

myfab Tools

Bibliothèque de fonctions et de méthodes packagées par 1Life à mettre en œuvre pour votre paramétrage avancé d'Open-Prod.

- [Règle générale](#)
- [Gestion de fichier](#)
- [Gestion FTP / SFTP / FTS TLS](#)
- [Impressions Jasper](#)
- [Création/Manipulation de fichier XML](#)
- [Traitement BL mf_do_partial_picking](#)

Règle générale

Globalement, et pour chaque méthode utilisable et présente dans **myfab Tools**, il faut passer par une instance du modèle **mf.tool** pour pouvoir appeler et utiliser ces méthodes. Par exemple :

```
self.env["mf.tools"].mf_get_file_name_extension(file_to_convert_name)
```

Dans le cas où il y a besoin d'utiliser plusieurs fois les "tools" dans la même méthode ou la même action serveur, on peut mettre l'instance dans une variable. Par exemple :

```
tools = self.env["mf.tools"]  
if self.mf_protocol == "ftp":  
    return tools.mf_login_to_ftp(mf_ftp_adress,mf_login,mf_password)  
elif self.mf_protocol == "sftp":  
    return tools.mf_login_to_sftp(mf_ftp_adress,mf_login,mf_password)
```

Les pages suivantes présentent des exemples d'utilisation des différents outils mis à disposition dans le module **myfab Tools**.

Gestion de fichier

Le tableau ci-dessous reprend les différentes méthodes utilisables, avec un exemple et une description ainsi que les données d'entrée et de sortie attendues.

Méthode / Exemple	Données d'entrée	Données de sortie
mf_get_file_name_extension(file_name) <pre>if self.env["mf.tools"].mf_get_file_name_extension(file_to_convert_name) in ['CSV','TXT']:</pre> <i>Dans ce cas-là on s'en sert pour vérifier si l'extension du fichier est CSV ou TXT pour faire un traitement spécifique pour ces fichiers.</i>	Prend une chaîne de caractères en entrée qui est un nom de fichier.	Retourne une chaîne de caractères qui a pour valeur l'extension de nom de fichier qui a été passé en entrée.
mf_get_file_name_without_extension (file_name) <pre>xlsx_file_name = self.env["mf.tools"].mf_get_file_name_without_extension(file_to_convert_name) + '.XLSX'</pre>	Prend une chaîne de caractères en entrée qui est un nom de fichier.	Retourne une chaîne de caractères qui a pour valeur le nom de fichier sans l'extension.
create_then_go_to (folder) <i>Créer le dossier si besoin puis déplace le curseur dedans.</i>	Prend une chaîne de caractères en entrée qui est un chemin absolu ou relatif vers un dossier qui existe ou non.	Ne retourne rien.
save_file (file_name,file_content) <i>Créer le fichier avec le nom et le contenu fourni.</i>	Prend en entrée une chaîne de caractères qui représente le nom du fichier avec son extension et une chaîne de caractères avec le contenu du fichier.	Ne retourne rien.
save_file_as_document (file_name,file_content,record) <i>Crée un document Open-Prod puis le lie avec la donnée passée en paramètre s'il y a un champ vers le modèle document.openprod.</i>	Prend en entrée une chaîne de caractères qui représente le nom du fichier avec son extension et une chaîne de caractères avec le contenu du fichier et la donnée sur laquelle lier le document (non obligatoire).	Retourne l'objet document qui a été créé.

Exemple action serveur pour imprimer un BL dans un fichier (avec le modèle jasper ayant l'ID=100)

Impression BL

Retirer du menu "Plus"

Modèle de base	Stock picking	Condition	True
Action à effectuer	Execute Python Code	Séquence	5

Code Python

Sécurité

Aide

Log parameterization

```
for record in objects:
    tools = env['mf.tools']
    jasper=env['jasper.document'].browse(100)
    report_file, report_format, model_report = tools.mf_print_report(jasper,record.id)
    tools.save_file("/home/openprod/Documents/"+str(record.id)+".pdf", report_file )
```

Gestion FTP / SFTP / FTS TLS

1. Protocole FTP Standard

Toutes les méthodes FTP fonctionnent globalement sur le même principe, aussi seules les méthodes pour le FTP standard sont explicitées ci-dessous.

Méthode / Exemple	Données d'entrée	Données de sortie
mf_login_to_ftp(ftp_adress,login,password)	Prend en entrée une adresse FTP, le login et le mot de passe qui sont tous des chaînes de caractères.	Retourne une instance de la connexion ftp qui sera utilisée par les méthodes suivantes.
mf_ftp_move_to_folder(ftp,folder)	Prend en entrée une instance de connexion FTP et une chaîne de caractères qui représente le dossier dans lequel on souhaite déplacer le curseur.	Ne retourne rien.
mf_send_file_to_ftp(ftp,file_name,delete)	Prend en entrée une instance de connexion FTP, une chaîne de caractères qui représente le nom du fichier local que l'on souhaite envoyer sur le serveur distant et un booléen qui sert à indiquer si l'on souhaite supprimer le fichier une fois l'envoi fait.	Ne retourne rien.
mf_quit_ftp(ftp) <i>Ferme la connexion FTP.</i>	Prend en entrée une instance de connexion FTP.	Ne retourne rien.
mf_ftp_get_all_files(ftp,delete) <i>Crée un document openprod puis le lie avec la donnée passée en paramètre s'il y a un champ vers le modèle document.openprod.</i>	Prend en entrée une instance de connexion FTP et un booléen qui sert à indiquer si l'on souhaite supprimer les fichiers du serveur distant une fois qu'ils ont été récupérés.	Retourne une liste avec tous les noms des fichiers récupérés.

Exemple d'utilisation d'une connexion FTP :

```
ftp = tools.mf_login_to_ftp(adress,login,password)
tools.mf_ftp_move_to_folder(ftp,folder)
tools.mf_send_file_to_ftp(ftp, str(self.number)+".csv",True)
```

2. Protocole FTP TLS

Le protocole FTP TLS utilise les mêmes méthodes que le FTP standard sauf pour le login :

mf_login_to_ftp_tls(ftp_adress,login,password)

3. Protocole SFTP

Variantes de méthodes propres au protocole SFTP :

- **mf_login_to_sftp(sftp_adress,login,password)**
- **mf_sftp_move_to_folder(sftp,folder)**
- **mf_send_file_to_sftp(sftp,file_name,delete)**
- **mf_sftp_get_all_files(sftp,delete)**
- **mf_sftp_get_file_by_name(sftp,file_name, delete)**

Impressions Jasper

Via la méthode ci-dessous, on peut manipuler l'impression de documents Jasper :

Méthode / Exemple	Données d'entrée	Données de sortie
mf_print_report(report_id,record)	Prend en entrée un enregistrement du modèle jasper.report et la donnée à partir de laquelle on veut imprimer le document.	Retourne le contenu du rapport PDF en chaîne de caractère, le format du document imprimé (PDF) et le nom du modèle du rapport.

Combiné avec la méthode de sauvegarde de fichier ci-dessus on peut créer un fichier pdf sur le serveur local. Exemple d'utilisation :

```
report_file, report_format, model_report =  
tools.mf_print_report(config.mf_invoice_report_to_print,invoice)  
tools.save_file(invoice.number.replace("/","_")+".pdf",report_file)
```

Exemple action serveur pour imprimer un BL dans un fichier (avec le modèle jasper ayant l'ID=100)

Impression BL

Retirer du menu "Plus"

Modèle de base	Stock picking	Condition	True
Action à effectuer	Execute Python Code	Séquence	5

Code Python

Sécurité

Aide

Log parameterization

```
for record in objects:  
    tools = env['mf.tools']  
    jasper=env['jasper.document'].browse(100)  
    report_file, report_format, model_report = tools.mf_print_report(jasper,record.id)  
    tools.save_file("/home/openprod/Documents/"+str(record.id)+".pdf", report_file )
```

Création/Manipulation de fichier XML

Les méthodes suivantes permettent de créer et manipuler les fichiers XML . Cela peut par exemple servir pour exporter un fichier XML à partir de données d'Open-Prod depuis une action serveur par exemple.

Méthode / Exemple	Données d'entrée	Données de sortie
mf_get_lxml_library() Exemple : lxml = env['mf.tools'].mf_get_lxml_library() etree=lxml.etree	--	Retourne la librairie lxml
lxml_element_get_text(xml)	Objet Element de la la librairie etree	Texte de l'objet Element donnée en paramètre
lxml_element_get_tag(xml)	Objet Element de la la librairie etree	Nom de l'objet Element donnée en paramètre
lxml_element_set_text(xml,text)	Objet Element de la la librairie etree ,Texte à mettre dans la balise	Objet Element modifié
lxml_element_set_ta(xml,tag)	Objet Element de la la librairie etree ,Nom à affecter à la balise	Objet Element modifié

Exemple : dans une action serveur

```
tools = env['mf.tools']
etree = tools.mf_get_lxml_library().etree

data_xml_de_base=""<?xml version="1.0" encoding="utf-8"?>
<ProcessMaterialInformation>
  <ApplicationArea>
    <Sender>
      <LogicalID>OPP</LogicalID>
    </Sender>
    <Receiver>
      <ID>QUBES</ID>
    </Receiver>
    <CreationDateTime>2010-07-22T11:20:33</CreationDateTime>
```



```

</ApplicationArea>
  <DataArea>
    <<Process>
    </Process>
    <<MaterialInformation>
      <<<ID>ItemsDefinition</ID>
      <<<Description>Material classes and material definitions for all the items</Description>
      <<<PublishedDate>2010-07-22T11:20:33</PublishedDate>
    </MaterialInformation>
  </DataArea>
</ProcessMaterialInformation>
""

```

```

xml=etree.fromstring(data_xml_de_base)
#incrément auto à chaque export : sequence code ITEM_QUBES
res=env['ir.sequence'].next_by_code('ITEM_QUBES')
BODID=etree.SubElement(xml,'BODID') # crée balise BODID dans la balise root du xml
tools.lxml_element_set_text(BODID,res) #affecte valeur texte à la balise créé

#Ajoute une balise article pour chaque produit en vie avec une sous balise id et description
MaterialInformations= xml.xpath("//MaterialInformation")
article_rcs=env['product.product'].search([('state','=', 'lifeserie')])
for MaterialInformation in MaterialInformations:
    for article_rc in article_rcs:
        # créer une sous balise MaterialDefinition dans MaterialInformation
        MaterialDefinition=etree.SubElement(MaterialInformation,'MaterialDefinition')# crée
balise MaterialDefinition
        # Créer balise ID avec la valeur -> code
        codep=etree.SubElement(MaterialDefinition,'ID')
        tools.lxml_element_set_text(codep,article_rc.code)
        # Créer balise Description avec la valeur -> name
        codep=etree.SubElement(MaterialDefinition,'Description')
        tools.lxml_element_set_text(codep,article_rc.name)

#Récupère le contenu en texte
file_content=etree.tostring(xml)
#sauvegarde le text dans fichier sur le serveur
env['mf.tools'].save_file("/home/openprod/Documents/toto.xml",file_content)
#affiche le contenu
#raise Warning(etree.tostring(xml))

```


Traitement BL

mf_do_partial_picking

Cette méthode permet de traiter un picking en précisant les ids des lignes à traiter ainsi que les quantités pour chaque ligne. Les lignes de mouvements du picking qui ne seront forcées avec une quantité à 0.

Méthode / Exemple	Données d'entrée	Données de sortie
<p>mf_do_partial_picking(picking_id, move_ids, qtys)</p> <p><i>Traitement des lignes 614 et 615 du picking d'id 68 avec quantité 5 et 3.</i></p> <pre>env["mf.tools"]. mf_do_partial_picking(68,[614,615],[5,3])</pre> <p>exemple utilisation API openprod :</p> <pre>URL : {{url serveur}} /web/api/endpoint { "db": {{base}}, "token": {{token}}, "model": "mf.tools", "method": "call_kw", "call_method": "mf_do_partial_picking", "args": [], "kwargs": [{"picking_id", 68}, ["move_ids", [614, 615]], ["qtys", [5, 3]] }</pre>	<p>picking_id : id du picking à traiter</p> <p>move_ids : liste des id des mouvements à traiter</p> <p>qtys : liste des quantités à traiter (correspondant aux mouvements de la liste move_ids)</p>	<p>True sur l'opération à succès.</p>