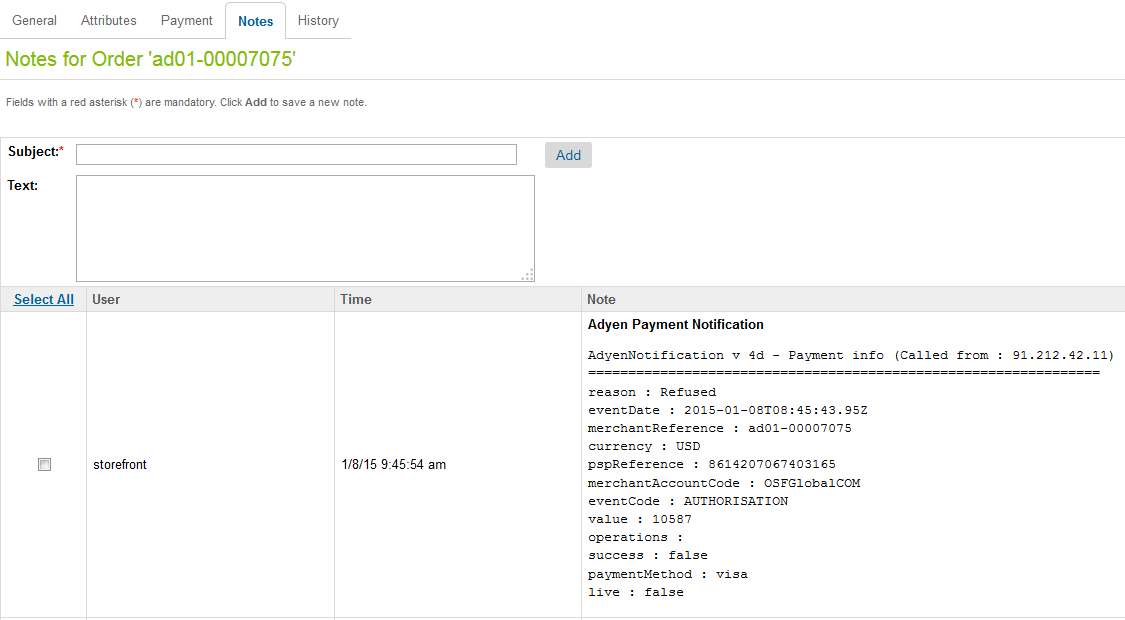
If payment authorization was refused, you will see the order confirmation page where the following error message will be displayed:



In Business Manager, the order will be marked as failed in this case:

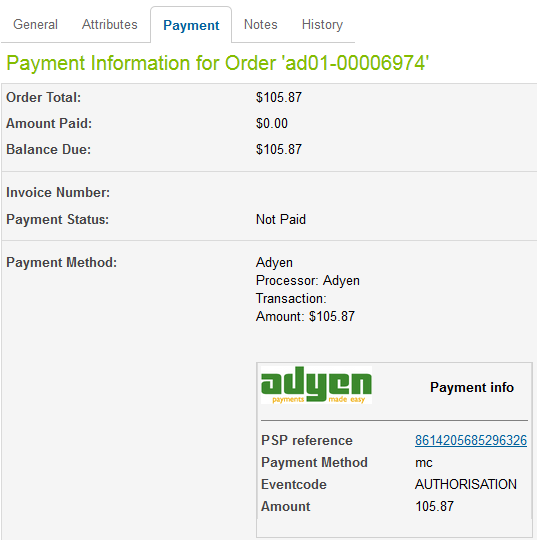


Orders can have the failed status for many reasons, and in order to check that a payment was refused, you should check the notifications tab of the order detail page. If the credit card payment method was used, the notification should be similar to the one provided below:



There may be a few notifications, you should check the newest one.

Refused payments always have an Eventcode that equals to AUTHORISATION, regardless of the payment method used:



In the Adyen back office, refused payments always have the appropriate status:



### 2.2.4 Use Case: 3-D Secure Credit Card

3-D Secure (Verified by VISA / MasterCard SecureCode™) is an additional authentication protocol that involves the shopper being redirected to their card issuer where they authenticate themselves before the payment can proceed to an authorization request.

You can find the 3-D Secure test credit card numbers here: <https://www.adyen.com/home/support/knowledgebase/implementation-articles.html?article=kb_imp_17>

For placing an order you have to follow the same steps like usual, the difference is that before you will be directed to the order confirmation page you will be prompted to enter your 3-D Secure credit card credentials.

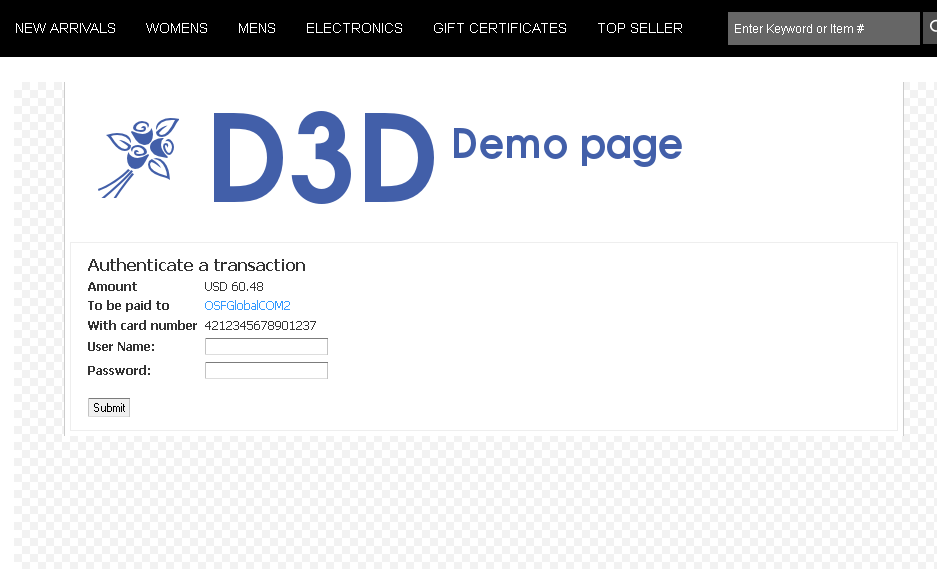
The order payment details can be found on the “Payment” tab from the order details page in Business Manager > Site > Ordering > Orders > Orders. Here you can find the information in the Adyen payment info section the PSP reference number with a link for the corresponding payment entry stored in the Adyen Customer Area (CA).

In the CA you can find the details about the payments in general, but also specific information about 3-D Secure payments like 3-D Secure Offered (yes or no) and 3-D Secure Authenticated (yes or no).

**Note.** For 3-D Secure, no additional changes of the storefront are required.

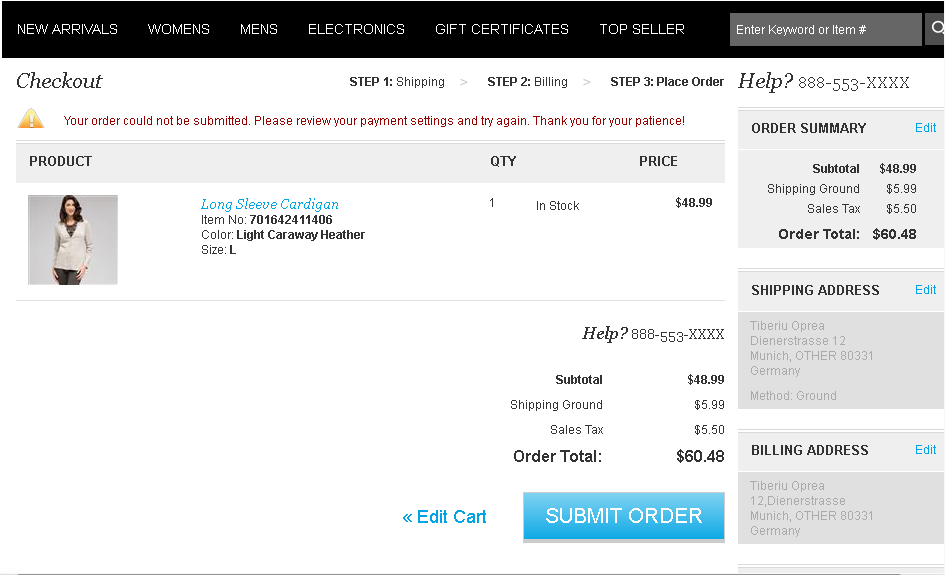
This use case describes the 3-D Secure credit card payments, when the card details are entered on Demandware side and are sent to Adyen via the Adyen API and PAL adapter. Make sure the settings for Adyen PAL adapter are correct.

Navigate to the place order page following the usual credit card checkout steps. Click on the 'SUBMIT ORDER' button to be directed to the 3-D Secure Authentication Page.



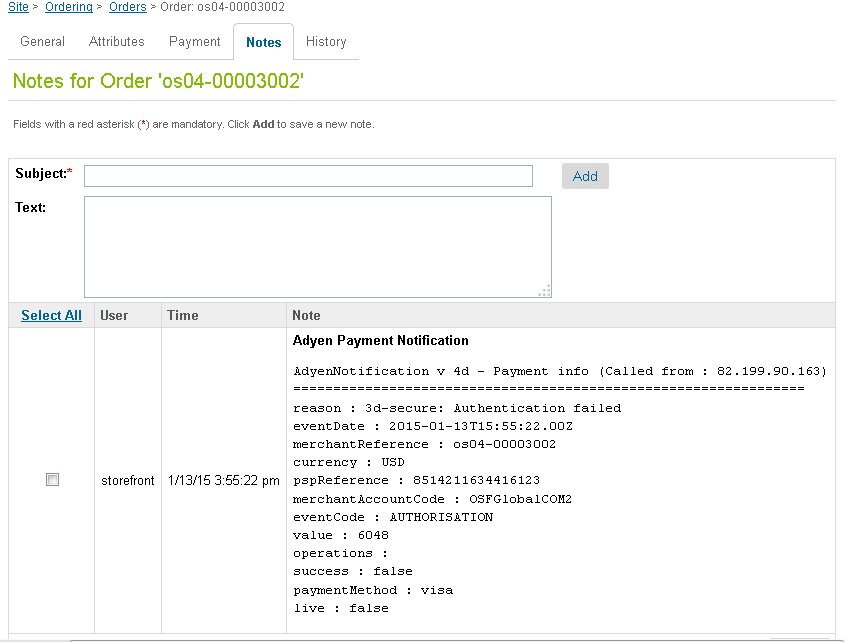
You have to enter your 3-D Secure credit card credentials there.

If you enter the correct credentials in the 3-D Secure authentication page, you will be directed to the order confirmation page. If these credentials are wrong you will be redirected back to the place order page and you will see an error message on top.



In Business Manager the order status will be failed in this case.

Orders can have the failed status for many reasons. In order to check that a payment was refused, you should check the notifications tab of order details page. If the credit card payment method was used, the notification should be similar to the one provided below:



There may be a few notifications, you should check the newest one.

Refused payments always have the Eventcode that equals to AUTHORISATION, regardless of the payment method used.

In Adyen back office, you can find details about payments in general, but also specific information about 3-D Secure payments like 3-D Secure Offered (yes or no) and 3-D Secure Authenticated (yes or no).

Refused payments always have the appropriate status:

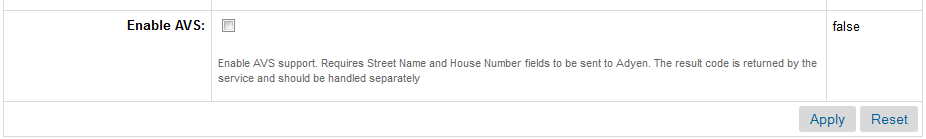
refusedCA

### 2.2.5 Use Case: AVS

#### 2.2.5.1 How to enable/configure AVS on Business Manager

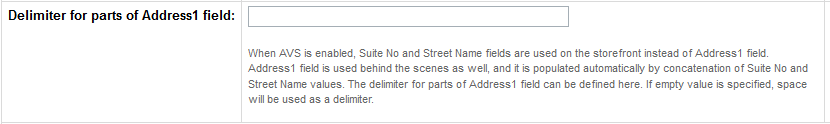
###### How to enable/disable AVS

1. Go to BM
2. Select a site
3. Go to Site Preferences > Custom Preferences > Adyen
4. Check/uncheck Enable AVS flag



###### Other configurations

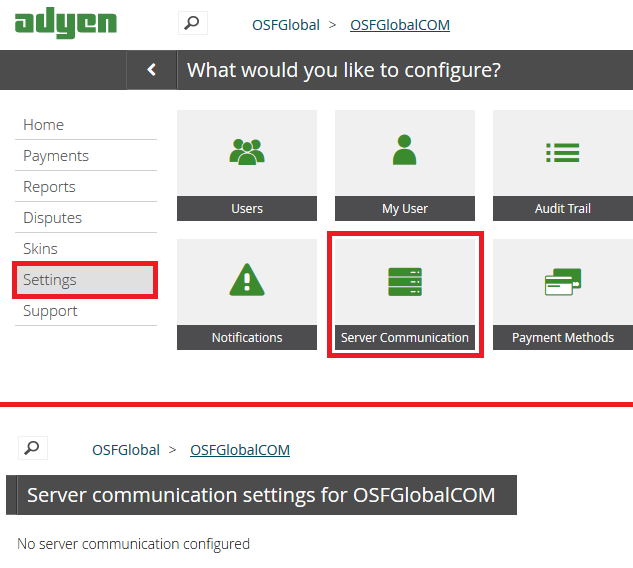
There is a field ‘Delimiter for parts of Address1 field’ that you can use for configuring the handling of address fields. There is a comment to this field in BM in which the meaning of the field is described, also this field is described in one of the next sections of this document.



###### Adyen Back Office

When billing details are sent to Adyen, the AVS result code is defined, and stored internally in the payment record, without any extra configurations.

Adyen can also send the AVS result code back to Demandware site, which may be needed for some site functionality. This possibility is disabled by default. In order to enable AVS, the server communication section has to be used and an appropriate flag has to be set there. This section is disabled by default and in order to enable it, you have to contact Adyen support. Also, when you use server communication, the notifications section has to be turned off, otherwise notifications will be used instead.



##### General explanation

1. In order to enable/disable AVS on the site, you have to select/deselect the flag in BM
2. In order to have an AVS code returned back from Adyen to the Demandware site, you have to configure Adyen back office properly
3. The settings in the Adyen back office mean only that AVS is allowed/disallowed to be used, but they do not require to use it, only the flag in BM turns on/off AVS

#### 2.2.5.2 AVS use cases

##### AVS is disabled

###### Storefront

Storefront works as usually, there are no visible changes for user

###### Business Manager

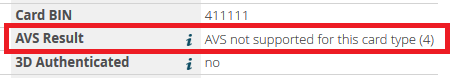
BM works as usually, there are no visible changes for merchant

###### Adyen Back Office

There is a field AVS Result on payment page, where you can check the result code.

There are 2 types of credit cards from an AVS standpoint: cards which support AVS and cards that do not support AVS.

1. If AVS is not supported for credit cards, an appropriate message will be shown in Adyen back office

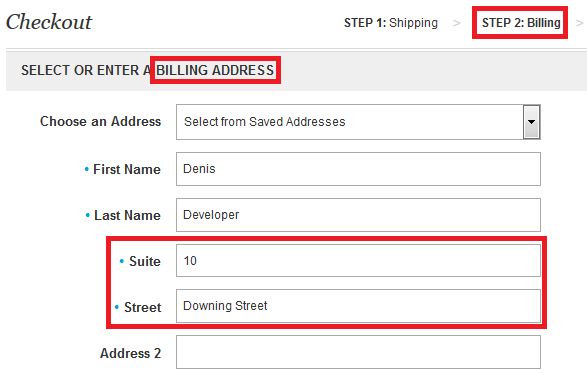


1. If AVS is supported for credit cards, the message will be the following



##### AVS is enabled

###### Storefront

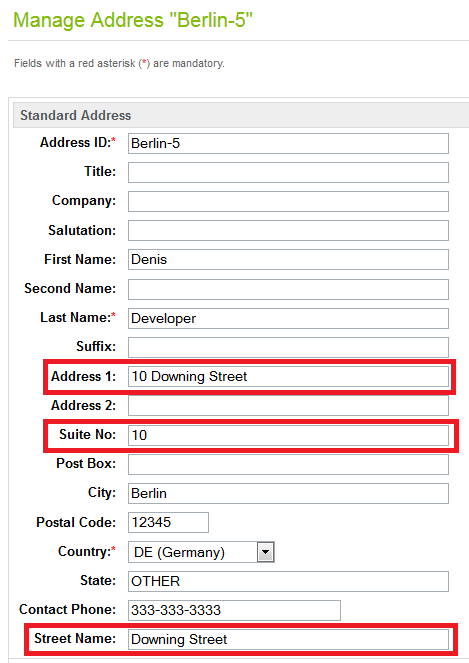
When AVS is enabled, there should appear two new fields instead of one field called Address1 on all site forms: Suite and Street. These two new fields should appear on all site forms where the user enters his address data. In the current implementation of the Adyen cartridge, only the billing address form from the checkout process is updated to meet this requirement, the other forms are remained untouched 

The billing form should work normally with two new fields. The selection of saved addresses from a dropdown list, storing of new addresses and the other functions should normally work on the standard Site Genesis site.

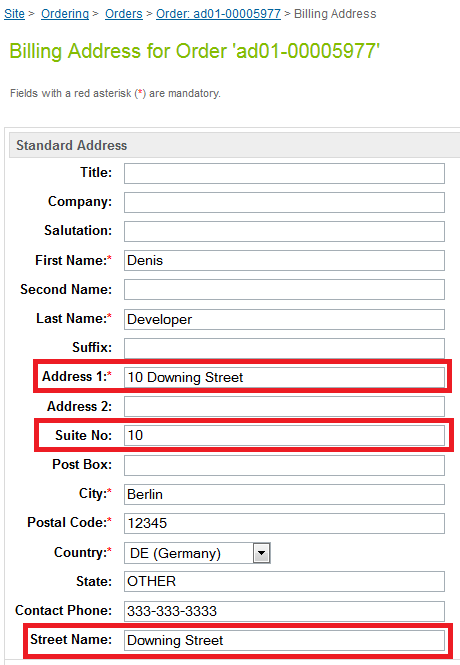
###### Business Manager

When AVS is enabled, two new fields are used for storing address details: Suite No and Street Name. There are three different places where address data is stored: Customer addresses, order billing and shipping addresses. In the current cartridge implementation, only customer address and order billing address are configured to be used/updated by the cartridge.

1. Customer address



1. Order billing address



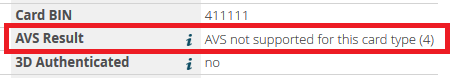
The logic of the address fields usage and update is like this: the values of Suite and Street fields which the user updates on the storefront are stored in appropriate fields in BM, but they are also concatenated and stored in field Address1. The concatenation of the values is performed behind the scenes, the delimiter for concatenated strings can be changed in Site Preferences, the concatenation of the values is required in order to make the Adyen cartridge compatible with other site functionality and not to interfere with other site functionality.

###### Adyen Back Office

There is a field AVS Result on the payment page, where you can check the result code.

There are two different types of credit cards from an AVS standpoint: cards for which AVS is supported and for which AVS is not supported.

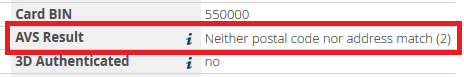
1. If AVS is not supported for credit card, an appropriate message will be shown in Adyen back office



1. If AVS is supported for credit card, one of possible result codes will be shown. Here are a few examples:
2. Successful address verification result



1. Not successful address verification result



#### Summary

1. The AVS result code is viewable for the merchant only in Adyen back office.
2. The AVS feature can be enabled in Business Manager.
3. If AVS is enabled, new fields will be shown on the storefront, otherwise there will not be any viewable changes for the user.
4. If AVS is enabled, new fields will be used for storing addresses in BM, but the old field Address1 will be used as well. If AVS is disabled, there will not be any changes visible for the merchant in BM.
5. The AVS result code can be received by the cartridge from the Adyen server, if an appropriate configuration of Adyen back office is done. If the AVS result code is received by the Adyen cartridge, it will not be shown to the merchant in the current implementation of the feature, the code can be handled only programmatically:
   1. The developer has a possibility to handle AVS result code programmatically if needed
   2. The merchant can see the AVS result code only in Adyen back office

## Limitations, Constraints

* The merchant will need a configured Adyen account
* The merchant can style the Adyen HPP (skin), to make it matching the look and feel of their store. See the Adyen skin manual or contact Adyen support for more info
* Successful payment authorizations lead to a 'Paid' order payment status and ‘Ready for Export’ export status
* Refused payment authorizations 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
* Refused payments have the status ‘Refused’ in Adyen back office, but ‘AUTHORISATION’ Eventcode in Business Manager
* In the cartridge these API methods are implemented: ‘Capture’, ‘Cancel Before Capture’, and ‘Cancel Or Refund’. The statuses require custom integration into the end site, they are optional functions and are not described in this document from a position of integration
* The AVS feature requires that all address forms of the site have two new fields ‘Building number’ and ‘Street name’ instead of the old field ‘Address 1’. When AVS is enabled, all site forms should have two new fields, otherwise either the new or the old approach can be used. Also, ‘Address 1’ field has to be updated behind the scenes in order to not break existing site functionality. In this guide it is described how to change Billing address form, it’s an example of how should work all the other address forms of your site

## Compatibility

Based on DW version 13.3.1 and SiteGenesis 2.0

## Privacy, Payment

* If only the redirect method is used, all payment data is entered into the Adyen HPP by the shopper and no Credit Card data will be stored inside Demandware (except the brand of used card)
* The shoppers email address and shopper ID used in Demandware are sent to Adyen to allow ‘recurring’ and/or 'one click' payments

# Implementation Guide

## Setup

### 3.1.1 Installation

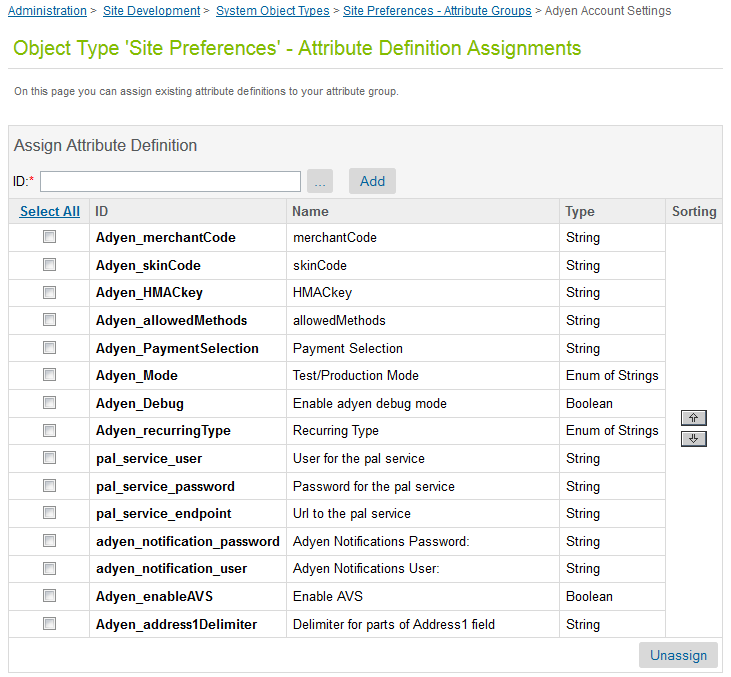
Install ‘int\_adyen’ and ‘bm\_adyen’ cartridges from the distributive zip-archive in a regular way using Demandware UX-studio.

### 3.1.2 Metadata Import

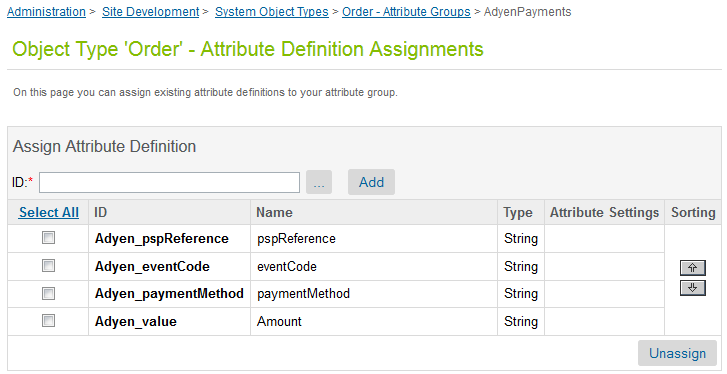
Find the archive ‘site-template.zip’ in the installation package, it contains metadata files. Please review the archive, do the necessary modifications, and import it through BM Administration > Site Development > Site Import & Export section. The only required modification is renaming the root folder of the archive to the ID of your site.

Before the import of ‘site-template.zip’, check and save cartridge paths of your site and Business Manager, because they will be modified, so you will be able to configure them manually if something goes wrong. Also, check the fields which will be updated after importing file ‘site-template.zip’, and be sure that there won’t be any conflict with the existing fields.

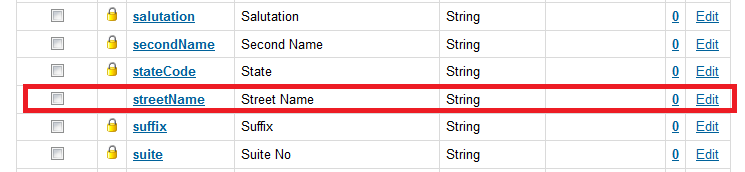
The following configuration items will be added after you import ‘site-template.zip’ archive:



Besides configuration parameters, the following order attributes will be added:



Also the new custom field ‘streetName’ to ‘CustomerAddress’ and ‘OrderAddress’ system objects will be added:

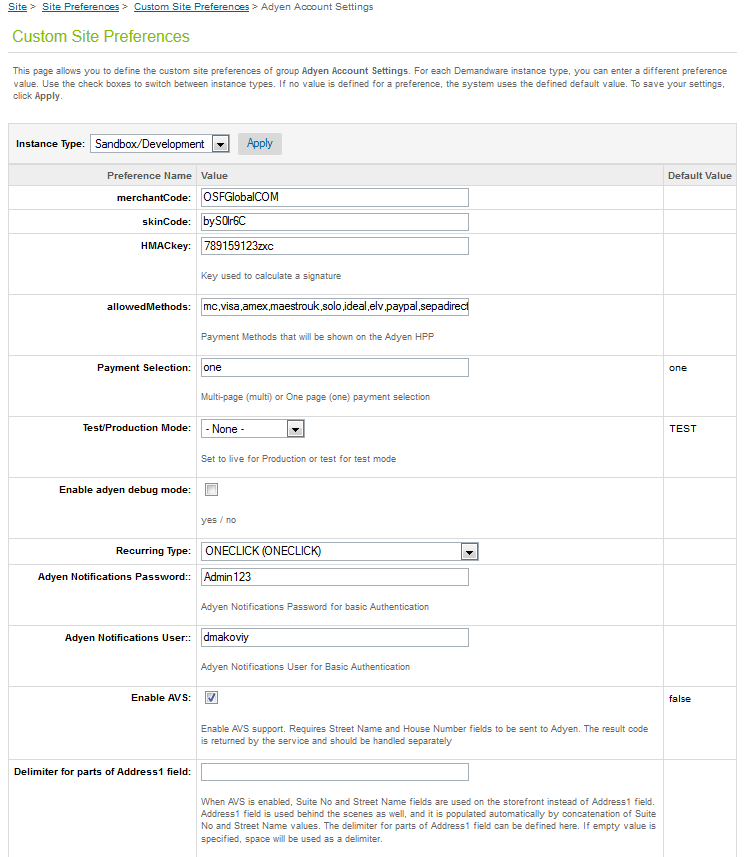


## Business Manager Configuration

In order to use Adyen service for processing credit cards you have to enable ‘Credit Card’ payment method and set payment processor to ‘ADYEN\_CREDIT’ for it.

In order to use ‘Adyen redirect’ feature, ‘Adyen’ payment method must be enabled.

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



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

|  |  |
| --- | --- |
| **Parameter Name** | **Description** |
| **merchantCode** | Adyen merchant account name. The account for which the payments will be processed. |
| **skinCode** | Code of the skin which you want to use. There can be defined few skins, each having a different configuration, style, language etc. |
| **HMACkey** | Secret key used to calculate a signature over the payment fields sent to Adyen. The value must be the same in Site Preferences and in Adyen Account Settings |
| **allowedMethods** | Payment Methods that will be shown on the Adyen HPP |
| **Payment Selection** | Multi-page (multi) or One page (one) payment selection |
| **Test/Production Mode** | Define the live (Production) or test mode |
| **Enable adyen debug mode** | Enables debug mode which shows a 'pay' button before doing the redirect to Adyen. This allows you to check the parameters sent to the HPP |
| **Recurring Type** | Either RECURRING, or ONECLICK or RECURRING,ONECLICK |
| **Adyen Notifications User** | The username used by the Adyen notification system in order to authenticate the calls with Demandware. Should be the same as the one set in Adyen system |
| **Adyen Notifications Password** | The password by the Adyen notification system in order to authenticate the calls with Demandware. Should be the same as the one set in Adyen system |

**Services Configuration**

Go to **Administration > Operations > Services** and configure Adyen services by entering PAL details: url, username and password

## Adyen Account Configuration

Here are some items that need to be configured in the Adyen Customer Area (CA) for the Adyen Integration to work correctly. You can find the CA at <https://ca-test.adyen.com>.

|  |  |
| --- | --- |
| **Parameter Name** | **Description** |
| **HMAC key** | Secret key used to calculate a signature over the payment fields sent to Adyen. This should match what is used in the Adyen configuration settings in your DW site |
| **SkinCode** | Code of the skin you want to use. You can use different skins, each having a different configuration, style, language etc. |
| **Notification URL and test page** | URL where Adyen should send notification messages to. The script at this URL should always reply with [accepted], otherwise Adyen will keep sending messages. The URL will point to your DW store e.g.:  https://adyen01.tech-prtnr-eu01.dw.demandware.net/on/demandware.store/Demos-SiteGenesis-Site/default/Adyen-Notify  The Notification URL and it's response can be tested from within the CA |
| **Payment Methods** | You can select which payment methods should be displayed by configuring your skin. This can also be influenced with the 'allowedMethods' parameter that can be sent to Adyen for each payment |
| **Test HPP and skin** | The skin and HPP can be tested from within the CA, to check the look and feel, translations, configured payment methods etc. |
| **Settings for the PAL adapter** | To send Credit Card payments directly (without redirecting to the HPP) you can use the PAL adapter. The user, password and location can be configured here |
| **Notifications User Name** | The username used by the Adyen notification system in order to authenticate the calls with Demandware. Should be the same as the one set in Demandware system under “Adyen Notifications User” preference. (Found under : Settings -> Notifications -> User Name) |
| **Notifications Password** | The password used by the Adyen notification system in order to authenticate the calls with Demandware. Should be the same as the one set in Demandware system under “Adyen Notifications Password” preference. . (Found under : Settings -> Notifications -> Password) |

## Integration Efforts

The following steps are needed to complete the integration:

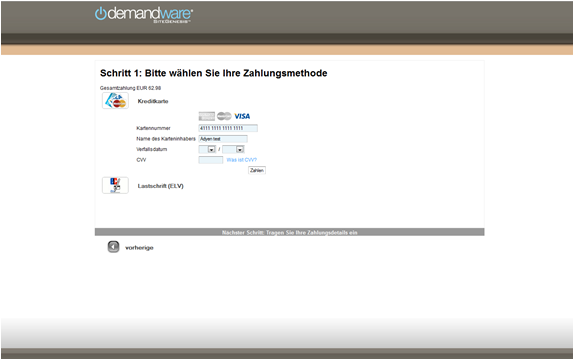
* Install the cartridges
* Change and import ‘site-template.zip’ (system objects)
* Do the required code changes
* Configure your merchant's Adyen Account in the Adyen back office
* Configure Adyen parameters in Demandware
* Test
* Create Skin for Adyen HPP

These steps should not require more than a few hours (creating the Skin can require several hours depending on complexity of styling).

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 'Adyen' and 'Credit Card' and the payment processors 'Adyen' and ‘ADYEN\_CREDIT’ will be created or updated from the ‘site-template.zip’.

Below is an example of a skin styled to look like the old Site Genesis store (note that this page is hosted on an Adyen Server):



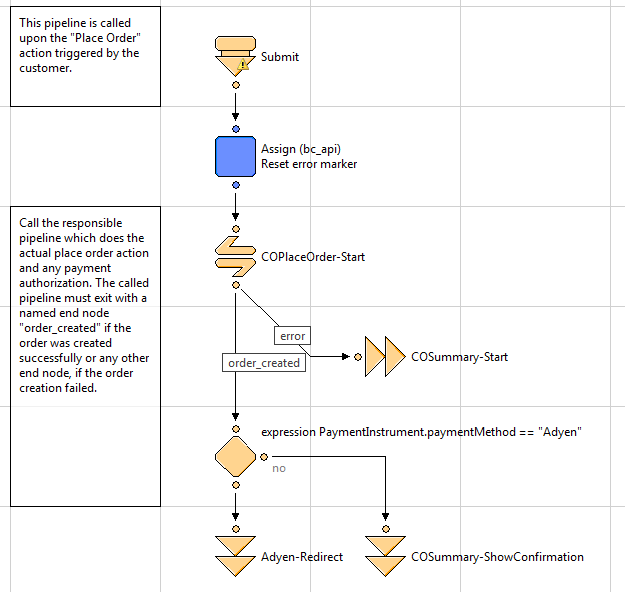
## Changes to pipelines, templates and others

### 3.5.1 COSummary pipeline

When the Adyen payment method is used, instead of showing an order confirmation page, the shopper has to be redirected to Adyen using the Adyen-Redirect pipeline.

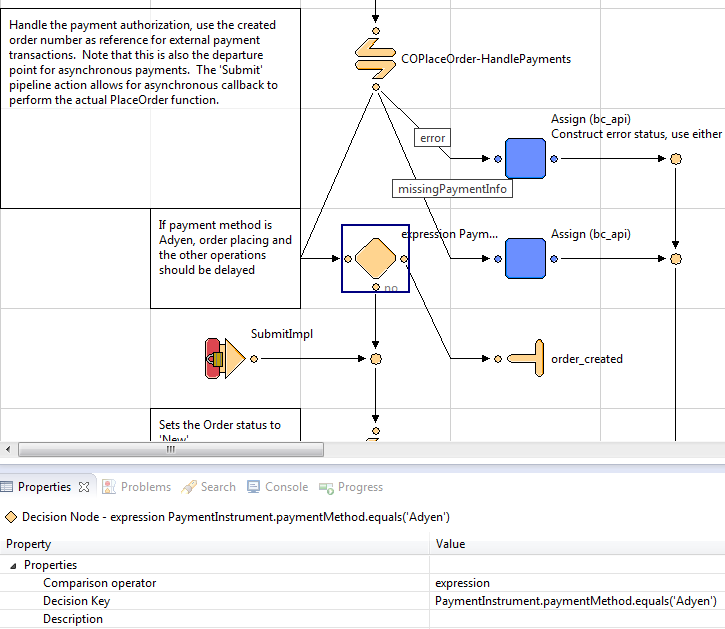
The order summary page is shown immediately only if the amount is covered by a Gift Certificate.

Change the Submit pipeline so it will look like the one below:



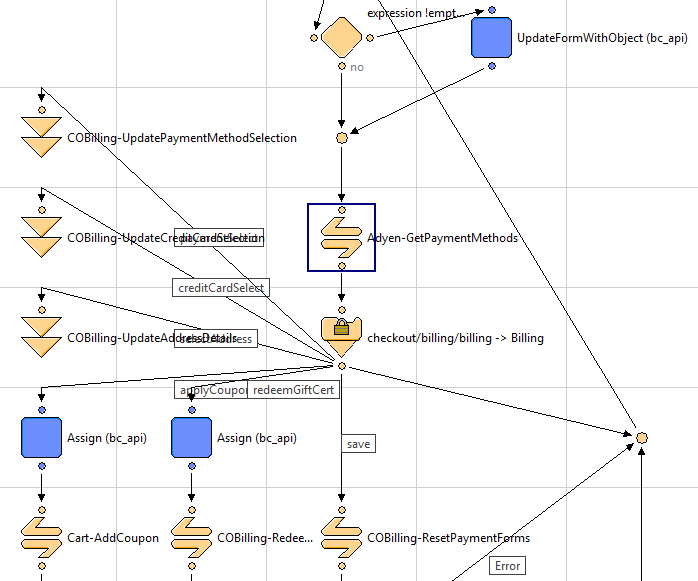
### 3.5.2 COPlaceOrder pipeline

Add a decision node to the pipeline like shown below:



### 3.5.3 COBilling pipeline

Add a call node to COBilling-Start pipeline right before interaction continue node, the node has to call the Adyen-GetPaymentMethods pipeline:



### 3.5.4 Form billingaddress.xml

Increase the max length of field address1 and add 2 new fields to the form, here is the expected result:

<field formid="address1" label="forms.address1" type="string" mandatory="true" binding="address1" max-length="105" missing-error="forms.addresserror" range-error="forms.address.field.invalid" />

<field formid="suite" label="forms.suite" type="string" mandatory="true" binding="suite" max-length="50" missing-error="forms.suiteerror" range-error="forms.suite.field.invalid" />

<field formid="streetName" label="forms.streetName" type="string" mandatory="true" binding="custom.streetName" max-length="50" missing-error="forms.streetnameerror" range-error="forms.streetname.field.invalid" />

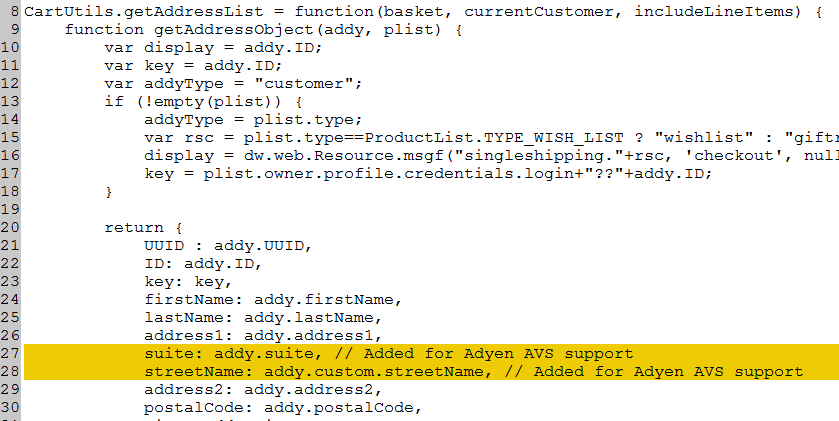
### 3.5.5 Script cart/CartUtils.ds

Find method CartUtils.getAddressList(), find function getAddressObject() inside of it, and add these 2 new fields to the returned object:

suite: addy.suite, // Added for Adyen AVS support

streetName: addy.custom.streetName, // Added for Adyen AVS support

Here is the expected result:



### 3.5.6 Script checkout/AddAddressToAddressBook.ds

Replace the line of code:

address.setAddress1( usedAddress.address1 );

on the following code:

// Added for Adyen AVS support. If AVS is enabled, 2 new fields are used on the site

if (dw.system.Site.getCurrent().getCustomPreferenceValue("Adyen\_enableAVS")) {

address.setSuite( usedAddress.suite );

address.custom.streetName = usedAddress.custom.streetName;

address.setAddress1( usedAddress.suite + (!empty(dw.system.Site.getCurrent().getCustomPreferenceValue("Adyen\_address1Delimiter")) ? dw.system.Site.getCurrent().getCustomPreferenceValue("Adyen\_address1Delimiter") : " ") + usedAddress.custom.streetName );

} else {

address.setAddress1( usedAddress.address1 );

}

### 3.5.7 Script checkout/UpdateBillingAddress.ds

Replace the line of code:

billingAddress.setAddress1( addressFields.address1.value );

on the following code:

// Added for Adyen AVS support. If AVS is enabled, 2 new fields are used on the site

if (dw.system.Site.getCurrent().getCustomPreferenceValue("Adyen\_enableAVS")) {

billingAddress.setSuite( addressFields.suite.value );

billingAddress.custom.streetName = addressFields.streetName.value;

billingAddress.setAddress1( addressFields.suite.value + (!empty(dw.system.Site.getCurrent().getCustomPreferenceValue("Adyen\_address1Delimiter")) ? dw.system.Site.getCurrent().getCustomPreferenceValue("Adyen\_address1Delimiter") : " ") + addressFields.streetName.value );

} else {

billingAddress.setAddress1( addressFields.address1.value );

}

### 3.5.8 Template checkout/billing/billing.isml

Replace the line of code:

<isinputfield formfield="${pdict.CurrentForms.billing.billingAddress.addressFields.address1}" type="input"/>

on the following code:

<iscomment>Added for Adyen AVS support. If AVS is enabled, 2 new fields are used on the site</iscomment>

<isif condition="${dw.system.Site.getCurrent().getCustomPreferenceValue('Adyen\_enableAVS')}">

<isinputfield formfield="${pdict.CurrentForms.billing.billingAddress.addressFields.suite}" type="input"/>

<isinputfield formfield="${pdict.CurrentForms.billing.billingAddress.addressFields.streetName}" type="input"/>

<iselse>

<isinputfield formfield="${pdict.CurrentForms.billing.billingAddress.addressFields.address1}" type="input"/>

</isif>

### 3.5.9 Template checkout/billing/paymentmethods.isml

Add the following code right before the closing </fieldset> tag:

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

<isinclude template="hpp"/>

</div>

### 3.5.10 Template checkout/summary/summary.isml

Add the following code right before the closing </fieldset> tag:

<iscomment>

Set the brandcode and issuerId in session for when user hits the back button on Adyen hpp

</iscomment>

<isif condition="${!empty(pdict.CurrentHttpParameterMap.brandCode.value) || !empty(session.custom.brandCode)}">

<isset name="brandCode" value="${!empty(pdict.CurrentHttpParameterMap.brandCode.value) ? pdict.CurrentHttpParameterMap.brandCode.value : session.custom.brandCode}" scope="session"/>

<input type="hidden" name="brandCode" value="${session.custom.brandCode}" />

<isif condition="${!empty(pdict.CurrentHttpParameterMap.issuerId.value) || !empty(session.custom.issuerId)}">

<isset name="issuerId" value="${!empty(pdict.CurrentHttpParameterMap.issuerId.value) ? pdict.CurrentHttpParameterMap.issuerId.value : session.custom.issuerId}" scope="session"/>

<input type="hidden" name="issuerId" value="${session.custom.issuerId}" />

</isif>

</isif>

### 3.5.11 Template components/header/htmlhead.isml

Add the following code after including file style.css:

<link rel="stylesheet" href="${URLUtils.staticURL('/css/checkout.css')}" />

### 3.5.13 Script billing.js

1. Add the following function method:

/\*\*

\* @function

\* @description Changes the payment type or issuerId of the selected payment method

\* @param {String, Boolean} value of payment type or issuerId and a test value to see which one it is, to which the payment type or issuerId should be changed to

\*/

function updatePaymentType(selectedPayType, test) {

if (!test) {

$('input[name="brandCode"]').removeAttr('checked');

} else {

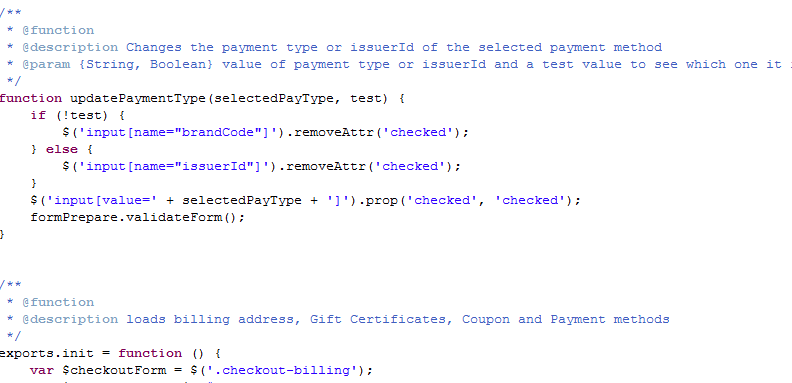
$('input[name="issuerId"]').removeAttr('checked');

}

$('input[value=' + selectedPayType + ']').prop('checked', 'checked');

formPrepare.validateForm();

}



1. In the exports.init function, after this code:   
   var selectedPaymentMethod = $selectPaymentMethod.find(':checked').val();
2. Add the next variable definitions:

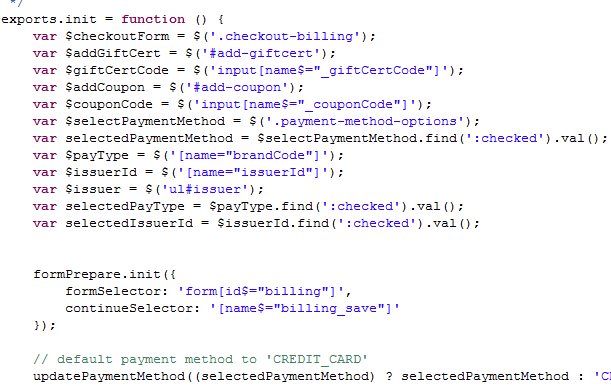
var $payType = $('[name="brandCode"]');

var $issuerId = $('[name="issuerId"]');

var $issuer = $('ul#issuer');

var selectedPayType = $payType.find(':checked').val();

var selectedIssuerId = $issuerId.find(':checked').val();



1. Change method $selectPaymentMethod.on('click', 'input[type="radio"]', function () {, add the following code to it:

if ($(this).val() == 'Adyen' && $payType.length > 0) {

//set payment type of Adyen to the first one

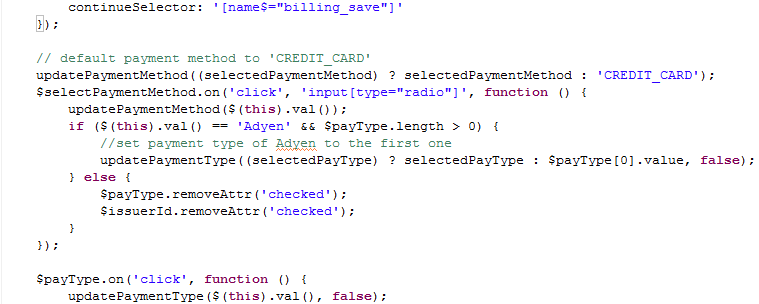
updatePaymentType((selectedPayType) ? selectedPayType : $payType[0].value, false);

} else {

$payType.removeAttr('checked');

$issuerId.removeAttr('checked');

}



1. Before this line:

// select credit card from list  
$('#creditCardList').on('change', function () {

Add the following code:

$payType.on('click', function () {

updatePaymentType($(this).val(), false);

//if the payment type contains issuerId fields, expand form with the values

if ($(this).siblings('#issuer').length > 0) {

$issuer.show();

updatePaymentType(

(selectedIssuerId) ? selectedIssuerId : $issuerId[0].value, true);

} else {

$issuer.hide();

$('input[name="issuerId"]').removeAttr('checked');

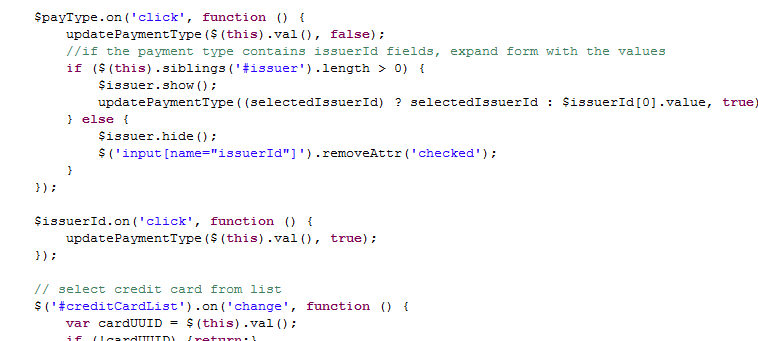
}

});

$issuerId.on('click', function () {

updatePaymentType($(this).val(), true);

});



### 3.5.14 File resources/forms.properties

Add the following lines to the file:

forms.suite=Suite

forms.streetName=Street

forms.suiteerror=Please enter Suite number

forms.streetnameerror=Please enter Street name

forms.suite.field.invalid=Please enter a valid value

forms.streetname.field.invalid=Please enter a valid value

adyentest.missedordernumber=Missed order number

adyentest.wrongordernumber=Wrong order number

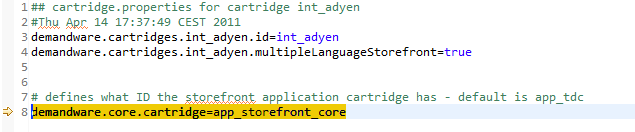
adyentest.ordernumber=Order number

## Additional configuration

You may need to change the ID of the storefront application. This can be defined in the file int\_adyen.properties in the root directory of the int\_adyen cartridge.

Example:  
# defines what ID the storefront application cartridge has - default is app\_tdc  
demandware.core.cartridge=SiteGenesisNew

Please open the 'int\_adyen.properties' file and edit the value to your storefront cartridge name to make the adyen cartridge work.



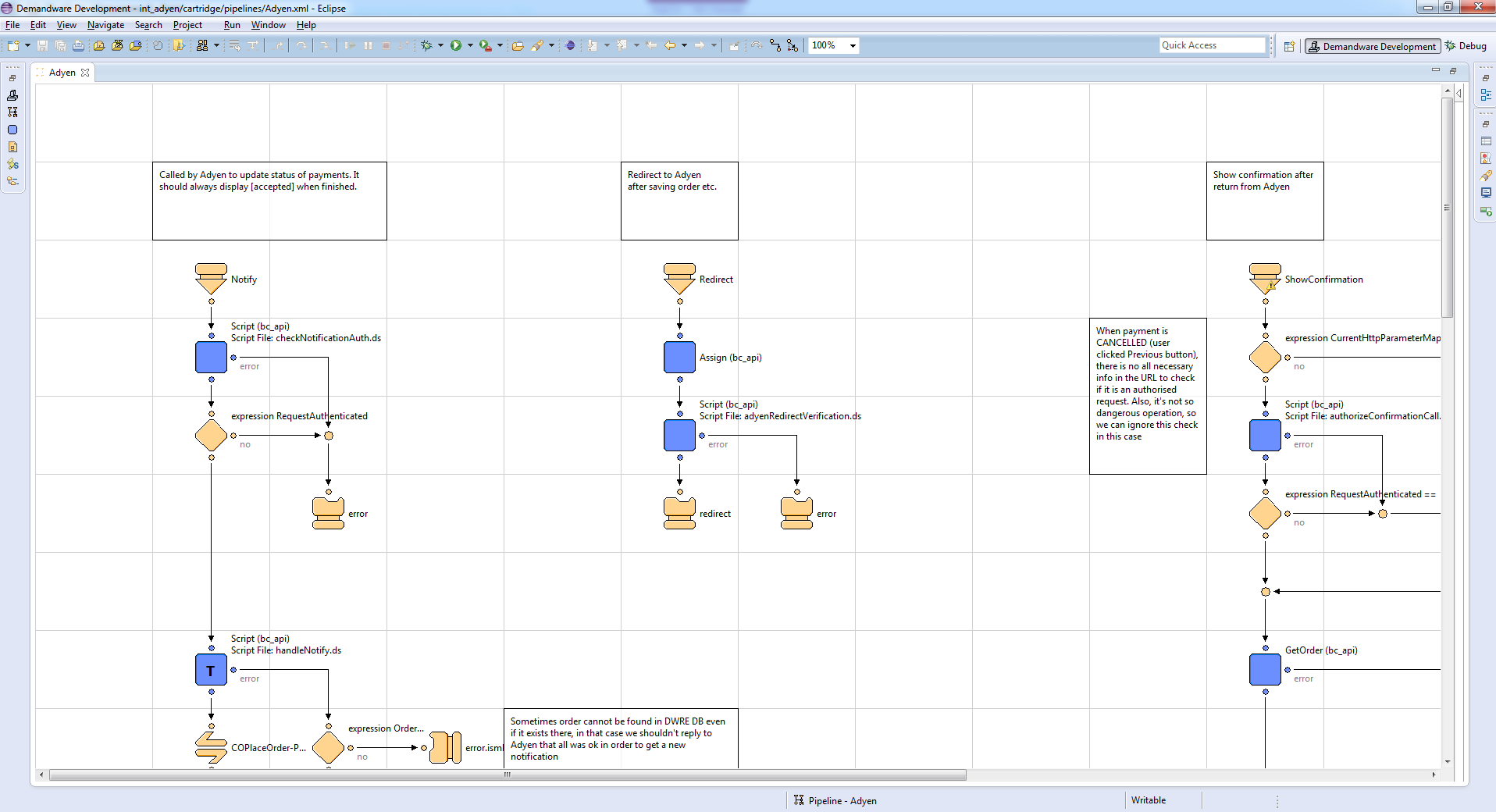
## Pipelines in the Adyen Cartridge

The cartridge contains pipelines to handle the redirect to Adyen, to process notifications, and to call Adyen API methods.

### 3.7.1 Adyen

Adyen redirect and the common functionality is gathered in Adyen pipeline.

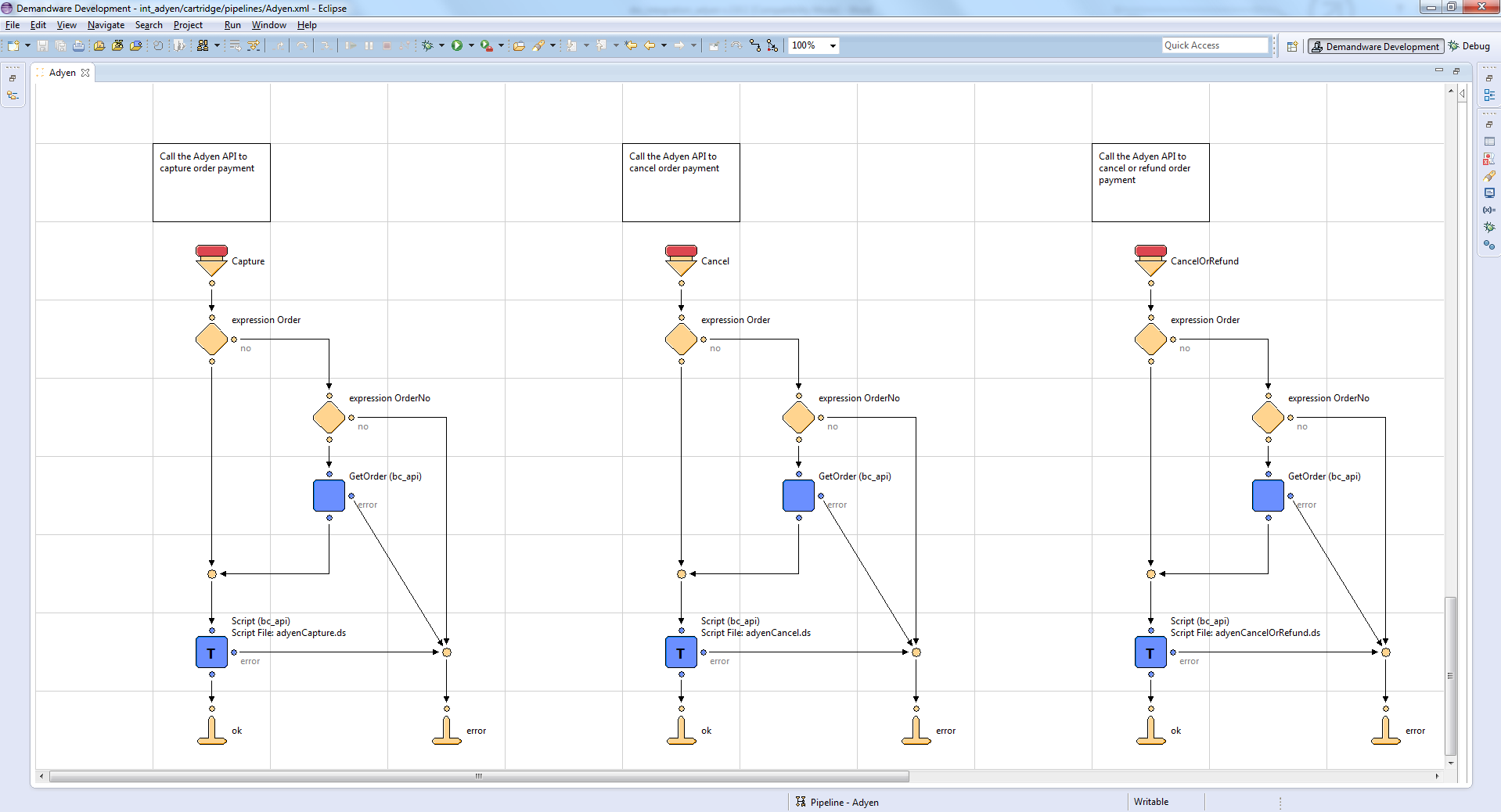
The COSummary-ShowConfirmation pipeline has been recreated here and will be called by Adyen when the shopper returns. You will not need to change anything here, unless your site has some special custom logic in this place.



#### IP check

You can enhance the security of the notifications delivery by checking the IP address of the incoming call against the IP address of the Adyen servers. In the cartridge the script ipCheck.ds can be used for that – you should insert it in the Notify pipeline (after the Start Node).

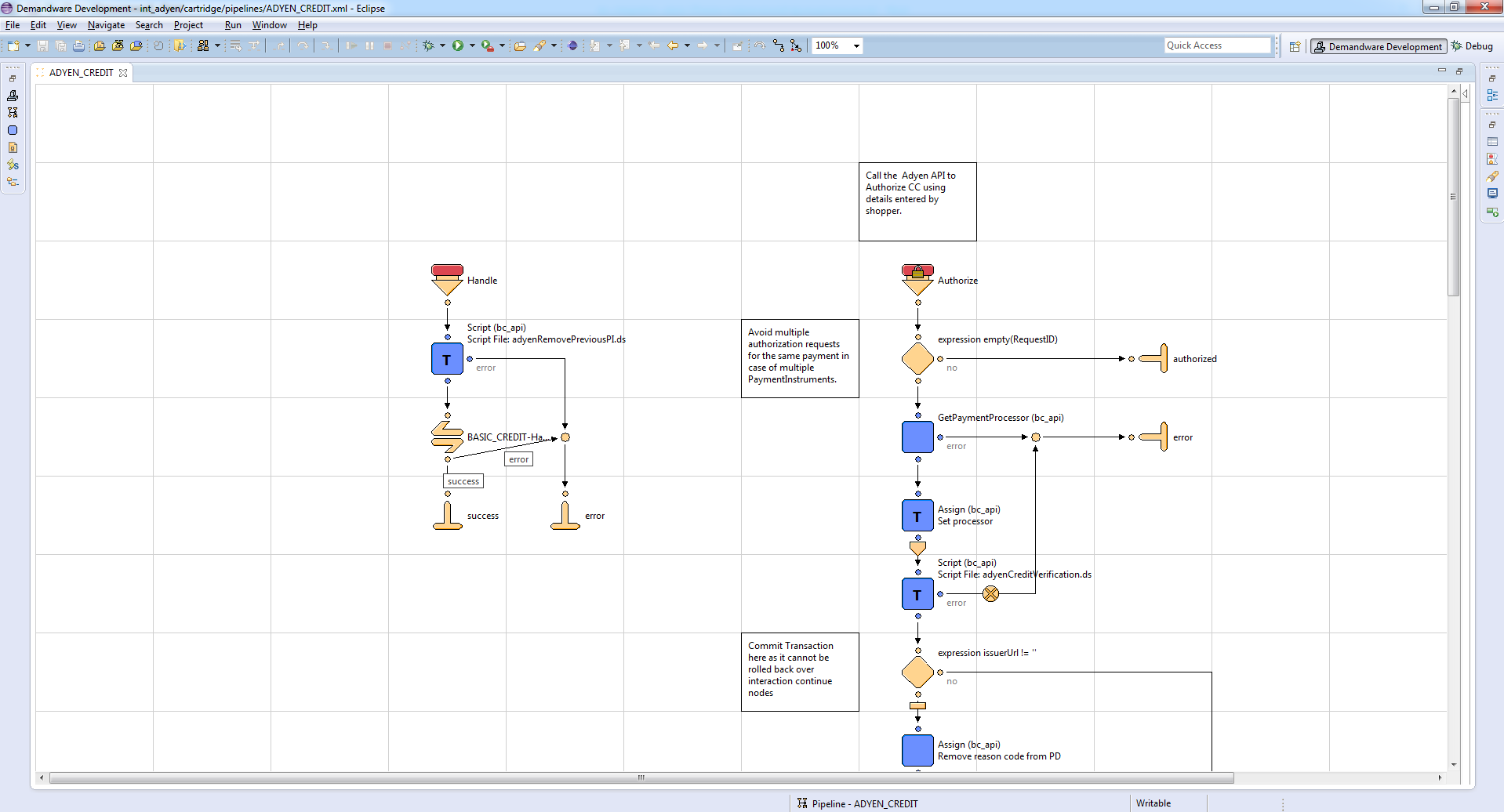
API payment modification pipelines



There are three pipelines which can be used for doing operations over payments. The pipelines are documented and need to be quickly investigated before the usage. Only the end developer can do a decision in which part of the site these pipelines have to be integrated in order to achieve needs of the client.

### 3.7.2 ADYEN\_CREDIT

Adyen Credit Card payments are handled in the separate ADYEN\_CREDIT pipeline:



## External Interfaces

None

# Testing

Make sure you have an Adyen test account. Configure your account and then configure the Adyen settings in the BM.

## Test Credit Cards

In case you want to test Credit Cards payment method, Adyen provides some test numbers you can use. Please check the following link as these numbers may change:

<https://support.adyen.com/index.php?/Knowledgebase/Article/View/11/0>

## Business Manager Configuration

Please read the Business Manager Configuration section for the info about configuring BM.

The Debug Setting set to 'yes' shows a 'pay' button before doing the redirect to Adyen. This allows you to check the parameters sent to the HPP.

## Testing API methods

There is a storefront cartridge in the distributive package, where there are added additional pages for testing API methods implemented in the cartridge. The current implementation of the cartridge includes the following APIs:

* Capture
* Cancel before the capture
* Cancel or refund

Here is the instruction for testing API methods.

### 4.3.1 Capture & Cancellation Payment

#### Storefront

1. In both cases you can see the form with the field where you should enter the order number
2. Once you provide the order number, you can click the submit button
3. You’ll be redirected to the result page. On this page, in the grey area, you’ll see the result of the operation you asked to perform
4. There are only 2 possible results of operation which you may see on result page:
   1. ‘Ok’ – everything was ok, the data were processed correctly. This status means that the data were handled properly and no any unexpected situation happened, but it does not necessary means that the payment will be captured, see below the info about event codes which are available in BM
   2. ‘Error’ – either the order was not found in DWRE system or some error occurred. If you see this message, nothing should be changed in BM, but theoretically something could be changed on Adyen side. See below the info about event codes which are available in BM

#### Business Manager

Once you see some result on the storefront, you may check the BM:

1. Find the order which you modified
2. Open it and go to Payment tab
3. The default value of Eventcode field is AUTHORISATION
4. If order payment was requested to be captured the following event codes are possible:
   1. CAPTURING – the data were successfully handled and the request was sent to Adyen
   2. CAPTURING REFUSED – Adyen refused payment capturing
5. If order payment was requested to be canceled the following event codes are possible:
   1. CANCELLATION – the data were successfully handled and the request was sent to Adyen
   2. CANCELLATION REFUSED – Adyen refused payment cancellation
6. If order payment was requested to be canceled or refunded the following event codes are possible:
   1. CANCELLATION OR REFUND – the data were successfully handled and the request was sent to Adyen
   2. CANCELLATION OR REFUND REFUSED – Adyen refused payment cancellation
7. After you send a request to Adyen, you see immediately only the result of this request, but not the result of operation which will be performed on Adyen side. Adyen works not in real time, so your request won’t be handled immediately on Adyen side, but when the request will be handled, Adyen will change the status of payment in its back office and also the Eventcode will be changed on Demandware site. So you should check after some time which is the end result of the operation, in BM and in Adyen back office

#### Adyen back office

1. Once you created new order it’ll be placed into Payment list in Adyen back office
2. When you perform some operations on payment, the status of payment is changed in an appropriate way. But this works not in real time, so when you send some request, wait a minute and only then check the result of operation. Also, when payment modification request is made from Adyen back office, an appropriate status of payment is shown in Adyen back office, but when I was working with the API, couldn’t see some statuses (Sent for Settlement for example, or Sent for Cancellation)
3. Possible statuses of payment in Adyen back office should be searched for in Adyen documentation

# Operations, Maintenance

## Data Storage

For each order which results in a payment attempt on the Adyen HPP, the payment results are available in the following fields:

* PSP reference - unique Adyen ID of this payment. It’s a link between Demandware order and Adyen payment
* Payment Method
* Eventcode - AUTHORISED, CANCELLATION, REFUND etc.
* Amount – Amount paid

They are handled by the Adyen-Notify pipeline, which receives the notification messages from Adyen. In one case, when a payment is refused on Adyen HPP, these fields are updated in Adyen-ShowConfirmation pipeline.

On top of that, all the communication to/from Adyen is logged in the log file directory available via webDAV. Please check this first in case you have any problems.

Also, all the notifications sent by Adyen are logged on the order level and can be found on Notifications tab of Order Details page in BM.

## Availability

In case of problems with the connection to Adyen please contact the Adyen support department.

Please supply as much information as possible (Merchant account, skin code, time, order number, PSP reference etc.). Also check the log file mentioned above.

## Support

In case of problems with the integration, missing features, etc. please contact the Adyen support department or your Adyen account manager.

# User Guide

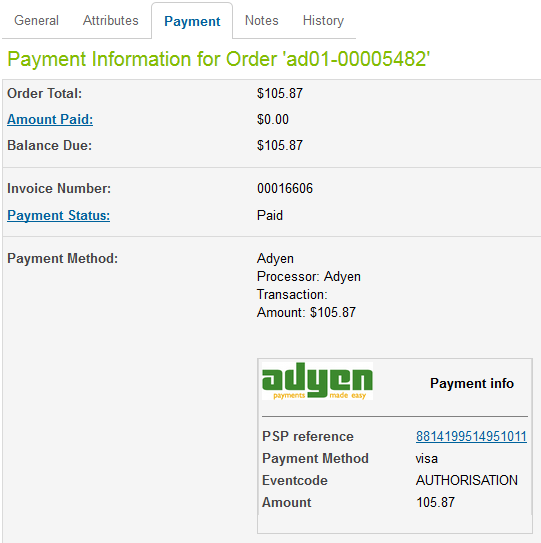
## Roles, Responsibilities

The store administrator should check the correct configuration of the Adyen Merchant account in the Adyen back office (CA) and should check the receiving of the Adyen notification messages on a regular basis.  
Also, check the payments received to see if they contain the expected data (currencyCode, amount, shopperEmail, etc)

## Business Manager

Please configure the settings relevant to the Adyen integration as described in section 3.2.

After an authorized payment, the Payments tab of the order should look like this:



All possible Eventcodes are described in a separate document from the distributive package and partially described on section 2.2. Use Cases.

Note that you can click on the PSP reference to go directly to the payment details in the Adyen back office (if you have logged in first).

In the Adyen back office you can also link back to DW by setting Merchant Settings > Merchant Reference Link to:

[https://adyen01.tech-prtnr-eu01.dw.demandware.net/on/demandware.store/Sites-Site/default/ViewOrder\_52-FindByNumber?OrderNo=](https://adyen.cloud01.developer-eu.demandware.net/on/demandware.store/Demos-Site/default/ViewOrder_52-FindByNumber?OrderNo=) (change to your own site name)

## Storefront Functionality

New storefront functionality is described in section 2.2 Use Cases

# Known Issues

* 'Session validity' and 'Shipbeforedate' are not used
* Partial payments are not supported

# Release History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| **1.0.0** | April 27, 2011 | Initial release |
| **1.0.1** | May 20, 2011 | GC changes |
| **2** | June 06,2013 | Plug and play, API payments |
| **2** | August 16,2013 | Minor changes to documentation |
| **3** | January 8, 2015 | Full refactoring and improvement of the document, fixed a lot of mistakes. Added description of new API methods, AVS, 3-D Secure |