


	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022



# Guide Web Services

## SLS / Karizma Conseil

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

## Table de matières

- I. Généralités sur le XML-RPC ODOO
- II. Architecture des Objets sur SLS
- III. Connexion à l'API Odoo
- IV. Exemple de listing de colis
- V. Exemple de listing des adresses associés au profil connecté
- VI. Exemple de listing des modèles de colis associés au profil connecté
- VII. API raccourcis

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022



## I. Généralités sur le XML-RPC ODOO

XML-RPC est un protocole basé sur XML, utilisé pour transmettre des informations entre des systèmes informatiques via le réseau. RPC signifie appel de procédure à distance et utilise XML pour coder les appels. Pour transmettre des informations, il utilise la requête HTTP (HyperText Transfer Protocol).



Odoo est généralement étendu en interne via des modules, mais bon nombre de ses fonctionnalités et toutes ses données sont également disponibles de l'extérieur pour une analyse externe ou une intégration avec divers outils. Une partie de l'API Models est facilement disponible via XML-RPC et accessible à partir d'une variété de langues.

## II. Architecture des Objets sur SLS

Objets	Champs	Nom technique	Description	Type de champ
Demande "sochepress.customer.request"	Client	customer_id	requis	Relation avec une autre table (Identifiant)
	Date de de demande	demand_date	requis	Date
	Type de demande	type	requis	Selection('normal' pour <b>Normal</b> , 'transport' pour <b>Transport</b> , 'course' pour <b>Course urgente</b> , et 'passage' pour <b>Passage</b> )
	Type d'envoi	send_type	non requis	Selection('send' pour <b>Envoi</b> et

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022



				'return_on_requests' pour Retour)
	statut de la demande	state	non requis	Selection('draft' pour <b>Brouillon</b> , 'waiting' pour <b>En Attente</b> , 'verified' pour <b>Vérifié</b> , 'accepted' pour <b>Acceptée</b> , 'closed' pour <b>Fermé</b> , et 'canceled' pour <b>Annulé</b> )
	Numéro de la demande	name	non requis	Char (Caractère)
Colis "sochepress.customer.request.line"	Numéro du colis	name	non requis	Char (Caractère)
	Type de colis	type_colis_id	requis	Many2One
	Poids	weight	requis	Float
	Volume	volume	requis	Float
	Source	source_id	non requis	Relation avec une autre table (Identifiant)
	Destination	destination_id	non requis	Relation avec une autre table (Identifiant)
	Destinataire	destinator_id	non requis	Relation avec une autre table

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

				(Identifiant)
	Expéditeur	expeditor_id	non requis	Relation avec une autre table (Identifiant)
	Position	current_position	non requis	Relation avec une autre table (Identifiant)
	Etape	step	non requis	Selection(['new', 'charged', 'discharged', 'delivered', 'in_progress', 'non_delivered', 'not_pickup', 'refused', 'reported', 'retracted', 'closed'])
	Référence externe	ref_ext	non requis	Char (Caractère)
	Statut de livraison	Delivery_state	Non requis	Selection('in_progress', 'delivered', 'returned_to_expeditor', 'in_return', 'non_delivered')

### III. Connexion à l'API Odoo

En se référant à la documentation d'Odoo, la connexion à l'API Odoo peut se faire par plusieurs langages de programmation notamment Python, Ruby, PHP, Java. Selon les différents langages cités ci-dessus voici comment se fait la connexion à l'API Odoo.

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

Python	Ruby	PHP	Java
<pre>url = &lt;insert server URL&gt; db = &lt;insert database name&gt; username = 'admin' password = &lt;insert password for your admin user (default: admin)&gt;</pre>	<pre>url = &lt;insert server URL&gt; db = &lt;insert database name&gt; username = "admin" password = &lt;insert password for your admin user (default: admin)&gt;</pre>	<pre>\$url = &lt;insert server URL&gt;; \$db = &lt;insert database name&gt;; \$username = "admin"; \$password = &lt;insert password for your admin user (default: admin)&gt;;</pre>	<pre>final String url = &lt;insert server URL&gt;,     db = &lt;insert database name&gt;,     username = "admin", password = &lt;insert password for your admin user (default: admin)&gt;;</pre>

Par la suite nous aurons besoin de l'id de l'utilisateur, nous le récupérons donc ainsi

Python	Ruby	PHP	Java
<pre>uid = common.authenticate(db, username, password, {})</pre>	<pre>uid = common.call('authenticate', db, username, password, {})</pre>	<pre>\$uid = \$common-&gt;authenticate(\$db, \$username, \$password, array());</pre>	<pre>int uid = (int)client.execute(common_config, "authenticate", asList( db, username, password, emptyMap()));</pre>

Le deuxième point de terminaison est `xmlrpc/2/object`, est utilisé pour appeler des méthodes de modèles odoo via la `execute_kw` fonction RPC.

Chaque appel à `execute_kw` prend les paramètres suivants :

- La base de données à utiliser, une chaîne
- L'ID utilisateur (récupéré via `authenticate`), un entier
- Le mot de passe de l'utilisateur, une chaîne
- Le nom du modèle, une chaîne
- Le nom de la méthode, une chaîne
- Un tableau / liste de paramètres passés par position
- Un mapping / dict des paramètres à passer par mot-clé (optionnel)

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

## IV. Exemple de listing de colis

### Python code :

```
import xmlrpc.client

# Connexion à l'API OD00
# Url contient l'adresse ip avec ou sans port
url = 'http://localhost:8013'
db = "SLM_PROD_27"
username = "admin"
password = "1234"



# Recuperation de l'ID de l'utilisateur
common =
xmlrpc.client.Server('{}xmlrpc/2/common'.format(url))

uid = common.authenticate(db, username, password, {})

# Appel des méthodes
models =
xmlrpc.client.Server('{}xmlrpc/2/object'.format(url))

#liste des colis dans différents états : nouveau, chargé...
list_colis0 = models.execute_kw(db, uid, password,
'sochepress.customer.request', 'get_list_colis', ['new'])

list_colis1 = models.execute_kw(db, uid, password,
'sochepress.customer.request', 'get_list_colis', ['charged'])
```

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

### **Java Code :**

```

final String url = "http://localhost:8013";
final String db = "SLM_PROD_27";
final String username = "admin";
final String password = "1234";

final XmlRpcClient client = new XmlRpcClient();
final XmlRpcClientConfigImpl common_config = new
XmlRpcClientConfigImpl();

common_config.setServerURL(new
URL(String.format("%s/xmlrpc/2/common", url)));

client.execute(common_config, "version", emptyList());



int uid = (int)client.execute(common_config, "authenticate",
asList(db, username, password, emptyMap()));

asList((Object[])models.execute("execute_kw", asList(
db, uid, password, "sochepress.customer.request",
"get_list_colis", asList("new"))));

asList((Object[])models.execute("execute_kw", asList(
db, uid, password, "sochepress.customer.request",
"get_list_colis", asList("charged"))));

```



	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

### **PHP Code :**


```
<?php
require_once('ripcord.php');
$url = "http://localhost:8013";
$db = "SLM_PROD_27";
$username = "ibda3way.o@gmail.com";
$password = "1234";
$common = ripcord::client("$url/xmlrpc/2/common");
$common->version();
$suid = $common->authenticate($db, $username,$password,
$version);
$models = ripcord::client("$url/xmlrpc/2/object");
$models->execute_kw($db, $uid, $password,
'sochepress.customer.request', 'get_list_colis',array('new'));
$models->execute_kw($db, $uid, $password,
'sochepress.customer.request', 'get_list_colis',
array('charged'));
?>
```

## **V. Exemple de listing des adresses associés au profil connecté**

### **Python code :**

```
# Récupérer le record de l'utilisateur connecté
user = models.execute_kw(db, uid, password, 'res.users',
'search_read', [[['id', '=', uid]]])

# Les adresses associés à l'utilisateur connecté
partners = models.execute_kw(db, uid, password, 'res.partner',
'search', [[['parent_id', '=', user[0]['partner_id'][0]]]])
# le paramètre 'limit' permet de limiter le nombre d'adresses
qu'on récupère à un
```

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

```
# nombre donné
five_partners = models.execute_kw(db, uid, password,
    'res.partner', 'search', [[['parent_id', '=', user[0]['partner_id'][0]]], {'limit': 5})
```


### **Java Code :**

```
final List<Dictionary<String, Object>> user =
asList((Object[])models.execute("execute_kw", asList(
    db, uid, password, "res.users", "search_read",
asList( asList( asList("id", "=", uid ))))));
asList((Object[])models.execute("execute_kw", asList(
    db, uid, password, "res.partner", "search",
asList( asList(asList("parent_id", "=",
user.get(0).get("partner_id").get(0))))
    )));
asList((Object[])models.execute("execute_kw", asList(
    db, uid, password, "res.partner", "search",
asList( asList( asList("parent_id", "=",
user.get(0).get("partner_id").get(0)))), new HashMap()
{{put("limit", 5); }}
    )));
```

### **PHP Code :**

```
$user = $models->execute_kw($db, $uid, $password, 'res.users',
    'search_read',array(array(array('id', '=', uid))));
$models->execute_kw($db, $uid, $password, 'res.partner',
    'search',array(array(array('parent_id', '=',
$user[0]['partner_id'][0]))));

$models->execute_kw($db, $uid, $password, 'res.partner',
    'search',array(array(array('parent_id', '=',
$user[0]['partner_id'][0]))), array('limit'=>5));
```

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

## **VI. Exemple de listing des modèles de colis associés au profil connecté**

### **Python code :**

```
domain = [user[0]['partner_id'][0]]
if user[0]['parent_id']:
    domain.append(user[0]['parent_id'][0])
modele_de_colis = models.execute_kw(db, uid, password,
'product.template', 'search', [[['client_id', 'in', domain],
['hidden', '=', False]]])
```



### **Java Code :**

```
final List<Integer> domain = new List<Integer>();
    domain.add(user.get(0).get("partner_id").get(0))
    if (user.get(0).get("parent_id")){
        domain.add(user.get(0).get("parent_id").get(0))
    }

asList((Object[])models.execute("execute_kw", asList(
    db, uid, password, "product.template", "search",
asList( asList(asList("client_id", "in", domain),
asList("hidden", "=", False)))
    )));
```



### **PHP Code :**

```
$domain = array($user[0]['partner_id'][0]);
if ($user[0]['parent_id']) {
    array_push($domain, $user[0]['parent_id'][0]);}
$model->execute_kw($db, $uid, $password, 'product.template',
'search',array(array(array('client_id', 'in', domain),
array('hidden', '=', False))));
```



	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

## VII. API raccourcis

API	Utilité	Nom technique	Input	Output
Liste des colis	Cette méthode a pour objectif de retourner la liste des colis du client connecté en se basant sur leurs états	get_list_colis	etat_colis: 'new' 'charged' 'discharged' 'delivered' 'in_progress' 'non_delivered' 'not_pickup' 'refused' 'reported'	Dictionnaire avec les clés : - Nom, - ID, - Étape, - Poids
Chercher un colis	Cette méthode a pour objectif de retourner le colis du client connecté en se basant sur son nom	get_colis_by_name	name : nom du colis	Dictionnaire avec les clés : - Nom, - ID, - Demande, - Étape, - Poids, - Volume, - Référence externe
Informations d'un colis	Cette méthode a pour objectif de retourner les colis du client connecté en se basant sur 3 critères : ref de colis, id de la demande de colis ou bien le nom du colis	get_colis	dict_colis : { "ref" : Référence externe du colis, "colis_name" : Nom du colis, "request_id" : ID de la demande du colis, }	Dictionnaire avec les clés: - ID - Demande - Étape - Poids - Volume - Référence externe

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

Créer un destinataire	Cette méthode a pour objectif de créer un destinataire en passant les informations nécessaires	create_destinator	dictionnaire :{ "nom_destinataire": Destinator name, "rue": Street, "rue2": Street2, "ville" : City, "pays" : Country, "region": Area, "zip": Zip Code, "destination": Destination, "phone": Phone, "mobile": Mobile, "email": Mail, "customer_id": Customer ID, }	Id du destinataire
Créer une demande	Cette méthode a pour objectif de créer une demande en passant les informations nécessaires	create_demand	{ "customer_id" : Customer ID, "type" : Type , "date" : Date, "colis" : { "ref_ext" : Référence externe, "type_colis": Type de colis, "expediteur": Expéditeur, "source": Source, "destinataire": Destinataire, "destination": Destination, "modele": Modèle de colis, "poids_colis": Poids, "methode_contre_remboursement":Retour de	Id de la demande créée

	Karizma Conseil		Auteur : Abdoul-Ashraf SALAMI
	Projet : Mise en place des Web Services plateforme SLS Odoo		Date de dernière mise à jour : 29/03/2022

			fonds, "montant_contre_remboursement": Montant, }	
--	--	--	---	--

## Références

Documentation Odoo :

<https://www.odoo.com/documentation/13.0/developer/misc/api/odoo.html>