Developers' guide

For PHP payment library

In this documentation we are detailing the developers' tasks for integration.

1. Realization of a general payment procedure

1.1 Choosing a payment method on the website

Make it possible in the webshop /website for your customers to choose a prefered payment method.

1.2 Finalization of order on the website

If you collected all the necessary data from your customer, save the order information in your database and call Payment Gateway **Init** method to initialize the transaction.

1.3 Initialisation of transaction at Payment Gateway

You can do it by integrating the next code detail into your software.

```
$request = new PaymentGateway_Init_Request();
$request->setProviderName("Sofort")
        ->setResponseUrl("http://www.webshop.com/response_url.php")
        ->setAmount(10)
        ->setCurrency("EUR")
        ->setOrderId("order 123")
        ->setUserId("user 123")
        ->setLanguage("EN");
$response = PaymentGateway::init($request);
```

The whole parameter list and more explanation is available in "Payment Gateway technical information" document.

IMPORTANT!

Payment Gateway will send the Customer to response URL after payment.

It often happens, that Customer finalizes payment at the website of the bank (or other payment service provider), closes browser here so does not return to your website.

In this case Payment Gateway automatically detects that transaction was not responsed to your system so it will call the response URL in the background, that's why it is important not to connect URL call to Customer's session or any other authentication.

1.4 Processing the result of the initialisation of transaction

After calling the Init method, Payment Gateway create an unique transaction ID **(TransactionID)** that will be returned to your system.

You can process the result of the initialisation of transaction by customizing the code detail bellow:

```
if ($response->ResultCode == "SUCCESSFUL" && $response->TransactionId)
{
        // TODO: store the TransactionId
        $request = new PaymentGateway_Start_Request($response->TransactionId);
        PaymentGateway::start($request);
}
 else
{
        // There was an error, display the error message for the user
        $errorMessage = $response->ResultCode.":".$response->ResultMessage;
}
```

If the transaction initialisation was successful, the value of **ResultCode** is „SUCCESSFUL", and TransactionID parameter has a value.

Before you call **Start** method, save TransactionID in your database and connect it to your order data, because Payment Gateway will refer to this ID later, and your system will be able to call the result of transaction from Payment Gateway using this ID.

1.5 Directing Customer to payment page

If you call Start method that you can find in the code detail above, you can direct the Customer to Payment Gateway, then Payment Gateway will forward them to the payment page of the payment service provider chosen by the Customer.

The procedure will be interrupted in your system here, Customer will realize the payment in the site of the payment service provider.

1.6 Customer will return to your webshop

After Customers have paid successfully or unsuccessfully, they return to the defined response URL **(ResponseURL)**. Payment Gateway will complete response URL with the unique transaction ID **(TransactionID)** that you received and saved in your database at the initialisation of the transaction.

An example for a response URL completed with the unique transaction ID:

http://www.yourdomain.com/response.php?TransactionId=0a62408585bdfbb54851fd454078641d

1.7 Querying the result of transactions

When Payment Gateway has called the response URL in your system, you have to query the result of the transaction in the background.

It is important not to connect the calling of response URL to the customer's session or to any other authentications because it may happen that Customer will close the browser at the payment page of the bank and will not return to the webshop, so Payment Gateway will call the response URL in the background.

The next code detail is for querying the result of payment transaction:

```
$request = new PaymentGateway_Result_Request($_GET["TransactionId"]);
$response = PaymentGateway::result($request);
if ($response->ResultCode == "SUCCESSFUL") {
        // Transaction successful
} else {
        // There was an error, display the error message for the user
        $errorMessage = $response->ResultCode.":".$response->ResultMessage;
}
```

In case the value of result code is „SUCCESSFUL”, payment was successfully realized, order can be fulfilled / transaction can be closed.

Please save the result of the transaction in your database + the values of – the most important - parameters bellow, and show them to your Customer:

- **ResultMessage:** explanation text of the result.
- **ProviderTransactionID:** unique ID registered by the payment service provider. If you may have any problem or question regarding the transaction, you can clarify it with the payment service provider referring to this ID.
- **Anum:** authorisation number given by the bank.

TIP!

In case the value of the result code (ResultCode) is NOT „SUCCESSFUL”, you should offer your Customer to try to pay again or choose another payment method. It is important because it might happen that the Customers gave any data not correctly (typo) or maybe they accidentally chose a payment service provider where they can not pay (because for example Customers need to be registered there in advance).

2. Support

If you have any questions or proposals, please do not hesitate to contact us:

BIG FISH Internet-technológiai Kft.

+36 1 209 0760

paymentgateway@bigfish.hu

www.paymentgateway.hu

www.bigfish.hu