

23/03/2023

SERIES

M/Adapter

Utilisation et configuration

Table des matières

| | | |
|------|--|----|
| I. | Introduction | 4 |
| II. | Principe..... | 4 |
| III. | Paramétrage | 4 |
| A. | Formats des fichiers de configuration..... | 4 |
| 1. | Configuration parente | 5 |
| B. | Structure de dossiers..... | 5 |
| C. | Connexion Series/M..... | 5 |
| D. | Synchronisation..... | 5 |
| E. | Configuration des documents..... | 6 |
| 1. | Identification | 6 |
| 2. | Template | 6 |
| 3. | Datasource | 6 |
| a) | Datasources additionnelles | 6 |
| 4. | Fichier de contrôle..... | 7 |
| 5. | Destination..... | 7 |
| 6. | Metadonnées | 7 |
| 7. | Paramètres M/OMS..... | 7 |
| 8. | Filtres..... | 7 |
| a) | Xslt..... | 8 |
| b) | Commande | 8 |
| F. | Scanners..... | 8 |
| 1. | Filtres..... | 8 |
| a) | Commande | 8 |
| b) | JDE | 8 |
| c) | XSLT..... | 8 |
| G. | Parsers..... | 9 |
| 1. | RDI | 9 |
| 2. | StreamIn | 9 |
| 3. | FieldIn..... | 10 |
| 4. | Documents formatés..... | 10 |
| d) | Fichier de contrôle | 11 |
| 5. | XML..... | 11 |
| e) | Pattern | 11 |
| f) | Split | 11 |
| H. | Events..... | 12 |
| I. | Purges | 12 |
| J. | Récapitulatif des paramètres..... | 12 |

| | | |
|-----|--------------------------------|----|
| K. | Contexte Freemarker | 14 |
| IV. | Lancement..... | 15 |
| A. | Scan directory | 15 |
| B. | Service Windows..... | 15 |
| C. | Lancement unitaire | 15 |
| D. | Impression PDF vers M/OMS..... | 16 |

I. Introduction

Le module Adapter a été créé afin de faciliter les migrations depuis les projets StreamServe.
L'Adapter reprend le fonctionnement en mode scandirectory, ainsi que les possibilités de StreamIn, FieldIn, RDI, PDFIn et OTFIn.
Hors migration StreamServe il permet également de traiter les flux dont le format n'est ni XML ni Json.

II. Principe

L'Adapter se charge de capturer les flux de données déposées dans un ou plusieurs dossiers scannés.

Les flux sont :

Identifiés

Parsés pour :

Découpage en documents unitaires

Création d'un dictionnaire de données rattaché à chaque document

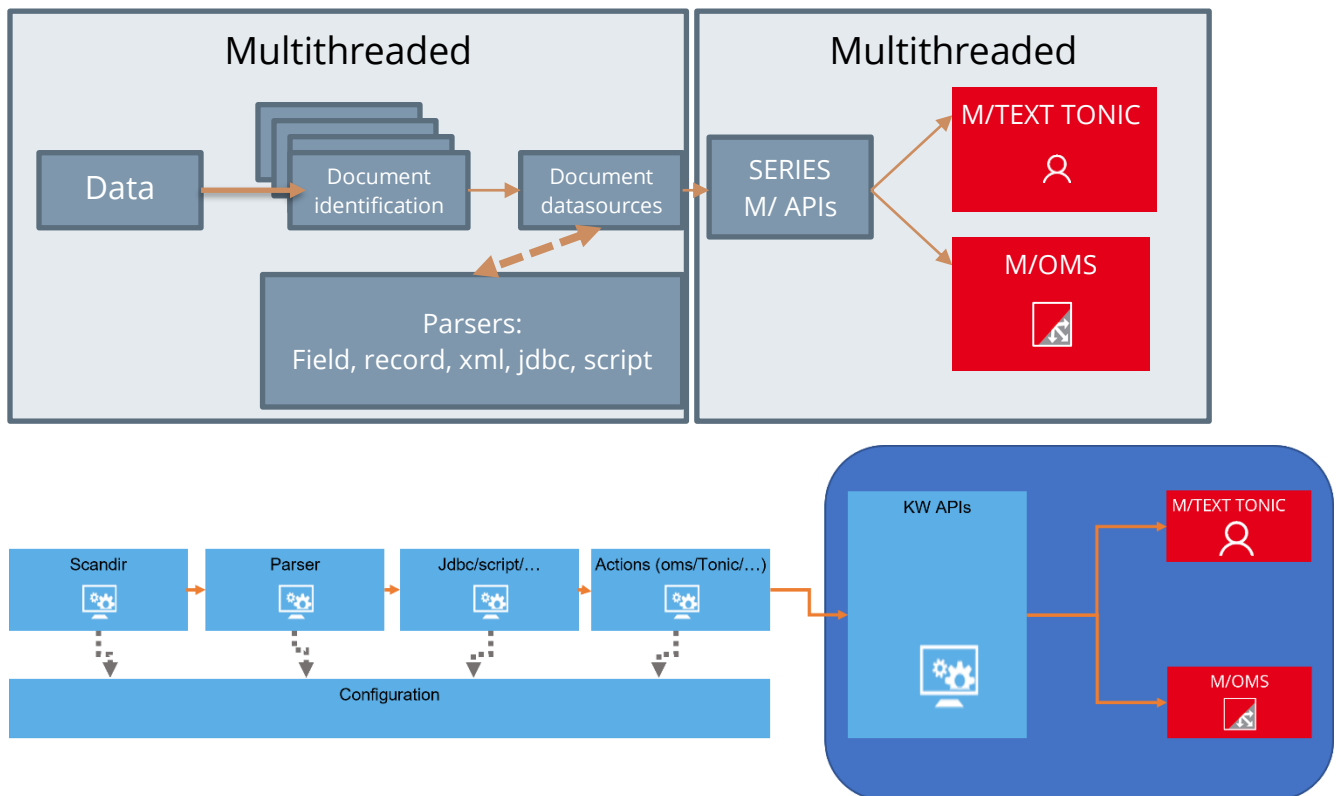
En fonction des settings, les documents seront envoyés vers KW pour :

Un retour PDF unitaire

Impression M/OMS

Création d'un Document dans Tonic

L'Adapter s'appuie sur les API Rest disponibles pour M/Text et M/OMS, ainsi que les librairies java fournies par KW.



III. Paramétrage

L'Adapter s'appuie sur plusieurs fichiers de paramétrage organisés suivant une structure de répertoire précise.

La plupart des settings peuvent être modifiés à chaud et/ou synchronisés depuis un projet dédié dans le Workbench. Ce projet de Settings devra être déployé de la même façon que les autres projets KW.

A. Formats des fichiers de configuration

Les fichiers sont au format **properties** Java (cf. <https://fr.wikipedia.org/wiki/properties>).

Ces fichiers sont encodés en ISO-8859-1, les paramètres sont stockés sous la forme « CLE=VALEUR », les caractères spéciaux ne pouvant être représentés en ISO-8859-1 doivent être codés sous la forme unicode escapes.

Unicode escapes :

UnicodeInputCharacter:

UnicodeEscape:

\ UnicodeMarker HexDigit HexDigit HexDigit HexDigit

UnicodeMarker:

u {u}

HexDigit:

(one of)

0 1 2 3 4 5 6 7 8 9 a b c d e f A B C D E F

Ex : \u2122 = ""

Quelques encodages spéciaux :

| Valeur à encoder | Encodage |
|--------------------------|----------|
| \ (anti-slash) | \\ |
| Tabulation | \\t |
| Saut de page (form feed) | \\f |
| nouvelle ligne | \\n |
| Retour chariot | \\r |

1. Configuration parente

Chaque fichier de configuration de l'adapter peut faire référence à un fichier de configuration parent. Si un paramètre n'est pas trouvé dans un fichier, il sera recherché dans la configuration parente.

Ex :

PARENT=common\\common.properties

B. Structure de dossiers

- <Adapter root folder>
 - bak : dossier où sont créés les backups des flux traités.
 - dictionaries : contient les fichiers ***.event** décrivant la structure des dictionnaires de données qui seront ensuite converties en XML.
 - dsc : fichiers descripteurs de type StreamIn, FieldIn, PDFIn, SAPGOFIn, RDI, XML.
 - Les extensions sont :
 - ***.dsc** : StreamIn, FieldIn, RDI, XML
 - ***.pdfin** : PDFIn
 - ***.sapgofin** : OTFIn
 - log : logs de l'adapter
 - scanners : Ce dossier contient des configurations permettant de faire des scans de dossiers dans l'attente de flux à traiter. Les fichiers de configurations dans ce dossier ont l'extension **.properties**.
 - tmp : dossier pour fichiers de travail temporaires.

xf : Ce dossier contient les templates des fichiers de contrôle à envoyer avec les documents préformatés (PDF, OTF). Les templates sont écrits en syntaxe Freemarker (<https://freemarker.apache.org/>). L'extension des fichiers templates est **.ftl**.

- **<main config>.properties** : fichier de configuration principal. Ce fichier contiendra généralement les informations de connexion au serveur SERIES M/, les paramètres de purge, les paramètres de synchronisation avec le serveur SERIES M/.

C. Connexion Series/M

L'Adapter devra se connecter à Series M/ pour soumettre les documents à traiter et optionnellement pour mettre à jour sa configuration.

Les paramètres de connexion sont placés dans le fichier de configuration principal, à la racine du dossier de travail de l'adapter (ex : main.properties).

Les paramètres sont :

KW_SERVER : le nom ou l'adresse IP du serveur

KW_SERVER_PORT : le port d'écoute du serveur

SSL_ENABLED : true si HTTPS, false si HTTP

KW_USER : l'utilisateur de connexion à KW

KW_PASSWORD_PLAIN : le mot de passe en clair

KW_PASSWORD_CRYPTED : le mot de passe crypté avec les outils fournis par KW.

Ex:

```
KW_SERVER=localhost
KW_SERVER_PORT=8080
SSL_ENABLED=false
KW_USER=mtext
KW_PASSWORD_CRYPTED=A050FB1827A63EED
```

D. Synchronisation

La configuration de l'adapter, hormis le fichier de configuration principal, peut être gérée comme un projet standard dans le Workbench. Ce projet sera déployé vers le serveur SERIES M/ lors des opérations de synchronisation (=déploiement) sur les serveurs.

Dans ce cas, on configure l'Adapter pour lui indiquer quel est le projet qui contient sa configuration. Il se connectera alors de façon régulière au serveur SERIES M/ et mettra sa configuration à jour si elle a été modifiée.

Il est également possible de demander à l'adapter une synchronisation via un dossier où sera livrée sa nouvelle configuration, on peut aussi directement mettre à jour ses fichiers de configuration.

Paramètres :

KW_SYNCHRO_SETTINGS_FOLDER : si la mise à jour se fait via la dépose de la nouvelle configuration dans un dossier de livraison dédié.

KW_SYNCHRO_SETTINGS_PROJECT : le projet contenant la configuration de l'adapter.

KW_SYNCHRO_SETTINGS_SCHEDULE : la période de mise à jour des settings.

Ce paramètre autorise deux types de syntaxe :

1. CRON : une syntaxe de type CRON suivant l'implémentation de Quartz (<http://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/crontrigger.html>)
2. Une fréquence de scan de type <nombre entier><unité>.

Les unités possibles sont :

- a. Millisecondes : ms
- b. Secondes : s
- c. Minutes : mn
- d. Heures : h
- e. Jours : d

Ex :

```
KW_SYNCHRO_SETTINGS_PROJECT=Adapter
KW_SYNCHRO_SETTINGS_SCHEDULE=1mn
```

E. Configuration des documents

Chaque document doit se voir attribuer un certain nombre de paramètres pour définir son traitement.

1. Identification

L'identification des documents est confiée à des « parsers ».

Les parsers sont les fichiers de description présents dans le dossier « dsc ». Ils existe des parsers de type « StreamIn », « FieldIn », « RDI », « PDFIn », « SAPGOFin », « XML ».

Les flux à traiter par l'Adapter vont être analysés en utilisant les parsers indiqués en paramètre. Si un parser n'identifie pas de document, le parser suivant dans la liste est utilisé.

Le rôle du parser est de reconnaître/identifier les documents dans le flux et de l'analyser pour produire une liste de champs afin de constituer un dictionnaire de données.

Exception : pour le XML, seule l'identification est réalisée, il n'est pas constitué de dictionnaire, le format XML étant déjà considéré comme un dictionnaire.

Lorsque le parser a identifié le document, le nom du document est utilisé pour récupérer la configuration qui lui est propre (template KW, datasources,...).

2. Template

Le template est le modèle de document qui sera utilisé pour la mise en forme des données.

La syntaxe à utiliser est : <document>_TEMPLATE=<chemin du template>

Ex :

```
Facture_TEMPLATE=/Facturation/Templates/Facture.template
```

Un seul template peut être attribué pour un document.

3. Datasource

Les templates KW sont associés à un ou plusieurs dictionnaires de données (data model). Le data model est vu par le document comme une datasource avec un nom.

L'appel d'un template avec un flux de données doit donc préciser le nom de la data source vers laquelle le flux de données sera fusionné.

Pour indiquer la data source d'un document, la syntaxe est :

<document>_DATASOURCE=<nom de la data source>

Ex :

```
Facture_DATASOURCE=DATA
```

a) Datasources additionnelles

- Fichiers additionnels

Un document peut être associé dans KW à plusieurs dictionnaires de données et donc plusieurs data sources. Il est possible dans la configuration du document d'aller chercher des flux de données supplémentaires pour les associer aux différents data modèles du document.

Il faut indiquer les fichiers supplémentaires à récupérer, ceux-ci seront « parsés » comme un flux de données classique et seront associés à un type de document.

Syntaxe :

- 1- Au niveau du document « maître », déclarer les fichiers supplémentaires à analyser :

<document>_ADDITIONAL_FILES=<fichier1> ;<fichier2> ;etc...

- 2- Déclarer pour chaque fichier supplémentaire, les documents qui ont doivent être trouvés :

<document additionnel>_ROUTAGE=NONE

<document additionnel>_DATASOURCE=<data source additionnelle>

- 3- Dans le document maître, ajouter les data sources additionnelles :

<document>_ADDITIONAL_DATASOURCES=<document additionnel1> :<data source additionnelle1> ;<document additionnel2> :<data source additionnelle2> ;etc...

Exemple :

Ici, un fichier de routage accompagne le document :

```
Facture_TEMPLATE=/Facturation/Templates/Facture.template
Facture_DATASOURCE=DATA
Facture_ADDITIONAL_DATASOURCES=ROUTAGE:ROUTAGE_DS
Facture_ADDITIONAL_FILES=..\routage\${originBaseName!}.rtg
```

```
ROUTAGE_TEMPLATE=NONE
ROUTAGE_DATASOURCE=ROUTAGE_DS
```

Explication de ce paramétrage :

A la réception d'une facture, on va chercher dans le dossier « ..\routage », un fichier ayant le même nom que le fichier reçu mais avec l'extension « .rtg ». La variable \${originBaseName !} utilise la syntaxe Freemarker.

Le fichier additionnel sera récupérer et analysé avec un parser. Le parser doit identifier un document nommé « ROUTAGE ». Ce document est paramétré avec un template à « NONE » pour indiquer qu'il s'agit d'un document additionnel.

On associe à ce document la data source « ROUTAGE_DS » qui constituera un data model supplémentaire pour le document maître.

Le document facture va ensuite parcourir les data sources additionnelles. Elle va alors trouver le document « ROUTAGE », associé à la data source « ROUTAGE_DS ».

Lors de l'envoi du document, vers KW, deux flux de données seront joints, l'un pour la data source principale « DATA », l'autre pour la data source additionnelle « ROUTAGE_DS ».

- Sources JDBC

Demander une assistance spécifique

- Sources Script

Demander une assistance spécifique

4. Fichier de contrôle

Dans le cas de documents préformattés (pdf, otf), un fichier de contrôle doit être constitué pour l'envoi vers M/OMS. Les fichiers de contrôle sont placés dans le dossier « xf » avec l'extension « .ftl ».

Par défaut le template du fichier de contrôle est recherché sous la forme <document>.ftl. On peut toutefois indiquer un autre template avec le paramètre <document>_OMS_CONTROL_JOB_TEMPLATE.

Ex :

```
Facture_OMS_CONTROL_JOB_TEMPLATE=commonControlJob.ftl
```

5. Destination

1- M/OMS

Pour envoyer le document vers M/OMS, il faut ajouter au document la propriété <document>_PRINTER=OMS.

Ex :

```
Facture_PRINTER=OMS
```

2- TONIC

Pour stocker le document dans TONIC, renseigner le paramètre <document>_TONIC_PATH=<chemin complet du document dans Tonic>

Le chemin du document dans Tonic est analysé par Freemarker et peut utiliser des variables issues du contexte du document (cf. Contexte Freemarker)

Ex :

```
Facture_TONIC_PATH=/Tonic/Facturation/${docType}/${date}/${originBaseName}
```

3- Retour PDF

Attention, les réponses PDF sont unitaires (par document).

Indiquer une valeur pour le paramètre <document>_RESPONSE_PATH.

Ex :

```
Facture_RESPONSE_PATH=./Facturation/${docType}/${originBaseName}.pdf
```

6. Metadonnées

Lors de la soumission des documents vers KW, des metadonnées peuvent être indiquées. Elles doivent correspondre à des metadonnées existante au niveau du document :

The screenshot shows a web interface for configuring metadata. On the left, there's a tree view under 'Metadata' with categories: Standard, Editor, and Custom. Under 'Custom', several metadata items are listed: jobid, inputFile, docIdx, inputFileName, and timestamp. On the right, under 'Properties: jobid', there are input fields for 'Value' and 'Value description'.

Pour indiquer une valeur à une meta donnée, renseigner le paramètre <document>_METADATA=<metadata name> :<metadata value> ;<metadata name> :<metadata value> ;etc...

Plusieurs metadata peuvent être ajoutées en les séparant par des « ; ». Ces valeurs sont analysées par Freemarker et peuvent donc contenir des variables issues du contexte du document.

Ex :

```
Facture_METADATA=jobId:${jobId!};inputFile:${P{origin!}};inputFileName:${originBaseName!};docIdx:${docIdx!};timestamp:${timestamp!}
```

7. Paramètres M/OMS

Une fois tous les documents soumis, s'ils ont été envoyés vers M/OMS, il est possible de mettre à jour les paramètres M/OMS des documents. C'est en particulier utilisé pour mettre à jour un statut permettant à M/OMS de ne traiter les documents que lorsque tous ont été soumis par l'adapter (cas d'un flux contenant plusieurs documents).

L'update n'est effectué que si tous les documents ont été soumis sans erreur.

On utilise pour ça le paramètre <document>_UPDATE_OMS_PARAMETERS=<parameter> :<value> ;<parameter> :<value> ;etc...

Ex :

```
ZAPP_APPNOTE_UPDATE_OMS_PARAMETERS=JOB_ID:${jobId!};JOB_SUBMIT_STATUS:DONE
```

Dans cet exemple, le fait de mettre JOB_SUBMIT_STATUS à DONE permettra aux stacks de traiter le document. Si une erreur se produit durant l'envoi des documents, cette mise à jour n'est pas effectuée et les documents resteront soumis dans la stack avec un statut intermédiaire.

8. Filtres

Lorsque le dictionnaire de données a été créé pour le document, celui-ci peut être représenté sous la forme d'un flux XML. Ce flux XML représente le Data model qui sera fourni à Series M/ pour générer le document.

Il est possible de transformer ce XML avant qu'il ne soit envoyé à KW en paramétrant un filtre au niveau du document. Le filtre peut être de type XSLT ou bien une commande externe fonctionnant en stdin/stdout

a) Xslt

Indiquer une feuille de style XSLT au document avec le paramètre suivant : <document>_FILTER=<chemin de la feuille de style>
Le chemin doit obligatoirement indiquer un fichier dont l'extension est « .xsl » ou « .xslt » afin que l'Adapter reconnaisse un filtre de type XSLT.
Ex :

```
Facture_FILTER=../xslt/Facture.xslt
```

b) Commande

Un programme externe peut être appelé pour transformer le XML du data modèle. Le programme lira le data model XML en « stdin » et devra renvoyer le résultat en « stdout ».

Il suffit d'indiquer la commande en paramètre :
<document>_FILTER=<commande du filtre externe>

Ex :

```
Facture_FILTER=../filters/FactureFilter.exe
```

F. Scanners

Les configuration des « scanners » doivent se situer dans le dossier <root folder>/scanners, sous la forme de fichier .properties.

Un fichier scanner doit contenir à minima les entrées suivantes :

FOLDER=<le dossier à scanner>

FILTER=<filtre sur les fichiers, syntaxe MS DOS>

SCHEDULE=<fréquence de scan>

Ex :

```
FOLDER=scanners/scan  
FILTER=*.*  
SCHEDULE=5s
```

Autres entrées :

PARSERS=<parser1>;<parser2>;etc...

Il s'agit de la liste des parsers qui seront utilisés pour analyser les flux entrants. L'ordre est important car les parsers sont utilisés de façon séquentielle jusqu'à ce qu'un document soit reconnu.

Ex :

```
PARSERS=dsc/Facture.dsc;dsc/routage.dsc
```

ENCODING=<encoding name, UTF-8, ISO8859-1,...>

The encoding that should be used to read the files received. It will be ignored for XML or PDF.

Ex :

```
ENCODING=ISO8859-1
```

PARENT=<Fichier de configuration parent>

L'option « PARENT » peut être utilisée dans tous les fichiers .properties, elle indique un fichier de configuration à utiliser si une propriété n'est pas présente.

Ex :

```
PARENT=../main.properties
```

DELETE_FILES=true/false

Indique si le scanner doit supprimer les fichiers du dossier scanné. Généralement oui.

Ex :

```
DELETE_FILES=true
```

1. Filtres

Les fichiers trouvés par le scanner peuvent être transformés par un filtre. Soit le filtre JDE (spécifique aux PDF taggés de JDE), soit via une commande externe fonctionnant en stdin/stdout. Dans le cas où le scanner reçoit des flux XML, on peut également indiquer un filtre de type XSLT.

Pour sauvegarder le fichier avant sa transformation par le filtre, indiquer une valeur pour le dossier de backup.
Indiquer le type de données que le filtre est supposé produire.

a) Commande

Ex :

```
FILTER_TYPE=COMMAND  
FILTER_ARGUMENTS=<programme à lancer>  
FILTER_BACKUP_FOLDER=bak  
FILTER_RESULT_EXTENSION=txt
```

b) JDE

Ex :

```
FILTER_TYPE=JDE  
FILTER_ARGUMENTS=-arg dsc/jde.txt -field  
FILTER_BACKUP_FOLDER=bak  
FILTER_RESULT_EXTENSION=txt
```

c) XSLT

Ex :

```
FILTER_TYPE=XSLT  
FILTER_ARGUMENTS=<chemin de la feuille de style XSLT>  
FILTER_BACKUP_FOLDER=bak  
FILTER_RESULT_EXTENSION=txt
```


G. Parsers

Les parsers sont de différentes natures en fonction des flux à analyser. Certains fonctionnent en mode « ligne à ligne » (fieldin, streamin, rdi), d'autres nécessitent des outils spécifiques (PDF, OTF, XML).

Les extensions des fichiers de configuration pour les parsers sont :

.dsc : rdi, fieldin, streamin, xmlin

.pdfin : PDF

.sapgonfin : OTF

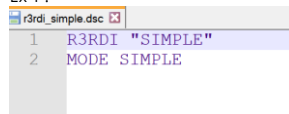
1. RDI

Le format RDI est supporté nativement dans l'Adapter. Son fichier de description contient juste un mot clé pour indiquer si l'on fait du RDI « standard » ou du RDI « simplifié » (plus compact).

Le nom du document qui servira de référence pour le paramétrage est le nom du formulaire SAP que l'on trouve en entête du rdi (cf. image ci-après).



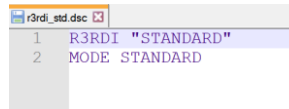
Ex 1 :



La première ligne « R3RDI "SIMPLE" » permet simplement de donner un nom au parser. Ce nom pourra être utilisé dans les messages de log.

Le mode indique ensuite que l'on parse un RDI « simplifié ».

Ex 2 :



2. StreamIn

Les parsers « StreamIn » utilisent la syntaxe des fichiers descripteurs de StreamServe. Des scripts peuvent être utilisés mais avec la syntaxe javascript. La ligne en cours d'analyse est disponible via la variable « line ». Le document en cours est disponible via « currentEvent » (peut être null).

Exemple de descripteur StreamIn avec du javascript pour l'identification. Dans cet exemple on parse des records identiques mais chaque document doit être constitué de deux records. On compte les lignes et une fonction modulo déclenche le premier record à chaque ligne impaire. Le premier record est quasi identique au second (mêmes champs) mails utilise le mot-clé « NewEvent » qui déclenche un nouveau document.

```
StreamIn "InFile"
Begin
    Record "AVIS" 1 ChrSep ";";
    Match SCRIPT {
        if (lineCounter == null || lineCounter == undefined){
            var lineCounter = 0;
            //var callCounter = 0;
        }
        function check(aLine){
            var matchAvis = aLine.startsWith('AVIS');
            if (! matchAvis){
                return false;
            }
            if ((lineCounter % 2) == 0){
                lineCounter = 1;
                return true;
            }else{
                return false;
            }
        }
        check(line);
    }
    NewEvent "AvisCoupure";
    Fields
        "ID";
        "Adr_Coupure";
        "Adr_Client_1";
        "Adr_Client_2";
        Etc...(liste de champs)
    End
End
Record "AVIS" 1 ChrSep ";";
Match SCRIPT{
    function check(aLine){
        var matchAvis = aLine.startsWith('AVIS');
        if (! matchAvis){
            return false;
        }
        lineCounter++;
        return true;
    }
    check(line);
}
InEvent "AvisCoupure";
Fields
    "ID";
    "Adr_Coupure";
    "Adr_Client_1";
    "Adr_Client_2";
    Etc...(liste de champs)
```

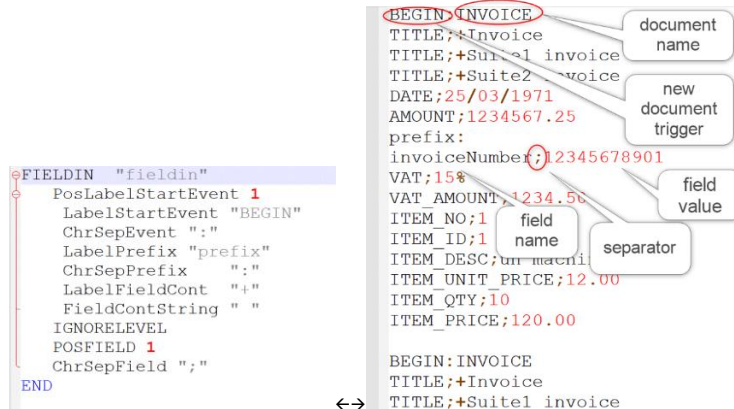
```

End
End
End

```

3. FieldIn

Les parsers FieldIn utilisent la syntaxe des fichiers descripteurs FieldIn de StreamServe, en revanche les scripts ne sont pas supportés.
Ex :



4. Documents formatés

Les parsers pour les documents formatés permettent d'identifier les documents en fonction des données contenues sur la page. La syntaxe est proche des event de type PreformatIn de StreamServe mais n'est pas strictement la même pour indiquer les types de page.

- ✓ Les scripts sur les champs sont supportés.
- ✓ La suppression de textes n'est supportée que pour l'OTF.
- ✓ La suppression d'images n'est pas supportée.
- ✓ Les parsers pour les PDF doivent avoir l'extension « .pdfin ».
- ✓ Les parsers pour l'OTF doivent avoir l'extension « .sagofin ».

Ex :

```

Event "Mail_in"
PageIn
// MessageID "2942224"

Field "Facture No" 51 229 111 241 $Facture No;
Field "Client No" 50 249 114 260 $Client No;
Field "Echeance" 51 266 115 279 $Echeance Format "Ddd.mm.yyyy";
Field "Compte" 52 286 120 299 $Compte;
Field "Montant" 51 305 118 318 $Montant Format "Nk=,d=";
Field "From" 52 324 449 337 $From;
Field "To" 52 343 451 356 $To;
Field "Niveau" 52 362 69 376 $Niveau;
LxfsSelection
  Begin
    <?xml version="1.0"?>
    <!DOCTYPE s-lxfin SYSTEM "s-lxfin.dtd">

    <s-lxfin>
      <page size="595.3,841.9" order="1" xoffset="15" type="FIRST">
        <pattern name="Mail" position="17.0,2.0" size="21.0,24.0" options="276" match="E"/>
        <field name="MailPattern" position="17.0,2.0" size="21.0,24.0" options="20" />
        <field name="Facture No" description="Numéro de facture" position="50.0,228.0" size="60.0,12.0" variable="Facture No" options="20"/>
        <field name="Client No" description="Numéro de client" position="49.0,248.0" size="64.0,11.0" variable="Client No" options="20"/>
        <field name="Echeance" description="Echéance" position="50.0,265.0" size="64.0,13.0" variable="Echeance" format="Ddd.mm.yyyy" options="20"/>
        <field name="Compte" description="Compte de contrat" position="51.0,285.0" size="68.0,13.0" variable="Compte" options="20"/>
        <field name="Montant" description="Montant" position="50.0,304.0" size="67.0,13.0" variable="Montant" format="Nk=,d=" options="20"/>
        <field name="From" description="Emetteur du mail" position="51.0,323.0" size="397.0,13.0" variable="From" options="20"/>
        <field name="To" description="Destinataire du mail" position="51.0,342.0" size="399.0,13.0" variable="To" options="20"/>
        <field name="Niveau" description="Type de mail de relance" position="51.0,361.0" size="17.0,14.0" variable="Niveau" options="20"/>
      </page>
    </s-lxfin>
  End
End

```

Seule la partie XML est utilisée.

Toutes les dimensions sont en points, à partir du coin supérieur gauche de la page.

<page> : représente un type de page.

Attribut **type** : Il faut indiquer le type de page : FIRST, NEXT, LAST.

<pattern> : identification de la page

Attribut **match** : pattern d'identification de la page. Si les wildcards sont activées, la syntaxe est celle des expressions régulières java (<http://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/crontrigger.html>).

Attribut **options** :

20 = pas d'utilisation de wildcards

Autre valeurs = utilisation de wildcards dans l'expression de l'attribut « **match** »

<field> : définition des champs de la page

Attributs **options** :

20 = garder les espaces, ne pas supprimer le texte

21 = garder les espaces, supprimer le texte

18 = supprimer les espaces, ne pas supprimer le texte

19 = supprimer les espaces, supprimer le texte

Du **javascript** peut être utilisé pour calculer le champ :

Ex :

Ici le javascript utilise des bibliothèques java (Apache commons Lang) pour faire des remplacements.

A noter : la section CDATA qui permet d'utiliser des caractères spéciaux pour éviter d'utiliser les entités xml à la place des caractères spéciaux.

<field name="QR" position="180.0,590.0" size="160.0,150.0" variable="QR" options="21"><![CDATA[

```

QR = org.apache.commons.lang3.StringUtils.replace(QR, "\n", "");
QR = org.apache.commons.lang3.StringUtils.replace(QR, "\r", "");
QR = org.apache.commons.lang3.StringUtils.replace(QR, "\\c013\\", "/r");
QR = org.apache.commons.lang3.StringUtils.replace(QR, "\\c010\\", "/n");
return QR;

```

]]></field>

d) Fichier de contrôle

Les documents préformatés (hormis les PDF taggés JDE) sont envoyés vers M/OMS. Cela nécessite un fichier de contrôle qui indiquera à M/OMS les target (KW_MEDIUM) et les divers paramètres des documents.

Les templates pour ces fichiers de contrôle doivent être placés dans le dossier <root folder>/xf, sous la forme <document>.ftl.

Les templates des fichiers de contrôle seront évalués par Freemarker qui remplacera les données variables en fonction du contexte de chaque document. Les champs mappés sur les documents peuvent être utilisés dans le fichier de contrôle.

Ex :

```

<#ftl output_format="XML">
<?xml version="1.0"?>
<document>
  <parameter name="DOCTYPE" value="${_docType}"/>
  <parameter name="JOB_ID" value="${jobId}"/>
  <parameter name="DOC_POSITION" value="${_docNum!}"/>
  <parameter name="OUT_FILE_NAME" value="-"/>
  <parameter name="JOB_DATE" value="${_date!}"/>
  <parameter name="JOB_TIME" value="${_time!}"/>
  <parameter name="PRINTER_NAME" value=""/>
  <parameter name="DOCUMENT_NO" value="${No_Matricule}"/>
  <parameter name="EMAIL_TO" value="${Email}"/>
  <parameter name="CLIENT_NO" value="${No_Matricule}"/>
  <parameter name="PERIODE_SALAIRE" value="${Periode!}"/>
  <target medium="SALAIRE_ARCHIVE" pages="1-" number="01" type="orig">
  </target>
  <#if Email?has_content>
    <target medium="SALAIRE_MAIL" pages="1-" number="01" type="orig">
    </target>
  <#else>
    <target medium="SALAIRE_PRINT" pages="1-" number="01" type="orig">
    </target>
  </#if>
  <pagedefinitions>
    <page number="1">
    <ocl value="PAGE_CONTROL DUPLEX_MODE=DUPLEX" /> </page>
  </pagedefinitions>
</document>

```

5. XML

Les flux XML peuvent normalement être gérés directement par KW. Toutefois dans le cas où le XML doit être splitté en documents individuels, on peut paramétrer l'Adapter pour reconnaître le flux XML et utiliser les APIs de KW pour découper le flux XML global en documents unitaires.

Le fonctionnement global est le suivant :

- 1- Identification du XML via un pattern de type XPATH.
- 2- Si le document reconnu a une option de split :
 - a. L'Adapter utilise une API de Series M/ pour découper le XML d'origine
 - b. Chaque XML unitaire constitue un data model qui est envoyé à Series M/ pour constituer un document
 - c. Le XML d'origine est ignoré et n'est pas envoyé à Series M/

e) Pattern

Les patterns pour identifier les XML sont placés dans un fichier descripteur.

L'extension du fichier descripteur est .dsc, il doit être placé dans le dossier « dsc ».

La première ligne contient le nom du descripteur. Ce nom est utilisé dans les logs.

XMLIN=<nom>

Les lignes suivantes sont de la forme :

<document>=<XPATH expression>

Si l'expression XPATH appliquée au XML reçu renvoie un résultat, le document est considéré comme identifié.

Exemple de descripteur XML :

```

XMLIN=xmlin

#Si un des xpath "matche", la clé indique le nom du document et est utilisé dans
configuration.properties pour lire les settings correspondants au document.
#La syntaxe est en XPATH

R584302C=/R584302C/Vue_triple_sur_F0411_F4311_F4101_S1_Group/LBH_On_Ship_To_S3
R580301=/R580301/Vue_sur_F03B21_et_F03B11_S1_Group/LBH_On_Address_Number_S23
R5519002=/R5519002/Vue_logique_sur_F4211_F4074_S1_Group/On_N_référence_S2

```

f) Split

Si le XML doit être splitté, il faut renseigner le paramètre <document>_XML_SPLITTER=<splitter>.

La valeur pour <splitter> correspond au chemin complet d'une configuration de Split déployée sur le serveur KW. Les « splitters » sont définis dans le Workbench et sont déployés avec le projet KW comme les autres ressources du projet (templates, data models, images,...).

Pour la définition du splitter, se référer à : *MWorkbench_En.pdf, 12.9 The Data Splitting*

Ex :

```
R584302C_XML_SPLITTER=/Configuration/splitters/R584302C.splitting
```

H. Events

Les fichiers .event, stockés dans le dossier « dictionaries » utilisent la syntaxe StreamServe.

Les formats sont ignorés, seuls les noms de blocs et de champs sont pris en compte.

Il s'agit de définir la structure du dictionnaire de données (blocs/sous-blocs) qui fournira au final la structure du XML envoyé à KW.

Les noms de champs et de blocs ne peuvent pas commencer par un nombre. Si besoin l'Adapter ajoutera la lettre « F » en préfixe.

Le nom du fichier doit correspondre au nom du document, son appel est implicite : lorsqu'un document est identifié (en RDI, FieldIn ou StreamIn), l'Adapter va rechercher si un fichier event correspondant existe.

S'il n'y pas de fichier, le XML produit respectera la structure de blocs initiale du rdi ou du streamin. Dans le cas du fieldIn, tous les champs seront à la racine.

Lorsqu'un champ créé par le parser n'est pas présent dans le fichier event, il sera créé tout de même, soit dans son bloc d'origine (cas du rdi ou du StreamIn), soit à la racine du document (dans ce dernier cas, si le champ est trouvé plusieurs fois, une seule instance sera retenue dans le xml final).

Ex :

```
StreamIn
// Generated with StreamIN
Root Name "Message"
  Field "HEADER_mode" $HEADER_mode
  Field "HEADER_dest" $HEADER_dest
  Field "HEADER_MastCctr" $HEADER_MastCctr Format "Nk= d=,"
  //Liste de champs...
  Field "CORE_END_editKey" $CORE_END_editKey
  Block "start"
    Field "CORE_START_editKey" $CORE_START_editKey
    Field "CORE_START_hSoldePrec"
    Field "CORE_START_hsPrec"
    Field "CORE_START_hCongeAnPrc"
    Field "CORE_START_proghebdo" $CORE_START_proghebdo
  End
  Block "core"
    Field "CORE_date" $CORE_date
    //Liste de champs...
    Field "CORE_libelleEvene"
    Field "CORE_hCongeAnRep"
  End
  Block "Total"
    Field "CORE_END_hTotal" $CORE_END_hTotal
    Field "CORE_END_indPiqTotal" $CORE_END_indPiqTotal
    Field "CORE_END_indRepTotal" $CORE_END_indRepTotal
    Field "CORE_END_indDepTotal" $CORE_END_indDepTotal
  End
  Block "Heure_due"
    Field "CORE_END_hDues" $CORE_END_hDues
    Field "CORE_END_hNuitTotal" $CORE_END_hNuitTotal
  End
  Block "nouveau_solde"
    Field "CORE_END_hSolde" $CORE_END_hSolde Format "Nk=,d=."
    Field "CORE_END_hEcret" $CORE_END_hEcret
    //Liste de champs...
    Field "CORE_END_hCongeAnTotal" $CORE_END_hCongeAnTotal
  End
End
End
```

I. Purges

Il est possible de paramétrer des purges périodiques, la syntaxe est la suivante :

PURGE=<purge1>,<purge2>,etc..

PURGE_<purge1>_FOLDER=<dossier à purger>

PURGE_<purge1>_FILTERS=<filtre sur les noms de fichiers (syntaxe MSDOS)>

PURGE_<purge1>_DAYS=<Age en jour des fichiers : les fichiers plus vieux sont supprimés>

Alternative : PURGE_<purge1>_AGE=<nombre><unité de durée : jour=d,heure=h,minute=mn>

PURGE_<purge1>_SCHEDULE=<expression CRON Quartz> ou <nombre><unité de durée (idem ci-dessus)>

Pour CRON Quartz : <http://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/crontrigger.html>

PURGE_<purge1>_RECURSIVE=true/false (parcours des sous-dossiers)

Ex :

```
PURGE=bak,InputJDE
PURGE_bak_FOLDER=.\bak
PURGE_bak_FILTERS=*
PURGE_bak_AGE=60d
PURGE_bak_RECURSIVE=true
PURGE_bak_SCHEDULE=1d
```

```
PURGE_InputJDE_FOLDER=.\input\jde
PURGE_InputJDE_FILTERS=*
PURGE_InputJDE_AGE=10d
PURGE_InputJDE_RECURSIVE=true
PURGE_InputJDE_SCHEDULE=1d
```

J. Récapitulatif des paramètres

| Paramètre | Description | Emplacement |
|-----------|-------------|-------------|
|-----------|-------------|-------------|

| Paramètre | Description | Emplacement |
|-------------------------------------|--|--|
| KW_SERVER | Nom/adresse IP du serveur KW | Fichier .properties principal Ex : main.properties |
| KW_SERVER_PORT | Port d'écoute du serveur KW | Fichier .properties principal |
| SSL_ENABLED | true/false (protocole https ou http) | Fichier .properties principal |
| LOG_LEVEL | Niveau de log, terminologie Java (https://docs.oracle.com/javase/7/docs/api/java/util/logging/Level.html) ALL, FINEST, FINER, FINE, CONFIG, INFO, WARNING, SEVER, OFF) | Fichier .properties principal |
| PARENT | Emplacement d'un fichier de configuration parent qui sera utilisé si un paramètre est absent. | N'importe quel fichier .properties. Il ne doit pas y avoir de « cycle » entre les fichiers de configuration (boucle infinie lors d'une recherche de paramètre) |
| KW_USER | Le user KW qui appelle les webs services | Fichier .properties principal |
| KW_PASSWORD_CRYPTED | Le mot de passe crypté (algorithme KW) | Fichier .properties principal |
| KW_PASSWORD_PLAIN | Le mot de passe en clair. | Fichier .properties principal |
| KW_SYNCHRO_SETTINGS_PROJECT | Le projet du workbench dédié aux settings de l'adapter. | Fichier .properties principal |
| KW_SYNCHRO_SETTINGS_SCHEDULE | Période de mise à jour des settings (Syntaxe CRON librairie java Quartz ou une période en indiquant l'unité d, h, mn) http://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/crontrigger.html | Fichier .properties principal |
| KW_SYNCHRO_SETTINGS_FOLDER | Dossier pour la mise à jour des settings, méthode alternative à KW_SETTINGS_PROJECT | Fichier .properties principal |
| KEEP_PARSED | true/false Indique s'il faut faire un backup ou pas des fichiers produits par les parsers (Fichier de données XML, contexte Freemarker du document, récapitulatif du document (statut, message d'erreur, routage,...)) | Fichier .properties principal |
| BACKUP | true/false Indique si les fichiers traités doivent faire l'objet d'un backup ou pas. | Fichier .properties principal |
| PURGE | Liste des purges, séparées par des virgules « , » | Fichier .properties principal |
| PURGE_<purge name>.FOLDER | Dossier à purger | Fichier .properties principal |
| PURGE_<purge name>.FILTERS | Filtres sur les fichiers à purger. Plusieurs filtres peuvent être indiqués (séparateur « , »). | Fichier .properties principal |
| PURGE_<purge name>.DAYS | Age maximal des fichiers en jours. | Fichier .properties principal |
| PURGE_<purge name>.AGE | Age maximal des fichiers en précisant l'unité (d, h, mn). Alternative à PURGE_<purge name>.DAYS. | Fichier .properties principal |
| PURGE_<purge name>.RECURSIVE | true/false Recherche ou pas dans les sous dossiers | Fichier .properties principal |
| PURGE_<purge name>.SCHEDULE | Ordonnancement des purges. Nombre + unité (d, h, mn) OU expression CRON librairie java Quartz. http://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/crontrigger.html | Fichier .properties principal |
| PROCESSING_FOLDER | Dossier de travail de l'Adapter pour les flux en cours de traitement. (tmp\processing) | Fichier .properties principal |
| <document>_TEMPLATE | Le template KW à appeler pour ce document. Ex : Facture_TEMPLATE=//Facturation/templates/Facture.template | Fichier .properties principal |
| <document>_DATASOURCE | La datasource à utiliser pour le flux de données. Ex : Facture_DATASOURCE=DATA | Fichier .properties principal |
| <document>_METADATA | Les metadata à valoriser pour le document soumis. Syntaxe : <nom> :<valeur>;<nom>:<valeur> ;etc... On peut utiliser des variables « Freemarker ». Ex : Facture_METADATA=JobId :\${jobId !} ;InputFile:\${originBaseName!} | Fichier .properties principal |
| <document>_OMS_CONTROL_JOB_TEMPLATE | Template qui servira à générer le fichier de contrôle dans le cas d'un document préformaté. Si cet argument n'est pas renseigné, on recherche <document>.ftl. Les templates sont obligatoirement dans le dossier « xf ». | Fichier .properties principal |
| <document>_ADDITIONAL_FILES | Fichiers additionnels à rechercher et à parser pour ce document. Cela va généralement créer de nouvelles data sources pour <document>_ADDITIONAL_DATASOURCES. | Fichier .properties principal |
| <document>_ADDITIONAL_DATASOURCES | Sources de données supplémentaires à envoyer pour ce document. Elles peuvent être issues de documents précédents dans le flux (ex : une entête technique parsée de façon indépendante), ou bien provenir de fichiers additionnels indiqués par <document>_ADDITIONAL_FILES. | Fichier .properties principal |
| <document>_PRINTER | Imprimante vers laquelle M/Text devra envoyer le document (normalement « OMS ») | Fichier .properties principal |

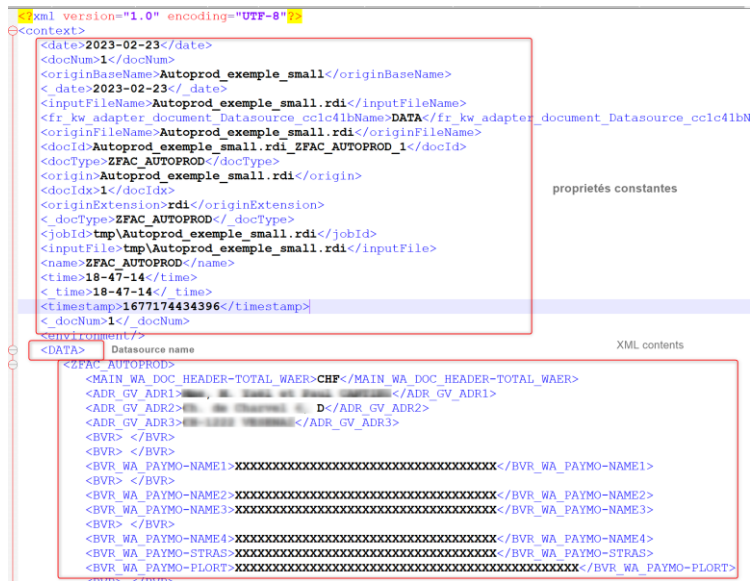
| Paramètre | Description | Emplacement |
|----------------------------------|---|---|
| <document>_UPDATE_OMS_PARAMETERS | Lise de paramètres à mettre à jour dans M/OMS. Utilise Freemarker. Ces paramètres ne sont mis à jour que si TOUS les documents ont été traités sans erreur. Sert habituellement à mettre à jour un statut « SUCCESS » pour permettre le traitement par M/OMS. Ex : Facture_UPDATE_OMS_PARAMETERS=JOB_ID:\${jobId};JOB_SUBMIT_STATUS:SUCCESS | Fichier .properties principal |
| <document>_TONIC_PATH | Chemin complet du document à créer dans Tonic. Utilise Freemarker. | Fichier .properties principal |
| <document>_RESPONSE_PATH | PDF à créer en réponse à l'appel du Web Service. Utilise Freemarker. Ne peut que créer des documents unitaires (pas de réponse globale si plusieurs documents dans le flux d'entrée). | Fichier .properties principal |
| <document>_FILTER | Filtre à utiliser sur la datasource principale du document. Peut être un programme externe fonctionnant en stdin/stdout, ou une feuille de style XSLT. | Fichier .properties principal |
| <document>_XML_SPLITTER | Indique la configuration XML Splitter à récupérer dans KW (fichier géré dans le workbench). | Fichier .properties principal |
| PARSERS | Liste des parsers disponibles pour un scanner. Plusieurs parsers peuvent être donnés, séparés par des « ; ». Ex : PARSERS=dsc/Facture.dsc;dsc/rdi;dsc/routage.dsc | Fichier .properties dans dossier « scanners » |
| ENCODING | Codepage à utiliser pour parcourir les flux reçus (streamin, fieldin, rdi). (https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html) | Fichier .properties dans dossier « scanners » |
| FILTER_TYPE | Type de filtre à appliquer sur les flux reçus avant leur parsing. COMMAND, JDE, XSLT | Fichier .properties dans dossier « scanners » |
| FILTER_ARGUMENTS | Si type JDE : Arguments du filtre JDE (Ex : -arg dsc/jde.txt -field) Si type XSLT : emplacement de la feuille de style XSLT Si type COMMAND : exécutable à appeler | Fichier .properties dans dossier « scanners » |
| FILTER_BACKUP_FOLDER | Dossier de backup pour sauvegarder les flux originaux (avant filtrage) | Fichier .properties dans dossier « scanners » |
| FILTER_RESULT_EXTENSION | Extension du type de flux produit par le filtre. Peut servir à l'adapter pour déterminer le traitement du fichier (ex. Si « pdf » le flux pourra être traité comme un pdf en écartant les parsers de type « texte » ou « XML ») | Fichier .properties dans dossier « scanners » |
| PARSE_ONLY | true/false Si « true », les flux ne sont pas envoyés vers KW, seulement sauvegardés (si FILTER_BACKUP_FOLDER a une valeur) | Fichier .properties dans dossier « scanners » |
| DELETE_FILES | true/false Si « false », les fichiers ne sont pas supprimés du dossier scanné. | Fichier .properties dans dossier « scanners » |
| FOLDER | Dossier à scanner | Fichier .properties dans dossier « scanners » |
| FILTER | Filtre des fichiers à rechercher (syntaxe MS DOS) | Fichier .properties dans dossier « scanners » |
| SCHEDULE | Période de scan. Syntaxe CRON Quartz (http://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/crontrigger.html) ou un nombre avec l'unité de temps (d,h,mn,s,ms) | Fichier .properties dans dossier « scanners » |

K. Contexte Freemarker

Les paramètres supportant les expressions Freemarker ou fichiers de contrôle M/OMS ont à leur disposition le contexte du document pour insérer des valeurs dynamiques.

Consulter la documentation Freemarker (<https://freemarker.apache.org/docs/index.html>) pour plus de détails sur les possibilités offertes (boucles, calculs, fonctions, tests,...).

| Variable | Commentaire |
|--|---|
| <code>\${docNum}</code> <code>\${_docNum}</code> <code>\${docIdx}</code> | Position du document au sein du flux initial. |
| <code>\${originBaseName}</code> | Nom du fichier entrant sans chemin, sans extension |
| <code>\${originFileName}</code> <code>\${inputFileName}</code> <code>\${origin}</code> | Nom du fichier entrant sans chemin, avec extension |
| <code>\${originExtension}</code> | Extension du fichier reçu |
| <code>\${_date}</code> <code>\${date}</code> | Date du jour (yyyy-mm-dd) |
| <code>\${docId}</code> | Identifiant de document généré par l'adapter |
| <code>\${docType}</code> <code>\${_docType}</code> <code>\${name}</code> | Type de document |
| <code>\${jobId}</code> | Identifiant du job généré par l'Adapter |
| <code>\${time}</code> <code>\${_time}</code> | Heure (hh-mm-ss) |
| <code>\${timestamp}</code> | Timestamp en ms |
| <code>\${<datasource>.<root>.<field>}</code> | Accès aux données du document. Ex : \${DATA.ZFAC_AUTOPROD.HEAD_P_ADRESSE01} |



IV. Lancement

A. Scan directory

```
java -cp KWAdapter.jar;KWAdapter_lib\* -Dksoft.env.globalcontext=ini\client.ini -Djava.io.tmpdir=tmp -Djavax.net.ssl.trustStore=ini\cacerts fr.kw.adapter.engine.scanner.StartScanners -config main.properties
```

- Le fichier client.ini est disponible dans l'installation de KW, il est utilisé par certaines librairies KW utilisées dans l'Adapter
- L'option -Djavax.net.ssl.trustStore=ini\cacerts.fr peut être utile dans un contexte SSL pour gérer une liste de certificats approuvés

B. Service Windows

Il est possible d'installer l'Adapter en tant que service Windows avec « Apache Commons Daemon» <https://commons.apache.org/proper/commons-daemon/procrun.html>.

Il est possible de renommer l'exécutable Apache en « KWAadapter-service.exe » car cet utilitaire se base sur son nom pour la création du service.

Voici un exemple d'utilisation :

```
KWAdapter-service.exe //IIS/KWAdapter-service
--Install=D:\KWSOFT\adapter\WindowsServiceTool\KWAdapter-service.exe
--Description="KWAdapter"
--Jvm="C:\Program Files\Eclipse Adoptium\jdk-11.0.14.101-hotspot\bin\server\jvm.dll"
--Classpath=D:\KWSOFT\adapter\KWAdapter.jar;D:\KWSOFT\adapter\KWAdapter_lib\*
--StartMode=jvm --StartClass=fr.kw.adapter.engine.scanner.StartScanners
--StartMethod=main
--StartParams=start
++StartParams=-config=D:\KWSOFT\adapter\main.properties
--StopMode=jvm --StopClass=fr.kw.adapter.engine.scanner.StartScanners
--StopMethod=stopService
--StopParams=stop --LogPath=D:\KWSOFT\adapter\log
--StdOutput=auto --StdError=auto
--StartPath=D:\KWSOFT\adapter
--JvmOptions=-Dkwsoft.env.globalcontext=D:\KWSOFT\adapter\ini\client.ini
++JvmOptions=-Djava.io.tmpdir=D:\KWSOFT\adapter\tmp
++JvmOptions=-Djavax.net.ssl.trustStore=D:\KWSOFT\adapter\ini\cacerts
```

Une fois les arguments corrects, lancer `KWAdapterService.bat` depuis une fenêtre MSDos lancée en mode administrateur. Le service est créé sous le nom « **KWAdapter-service** »

Si après essai, le service ne fonctionne pas correctement, il est possible de le supprimer avec la commande : « **sc delete KWAdapter-service** »

C. Lancement unitaire

L'Adapter peut être lancé en ligne de commande pour traiter un flux de façon unitaire.

Ex :

kw.adapter.engine.ProcessFile

```
java -cp KWAdapter.jar;KWAdapter_lib\* -Dksoft.env.globalcontext=ini\client.ini -Djava.io.tmpdir=tmp -Djavax.net.ssl.trustStore=ini\certs fr.kw.adapter.engine.ProcessFile -config scanners\<mon scanner>.properties -baseConfig main.properties -in <file to process> [-noRefresh] [-noPurge] [-refreshOnly]
```

A noter que la configuration part généralement d'une configuration de type « scanner » car elle contiendra les informations sur les parsers à utiliser.

L'option `-baseConfig` n'est pas obligatoire (mais conseillée) si la configuration du parser référence `main.properties` via la propriété « `PARENT` ».

-noRefresh : on demande à l'Adapter de ne pas chercher à rafraichir ses settings en se connectant au serveur KW.

-noPurge : on demande à l'Adapter de ne pas se préoccuper des paramètres de purge.

- refreshOnly : L'Adapter fait juste une mise à jour de ses settings (en se connectant au serveur KW).

D. Impression PDF vers M/OMS

L'Adapter peut être utilisé pour envoyer des PDF vers M/OMS avec un fichier de contrôle. Cette fonctionnalité offre aussi la possibilité de redimensionner le pdf.

Exemple d'utilisation (contexte Gerflor) :

- le PDF JDE reçu est sauvegardé par l'adapter et converti en XML par le filtre JDE.
- Le XML génère un document « dummy » qui part vers M/OMS dans un stack « PRE-PRINT »
- La stack PRE-PRINT lance une commande externe qui va utiliser les possibilités de l'adapter pour envoyer le PDF d'origine vers M/OMS (pour traitements par les stacks classiques de mail, archivage, impression,...). Au passage le pdf peut être redimensionné via un paramétrage (cf. Common\scripts\OMS\getPPDFScale.js)

Ex :

```
java -cp KWAdapter.jar;KWAdapter_lib\* -Dkwsoft.env.globalcontext=ini\client.ini -Djava.io.tmpdir=tmp -Djavax.net.ssl.trustStore=ini\cacerts fr.kw.adapter.engine.ProcessFile -config main.properties -noPurge -noRefresh -pdfDirect -pdfScale <redimensionnement> -input <fichier PDF> -pdfControlJob <fichier de contrôle M/OMS>
```

-pdfscale : échelle à appliquer pour redimensionner le pdf

Ex :

1.0= pas de redimensionnement

0.5 = réduction de 50%

-input : le PDF à traiter

-pdfControlJob : le fichier de contrôle pour M/OMS

-config <main.properties> : le fichier de configuration qui contient les informations de connexion au serveur KW.

-noPurge : ne pas exécuter les jobs de purges ayant pu être configuré dans main.properties

-noRefresh : ne pas se connecter au serveur KW pour mettre les settings à jour.

Les arguments en italique sont facultatifs et dépendent du contexte :

-Djava.io.tmpdir=tmp : on indique que le dossier temporaire est le dossier « tmp »

-Djava.nst.ssl.trustStore=ini\cacerts : utiliser le fichier ini\cacerts pour la liste des certificats numériques approuvés. Ceci est utilisé lorsque le serveur KW utilise du HTTPS, dans ce cas, le certificat du serveur doit faire partie des certificats approuvés afin que le programme client (ici Java et l'Adapter) accepte de s'y connecter.