# SDK PHP

Contenu


- Présentation
- Installation
  - Le répertoire 'lib'
  - Le répertoire 'logs'
- Fonctions
- Configuration du SDK
- Compatibilité du SDK
- Intégration
- Pages associées


---

## Présentation

Un kit d'intégration est une librairie logicielle (sdk) qui facilite le développement informatique que vous allez devoir réaliser pour intégrer la solution de paiement Payline dans votre site.

⚠️ Vous devez consulter la documentation Comment démarrer et l'intégration Paiement Page Web pour développer votre Paiement Page Web.

Le SKD PHP est composé des éléments suivants :

- une notice d'installation du SDK de développement ;
- un descripteur **composer.json** pour le téléchargement via Composer de la librairie PHP qui permet d'utiliser les fonctions de l'API Payline ;
- un exemple d'application web utilisant le SDK.

## Installation

Exécuter les étapes suivantes :

**1. Télécharger le SDK de l'API Payline :**

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

**2. Décompressez l'archive à la racine de votre serveur web.**

Si vous n'utilisez pas encore Composer, téléchargez l'exécutable composer.phar via https://getcomposer.org/download/
Déposez l'exécutable composer.phar au même niveau que le fichier composer.json inclu dans l'archive payline-php-sdk.

**3. Via une invite de commande :**

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

Composer créée le répertoire vendor et y dépose la dernière version du SDK PHP Payline, ainsi que les librairies liées.

En montant le fichier avec composer on trouve trois répertoires :

### *Le répertoire 'lib'*

*Il regroupe les classes qui décrivent les demandes, réponses et objets de l'API SOAP Payline.*

- *paylineSDK.php : Ce fichier contient la classe principale qui permet la création de messages SOAP ainsi que les autres classes spécifiques qui réalisent gèrent les demandes et réponses métier. Par exemple : doWebPayment, getWebPaymentDetails.*

- *lib_debug.php : C'est la librairie de debug qui permet l'affichage sous forme de tableau imbriqué du message retour de vos demandes à l'API SOAP Payline.*
- *v4.XX.wsdl : Le descripteur des web services Payline, à partir duquel les requêtes/réponses sont construites. Ce fichier est remplacé à chaque nouvelle release de Payline, la compatibilité ascendante est assurée.*
- *CONFIG.php : Ce fichier introduit dans la version V4.64.1 remplace le répertoire configuration et son contenu. Il contient les paramètres de connexion à Payline, ainsi que les paramètres (contrats, version, devise,...) affichés par défaut dans les pages d'exemple.*
  *Le fichier CONFIG.php peut être édité directement, ou via le formulaire inclut dans la rubrique Home.*

### *Le répertoire 'logs'*

*Au sein de l'archive, ce répertoire est vide. Il correspond au répertoire par défaut d'écriture des traces.*

*Un fichier <date>.log (<date> étant au format aaaa-mm-jj) est créé et/ou alimenté lors de chaque appel à une fonction de la librairie.*
*La fonction d'ajout d'une ligne dans ce fichier se nomme writeTrace. C'est une fonction publique de la classe paylineSDK, qui peut donc être réutilisée.*

# Fonctions

L'ensemble des fonctions Payline sont couvertes par le SDK PHP.
La classe paylineSDK propose une fonction correspondant à chaque web service, portant le même nom que ce dernier (doWebPayment, doAuthorization, …).

## Appel

Le passage de paramètre entre votre boutique et ces fonctions se fait via un tableau associatif. Examinez les scripts d'exemple contenus dans le SDK pour comprendre ce mécanisme.

## Retour

La réponse d'un web service Payline est également stockée par le SDK dans un tableau associatif. Examinez les scripts d'exemple contenus dans le SDK pour comprendre la lecture d'une réponse.

## Cas particulier

Si lors de l'appel à une fonction du SDK une exception se produit (par exemple en raison d'une absence de communication avec Payline du fait du mauvais paramétrage de l'authentification ou de proxy), une réponse contenant le code XXXXX ainsi que le libellé de l'exception est renvoyée. Cette réponse n'est pas retournée par Payline, mais par le SDK PHP.

# Configuration du SDK

***Le fonctionnement du SDK nécessite l'activation des extensions PHP suivantes sur votre serveur : php_curl, php_http, php_openssl, php_soap***

Une fois le SDK décompressé sur votre serveur, vous devez renseigner les paramètres dans le formulaire "configuration" de la rubrique "Home", ou directement dans le fichier CONFIG.php situé sous le répertoire lib.

**MERCHANT_ID** : l'identifiant de votre compte commerçant
**ACCESS_KEY** : la clé d'accès associé à votre compte commerçant
**ACCESS_KEY_REF** : la référence web2token associée à votre clé d'accès
**PROXY_HOST** : l'URL de votre proxy Internet
**PROXY_PORT** : le port de communication de votre proxy Internet
**PROXY_LOGIN** : l'identifiant utilisateur requis par votre proxy Internet
**PROXY_PASSWORD** : le mot de passe utilisateur requis par de votre proxy Internet
**ENVIRONMENT** : indicateur qui permet déterminer l'environnement Payline vers lequel les requêtes sont adressées. Deux valeurs sont utilisables par les commerçants :
HOMO (environnement d'homologation, pour réaliser les tests d'intégration) et PROD (environnement de production).
**WS_VERSION** : version applicative des web services Payline.
**PAYMENT_CURRENCY** : le code ISO de la devise à utiliser par défaut pour le paiement
**ORDER_CURRENCY** : le code ISO de la devise à utiliser par défaut pour la commande

**SECURITY_MODE** : le code du mode de sécurité à utiliser par défaut
**LANGUAGE_CODE** : le code ISO de la langue à faire afficher par défaut
**PAYMENT_ACTION** : le code de la méthode de paiement à utiliser par défaut
**PAYMENT_MODE** : le mode de paiement à utiliser par défaut
**CANCEL_URL** : l'URL d'annulation utilisée lorsque votre client a annulé le paiement
**NOTIFICATION_URL** : l'URL de notification utilisée lorsque Payline vous notifie d'un paiement effectué
**RETURN_URL** : l'URL de retour utilisée lorsque le paiement a été accepté ou refusé
**CUSTOM_PAYMENT_TEMPLATE_URL** : l'URL du template dynamique à appliquer aux pages web de paiement
**CUSTOM_PAYMENT_PAGE_CODE** : le code de personnalisation des pages de paiement Payline à utiliser par défaut.
**CONTRACT_NUMBER** : le numéro de contrat qui identifie votre point de vente et votre moyen de paiement par défaut,
**CONTRACT_NUMBER_LIST** : la liste des numéros de contrat à faire afficher si vous en possédez plusieurs. Votre numéro de contrat unique. La balise Payline SelectedContractList du doWebPayment.
Pour une version supérrieure ou égale à 4 du web service, cette balise est obligatoire.
Exemple : $array['contracts'] = array('1234567','7777777');
**SECOND_CONTRACT_NUMBER_LIST** : la liste des numéros de contrat à faire afficher lorsque la première tentative de paiement est un échec.

# Compatibilité du SDK

Le SDK PHP est compatible avec l'environnement suivant :

- le système Windows 7 Professionnel
- le serveur Apache 2.2.22
- la version PHP : voir le fichier README.md sur github

# Intégration

Veuillez consulter la documentation Comment démarrer et l'intégration Paiement Page Web pour développer votre Paiement Page Web.

Lors de l'intégration, veuillez bien à nommer les balises comme indiqué dans le kit. Elles peuvent être différentes de l' API Payline :

> ***Pour la balise <authentication3DSecure>, il faut appeler <3DSecure>***
>
> - *Le commerçant doit valoriser le tableau en remplaçant ['authentication3DSecure'] par ['3DSecure']*
>
> ***Pour la balise <selectedContractList> s'appelle <contracts>***
>
> - *Le commerçant doit valoriser le tableau en remplaçant [' selectedContractList '] par ['contracts']*

Exemple d'utilisation de l'initialisation de paiement en mode page web :

**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'])
        );
```

Exemple d'utilisation d'une demande d'autorisation de paiement en mode direct

**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 associées

- SDK PHP