Custom payment gateways

Custom payment gateway implementations allow you to implement payment gateways not natively supported in Resello. The basic flow for this is the following:

Resello -> You -> Payment provider -> You -> Resello

Basically you are a proxy between Resello and the payment provider, translating the messages from the payment provider to messages Resello understands and messages Resello sends to messages the payment provider understands.

Creating your custom payment gateway

In order to use a custom payment gateway you have to create one in our system. You can create a custom payment gateway in your reseller account with the following steps:

- Login to your reseller account.
- In the menu select **Settings** and click **Payment gateways**.
- Click the **Create** button and click on **Basic** in the drop down menu that appeared.
- Enter a unique name for this payment gateway and enter the URL to which the user should be redirected upon starting the payment. Click **Save** to create the payment gateway.

Creating a new payment method

To allow customers and resellers to pay using the custom payment gateway you just created you have to create a new payment method.

- Login to your reseller account.
- In the menu click **Storefront**, select **Settings** in the storefront menu and click **Payment options**.
- Click the **Create** button.
- Select type Automatic, for gateway choose the gateway you created in the previous step, enter a name and a description and optionally a transaction fee. Click **Save** to create the payment method.

Secret keys and notification URL

You've now created a custom payment gateway and custom payment method. During the creation we've generated a pair of secret keys and a notification URL for the payment gateway. You will need these values to integrate your custom payment gateway.

- Login to your reseller account.
- Select the menu **Settings** and click on **Payment gateways**.
- Click on your custom payment gateway.
- You should now see a page presenting a **Secret key 1**, **Secret key 2** and **Notification URL**.

Creating a payment page

On our own server you should create the page that you entered as the payment gateway URL. When the user starts the payment, a POST request is performed to that page. With this POST request the following parameters are send.

Parameter name	Type	Description
reference	string	This is the payment reference. This is unique for each payment.
currency	string	This is the payment currency. It's the ISO 4217 code.
amount	integer	This is the payment amount as an integer. In order to get the proper amount you have to divide it by 100.
customer	integer	This is the id of the customer that started this payment.
started	string	This is the date and time of the start point of this payment.
expires	string	This is the date and time of the expiration point for this payment.
gateway	integer	This is the id of the selected payment gateway.
return_url	string	This is the URL you should redirect the customer to for an instant payment status update.
signature	string	This is a signature that is hashed with secret key 1 and secret key 2. Both keys were generated for you when your payment gateway was created. This signature is built of the above keys in the order they are defined in this table.

Calculating the signature

In order to send messages back to our system and validate the data send to your payment page we have secured the data with a signature. The signature is created the following way.

Python:

```
import hashlib
import hmac
def sign(secret_key_1, secret_key_2, data):
```

```
data = ''.join(data) + secret_key_1
return hmac.new(key=secret_key_2, msg=data,
digestmod=hashlib.sha512).hexdigest()
```

PHP:

```
function($secret_key_1, $secret_key_2, $data) {
$data = implode('', $data) . $secret_key_1;
return hash_hmac('sha512', $data, $secret_key_2);
}
```

Redirecting the customer back

When redirecting the customer back to our system you have to pass three variables.

Parameter name	Type	Description
reference	string	This is the payment reference. This is unique for each payment.
status	string	This is the payment status. You can send either AUTHORISED to indicate the payment has succeeded, FAILED in case the payment failed or STARTED in case the payment is not yet completed, but also didn't fail. In case of a STARTED status you should send notifications to complete or fail the payment.
signature	string	This is a signature that is hashed with secret key 1 and secret key 2. Both keys were generated for you when your payment gateway was created. This signature is built of the above keys in the order they are defined in this table.

Notifications

For payments that are not instantly completed or failed you can send notifications. This is done via a HTTP POST on a specific URL. This URL can be found in your payment gateway details.

The values you have to POST with the request are specified in this table.

Parameter name	Type	Description
reference	string	This is the payment reference. This is unique for each payment.
status	string	This is the payment status. You can send either AUTHORISED to indicate the payment has succeeded, FAILED in case the payment failed or STARTED in case the payment is not yet completed, but also didn't fail. In case of a STARTED status you should send notifications to complete or fail the payment.
signature	string	This is a signature that is hashed with secret key 1 and secret key 2. Both keys were generated for you when your payment gateway was created. This signature is built of the above keys in the order they are defined in this table.

Python:

```
import requests
signature = example_create_signature()
notification_url = 'https://example-notification-url'
requests.post(notification_url, data={
'reference': 'example-reference',
'status': 'AUTHORISED',
'signature': signature
})
```

PHP:

```
$signature = example_create_signature();
$notification_url = 'https://example-notification-url';
$fields = array(
'reference' => 'example-reference',
'status' => 'AUTHORISED',
'signature' => $signature
);
$fields_string = '';
foreach($fields as $key=>$value) {
$fields_string .= $key . '=' . $value . '&';
}
rtrim($fields_string, '&');
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $notification_url);
curl_setopt($ch, CURLOPT_POST, count($fields));
curl_setopt($ch, CURLOPT_POSTFIELDS, $fields_string);
curl_exec($ch);
curl_close($ch);
```