Payment Gateways

This document describes how to implement both Merchant and Non-merchant payment gateways. The difference between these two types is that Non-merchant gateways process payments on an external web site. That is, rather than enter payment details (credit card number or bank account information) within Blesta as is the case with Merchant gateways, Non-merchant gateways forward the user off to the payment processor's own website to enter their payment details, after which time they may be directed back to Blesta.

Merchant gateways are generally preferred over Non-merchant gateways as they allow the user to subscribe for automatic payment processing. Some Non-merchant gateways support subscription based payments, but these payments are initiated from the payment gateway rather than by Blesta, so they don't make a good fit for users with multiple services recurring at different intervals.

Did you know...

The terms "Merchant gateway" and "Non-merchant gateway" were coined by the Blesta development team just prior to the initial launch of version 1.0 in 2007. The word "merchant" refers to the requirement of a Merchant Account to process payments.

- The Merchant Gateways section explains how to implement a Merchant payment gateway
- The Non-merchant Gateways section explains how to implement a Non-Mechant payment gateway

Use the Extension Generator

As of Blesta 4.12 we've included a useful tool to help developers get started and save time. Blesta's Extension Generator can be used to generate many the files necessary for a gateway and will create basic code with an option to include comments to help you understand each part of the code.

Getting Started with Payment Gateways

For the purpose of this manual, the gateway name we'll refer to is **my_gateway**. Your gateway name will, of course, differ.

Gateway names must be unique

A user will not be able to install two gateways with the same name.

File Structure

Gateways are fully contained within their named gateway directory and placed in the */installpath/components/gateways/merchant/* or */installpath/components/gateways/nonmerchant/* directory (depending on their type). Each gateway is only required to contain a single class, representing the gateway, but the following is the recommended structure:

- */gateways/merchant/*
    - *my_gateway/*
        - *views/*
        - *language/*
        - **my_gateway.php**

Install Logic

If your gateway requires any code to execute when installed, place that logic in your gateway's **install()** method.

/gateways/merchant/my_gateway/my_gateway.php

```
<?php
class MyGateway extends MerchantGateway implements MerchantCc {

    ...

    public function install() {
        #
        # TODO: Place installation logic here
        #
    }
}
?>
```

Uninstall Logic

If your gateway required code to install, it likely requires code to uninstall. Place that logic in your gateway's **uninstall()** method.

There are two parameters passed to the uninstall() method. The first (**$gateway_id**) is the ID of the gateway being uninstalled. Because a gateway can be installed independently under different companies you may need to perform uninstall logic for a single gateway instance. The second parameter (**$last_instance**) identifies whether or not this is the last instance of this type of gateway in the system. If **true**, be sure to remove any remaining remnants of the gateway.

/gateways/merchant/my_gateway/my_gateway.php

```
<?php
class MyGateway extends MerchantGateway implements MerchantCc {

    ...

    public function uninstall($gateway_id, $last_instance) {
        #
        # TODO: Place uninstallation logic here
        #
    }
}
?>
```

Upgrade Logic

When the version of your code differs from that recorded within Blesta a user may initiate an upgrade, which will invoke your gateway's **upgrade()** method.

The **$current_version** is the version currently installed. That is, the version that will be upgraded from.

/gateways/merchant/my_gateway/my_gateway.php

```
<?php
class MyGateway extends MerchantGateway implements MerchantCc {

    ...

    public function upgrade($current_version) {
        #
        # TODO: Place upgrade logic here
        #
    }

}
?>
```

Error Passing

Blesta facilitates error message passing through the use of the **Input** component. Simply load the Input component into your gateway and set any errors you encounter using **Input::setErrors()**. The setErrors() method accepts an array of errors. This can be a multi-dimensional array (in the case of multiple errors triggered for the same input or criteria) or a single dimensional array. The first dimension should be the name of the input that triggered the error. The second dimension, if necessary, is used to identify the type of error encountered.

/gateways/merchant/my_gateway/my_gateway.php

```
<?php
class MyGateway extends MerchantGateway implements MerchantCc {

    public function __construct() {
        Loader::loadComponents($this, array("Input"));
    }

    ...

    public function upgrade($current_version) {
        // Ensure new version is greater than installed version
        if (version_compare($this->version, $current_version) < 0) {
            $this->Input->setErrors(array(
                'version' => array(
                    'invalid' => "Downgrades are not allowed."
                )
            );
            return;
        }
    }
}
?>
```