Module d'import de données

Structure générale

```
<?xml version="1.0" encoding="utf-8" ?>
<catalogfilter version="2">
        <filter name="Visible name" type="xls" />

        <option name="prefix" type="string" required="1" checkregex="^\d+$" />

        <zones>
                <zone name="work" level="work" col="1" isactivecol="C"
isactiveregex="^(\d+|)$">
                        <field name="title" col="A" ghregex="Title/Alternate Title\(s\)">
                                <helper type="regex" when="beforepost" find="\s+" replace=""
/>
                        </field>

                        <newrec>
                                <condition type="change" field="title" />
                        </newrec>

                        <onpost>
                                <![CDATA[
                                Begin
                                        if getstring('title')=getstring('subtitle') then
                                                putstring('subtitle','');
                                End.
                                ]]>
                        </onpost>
                </zone>
        </zones>
        <lookups>
                <lookup name="abatprix" valuetype="integer">
                        <entry key="FULL" value="1" />
```

```
                </lookup>
        </lookups>

</catalogfilter>
```

Filter

```
<filter name="filtername" type="xls|csv|fix" />
```

Attributs

Attribut	Obligatoire	Explication
name	Oui	Le nom du filtre tel qu'il apparaîtra dans la liste
type	Oui	Le type de fichier : xls ou csv ou fix
maxheader	Non	Le nombre de lignes où chercher les en-têtes (par défaut = 20)
delim	si type="csv"	Le délimiteur à utiliser (par défaut « , »).
		* tab ou $09 pour les tabulations
		* auto pour détection automatique
engine	Non	Le moteur de lecture à utiliser \ * si type= ''xls'' : ''xlsrw'' ou ''cx'' \ * si type= ''csv'' ou ''fix'' : ''fileread[.encoding]'' ou ''memorymap'' (où encoding peut être ''UTF8'', ''CP437'', ''ANSI'', ''ASCII'')
reconcile	Non	Par défaut : « Non ». Si le moteur de lecture est ''fileread'', essayer de « recoller » les lignes qui contiennent des retour chariot.
unquote	Non	Par défaut : « Non ». Nettoyer les champs s'ils commencent et finissent par des "

Zones

```
<zone name="zonename" attributes="..." />
```

Liste des attributs de zones

Attribut	Obligatoire	Explication
name	Oui	Le nom du filtre tel qu'il apparaîtra dans la liste
level	Oui	Le nom du niveau (facultatif si clone)
col	Oui	La position où commence la zone (ou auto si ghregex)
clone	Non	Nom de la zone à cloner
parent	Spécial	Sur les imports multi-niveaux, définit le nom de la zone parente
ghregex	Non	Permet de repérer la zone (mettre col=''auto'') et de définir les champs avec col=+0 ou col=+2
zonegroup	Non	Permet de définir une chaîne réutilisable avec ''%zg%'' dans les ghregex et les ''''
type	Non	Définit une zone de type ''table''

Lookups (tables de correspondances)

```
<lookups>
        <lookup name="tors" valuetype="string">
                <entry key="LP" value="S" />
                <entry key="VIDEO TRACK" value="T" />
        </lookup>
</lookups>
```

Champs (fields)

```
<field name="title" col="A" />
```

Liste des attributs de champs

Attribut	Obligatoire	Explication
name	Oui	Le nom du champ tel que défini par l'appelant

Attribut	Obligatoire	Explication
col	Oui	La position du champ\ \ numéro de colonne\ colonne Excel (A-Z, AA, AB, etc.)\ n,l : n = numéro de colonne, l=longueur (pour les types « fix »)\ +n : numéro de colonne relativement à la zone (+0 pour le premier champ)\ auto : la colonne sera trouvée avec ghregex\ 1%%
ghregex	Non	Regex de l'en-tête pour contrôle ou découverte (si col=auto)
readasstring	Non	Si « 1 » le champ sera lu/vu comme une chaîne jusqu'à son enregistrement ce qui permet de lui appliquer des regex, eval et autres helpers
defaultvalue	Non	Valeur par défaut si le champ est vide
forcedvalue	Non	Valeur du champ forcée
optional	Non	« 1 » si le champ n'est pas systématiquement présent dans les fichiers
checktype	Non	* regex\ * change
checkregex	Oui	Si checktype = regex
onfail	Oui	* null : le champ est nullifié \ * empty : le champ est vidé \ * ignorerec : l'enregistrement est ignoré \ * ignoretab : arrêt du traitement de l'onglet courant \ * fatalerror : le traitement est interrompu
onfailmessage	fatalerror	Message affiché en cas d'erreur fatale (%s pour afficher un contenu capturé par la regex)
acceptif	Non	Raccourci pour checktype=regex + readasstring + OnFail=IgnoreRec
ignoreif	Non	Raccourci pour checktype=regex + readasstring + OnFail=IgnoreRec (regex=!regex)
fatalif	Non	Raccourci pour checktype=regex + readasstring + OnFail=FatalError (regex=!regex) \ Il est possible de préciser un onfailmessage
fatalifnot	Non	Raccourci pour checktype=regex + readasstring + OnFail=FatalError + onfailmessage \ Il est possible de préciser un onfailmessage

Attribut	Obligatoire	Explication
keytype	Non	''primary'' : ce champ est unique et toujours présent \ ''unique'' : ce champ est unique ou vide \ ''fallback''
codetype	si keytype	Type de code à associer
sticky	Non	La valeur du champ est mémorisée pour les lignes suivantes où cette valeur serait vide jusqu'au <newrec> du niveau immédiatement supérieur
verysticky	Non	La valeur du champ est mémorisée pour les lignes suivantes où cette valeur serait vide jusqu'au <newrec> du niveau le plus haut
isactivecol	Non	Colonne où vérifier isactiveregex
isactiveregex	Non	regex à vérifier pour que la ligne soit prise en compte
fillbeforeheadersfound	Non	''1'' pour les champs Sticky dont les valeurs sont définies avant que tous les en-têtes aient été trouvés
clearhelpers	Non	Pour les zones clonées, ''1'' supprime les helpers hérités

Contrôles

Avant de prendre en compte une donnée, plusieurs contrôles peuvent être effectués :

^checkactive|Permet de vérifier, à un autre endroit de la ligne, que le champ qu'on s'apprête à lire est bien celui qu'on croit. Sur les fichiers de type « fix », cela permet par exemple de s'assurer que le type d'enregistrement est celui attendu.| ^check|Permet de vérifier la validité de la donnée et de déterminer quoi faire dans le cas contraire (ignorer la ligne, le fichier ou interrompre le traitement si l'erreur rencontrée indique un fichier corrompu ou invalide)|

Les traitements sont effectués dans cet ordre :

- Vérifier que le champ est bien défini (certains champs sont facultatifs)
- Appliquer les helpers checkactive
- Vérifier que le champ est bien actif (checkactive)
- Appliquer les helpers afterread
- Vérifier la validité de la donnée (check)
- Appliquer les helpers beforepost

Helpers

```
<helper type="regex" when="beforepost" find="xxx" replace="yyy" else="zzz" />
```

When

When	Description
afterheaderread	N’est utilisé que dans la phase de recherche d’en-têtes, avant de vérifier si ceux-ci répondent au(x) critère(s) attendu(s).
beforecheckactive	Traitement appliqué aux champs checkactive (donc généralement pas le champ lui-même mais peut être utilisé pour opérer une transformation de la donnée uniquement aux fins de vérification en la gardant intacte)
afterread	Effectué avant de vérifier la conformité de la donnée
beforepost	Effectué après vérification de conformité mais avant stockage
userdefined	Non utilisé actuellement

Types

Type	Description
lowercase	Force le champ en minuscules.
uppercase	Force le champ en capitales
trim	Élimine les espaces de début et de fin (déjà fait à la lecture du champ)
lpad	remplit à gauche avec un caractère pour atteindre une longueur prédéfinie. Si le champ est déjà trop long, il est tronqué. \ ''fill'' : caractère à utiliser pour combler \ ''length'' : longueur à atteindre
rpad	remplit à droite avec un caractère pour atteindre une longueur prédéfinie. Si le champ est déjà trop long, il est tronqué. \ ''fill'' : caractère à utiliser pour combler \ ''length'' : longueur à atteindre
round2	Arrondit le champ (float) à 2 chiffres après la virgule
round8	Arrondit le champ (float) à 8 chiffres après la virgule
eval	Évaluation mathématique du champ (par exemple 100/2)
regex	Recherche un motif et le remplace par une chaîne. Les portions isolées par des parenthèses sont remplacées par $1, $2, etc. \ ''find'' : motif à rechercher \ ''replace'' : chaîne par laquelle remplacer
regexeval	Combine regex et eval. Par exemple : \ ''find'' : ''^(.*)$'' \ ''replace'' : $1/2
regexelse	Recherche le motif find et remplace l’ensemble de la chaîne par replace en cas de succès ou par else en cas d’échec ou de champ vide/nul. \ ''find'' : motif à rechercher \ ''replace'' : chaîne par laquelle remplacer \ ''else'' : chaîne alternative

Type	Description
split	Récupère le N-ième élément en séparant le champ lu sur un ou plusieurs caractères \ ''splitchars'' : un ou plusieurs caractères qui serviront à séparer les éléments \ ''count'' : le N-ième élément à récupérer
userdefined	Permet au module appelant (ImportCatalogues) de définir un helper particulier à appliquer juste avant d'enregistrer les données. \ ''udname'' : nom du helper qui sera reconnu par l'appelant
lookup	Permet de transformer une valeur lue par une autre, définie dans une table de correspondance \ ''lookup'' : nom de la table de correspondance \ ''keep'' : si 1 on garde la valeur si correspondance non trouvée
concat	Concatène deux champs (avec un éventuel séparateur) \ ''addfield'' : champ à ajouter au champ courant \ ''separator'' : séparateur à ajouter (facultatif)
tablelookup	Effectue une recherche de ''key'' sur une zone de type ''table'' et retourne la valeur ''value'' \ ''table'' : zone de type ''table'' où faire la recherche \ ''key'' : donnée à rechercher \ ''value'' : valeur à récupérer
log	Ajoute la valeur courante du champ dans le journal \ ''prefix'' peut être utilisé pour distinguer différentes lignes

Code

Actuellement, chaque zone peut avoir un code associé, placé dans une directive . Outre le langage Pascal standard, les fonctions suivantes sont disponibles :

Fonction	Description
getstring(name) \ **getinteger**(name) \ **getfloat**(name)	Récupère la valeur d'un champ \ ''name'' : nom du champ à récupérer
putstring(name,value) \ **putinteger**(name,value) \ **putfloat**(name,value)	Stocke des données dans un champ \ ''name'' : nom du champ \ ''value'' : valeur à stocker
TestRegex(str,expression,options:String):Boolean	Vérifie si une chaîne matche une expression régulière \ ''str'' : chaîne à tester \ ''expression'' : regex \ ''options'' : ''i'' pour ignorer MAJ/min
Eval(str):Integer \ **DEval**(str):Double	Convertit une chaîne en nombre (ou zéro si la chaîne ne représente pas un nombre)
GetLastSerial(level:str):Integer	Retourne la dernière clé primaire pour le niveau indiqué
Reload(zonename:str; index:integer)	Recharge les champs de la zone zonename avec les données de la clé primaire index
Resave	Écrase les données de l'enregistrement rechargé (avec Reload) après modifications
abortzone	Invalide les données de la zone, comme si un check avait échoué et retourné ignorerec

Fonction	Description
abortthisline	Non implémenté
abortwholefile	Interrompt le traitement du fichier entier
debug(text)	Écrit sur la console lorsque l'utilisateur a choisi de voir les informations de débogage
log*(text)	Écrit sur la console
Fonctions liées aux lookups	
lookup_count(LookupName:str):Integer	Retourne le nombre de clés dans le lookup
lookup_clear(LookupName:str)	Vide la table de lookup
lookup_HasKey(Lookupname,Key:str):Boolean	Retourne **vrai** si la **Key** existe dans le lookup
lookup_GetKey(Lookupname:str; indice:Integer):str	Retourne la indice-ième **Key** du lookup
lookup_Get(Lookupname,Key:str):Variant	Retourne la valeur liée à la **Key** dans le lookup
lookup_Set(Lookupname,Key:str; Value:Variant	Place **Value** pour la **Key**
lookup_dump(LookupName:str)	Liste le contenu du lookup dans le journal

Revision #6
Created 2025-03-11 08:29:09 UTC by Admin
Updated 2026-03-17 08:37:06 UTC by Admin