



Table des matières

Table	des matières	2
Histor	rique des versions	4
1	Introduction	5
1.1	Présentation de Sips	5
1.2	Objectif du document	5
1.3	Prérequis	5
2	Cinématique de paiement	6
2.1	Principe général	6
2.2	Flux de paiement	7
3	Description du protocole	9
3.1	Champs POST	9
3.1.1	Syntaxe du champ Data	9
3.1.2	Syntaxe du champ Seal	9
3.1.3	Syntaxe du champ Encode	10
4	Comment implémenter un paiement ?	11
4.1	Requête de paiement	11
4.1.1	Champs de la requête de paiement	11
4.1.2	Exemple	11
4.1.3	Gestion des erreurs	11
4.2	Réponses de paiement	13
4.2.1	Réponse manuelle	13
4.2.2	Réponse automatique	14
4.2.3	Problèmes de réception des réponses Sips	14
4.2.4	Gestion des erreurs– Pas de signature dans la réponse	15
5	Comment SIGNER un MESSAGE ?	16
5.1	Pourquoi signer un message ?	16
5.2	Méthode utilisée pour signer un message	16



5.3	Exemples de code	17
5.3.1	Php 5	17
5.3.2	Java	17
5.3.3	.net	18
6	Description des messages	19
6.1	Requête de paiement	19
6.1.1	Champs génériques	19
6.1.2	Champs optionnels relatifs à la fraude	20
6.1.3	Champs optionnels pour le paiement en N fois	20
6.1.4	Champs optionnels relatifs aux pages de paiement	21
6.1.5	Champs optionnels relatifs aux moyens de paiement	21
62	Réponses (automatiques et manuelles)	21



Historique des versions

Date	Version	Description	
05/03/2013	1.0	Création du document	
07/05/2013 1.1 Ajou		Ajout des champs de la version 2.3 des connecteurs	

R:\RP Remote Payment\300 - Doc Applicative\Référentiel Documentation Clients\Parties génériques\Préface.docx



1 Introduction

1.1 Présentation de Sips

R:\RP Remote Payment\300 - Doc Applicative\Référentiel Documentation Clients\Parties génériques\à propos de Sips.docx

1.2 Objectif du document

Le présent document a pour objectif d'expliquer comment implémenter la solution de paiement Sips Payment Direct et comment démarrer les premiers tests de paiement.

Ce document s'adresse à tous les commerçants souhaitant souscrire à l'offre Sips et voulant bénéficier d'un connecteur basé sur des échanges HTTP(s) POST entre les sites web des commerçants et les serveurs Sips, tout en utilisant la passerelle Sips Payment Direct.

Ce connecteur se veut aussi plug-and-play que possible pour le commerçant.

1.3 Prérequis

Une connaissance de base des standards des langages de développement Web de l'industrie comme Java, Php ou .Net est nécessaire pour développer un client se connectant à la passerelle Sips Payment Direct.

Grâce à cette solution, les messages échangés entre le site Web du commerçant et les serveurs Sips sont sécurisés à l'aide de clés secrètes.

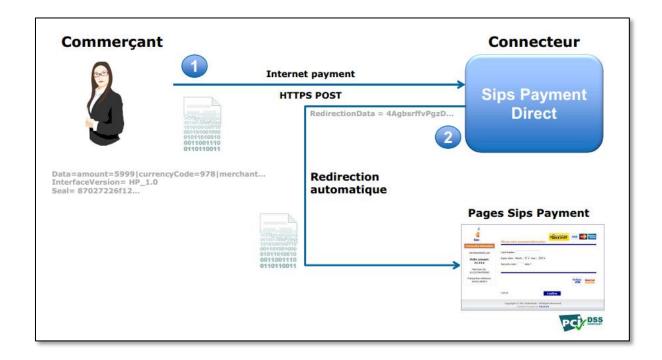
Ces clés doivent être stockées et gérées de façon sécurisée à la charge du commerçant.

Si la clé est compromise ou soupçonnée de l'être, il est de la responsabilité du commerçant de renouveler sa clé secrète via l'extranet Sips Download.



2 Cinématique de paiement

2.1 Principe général

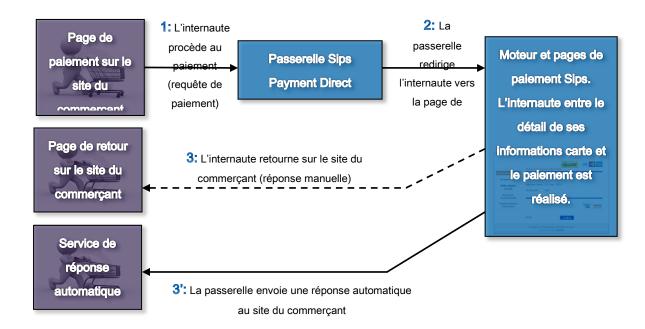


- 1 : Lorsque l'internaute valide son panier, il est redirigé vers les serveurs de Sips Payment. La requête de paiement est alors vérifiée et si elle est valide, elle est cryptée (nommé RedirectionData dans le schéma).
- 2 : L'internaute est alors redirigé de façon automatique vers les pages Sips Payment avec la requête cryptée. Cette dernière est décryptée et la page de Sips Payment invite l'internaute à saisir les informations de son moyen de paiement.



2.2 Flux de paiement

Il y a 3 flux à implémenter entre le site Web du commerçant et le serveur de paiement pour intégrer la solution.



Flux 1: Une fois que l'internaute procède à l'étape de paiement, une requête de paiement doit être envoyée à la passerelle Sips Payment Direct. L'URL de la passerelle est fournie au commerçant par ATOS. La meilleure façon de gérer cet appel est d'envoyer un formulaire en mode POST HTTPS, mais toute autre solution qui enverrait une requête POST HTTPS conviendrait.

Flux 2: La passerelle Sips Paiement Direct va rediriger l'application appelante vers les pages de paiement Sips. L'internaute devra entrer les coordonnées du moyen de paiement pour que le serveur de paiement Sips traite la transaction. A noter que la saisie des coordonnées de paiement peut être faite directement sur le serveur émetteur du moyen de paiement (exemple : Credit Transfert ou Paypal). A la fin du processus de paiement, qu'il soit réussi ou pas, deux réponses sont construites et envoyées aux URLs de réponses fournies au travers du flux 1.

Il y a deux processus séparés de notification :

• Flux 3: Les réponses manuelles sont envoyées en mode HTTP(S) POSTs par le serveur de paiement vers l'URL de réponse normale renseignée dans la requête de paiement quand l'internaute clique sur "Retour à la boutique" sur la page de paiement. C'est pourquoi l'URL de réponse normale est également la page de destination où l'internaute est redirigé à la fin du paiement. Il n'y a aucune



garantie que l'internaute clique sur ce lien et donc que la réception de la *réponse manuelle* soit garantie.

Flux 3': Les réponses automatiques sont envoyées indépendamment des réponses manuelles. Elles utilisent aussi des requêtes HTTP(s) POSTs envoyées par les serveurs de paiement Sips mais au travers de l'URL de réponse automatique renseignée dans la requête de paiement dans ce cas. Cela a pour conséquence que le commerçant va toujours recevoir cette réponse dès que le paiement est terminé sur les pages de paiement Sips.

Si le paiement échoue et après que l'internaute eut été redirigé sur le site web du commerçant, il n'y a aucun moyen de retourner sur les pages de paiement Sips pour réessayer ou corriger les détails des informations cartes. C'est de la responsabilité du site web du commerçant d'initialiser une nouvelle requête epaiement en commençant par un appel au connecteur Sips Payment Direct.



3 Description du protocole

3.1 Champs POST

3 champs obligatoires sont fournis dans les requêtes et réponses de paiement.

Data Contient toutes les informations relatives à la transaction, rassemblées dans une

chaine de caractère comme expliqué en § 3.1.1

InterfaceVersion Version de l'interface du connecteur.

Seal Utilisé pour valider l'intégrité des données échangées. Le champ Seal est calculé à

partir du champ Data et de la clé secrète comme expliqué en § 3.1.2

Un champ optionnel supplémentaire est disponible :

Encode Donne l'encodage utilisé dans le champ Data comme expliqué en § 3.1.3

A noter

Les noms des champs sont sensibles à la casse.

3.1.1 Syntaxe du champ Data

Le champ Data est construit suivant le format suivant :

<field name>=<value name>|<field name>=<value name>|<field name>=<value name>...

Tous les champs nécessaires à la transaction (voir le détail dans le dictionnaire des données) doivent être présents dans cette chaine de caractère. L'ordre des champs n'a pas d'importance.

Exemple de requête de paiement :

 $amount=2355 \mid \texttt{currencyCode}=978 \mid \texttt{merchantId}=011223744550001 \mid \texttt{normalReturnUrl}=\texttt{http://www.normalreturnurl.com/transactionReference}=534654 \mid \texttt{keyVersion}=1$

3.1.2 Syntaxe du champ Seal

La valeur du champ Seal est construite comme suit :

 Concaténation du champ Data et de la clé secrète (encodée si l'option d'encodage est utilisée. Voir § 3.1.3)



- Obtention de l'encodage UTF-8 des données du résultat précédent
- Cryptage SHA256 des octets obtenus

Ce procédé peut être résumé comme suit :

SHA256(UTF-8(Data+secretKey))

3.1.3 Syntaxe du champ Encode

Dans le cas où des caractères spéciaux peuvent apparaître dans le champ Data, la valeur de ce champ doit être encodée.

Deux formats d'encodage peuvent être utilisés : base64 ou base64Url.

Il est à noter que comme un calcul de la signature est réalisé sur le champ Data, quand un encodage est appliqué, c'est la valeur encodé du champ Data qui est utilisée pour ce calcul.



4 Comment implémenter un paiement ?

4.1 Requête de paiement

La requête de paiement est un appel en mode HTTP POST vers le connecteur de la passerelle de paiement. La façon la plus simple de réaliser cet appel est un formulaire HTML utilisant la méthode POST.

4.1.1 Champs de la requête de paiement

Toutes les données de la requête de paiement doivent être fournies comme décrit au chapitre § 3.

InterfaceVersion doit être fixée à HP_2.3.

Le dictionnaire des données et le chapitre de description des messages décrivent tous les paramètres de la requête de paiement, leur format ainsi que leur caractère obligatoire ou facultatif.

4.1.2 Exemple

Un exemple de formulaire est présenté ci-dessous :

4.1.3 Gestion des erreurs

Tous les champs reçus au travers du connecteur de la passerelle Sips Payment Direct sont vérifiés individuellement. La liste des messages d'erreur qui peuvent être affichés lors de cette étape de vérification ainsi que les solutions à apporter sont décrites dans le tableau ci-dessous.

A noter

Les messages ne sont affichés que sur la plateforme de simulation, pour valider l'intégration du site web du commerçant. Pour des raisons de sécurité, sur la plateforme de production, un message d'erreur beaucoup plus simple est affiché lors de problèmes comme "Erreur lors du traitement de la requête de paiement. Contacter le commerçant.



Message	Cause	Solution
Version d'interface inconnue :	La valeur <version> du champ POST</version>	Vérifiez la version d'interface
<version></version>	InterfaceVersion est inconnue	dans ce guide utilisateur
Mot clé invalide : <nom param<="" td=""><td>La requête contient un paramètre <nom< td=""><td>Vérifiez les paramètres de la</td></nom<></td></nom>	La requête contient un paramètre <nom< td=""><td>Vérifiez les paramètres de la</td></nom<>	Vérifiez les paramètres de la
>= <valeur param=""></valeur>	param> qui n'est pas attend dans la	requête de paiement dans le
	requête de paiement	dictionnaire des données
Taille de champ invalide : <nom< td=""><td>La valeur du paramètre <nom param=""> n'a</nom></td><td>Vérifiez la longueur des</td></nom<>	La valeur du paramètre <nom param=""> n'a</nom>	Vérifiez la longueur des
param >= <valeur param=""></valeur>	pas la bonne longueur	paramètres de la requête de
		paiement dans le dictionnaire
		des données
Valeur de champ invalide :	La valeur du paramètre <nom du="" param=""></nom>	Vérifiez le format des
<nom param="">=<valeur param=""></valeur></nom>	n'a pas le bon format	paramètres de la requête de
		paiement dans le dictionnaire
		des données
Champ obligatoire manquant :	Le paramètre obligatoire <nom du="" param<="" td=""><td>Vérifiez les paramètres</td></nom>	Vérifiez les paramètres
<nom param=""></nom>	>est manquant dans la requête de	obligatoires pour la requête de
	paiement	paiement dans le dictionnaire
		des données.
Version de sécurité inconnue :	La valeur <version> du paramètre</version>	Vérifiez les versions de clés
<version></version>	keyVersion est inconnu	disponibles dans l'interface
		commerçant
Signature invalide	La vérification de la signature de la requête	Vérifiez les règles de calcul de
	de paiement a échoué. Cela peut être du à	la signature dans le
	un mauvais calcul de la signature ou cela	dictionnaire des données.
	indique que certains champs ont été	
	falsifiés après calcul de la signature.	
Transaction déjà traitée :	Une requête de paiement avec le même	Assurez-vous que le
<transaction reference=""></transaction>	transactionReference a déjà été reçue et	transactionReference est
	traitée par les serveurs Sips	unique pour une transaction
		donnée
Autres messages		Contactez le support technique



4.2 Réponses de paiement

Il y a deux types de réponses. Bien que les protocoles, formats et contenus des deux réponses soient exactement les mêmes, les deux réponses doivent être gérées différemment car elles répondent à deux besoins.

4.2.1 Réponse manuelle

L'objectif principal de la réponse manuelle est de rediriger l'internaute vers le site web du commerçant avec le résultat du paiement, afin que le commerçant puisse prendre la bonne décision en ce qui concerne son client. Par exemple, en cas d'erreur, il peut proposer de retenter le paiement et relancer le processus, alors que dans le cas d'un succès, il peut afficher un message de remerciement et commencer à livrer la marchandise, le cas échéant.

La dernière étape du processus de paiement sur Sips Payment est d'afficher un lien de redirection au client. Lorsque l'internaute clique sur ce lien, le serveur Sips le redirige vers l'URL contenue dans le champ normalReturnUrl qui est fourni au début du processus de paiement. La redirection est une demande HTTP POST qui contient les paramètres de réponse comme décrit en § 4.2. C'est de la responsabilité du commerçant de récupérer ces paramètres, vérifiez la signature pour assurer de l'intégrité des données de la réponse et afficher les messages pertinents à son client en relation avec les détails de la réponse.

Il est important de noter que la réception de la réponse n'est pas garantie vue qu'elle est envoyée par le navigateur de l'internaute. Fondamentalement, l'utilisateur final a le choix de ne pas cliquer du tout sur le lien, ou la connexion internet de l'internaute pourrait avoir simplement un problème et bloquer l'envoi de cette réponse. Par conséquent les processus métier du commerçant ne doivent pas se reposer uniquement sur cette réponse.

Les noms des paramètres utilisés dans la réponse manuelle sont sensibles à la casse.

La version actuelle de **InterfaceVersion** est **HP_2.3**. Veuillez consulter le dictionnaire de données Sips pour une description complète des paramètres figurant dans la réponse.



4.2.2 Réponse automatique

Une réponse automatique est envoyée uniquement si le champ automaticResponseUrl a été envoyé dans la requête de paiement. Si c'est le cas, le serveur Sips envoie une réponse de type http POST vers l'URL reçue. Les champs de la réponse sont identiques à ceux envoyés vers la réponse manuelle. La seule différence entre les deux processus est que la réponse automatique est envoyée directement par le serveur Sips sans passer par le navigateur de l'internaute. En conséquence, il est beaucoup plus fiable car il sera toujours envoyé. Une autre conséquence est que le processus en charge de la réception de cette réponse ne doit pas essayer de répondre à l'application appelante. Fondamentalement, le serveur Sips n'attend aucune réponse suite à l'envoi de la réponse automatique.

Comme pour la réponse manuelle, les champs de la réponse automatique sont décrits en § 4.2. C'est de la responsabilité du commerçant de récupérer les paramètres de la réponse, de les enregistrer sous forme cryptée, de vérifier la signature pour s'assurer de l'intégrité des champs de la réponse et donc mettre à jour son backoffice commerçant en conséquence.

Les noms des paramètres utilisés dans la réponse automatique sont sensibles à la casse.

La version actuelle de **InterfaceVersion** est **HP_2.3**. Veuillez consulter le dictionnaire de données Sips pour une description complète des paramètres figurant dans la réponse.

4.2.3 Problèmes de réception des réponses Sips

Vous trouverez ci-dessous une liste des problèmes les plus couramment rencontrés empêchant la réception des réponses automatiques et manuelles. Veuillez-vous assurer de les avoir vérifié avant de déclencher un appel avec le service d'assistance technique.

- Vérifiez que les URLs de réponses sont fournies dans la requête de paiement et sont valides. Vous pouvez tout simplement copier/coller les URLs dans votre navigateur pour vérifier leur validité.
- Les URLs fournies doivent être accessible depuis l'extérieur et donc Internet. Un contrôle d'accès (login/mot de passe ou filtre IP) ou un pare-feu peut bloquer l'accès à votre serveur.
- Un accès aux URLs de réponses doit apparaitre dans le journal de notification de votre serveur web.
- Si vous utilisez un port non standard, il doit être dans la plage 80 à 9999 pour être compatible avec
 Sips.



 Vous ne pouvez pas ajouter de paramètres contextuels aux URLs de réponses. Le champ orderld est réputé pour recevoir ces paramètres additionnels, ou alternativement un sessionld qui permettra au commerçant de retrouver les informations du client à la fin du processus de paiement.

4.2.4 Gestion des erreurs- Pas de signature dans la réponse

Il y a des cas d'erreurs où le serveur Sips n'est pas en capacité de signer le message de réponse. Par exemple, lors d'une erreur « MerchantID inconnu » ou si la clé secrète est inconnue du référentiel Sips.

Pour ces raisons particulières, le serveur de paiement enverra la réponse sans signature dans le champ Seal.



5 Comment SIGNER un MESSAGE ?

5.1 Pourquoi signer un message?

Les paramètres de la transaction (requête de paiement) sont envoyés au travers du navigateur de l'internaute. Théoriquement, il est possible à un hacker de modifier les paramètres durant la transmission des données vers le serveur de paiement.

Il est par conséquent nécessaire d'ajouter de la sécurité pour assurer l'intégrité des paramètres transmis de la transaction. La solution Sips répond à ce besoin par l'échange de signatures.

Un contrôle réussi des signatures implique deux choses :

- L'intégrité des messages requête et réponse, pas d'altération durant l'échange
- L'authentification de l'émetteur et du receveur car ils partagent la même clé secrète.

Si la clé utilisée pour la signature est compromise ou suspectée de l'être, il est de la responsabilité du commerçant de demander le renouvellement de la clef en se connectant à Sips Download.

5.2 Méthode utilisée pour signer un message

L'opération de signature est réalisée en calculant une valeur cryptée à partir des paramètres de la transaction (le champ Data) auxquels on ajoute la clé secrète (inconnue de l'internaute). Toutes les chaînes de caractères sont converties en UTF8 avant d'être cryptées.

L'algorithme de cryptage (SHA256) produit un résultat non réversible. D'une manière générale, sur réception d'un tel message, le réceptionneur du message doit recalculer la valeur cryptée pour la comparer avec la valeur reçue. Toute différence indique que les données échangées ont été falsifiées.

Le résultat doit être envoyé sous forme hexadécimale dans le champ POST nommé Seal.



5.3 Exemples de code

5.3.1 Php 5

```
<?php
echo hash('sha256', $data.$secretKey);
?>
```

Data et secretKey doivent utiliser un jeu de caractère UTF-8. Se référer à la fonction **utf8_encode** pour convertir d'ISO-8859-1 vers UTF-8.

5.3.2 Java

```
import java.security.MessageDigest;
public class ExampleSHA256 {
          * table to convert a nibble to a hex char.
        static final char[] hexChar = {
            '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};
         * Fast convert a byte array to a hex string
         * with possible leading zero.
* @param b array of bytes to convert to string
         * @return hex representation, two chars per byte.
        public static String encodeHexString ( byte[] b )
            StringBuffer sb = new StringBuffer( b.length * 2 );
            for ( int i=0; i < b.length; i++ )</pre>
               // look up high nibble char
               sb.append( hexChar [( b[i] & 0xf0 ) >>> 4] );
               // look up low nibble char
               sb.append( hexChar [b[i] & 0x0f] );
            return sb.toString();
         * Computes the seal
         * @param Data the parameters to cipher
         * @param secretKey the secret key to append to the parameters
* @return hex representation of the seal, two chars per byte.
        public static String computeSeal(String Data, String secretKey) throws Exception
          MessageDigest md = MessageDigest.getInstance("SHA-256");
          md.update((Data+secretKey).getBytes("UTF-8"));
           return encodeHexString(md.digest());
```



5.3.3 .net

(Réalisé en utilisant un simple formulaire nommé "Form1" contenant 2 champs texte pour la saisie : txtSips, txtSecretKey et un autre pour l'affichage : lblHEX)

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System. Windows. Forms;
using System.Security.Cryptography;
namespace ExampleDotNET
    public partial class Form1 : Form
        public Form1()
        {
            InitializeComponent();
        private void cmdGO Click(object sender, EventArgs e)
            String sChaine = txtSips.Text + txtSecretKey.Text;
            UTF8Encoding utf8 = new UTF8Encoding();
           Byte[] encodedBytes = utf8.GetBytes(sChaine);
            byte[] shaResult;
            SHA256 shaM = new SHA256Managed();
            shaResult = shaM.ComputeHash(encodedBytes);
            lblHEX.Text = ByteArrayToHEX(shaResult);
        }
        private string ByteArrayToHEX(byte[] ba)
            StringBuilder hex = new StringBuilder(ba.Length * 2);
            foreach (byte b in ba)
               hex.AppendFormat("{0:x2}", b);
            return hex.ToString();
```

D:\My Documents\Private\Acceptance\BEECOM\05 Qualification\02 Provisional and Final Acceptance\Webshop integration\Sips Payment Direct - How to test Go Live_BENELUX_HP 2.3_1.1.docx



6 Description des messages

6.1 Requête de paiement

6.1.1 Champs génériques

Nom du champ	Présence (O/F)	Depuis la version	Commentaire
currencyCode	0	HP_1.0	
merchantId	0	HP_1.0	
normalReturnUrl	0	HP_1.0	
Amount	0	HP_1.0	
transactionReference	0	HP_1.0	
keyVersion	0	HP_1.0	
automaticResponseUrl	F	HP_1.0	
captureDay	F	HP_1.0	
captureMode	F	HP_1.0	
customerId	F	HP_2.0	
customerlpAddress	F	HP_2.1	
customerLanguage	F	HP_1.0	
expirationDate	F	HP_1.0	
fraudData	F	HP_2.2	Voir ci-dessous
hashAlgorithm1	F	HP_2.3	
hashAlgorithm2	F	HP_2.3	
hashSalt1	F	HP_2.1	
hashSalt2	F	HP_2.1	
installmentData	F	HP_2.2	Voir ci-dessous
invoiceReference	F	HP_2.0	
merchantSessionId	F	HP_2.0	
merchantTransactionDateTime	F	HP_2.0	
merchantWalletID	F	HP_2.2	
orderChannel	F	HP_2.1	
orderld	F	HP_1.0	
paymentMeanBrandList	F	HP_1.0	
paymentMeanData	F	HP_2.2	Voir ci-dessous
paypageData	F	HP_2.0	Voir ci-dessous
paymentPattern	F	HP_2.1	
returnContext	F	HP_2.0	



Nom du champ	Présence (O/F)	Depuis la version	Commentaire
statementReference	F	HP_2.3	
templateName	F	HP_2.1	
transactionActors	F	HP_2.2	
transactionOrigin	F	HP_2.0	

Tableau 1: Requête de paiement

6.1.2 Champs optionnels relatifs à la fraude

Champs	Présence (O/F)	Depuis la version	Commentaire
fraudData.allowedCardArea	F	HP_2.1	
fraudData.allowedCardCountryList	F	HP_2.1	
fraudData.allowedIpArea	F	HP_2.1	
fraudData.allowedIpCountryList	F	HP_2.1	
fraudData.bypass3DS	F	HP_2.1	
fraudData.bypassCtrlList	F	HP_2.1	
fraudData.bypassInfoList	F	HP_2.1	
fraudData.deniedCardArea	F	HP_2.1	
fraudData.deniedCardCountryList	F	HP_2.1	
fraudData.deniedIpArea	F	HP_2.1	
fraudData.deniedIpCountryList	F	HP_2.1	

Tableau 2: Détail des champs de fraude

6.1.3 Champs optionnels pour le paiement en N fois

Champs	Présence (O/F)	Depuis la version	Commentaire
instalmentData.number	F	HP_2.2	
instalmentData.datesList	F	HP_2.2	
instalmentData.transactionReferencesList	F	HP_2.2	
instalmentData.amountsList	F	HP_2.2	

Table 3: Champs relatifs au paiement récurrent



6.1.4 Champs optionnels relatifs aux pages de paiement

Champs	Présence (O/F)	Depuis la version	Commentaire
paypageData.bypassReceiptPage	F	HP_2.0	

Table 4: Détail des champs sur le comportement des pages de paiement

6.1.5 Champs optionnels relatifs aux moyens de paiement

Pour Paypal

Champs	Présence (O/F)	Depuis la version	Commentaire
paymentMeanData.paypal.landingPage	F	HP_2.2	
paymentMeanData.paypal.addrOverride	F	HP_2.2	
paymentMeanData.paypal.invoiceId	F	HP_2.2	
paymentMeanData.paypal.dupFlag	F	HP_2.2	
paymentMeanData.paypal.dupDesc	F	HP_2.2	
paymentMeanData.paypal.dupCustom	F	HP_2.2	
paymentMeanData.paypal.dupType	F	HP_2.2	
paymentMeanData.paypal.mobile	F	HP_2.2	

Table 5: Champs relatifs à Paypal

Pour SDD

Champs	Présence (O/F)	Depuis la version	Commentaire
paymentMeanData.sdd.mandateAuthentMet hod	F	HP_2.2	
paymentMeanData.sdd.mandateUsage	F	HP_2.2	

Table 6: Champs relatifs au SDD

6.2 Réponses (automatiques et manuelles)

Le contenu des réponses automatiques et manuelles de Sips Payment Web sont identiques. Le contenu luimême peut varier suivant le résultat du paiement (avec succès ou non).

Champs	Présence (O/F)	Depuis la version	Commentaire
acquirerResponseCode	0	HP_2.0	
Amount	0	HP_1.0	Valeur passée à la requête de paiement.
autorisationId	0	HP_1.0	Valeur passée à la requête de paiement.



Champs	Présence (O/F)	Depuis la version	Commentaire
captureDay	0	HP_1.0	Valeur passée à la requête de paiement.
captureLimiteDate	F	HP_2.3	
captureMode	0	HP_1.0	Valeur passée à la requête de paiement.
cardCSCResultCode	0	HP_2.0	
complementaryCode*	F	HP_1.0	
complementaryInfo*	F	HP_2.0	
currencyCode	0	HP_1.0	Valeur passée à la requête de paiement.
customerEmail	0	HP_2.0	Valeur passée à la requête de paiement.
			Uniquement disponible sur la version
			HP_2.0
customerId	0	HP_2.0	Valeur passée à la requête de paiement.
customerlpAddress	0	HP_2.0	Valeur passée à la requête de paiement.
customerMobilePhone	0	HP_2.1	Valeur passée à la requête de paiement.
			Uniquement disponible sur la version
			HP_2.1
dccStatus*	F	HP_2.3	
dccResponseCode*	F	HP_2.3	
dccAmount*	F	HP_2.3	
dccExchangeRate*	F	HP_2.3	
dccProvider*	F	HP_2.3	
dccCurrencyCode*	F	HP_2.3	
guaranteeIndicator	F	HP_2.0	
hashPan1	F	HP_2.3	
hashPan2	F	HP_2.3	
holderAuthentMethod*	F	HP_2.4	
holderAuthentRelegation*	F	HP_2.0	
holderAuthentStatus*	F	HP_2.0	
interfaceVersion*	F	HP_1.0	
issuerEnrollementIndicator*	F	HP_2.0	
keyVersion	0	HP_1.0	Valeur passée à la requête de paiement.
maskedPan*	F	HP_1.0	
merchantId	0	HP_1.0	Valeur passée à la requête de paiement.
merchantSessionId	0	HP_2.0	Valeur passée à la requête de paiement.
merchantTransactionDateTime	0	HP_2.0	Valeur passée à la requête de paiement.
merchantWalletID	0	HP_2.0	Valeur passée à la requête de paiement.
orderChannel	0	HP_2.0	Valeur passée à la requête de paiement.
orderld	0	HP_1.0	Valeur passée à la requête de paiement.
panEntryMode**	F	HP_2.4	



Champs	Présence (O/F)	Depuis la version	Commentaire
panExpiryDate*	F	HP_2.0	
paymentMeanBrand*	F	HP_1.0	
paymentMeanData.sdd*	F	HP_2.3	
paymentMeanType*	0	HP_1.0	
paymentPattern	0	HP_2.0	Valeur passée à la requête de paiement.
responseCode	0	HP_1.0	
returnContext	F	HP_1.0	Valeur passée à la requête de paiement.
scoreColor*	F	HP_2.0	
scoreInfo*	F	HP_2.0	
scoreProfile*	F	HP_2.0	
scoreThreshold*	F	HP_2.0	
scoreValue*	F	HP_2.0	
statementReference*	F	HP_2.4	Valeur passée à la requête de paiement
tokenPan*	F	HP_2.0	
transactionActor	0	HP_2.2	Valeur passée à la requête de paiement.
transactionDateTime	0	HP_1.0	
transactionOrigin	0	HP_2.0	Valeur passée à la requête de paiement.
transactionReference	0	HP_1.0	Valeur passée à la requête de paiement.
walletType	F	HP_2.4	

Table 7: Champ de la réponse automatique / manuelle du paiement

^{*:} ces champs sont fournis lorsqu'ils sont disponibles, selon le statut de la transaction et le moyen de paiement qui a été choisi