PARAMETRE DE CONFIGURATION DE L'API SMSECO V 2.0

Vous atteignez les mêmes objectifs en envoyant vos messages depuis :

- Votre Application
- Votre site web

Si vous disposez d'une solution logicielle ou applicative dans laquelle vous souhaiteriez intégrer ou automatiser l'envoi de SMS, SMSECO répondra à vos attentes.

SOMMAIRE

I. ENVOI DE SMS

L'API SMSECO permet d'automatiser l'envoi de sms vers un ou plusieurs destinataires.
Ces messages peuvent être envoyés immédiatement ou en programmé.
Nous mettons à votre disposition plusieurs formats d'envoi de vos sms

I-1. XML

L'URL utilisée pour écrire les données au format XML est la suivante:

http://www.smseco.com/api/xml/sendsms/

L'URL de base a changé, mais l'ancienne URLhttp://www.smseco.com/XML/XMLdatas.php reste toujours valable.

Les numéros en double dans une requête POST seront filtrés par le système ; ainsi, une personne ne peut pas recevoir plus de 1 message par demande.

I-1-1. Structure de la requête xml

```
XML=<SMS>
        <compte>
            <login></login>
            <password></password>
        </compte>
        <message>
            <msgid></msgid>
            <expediteur></expediteur>
            <msg></msg>
            <flash></flash>
            <unicode></unicode>
            <binaire></binaire>
            <datesend></datesend>
            <mwi></mwi>
            <pid></pid>
            <rpi></rpi>
            <alt_dcs></alt_dcs>
            <udhdata></udhdata>
        </message>
        <destinataires>
            <numero></numero>
            <numero></numero>
            <numero></numero>
        </destinataires>
</SMS>
```

La longueur de ***msgid*** doit être au maximum 30 caractères. Les paramètres en vert sont optionnels.

Paramètres utilisés

Paramètre		Description
compte	login	Le nom d'utilisateur de votre compte sur smseco.com. Il s'agit d'une adresse email ou d'un numéro de mobile ou de de votre identifiant smseco
	password	Le mot de passe relatif à votre compte.
message	msgid	Identifiant du message, utilisé pour la livraison des rapports d'envoi
	expediteur	expéditeur du message. Alphanumérique et 11 caractères maxi Si vide, c'est le numéro utilisé lors de l'inscription qui apparait par défaut.
	msg	Le texte du message à envoyer. si votre message fait plus de 160 caractères (standard SMS), il sera séparéen plusieurs messages et regroupé sur le téléphone du destinataire en unseul. Votre compte sera débité en conséquence. **N.B**. : le sms doit être en urlencode
	flash	Peut être 0 ou 1 0 : message envoyé en mode normal Si non nul : message envoyé en mode flash Le mode flash signifie l'ouverture immédiate du message chez le destinataire. Le message n'est pas stocké dans sa messagerie.
	unicode	Peut être 0 ou 1 0 : message envoyé au format standard Les caractères supportés sont les suivants: **!#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ_ abcdefghijklmnopqrstuvwxyz£¤¥§¿ÄÅÆÇÉÑÖØÜßàäåæèéìñòöøùü** Si non nul : message envoyé en format unicode . il prend en compte tous les caractères qui ne figurent pas dans la liste ci-dessus. La limitation de caractère pour un message en unicode est de 70.
	binaire	Peut être 0 ou 1 Si non nul : message envoyé en binaire
	datesend	Date d'envoi du message. Format : AAAA-MM-JJ HH:MM (Envoi différé)
	mwi	
	pid	
	rpi	
	alt_dcs	
	udhdata	
destinataires	numero	Le destinataire du message.

Réponse XML

Après le XML POST initié par le client, certains codes d'état sont disponibles.
La structure retour de la requête XML :

```
<REPONSE>
        <msgfid></msgfid>
        <statut></statut>
        <texte></texte>
        <destinataires></destinataires>
        <sms></sms>
        <credits></credits>
        <solde></solde>
        <messages>
                <sms>
                        <smsfid></smsfid>
                        <destination></destination>
                        <credits></credits>
                </sms>
        </messages>
</REPONSE>
```

Paramètres

Paramètre		Description
msgfid		Identifiant de l'envoi
statut		Si >0 Nombre de destinataires valides
texte		Texte expliquant le statut de l'envoi
destinataires		Nombre total de destinataires valides
sms		Nombre total de sms
credits		Coût de l'envoi en crédit (qui est l'unité de mesure de smseco)
Solde		Nbre de crédit restant
Messages (sms)	smsfid	Identifiant d'un sms
	destination	Le numéro du destinataire valide
	credits	Coût total du sms en crédit pour ce destinataire

N.B. : Nous aurons autant de nœud **sms** dans **messages** que de destinataires valides.

En cas d'erreur, voici la structure du XML retour :

```
<REPONSE>
<statut></statut>
<texte></texte>
</REPONSE>
```

Paramètres

Paramètres	Description
statut	Une valeur < 0 qui représente le code de l'erreur
texte	Explication du code erreur

Codes d'erreur

Statut	Signification
-1	XML invalide
-2	Login non spécifié
-3	Mot de passe non spécifié
-4	Echec de l'Authentification
-5	Aucun message spécifié
-6	Pas assez de crédits ou crédits insuffisants
-7	Aucun destinataire spécifié
-8	Expéditeur non enregistré
-9	Aucuns destinataires valides
-10	La longueur de l'Id du message dépasse les 30 caractères autorisés
-11	Envoi de sms non autorisé pour ce compte
-12	Compte non actif dans le système
-13	Heure d'envoi pour message différé incorrecte
-14	Commande inconnue
-15	Aucun statut sms disponible

I-2. HTTP GET

Structure de l'URL pour un envoi de message par GET

http://www.smseco.com/api/{format-sorti}/sendsms?login=xx&password=xx&expediteur=xx&msgid=xx&msg=xx&numero=xx&datesend=AAAA-MM-JJ HH :MM&flash=0&binaire=0&unicode=0

{format-sorti}peut prendre les valeurs suivantes : xml, json ou html

Les paramètres sont les même utilisés que ceux du **I-1 XML**

Pour un envoi vers plusieurs numéros, les numéros sont séparés par des virgules. Pour le format des numéros l'indicatif n'est pas obligatoire pour les numéros situés dans le même pays que l'expéditeur. Pour les numéros internationaux, renseigner selon le format **indicatif+numero**

(exemple : 22678657808)

N.B. : Pour un envoi massif, nous ne vous conseillons pas la méthode http GET compte tenu de la limitation de données par get

I-3. JSON

Les développeurs qui préfèrent le JSON seront servis avec cette version de l'API. En effet cette version de l'API de SMSECO prévoit le format JSON.

L'URL utilisée pour écrire les données au format JSON est la suivante:

http://www.smseco.com/api/json/sendsms/

I-3-1. Structure de la requête JSON

JSON={"compte":{"login":"xxx","password":"xxx"},"message":{"expediteur":"xx","msgid":"xx","msg":"xx","date send":"xx","flash":"" ,"unicode":"","binaire":""},"destinataires":[{"numero":"xx"}, {"numero":"xx"}]}

Les paramètres utilisés sont les mêmes que pour le format XML.

Réponse JSON

Après le JSON POST initié par le client, certains codes d'état sont disponibles.
La structure retour de la requête JSON :
{"msgfid":"xx","statut":x,"texte":"xx","destinataires":"xx","sms":"xx","credits":"xx","solde":"xx","messages":[{" smsfid":"xx","destination":"xx","credits":xx}]}

N.B. : Les paramètres retour sont les mêmes que pour le XML

En cas d'erreur, voici la structure du JSON retour :

{"statut":xx,"texte":"xx"}

I-3-2. Exemple d'un envoi au format JSON en CURL

```
<?php
$uri = "http://www.smseco.com/api/json/sendsms/";

$data =
"JSON={\"compte\":{\"login\":\"toto@toto.fr\",\"password\":\"123456\"},\"message\":{\"expediteur\":\"TOT
O\",\"msgid\":\"15\",\"msg\":\"Juste un  est\"},\"destinataires\":[{\"numero\":\"05442710\"},
{\"numero\":\"07352518\"}]}";

$ch = curl_init();

curl_setopt($ch, CURLOPT_URL,$uri);

curl_setopt($ch,CURLOPT_HTTPHEADER,array("Accept: application/json","Accept: application/json","Content-
Type: application/json", "Content-Length: ". strlen($data)));

curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $data);

curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$output = curl_exec ($ch);
curl_close ($ch); // close curl handle
```

?>

I-4. SOAP

La communication avec SMSECO se fait également à l'aide de webservices, développés suivant le protocole SOAP (Simple Object Access Protocol) et décrit dans le fichier
http://www.smseco.com/api/soap/wsdl/

Exemple d'envoi de SMS avec SOAP php

```
< ?php

$wsdl = "http://www.smseco.com /api/soap/wsdl/";

$server = "http://www.smseco.com/api/soap/server/";

$options = array("location" =>$server, "trace"=>true, 'style'=> SOAP_DOCUMENT, 'use'=> SOAP_LITERAL);

$api = new SoapClient($wsdl, $options);

$smsdatas = array("login"=>"particulier@smseco.com", "password"=>"123456", "msgid"=>"9876543210",
"msg"=>"Bonjour, Ceci est juste un test .", "expediteur"=>"TEST", "datesend"=>"2015-06-21 16:49",
"to"=>array("07352518", "05442710", "08559001"));

$api->Sendsms($smsdatas);

?>
```

II. ENVOI DE SMS AVEC PUBLIPOSTAGE

Le Publipostage vous permet de personnaliser vos messages.

II-1. XML

L'URL utilisée pour écrire les données au format XML est la suivante:
http://www.smseco.com/api/xml/sendsms/

II-1-1. Structure de la requête xml

```
XML=<SMS>
        <compte>
            <login></login>
            <password></password>
        </compte>
        <message>
            <msgid></msgid>
            <expediteur></expediteur>
            <msg>Bonjour {nom}, votre solde est {solde}</msg>
            <flash></flash>
            <unicode></unicode>
            <binaire></binaire>
            <datesend></datesend>
            <mwi></mwi>
            <pid></pid>
            <rpi></rpi>
```

```
            <alt_dcs></alt_dcs>
            <udhdata></udhdata>
        </message>
        <destinataires>
            <sms>
                <numero></numero>
                <datas>
                    <nom> le nom du premier contact</nom>
                    <solde> le solde du premier contact</solde>
                </datas>
            </sms>
            <sms>
                <numero></numero>
                <datas>
                    <nom> le nom du deuxième contact</nom>
                    <solde> le solde du deuxième contact</solde>
                </datas>
            </sms>
            <sms>
                <numero></numero>
                <datas>
                    <nom> le nom du premier contact</nom>
                    <solde> le solde du premier contact</solde>
                </datas>
            </sms>

        </destinataires>
</SMS>
```

Paramètres utilisés

Paramètre			Description
compte	login		Le nom d'utilisateur de votre compte sur smseco.com. Il s'agit d'une adresse email ou d'un numéro de mobile ou de de votre identifiant smseco
	password		Le mot de passe relatif à votre compte.
message	msgid		Identifiant du message, utilisé pour la livraison des rapports d'envoi
	expediteur		expéditeur du message. Alphanumérique et 11 caractères maxi Si vide, c'est le numéro utilisé lors de l'inscription qui apparait par défaut.
	msg		Le texte du message à envoyer. si votre message fait plus de 160 caractères (standard SMS), il sera séparé en plusieurs messages et regroupé sur le téléphone du destinataire en un seul. Votre compte sera débité en conséquence. Il devra contenir les variables qui seront remplacées pour chaque destinataire. Exemple : Bonjour {nom} **N.B. : les variables ne doivent pas avoir d'espace**
	flash		Peut être 0 ou 1 0 : message envoyé en mode normal Si non nul : message envoyé en mode flash Le mode flash signifie l'ouverture immédiate du message chez le destinataire. Le message n'est pas stocké dans sa messagerie.
	unicode		Peut être 0 ou 1 0 : message envoyé au format standard Les caractères supportés sont les suivants: **!#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ_ abcdefghijklmnopqrstuvwxyz£¤¥§¿ÄÅÆÇÉÑÖØÜßàäåæèéìñòöøùü** Si non nul : message envoyé en format unicode . il prend en compte tous les caractères qui ne figurent pas dans la liste ci-dessus. La limitation de caractère pour un message en unicode est de 70.
	binaire		Peut être 0 ou 1 Si non nul : message envoyé en binaire
	datesend		Date d'envoi du message. Format : AAAA-MM-JJ HH:MM (Envoi différé)
	mwi		
	pid		
	rpi		
	alt_dcs		
	udhdata		
destinataires	sms	numero	Le numéro du destinataire
		datas	C'est à ce niveau que nous allons définir les différentes variables du message. Exemple : <datas> <nom>Ange</nom> </datas>

II-2. JSON

L'URL utilisée pour écrire les données au format JSON est la suivante:

http://www.smseco.com/api/json/sendsms/

II-2-1. Structure de la requête JSON

JSON={"compte":{"login":"xx","password":"xx"},"message":{"expediteur":"xx","msgid":"xx", "msg":" Bonjour {nom}, votre solde est {solde}", "datesend":"xx", "flash":0, "unicode":0, "binaire":0},"destinataires":[{"numero":"xx", "datas":[{"nom":"Philo", "solde":"xx"}]}, {"numero":"xx", "datas":[{"nom":"Ange", "solde":"xx"}]}]}

Les paramètres utilisés sont les mêmes que pour le format XML.

II-3. SOAP

Le webservice est décrit dans le fichier suivant :

http://www.smseco.com/api/soap/wsdl/

Pour l'envoi de SMS avec publipostage avec le protocole SOAP nous allons utiliser la fonctionnumberwithdatas

Exemple d'envoi de SMS avec SOAP php

```
< ?php

class numberwithdatas {

        var $to;

        var $data1;

        var $data2;

        var $data3;

        var $data4;

        var $data5;

        var $data6;

        var $data7;

        var $data8;

        var $data9;

        var $data10;

        function __construct(array $parameters){

                $datas = array();
```

```
        foreach(array_keys(get_object_vars($this)) as $key)  $datas[strtolower($key)] = $key;
                foreach($parameters as $key => $value) if(isset($datas[$key])) {
                        $this->{$datas[strtolower($key)]} = utf8_encode($value);
                }
        }
}

$wsdl = "http://www.smseco.com /api/soap/wsdl/";

$server = "http://www.smseco.com /api/soap/server/";

$options = array("location" =>$server, "trace"=>true, 'style'=> SOAP_DOCUMENT, 'use'=>
SOAP_LITERAL);

$api = new SoapClient($wsdl, $options);

$publipostage = array();

$publipostage[0] = new numberwithdatas(array("to"=>"07352518", "data1"=>"Philo",
"data2"=>"Developpeur Web"));

$publipostage[1] = new numberwithdatas(array("to"=>"05442710", "data1"=>"MALAMIN",
"data2"=>"Financial Manager"));

$smsdatas = array("login"=>"particulier@smseco.com",
"password"=>"123456","msgid"=>"9876543210", "msg"=>"Bonjour {nom}, tu es {fonction}.",
"expediteur"=>"TEST", "datesend"=>"2015-06-21 16:49", "to"=>array("07352518",
"05442710"),"publipostage"=>$publipostage);

$api->Sendsms($smsdatas);

?>
```

III. SUIVI DES ENVOIS

L'API de SMSECO vous permet de suivre l'état de vos envois. Il faut souligner que les statuts des sms déjà consultés ne seront plus disponibles à la prochaine consultation.

III-1. HTTP GET

Structure de l'URL pour un suivi des envois de message par GET

http://www.smseco.com/api/{format-sorti}/getdlrs?login=xx&password=xx&msgfid=xx&smsfid=xx

{format-sorti}peut prendre les valeurs suivantes : xml, json ou html

N.B. : Les paramètres en vert sont optionnels. Ils sont ajoutés lorsqu'on veut avoir le statut d'un envoi particulier ou d'un numéro.

msgfid = Identifiant d'un envoi au niveau de SMSECO

smsfid = Identifiant d'un sms pour un numéro spécifique lors d'un envoi

Réponse XML

La structure retour du XML est la suivante :

```
<REPONSE>
        <accountid>xx</accountid>
        <statut>xx</statut>
        <texte>xxx</texte>
        <messages>
                <message>
                        <msgfid>xx</msgfid>
                        <dlrs>
                                <dlr>
                                        <smsfid>xx</smsfid>
                                        <destination>xx</destination>
                                        <date>xx</date>
                                        <statut>xx</statut>
                                        <libelle>xx</libelle>
                                        <texte>xx</texte>
                                </dlr>
                        </dlrs>
                </message>

        </messages>
</REPONSE>
```

Paramètres

Paramètre		Description
accountid		Votre Identifiant smseco
statut		Si =0 Requête traitée avec succès
texte		Texte expliquant le statut
messages (message)	msgfid	Identifiant d'un envoi
	smsfid	Identifiant d'un sms
	destination	Le numéro du destinataire
	date	Date à laquelle l'opération a été faite
	statut	Valeur numérique représentant l'état de l'envoi
	libelle	Explication du statut
	texte	Explication plus détaillée du statut

Définition des statuts des messages envoyés

Statut	Signification	Facturé
50	en Attente	NON
62	Rejeté	NON
65	File d'attente locale	OUI
54	non envoyé	NON
55	Transmis à l'Opérateur	OUI
61	File d'attente Opérateur	OUI
56	Accusé de Réception	OUI
59	Non Délivré par l'Opérateur	OUI
60	Erreur réseau opérateur	OUI
49	Erreur Crédits	NON

En cas d'erreur, voici la structure du XML retour :

```
<REPONSE>
<statut></statut>
<texte></texte>
<messages>
</messages>
</REPONSE>
```

Paramètres

Paramètres	Description
statut	Une valeur < 0 qui représente le code de l'erreur
texte	Explication du code erreur

Codes d'erreur

Statut	Signification
-1	XML invalide
-2	Login non spécifié
-3	Mot de passe non spécifié
-4	Echec de l'Authentification

III-2. XML

L'URL utilisée pour écrire les données au format XML est la suivante:

http://www.smseco.com/api/xml/getdlrs/

III-2-1. Structure de la requête xml

```
XML=<SMS>
        <compte>
            <login></login>
            <password></password>
        </compte>
        <message>
            <msgfid></msgfid>
            <smsfid></smsfid>
        </message>
</SMS>
```

N.B. : Les paramètres en vert sont optionnels. Ils sont ajoutés lorsqu'on veut avoir le statut d'un envoi particulier ou d'un numéro.

msgfid = Identifiant d'un envoi au niveau de SMSECO

smsfid = Identifiant d'un sms pour un numéro spécifique lors d'un envoi

La réponse du XML est la même que par l'envoi par GET

III-3. JSON

L'URL utilisée pour écrire les données au format JSON est la suivante:

http://www.smseco.com/api/xml/getdlrs/

III-3-1. Structure de la requête JSON

```
JSON={"compte":{"login":"xxx","password":"xxx"},"message":{"msgfid":"xx","smsfid":"xx"}}
```

Les paramètres utilisés sont les mêmes que pour le format XML.

III-3-2. Structure de la réponse JSON

{"accountid":"xx", "statut":xx, "texte":"xx", "messages":[{"msgfid":"xx", "dlrs":[{"smsfid":"xx", "destination":"xx", "date":"xx", "stastut":"xx", "libelle":"xx", "texte":"xx"}]}]}

III-4. SOAP

Le webservice est décrit dans le fichier suivant :

http://www.smseco.com/api/soap/wsdl/

Exemple d'envoi de SMS avec SOAP php

```
< ?php

$wsdl = "http://www.smseco.com /api/soap/wsdl/";

$server = "http://www.smseco.com /api/soap/server/";

$options = array("location" =>$server, "trace"=>true, 'style'=> SOAP_DOCUMENT, 'use'=> SOAP_LITERAL);

$api = new SoapClient($wsdl, $options);

$smsdatas = array("login"=>"particulier@smseco.com", "password"=>"123456", "msgid"=>"
55857bdb8d51b7-38880789", " smsid"=>" 55857bf80517a0-38726421");

$api-> Getdlrs ($smsdatas);

?>
```

Les paramètres utilisés sont les mêmes que pour le format XML.

IV. COMMANDES ADDITIONNELLES

IV-1. Consultation du solde

IV-1-1. HTTP GET

Structure de l'URL pour la consultation de son solde par GET

http://www.smseco.com/api/{format-sorti}/Getbalance?login=xx&password=xx

{format-sorti}peut prendre les valeurs suivantes : xml, json ou html

Réponse XML

La structure retour du XML est la suivante :

```
<REPONSE>
        <accountid>xx</accountid>
        <statut>xx</statut>
        <texte>xx</texte>
        <credits>xx</credits>
        <expire>xx</expire>
        <cumul>xx</cumul>
</REPONSE>
```

Paramètres

Paramètre	Description
accountid	Votre Identifiant smseco
statut	Si =0 Requête traitée avec succès
texte	Texte expliquant le statut
credits	Nombre de crédits restant
expire	Date d'expiration
cumul	Cumul Consommation

En cas d'erreur, voici la structure du XML retour :

```
<REPONSE>
<accountid></accountid>
<statut></statut>
<texte></texte>
</REPONSE>
```

Paramètres

Paramètres	Description
accountid	Votre Identifiant smseco
statut	Une valeur < 0 qui représente le code de l'erreur
texte	Explication du code erreur

Codes d'erreur

Statut	Signification
-1	XML invalide
-2	Login non spécifié
-3	Mot de passe non spécifié
-4	Echec de l'Authentification

IV-1-2. XML

L'URL utilisée pour écrire les données au format XML est la suivante:

http://www.smseco.com/api/xml/getbalance/

IV-1-2-1. Structure de la requête xml

```
XML=<SMS>
        <compte>
            <login></login>
            <password></password>
        </compte>
    </SMS>
```

La réponse du XML est la même que par l'envoi par GET

IV-1-3. JSON

L'URL utilisée pour écrire les données au format JSON est la suivante:

http://www.smseco.com/api/json/getbalance/

IV-1-3-1. Structure de la requête JSON

```
JSON={"compte":{"login":"xxx","password":"xxx"}}
```

IV-1-4. SOAP

Le webservice est décrit dans le fichier suivant :

http://www.smseco.com/api/soap/wsdl/

Exemple de consultation de son solde avec SOAP php

```
< ?php

$wsdl = "http://www.smseco.com /api/soap/wsdl/";

$server = "http://www.smseco.com /api/soap/server/";

$options = array("location" =>$server, "trace"=>true, 'style'=> SOAP_DOCUMENT, 'use'=> SOAP_LITERAL);

$api = new SoapClient($wsdl, $options);

$smsdatas = array("login"=>"particulier@smseco.com", "password"=>"123456");

$api->Getbalance ($smsdatas);

?>
```

Les paramètres utilisés sont les mêmes que pour le format XML.

IV-2. Consultation de la liste des expéditeurs

Nous verrons dans cette partie comment voir sa liste de noms expéditeurs

IV-2-1. HTTP GET

Structure de l'URL pour la consultation de son solde par GET

http://www.smseco.com/api/{format-sorti}/Getsenders?login=xx&password=xx

{format-sorti}peut prendre les valeurs suivantes : xml, json ou html

Réponse XML

La structure retour du XML est la suivante :

```
<REPONSE>

<accountid>xx</accountid>

<statut>xx</statut>

<texte>xx</texte>

<senders>

<sender>xx</sender>

<sender>xx</sender>
```

</senders>

</REPONSE>

N.B. : Nous aurons autant de nœud senderque de nom d'expéditeur

Paramètres

Paramètre	Description
accountid	Votre Identifiant smseco
statut	Si =0 Requête traitée avec succès
texte	Texte expliquant le statut
sender	Nom d'expéditeur

En cas d'erreur, voici la structure du XML retour :

```
<REPONSE>
<accountid></accountid>
<statut></statut>
<texte></texte>
<senders></senders>
</REPONSE>
```

N.B. : senders est vide en cas d'erreur.

Paramètres

Paramètres	Description
accountid	Votre Identifiant smseco
statut	Une valeur < 0 qui représente le code de l'erreur
texte	Explication du code erreur

Codes d'erreur

Statut	Signification
-1	XML invalide
-2	Login non spécifié
-3	Mot de passe non spécifié
-4	Echec de l'Authentification

IV-2-2. XML

L'URL utilisée pour écrire les données au format XML est la suivante:

http://www.smseco.com/api/xml/getsenders/

IV-2-2-1. Structure de la requête xml

```
XML=<SMS>
        <compte>
            <login></login>
            <password></password>
        </compte>
    </SMS>
```

La réponse du XML est la même que par l'envoi par GET

IV-2-3. JSON

L'URL utilisée pour écrire les données au format JSON est la suivante:

http://www.smseco.com/api/json/getsenders/

IV-2-3-1. Structure de la requête JSON

```
JSON={"compte":{"login":"xxx","password":"xxx"}}
```

IV-2-4. SOAP

Le webservice est décrit dans le fichier suivant :
http://www.smseco.com/api/soap/wsdl/
Exemple de consultation de sa liste de noms expéditeur avec SOAP php

```
< ?php

$wsdl = "http://www.smseco.com /api/soap/wsdl/";

$server = "http://www.smseco.com /api/soap/server/";

$options = array("location" =>$server, "trace"=>true, 'style'=> SOAP_DOCUMENT, 'use'=> SOAP_LITERAL);

$api = new SoapClient($wsdl, $options);

$smsdatas = array("login"=>"particulier@smseco.com", "password"=>"123456");

$api->Getsenders($smsdatas);

?>
```

Les paramètres utilisés sont les mêmes que pour le format XML.

V - OPTIMISATION DE BASE DE DONNEES (HLR)

L'outil HLR permet de vérifier à une période donnée la validité ou encore la disponibilité d'un numéro de mobile afin de réduire les coûts d'envois et évite les messages non délivrés.

N.B. : Seuls les Comptes Entreprises ont accès à ce service.

V-1. HTTP GET

Structure de l'URL pour le HLR par GET

http://www.smseco.com/api/{format-sorti}/sendhlr?login=xx&password=xx&numero=xx

{format-sorti}peut prendre les valeurs suivantes : xml, json ou html

Pour tester la disponibilité de plusieurs numéros, il faudra les séparés par une virgule.

```
<REPONSE>
        <msgfid>xx</msgfid>
        <statut>xx</statut>
        <texte>xx</texte>
        <destinataires>xx</destinataires>
        <sms>xx</sms>
        <credits>xx</credits>
        <solde>xx</solde>
        <messages>
                <sms>
                        <smsfid>xx</smsfid>
                        <destination>xx</destination>
                        <credits>xx</credits>
                </sms>
        </messages>
</REPONSE>
```

Paramètres

Paramètre		Description
msgfid		Identifiant de l'envoi
statut		Si >0 Nombre de destinataires valides
texte		Texte expliquant le statut de l'envoi
destinataires		Nombre total de destinataires valides
sms		Nombre total de sms
credits		Coût de l'envoi en crédit (qui est l'unité de mesure de smseco)
Solde		Nbre de crédit restant
Messages (sms)	smsfid	Identifiant d'un sms
	destination	Le numéro du destinataire valide
	credits	Coût total du HLR en crédit pour ce destinataire

N.B. : Nous aurons autant de nœud **sms** dans **messages** que de destinataires valides.

En cas d'erreur, voici la structure du XML retour :

```
<REPONSE>
<statut></statut>
<texte></texte>
</REPONSE>
```

Paramètres

Paramètres	Description
statut	Une valeur < 0 qui représente le code de l'erreur
texte	Explication du code erreur

Codes d'erreur

Statut	Signification
-1	XML invalide
-2	Login non spécifié
-3	Mot de passe non spécifié
-4	Echec de l'Authentification
-6	Pas assez de crédits ou crédits insuffisants
-7	Aucun destinataire spécifié
-9	Aucuns destinataires valides
-10	La longueur de l'Id du message dépasse les 30 caractères autorisés
-11	Envoi de sms non autorisé pour ce compte
-12	Compte non actif dans le système
-14	Commande inconnue
-15	Aucun statut sms disponible

V-2. XML

L'URL utilisée pour écrire les données au format XML est la suivante:

http://www.smseco.com/api/xml/sendhlr/

V-2-1. Structure de la requête xml

```
XML=<SMS>
        <compte>
            <login></login>
            <password></password>
        </compte>
        <message>
            <msgid></msgid>
        </message>
        <destinataires>
            <numero></numero>
            <numero></numero>
            <numero></numero>
        </destinataires>
</SMS>
```

La longueur de ***msgid*** doit être au maximum 30 caractères.

Paramètres utilisés

Paramètre		Description
compte	login	Le nom d'utilisateur de votre compte sur smseco.com. Il s'agit d'uneadresse email ou d'un numéro de mobile ou de de votre identifiant smseco
	password	Le mot de passe relatif à votre compte.
message	msgid	Identifiant du message, utilisé pour la livraison des rapports d'envoi
destinataires	numero	Le numéro a vérifier

V-3. JSON

L'URL utilisée pour écrire les données au format JSON est la suivante:

http://www.smseco.com/api/json/sendhlr/

V-3-1. Structure de la requête JSON

JSON={"compte":{"login":"xxx","password":"xxx"},"message":{"msgid":"xx"},"destinataires":[{"numero":"xx"}, {"numero":"xx"}]}

Les paramètres utilisés sont les mêmes que pour le format XML.

Réponse JSON

Après le JSON POST initié par le client, certains codes d'état sont disponibles.

La structure retour de la requête JSON :

{"msgfid":"xx","statut":x,"texte":"xx","destinataires":"xx","sms":"xx","credits":"xx","solde":"xx","messages":[{"smsfid":"xx","destination":"xx","credits":xx}]}

N.B. : Les paramètres retour sont les mêmes que pour le http GET

En cas d'erreur, voici la structure du JSON retour :

{"statut":xx,"texte":"xx"}

V-3-2. Exemple d'un envoi au format JSON en CURL

```
<?php
$uri = "http://www.smseco.com/api/json/sendhlr/";

$data =
"JSON={\"compte\":{\"login\":\"toto@toto.fr\",\"password\":\"123456\"},\"message\":{\"msgid\":\"15\"},\"destinataires\":[{\"numero\":\"05442710\"}, {\"numero\":\"07352518\"}]}";


$ch = curl_init();

curl_setopt($ch, CURLOPT_URL,$uri);

curl_setopt($ch,CURLOPT_HTTPHEADER,array("Accept: application/json","Accept: application/json","Content-Type: application/json", "Content-Length: ". strlen($data)));

curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $data);

curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$output = curl_exec ($ch);

curl_close ($ch); // close curl handle

?>
```

V-4. SOAP

Le webservice est décrit dans le fichier suivant :

http://www.smseco.com/api/soap/wsdl/

Exemple de HLR avec SOAP php

```
< ?php

$wsdl = "http://www.smseco.com /api/soap/wsdl/";

$server = "http://www.smseco.com /api/soap/server/";

$options = array("location" =>$server, "trace"=>true, 'style'=> SOAP_DOCUMENT, 'use'=> SOAP_LITERAL);

$api = new SoapClient($wsdl, $options);

$smsdatas = array("login"=>"particulier@smseco.com", "password"=>"123456", "msgid"=>"9876543210",
"to"=>array("07352518", "05442710", "08559001"));

$api-> Sendhlr($smsdatas);

?>
```

Tel : 225 22 42 73 64

CEL : 225 47 51 51 57

SKYPE SUPPORT : SUPPORT_SMSECO

SKYPE COMMERCIAL : SMSECO