Create Payment Provider OpenStore

Introduction

This document explains how to create a payment provider plugin for OpenStore v4. This document is a guide, your gateway will need extra work based on how it works.

You need to have Visual Studio and the VS template for the OS payment provider ("OS_PaymentGateway").

https://github.com/Open-Store-Project/OS_PaymentGateway/releases

Create Project

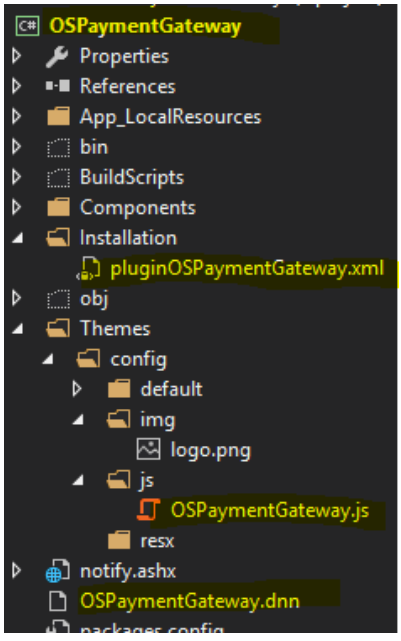
Create a new project in VS using the "OS_Paymentgateway" VS project template.

USE case match on next replace operation:
Rename ALL instances of "`os_paymentgateway_`" with "new gateway name_" (IMPORTANT: for this replace make sure you use LOWERCASE) **NOTICE: "_" on the end of this replacement.
If you forget the case sensitive replacement, it's easier to start again.

Ensure the xml ctrl field in the "plugin*.xml" file is the lowercase "new gateway name".
`<ctrl update="save">lowercase</ctrl>` This should match the replacement above (without "_").

Rename ALL instances of "OS_Paymentgateway" with "new gateway name" (Turn off match case).

Rename files and project to match new gateway.



You should be able to compile now. You'll need to move the assembly into the DNN bin folder.

Rename the JS file in the "settings.cshtml", it needs to match the file name in "`/DesktopModules/NBright/OS_PayPal/Themes/config/js/`". Usually this is updated by the VS template and therefore may not match in some cases.

“<script type="text/javascript"
src="/DesktopModules/NBright/OS_PayPal/Themes/config/js/<JS File Name>.js"></script>”

In the “settingsfields.cshtml” template there is a hardcoded ctrl name.

“<input id="ctrl" type="hidden" update="save" value="os_paymentgateway" />”
Make sure this is the same value as your gateway, in lowercase. (This is hardcoded so that any redisplay from ajax will get the correct value)

MSBUILD is included in the template and this can be activated by uncommenting the lines in the project .csproj file. (See below)

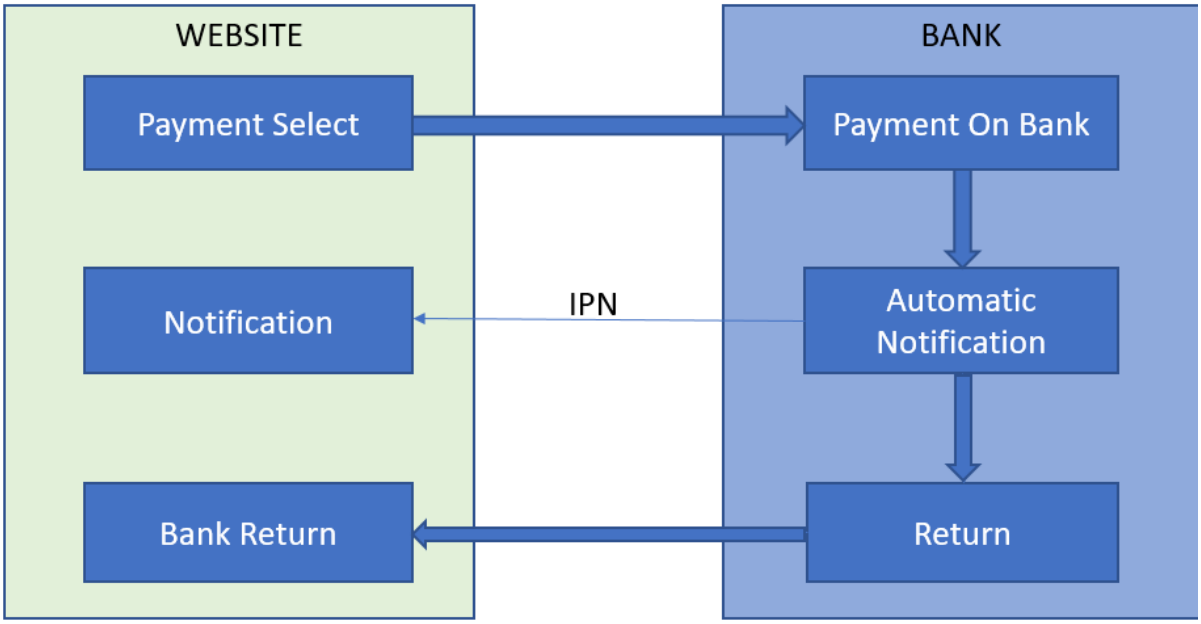
MSBUILD will automatically move your assembly to the DNN bin folder in Debug and release mode. In release mode it will also automatically create an install package for you. NOTE: The file names should match the assembly name. The MSBuild process uses the assembly name for the build process.

```
 <Import Project="BuildScripts\ModulePackage.Targets" />

 <Target Name="AfterBuild" DependsOnTargets="PackageAndDeploy">

 </Target>

 <Import Project="packages\MSBuildTasks.1.5.0.235\build\MSBuildTasks.targets"
Condition="Exists('packages\MSBuildTasks.1.5.0.235\build\MSBuildTasks.targets')" />

 <Target Name="EnsureNuGetPackageBuildImports" BeforeTargets="PrepareForBuild">

  <PropertyGroup>

   <ErrorText>This project references NuGet package(s) that are missing on this computer. Use NuGet Package Restore to download them.  For more
information, see http://go.microsoft.com/fwlink/?LinkID=322105. The missing file is {0}.</ErrorText>

  </PropertyGroup>

  <Error Condition="!Exists('packages\MSBuildTasks.1.5.0.235\build\MSBuildTasks.targets')" Text="$([System.String]::Format('$(ErrorText)',
'packages\MSBuildTasks.1.5.0.235\build\MSBuildTasks.targets'))" />

 </Target>
```

Building the gateway

From this point you can create the bespoke requirements for your gateway. The base template only offers a structure and the process required.

This is the process for most payment gateways. (not always)



We're usually only code for what the website is doing. The bank side is out of our control, however we often have to pass settings to the bank. These settings are sometimes in the admin of the bank system and don't need to be passed.

Add to Open Store Menu

The payment gateways are added to the OpenStore admin menu via the plugin interface. The easiest way to add a plugin is to copy the "plugin*.xml" of the project into the "\DesktopModules\NBright\NBrightBuy\Plugins" folder, then go into the Open-Store BO>Admin>Plugins and the plugin will be automatically added to the menu. You can also create the plugin manually through the plugin page if you want to.

NOTE: If you create the plugin manually you can use the XML field to get a copy of the XML required to create a plugin*.xml file.

Updating Setting

At this point you should be able to edit the plugin settings and they should save OK. If they don't then look at the JS file in the theme folder. The "settings.cshtml" should load this and the names should be matching, if the above replacements have been done correctly.

The setting page uses the OpenStore Ajax interface to save the setting. This update is triggered by the "`nbxget('*_savesettings', '.*data', '.*returnmsg');`" call and it actioned by the "AjaxProvider.cs" code.

To create extra settings required for your gateway edit the "settingsfields.cshtml" file and add your extra fields, this field data will automatically be saved via the Ajax call.

NOTE: Using "_" in the xpath (id) of the control input fields will alter the name of the field. The "_" is used to identify multiple rows which may have identical input ids, the "_" is used to stop this clash.

The Payment Provider

The payment provider code is defined by the plugin data in "BO>Admin>Plugins". By default the interface code is "PaymentProvider.cs".

Payment Gateway Template

The payment gateway displays a template to the client which has a button to select the payment method. Remember multiple payment methods can be listed on the payment page.

The payment template is displayed to the client via the "PaymentProvider.cs" code, by default the template is "methodselection.cshtml"

In this template is the button which has some special code to deal with the redirection.

```
<div id="cartactions" style="">
    <div style="clear: both;">
        <a href="?provider=sips1" id="cmdOS_Sips" class="primarybutton">@info.GetXmlProperty("genxml/lang/genxml/textbox/buttontext")</a>
    </div>
</div>
```

The link refreshed the payment page with a param of "provider", which is the plugin ref for the provider, this will inform the Payment module in OpenStore that a redirection to a bank is required.

The payment module will then call the "`RedirectForPayment`" method in the "PaymentProvider.cs" code. This will cause the redirection to the bank.