PHP SDK

Content

Introduction

An integration kit is a software library (sdk) that facilitates the IT development that you will have to carry out to integrate the Payline payment solution into your site.

⚠️You should review the Getting Started documentation and the Web Page Payment integration.

The PHP SKD is made up of the following elements:

- an installation guide for the development SDK;
- a **composer.json** descriptor for downloading via Composer from the PHP library which allows you to use the functions of the Payline API;
- an example web application using the SDK.

By decompressing the zip file, there are three directories:

The 'examples' directory

The “example” directory corresponds to the web application implementing all of the Payline functionalities available via the PHP library. Each feature comes in the form of two files:

- *a <function_name> Form.php file containing the presentation form.*
- *a <function_name> .php file which retrieves the data transmitted by the user via the form. The PHP code makes the call to the desired Web Service.*

For example, the doWebPayment service, which allows a Web payment to be made via the SDK, results in the presence of the following two files:

- *doWebPaymentForm.php: This file simulates the summary page of your customer's order. It constitutes step 0 in the web payment process. You can modify the values of the form fields: Amount, Currency and OrderRef to make test payments.*
- *doWebPayment.php: This file contains the PHP code which retrieves the information transmitted by the form and uses the library to initiate a web payment request.*

The 'lib' directory

It groups together the classes that describe the requests, responses and objects of the SOAP Payline API.

- *paylineSDK.php: This file contains the main class which allows the creation of SOAP messages as well as the other specific classes which manage business requests and responses. For example: doWebPayment, getWebPaymentDetails.*
- *lib_debug.php: This is the debug library which allows the display in the form of a nested table of the return message of your requests to the SOAP Payline API.*
- *v4.* XX *.wsdl: The descriptor of the Payline web services, from which the requests / responses are constructed. This file is replaced with each new release of Payline, backward compatibility is ensured.*

- *CONFIG.php: This file introduced in version V4.64.1 replaces the configuration directory and its content. It contains the Payline connection parameters, as well as the parameters (contracts, version, currency, etc.) displayed by default in the example pages.*
  The CONFIG.php file can be edited directly, or via the form included in the Home section.

The 'logs' directory

Within the archive, this directory is empty. *It corresponds to the default directory for writing traces.*

A <date> .log file (<date> being in yyyy-mm-dd format) is created and / or fed during each call to a library function.

The function to add a line to this file is called writeTrace. *It is a public function of the paylineSDK class, which can therefore be reused.*

Installation

Perform the following steps:

1. Download the Payline API SDK:

- https://github.com/PaylineByMonext/payline-php-sdk

2. Unzip the archive at the root of your web server.

If you are not yet using Composer, download the composer.phar executable via https://getcomposer.org/download/
Deposit the composer.phar executable at the same level as the composer.json file included in the payline archive -php-sdk .

3. Via a command prompt:

```
composer require monext/payline-sdk
```

Composer creates the vendor directory and deposits the latest version of the PHP Payline SDK there, as well as the related libraries.

Functions

All Payline functions are covered by the PHP SDK.
The paylineSDK class offers a function corresponding to each web service, bearing the same name as the latter (doWebPayment, doAuthorization,…).

Call

The parameter passing between your store and these functions is done via an associative array. Examine the sample scripts contained in the SDK to understand this mechanism.

Return

The response from a Payline web service is also stored by the SDK in an associative array. Examine the sample scripts in the SDK to understand how to read a response.

Particular case

If an exception occurs when calling an SDK function (for example due to a lack of communication with Payline due to incorrect authentication or proxy settings), a response containing the code XXXXX as well as the exception label is returned.
This response is not returned by Payline, but by the PHP SDK.

SDK configuration

The operation of the SDK requires the activation of the following PHP extensions on your server: php_curl, php_http, php_openssl, php_soap

Once the SDK has been unzipped on your server, you must enter the parameters in the "configuration" form of the "Home" section, or directly in the CONFIG.php file located under the lib directory.

MERCHANT_ID : the identifier of your merchant account
ACCESS_KEY : the access key associated with your merchant account
ACCESS_KEY_REF : the web2token reference associated with your access key
PROXY_HOST : the URL of your Internet proxy
PROXY_PORT : the communication port of your Internet proxy
PROXY_LOGIN : the user ID required by your Internet proxy
PROXY_PASSWORD : the user password required by your Internet proxy
ENVIRONMENT : indicator used to determine the Payline environment to which requests are sent. Two values can be used by traders:
HOMO (certification environment, to carry out integration tests) and PROD (production environment).
WS_VERSION : application version of Payline web services.
PAYMENT_CURRENCY : the ISO code of the currency to use by default for payment
ORDER_CURRENCY : the ISO code of the currency to use by default for the order
SECURITY_MODE : the code of the security mode to use by default
LANGUAGE_CODE : the ISO code of the language to display by default
PAYMENT_ACTION : the code of the payment method to use by default
PAYMENT_MODE : the payment method to use by default
CANCEL_URL : the cancellation URL used when your customer canceled the payment
NOTIFICATION_URL : the notification URL used when Payline notifies you of a payment made
RETURN_URL : the return URL used when the payment was accepted or refused
CUSTOM_PAYMENT_TEMPLATE_URL : the URL of the dynamic template to apply to the payment web pages
CUSTOM_PAYMENT_PAGE_CODE : the code for customizing the Payline payment pages to be used by default.
CONTRACT_NUMBER : the contract number that identifies your point of sale and your default payment method,
CONTRACT_NUMBER_LIST : the list of contract numbers to display if you have more than one. Your unique contract number. The Payline SelectedContractList tag of the doWebPayment.
For a version greater than or equal to 4 of the web service, this tag is mandatory.
Example : $ array ['contracts'] = array ('1234567', '7777777');
SECOND_CONTRACT_NUMBER_LIST : the list of contract numbers to display when the first payment attempt is unsuccessful.

SDK compatibility

The PHP SDK is tested with the following environment:

- Windows 7 Professional system
- the Apache server 2.2.22
- PHP version 5.3.13

Integration

Please see the Getting Started documentation and the Web Page Payment integration to develop your Web Page Payment .

When integrating, please name the tags as indicated in the kit. They may be different from the Payline API:

> ***For the <authentication3DSecure> tag, you must call <3DSecure>***
>
> - *The merchant must value the table by replacing ['authentication3DSecure'] by ['3DSecure']*
>
> ***For the <selectedContractList> tag is called <contracts>***
>
> - *The merchant should value the array by replacing ['selectedContractList'] with ['contracts']*

Example of using payment initialization in web page mode:

doWebPayment

```
public function doWebPayment(array $array)
    {
        $this->formatRequest($array);
        $WSRequest = array(
            'payment'                   => $this->payment($array['payment']),
            'returnURL'                 => $array['returnURL'],
            'cancelURL'                 => $array['cancelURL'],
            'order'                     => $this->order($array['order']),
            'notificationURL'           => $array['notificationURL'],
            'customPaymentTemplateURL'  => $array['customPaymentTemplateURL'],
            'selectedContractList'      => $array['contracts'],
            'secondSelectedContractList' => $array['secondContracts'],
            'privateDataList'           => $this->privateData,
            'languageCode'              => $array['languageCode'],
            'customPaymentPageCode'     => $array['customPaymentPageCode'],
            'buyer'                     => $this->buyer($array['buyer'], $array['shippingAddress'], $array
['billingAddress'], $array['merchantAuthentication']),
            'owner'                     => $this->owner($array['owner'], $array['ownerAddress']),
            'securityMode'              => $array['securityMode'],
            'contractNumberWalletList'  => $array['walletContracts'],
            'merchantName'              => $array['merchantName'],
            'subMerchant'               => $this->subMerchant($array['subMerchant']),
            'miscData'                  => $array['miscData'],
            'asynchronousRetryTimeout'  => $array['asynchronousRetryTimeout'],
            'threeDSInfo'               => $this->threeDSInfo($array['threeDSInfo'], $array['browser'],
$array['sdk'])
        );
```

Example of using a payment authorization request in direct mode:

doAuthorization

```
public function doAuthorization(array $array)
    {
        $this->formatRequest($array);
        $WSRequest = array(
            'payment'                  => $this->payment($array['payment']),
            'card'                     => $this->card($array['card']),
            'order'                    => $this->order($array['order']),
            'buyer'                    => $this->buyer($array['buyer'], $array['shippingAddress'], $array
['billingAddress'], $array['merchantAuthentication']),
            'owner'                    => $this->owner($array['owner'], $array['ownerAddress']),
            'privateDataList'          => $this->privateData,
            'authentication3DSecure'   => $this->authentication3DSecure($array['3DSecure']),
            'bankAccountData'          => $this->bankAccountData($array['bankAccountData']),
            'subMerchant'              => $this->subMerchant($array['subMerchant']),
            'asynchronousRetryTimeout' => $array['asynchronousRetryTimeout']
        );
```

Pages linked

- .NET SDK
- JAVA SDK

- JAVA version 2 SDK
- PHP SDK
- SDK & Plugins