

Manipulations sur les fichiers et les répertoires

Via les instructions natives

Dir()	Renvoie le nom d'un fichier ou d'un répertoire, ou encore de tous les fichiers et répertoires existant dans une unité ou un répertoire correspondant au chemin de recherche spécifié.
FileCopy()	Copie un fichier.
FileDateTime()	Renvoie une chaîne de caractères contenant la date et l'heure de création ou de dernière modification d'un fichier.
FileExists()	Renvoie True si un fichier ou un répertoire existe.
FileLen()	Renvoie la longueur d'un fichier (en octets).
GetAttr()	Renvoie le type d'un fichier, volume ou répertoire.
GetPathSeparator()	Renvoie le séparateur de chemins pour le système courant.
Kill()	Supprime un fichier d'un disque.
MkDir()	Crée un nouveau répertoire.
Name()	Renomme un fichier ou répertoire existant.
Rmdir()	Supprime un répertoire existant.
SetAttr()	Définit les attributs d'un fichier particulier.

Via un objet SimpleFileAccess

Méthodes du service com.sun.star.ucb.SimpleFileAccess :

```
oSFA = createUNOService("com.sun.star.ucb.SimpleFileAccess")
```

copy	Copie un fichier.
createFolder	Crée un nouveau répertoire.
exists	Vérifie si un fichier ou un répertoire existe.
getContent	Retourne le type de contenu d'un fichier.
getDateModified	Retourne la date de dernière modification d'un fichier.
getFolderContents	Retourne le contenu d'un répertoire.
getSize	Retourne la taille d'un fichier.
isFolder	Vérifie si une URL est celle d'un répertoire.
isReadOnly	Vérifie si un fichier est en lecture seule.
kill	Supprime un fichier ou un répertoire. Un répertoire, même non vide, est supprimé.
move	Déplace un fichier.
setInteractionHandler	Déclare un gestionnaire d'interactions pour d'autres opérations.
setReadOnly	Positionne le drapeau de lecture seule d'un fichier (droits nécessaires).

Chemins des fichiers

Pour être multi plate-formes, les chemins des fichiers sont exprimés au format URL :
 file:///support/chemin/vers/fichier.ext

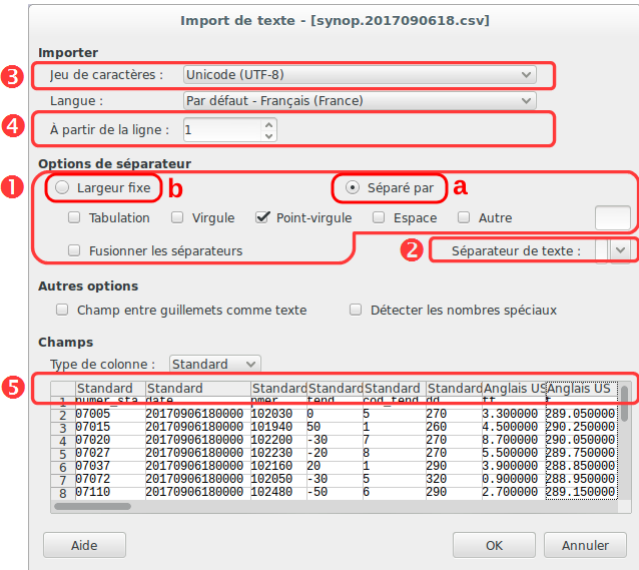
Fonctions de conversion

De natif à URL	NomUrl = ConvertToURL(NomFichierNatif)
De URL à natif	NomOS = ConvertFromURL(NomFichierURL)

Importer des fichiers texte (CSV)

Manuellement

Voici le dialogue d'import affiché par Calc :



Éléments des filtres CSV

Les filtres CSV nécessitent 5 paramètres (pour référence, voir le dialogue ci-dessus).

1. Séparateur de champ

a. Format variable

Code Ascii du séparateur (voir tableau Ascii).	"9"
Si alternatives, les séparer par /	"9/36"
Si plusieurs caractères fusionnés, ajouter /MRG	"36/36/MRG"

b. Format fixe "FIX"

2. Délimiteur de texte

Code Ascii du délimiteur (voir tableau Ascii) "34" (" si aucun)

3. Jeu de caractères

Jeux de caractères fréquents (" possible si UTF-8) :

Windows-1252/WinLatin1	ANSI	ISO-8859-15/EURO	22
ISO-8859-1	12	UTF-8	76

Liste à jour des jeux supportés : service "com.sun.star.document.FilterFactory"

4. Première ligne à traiter

Numéro de cette ligne (commence à 1) "2" ("1" ou "" si 1^{re} ligne)

5. Format des colonnes

a. Variable	Pour chaque colonne, séquence de 4 car. : rang (base 1) / format / (voir tableau Formats)	"1/2/ 2/2/ 3/1/" (" si uniquement valeurs par défaut)
b. Fixe	Pour chaque colonne, séquence de 4 car. pos. 1 ^{er} car (base 0) / format / (voir tableau Formats)	"0/1/ 8/2/ 13/1/"

- Vous pouvez intercaler des espaces pour rendre la séquence plus lisible.
- Vous pouvez ne spécifier que les seules colonnes utiles.

Le filtre d'import CSV

Le filtre à utiliser est constitué de la concaténation de ces 5 paramètres, séparés par des virgules :

① ② ③ ④ ⑤
Filtre = "9,34,76,1,1/2/2/2/3/1"

Selon le contexte, certaines valeurs peuvent être omises (voir détails des paramètres) : Filtre = "59,,76,,,"

Informations complémentaires

Codes Ascii fréquents (décimaux)

9 Tabulation	34 "	36 \$	44 ,	59 ;
32 Espace	35 #	39 '	58 :	

Formats des colonnes

1 Standard (choix automatique Calc)	5	YY/MM/DD
2 Texte	9	(ignorer la colonne)
3 MM/DD/YY	10	Format US (sép. décimal, milliers)
4 DD/MM/YY		

Importer un fichier CSV dans une feuille Calc

Soit un fichier CSV nommé MonFichier.csv. On veut copier le contenu de ce fichier dans la feuille nommée Feuille du classeur Calc courant. L'opération se fait par création d'un lien puis par la suppression de ce lien.

```
Dim CsvURL As String 'l'adresse du fichier .csv source
Dim Filtre As String
Dim Feuil As Object 'la feuille cible dans le classeur

Feuil = ThisComponent.Sheets.getByIndex("Feuille")
CsvURL = ConvertToURL("C:\chemin\vers\MonFichier.csv")
'options de lecture du fichier .csv
Filtre = "9,34,ANSI,1,1/2/ 2/2/ 3/1/ 4/1/"
'importation à travers un lien entre la feuille et la source .csv
Feuil.Link(CsvURL, "", "Text - txt - csv (StarCalc)", Filtre, com.sun.star.sheet.SheetLinkMode.VALUE)
'libération du lien pour rendre le classeur autonome
Feuil.setLinkMode(com.sun.star.sheet.SheetLinkMode.NONE)
```

Le contenu préexistant sur la feuille est écrasé sans avertissement.

Créer un classeur Calc à partir d'un CSV

Soit un fichier CSV nommé MonFichier.csv. On veut créer un nouveau classeur Calc à partir de ce fichier.

```
Dim props1(1) As New com.sun.star.beans.PropertyValue
Dim props2()
Dim CsvURL As String 'l'adresse du fichier .csv source
Dim LeDocURL As String 'l'adresse du fichier .ods cible
Dim oDoc As Object 'le classeur cible

CsvURL = ConvertToURL("C:\chemin\vers\MonFichier.csv")
'options de lecture du fichier .csv
props1(0).Name = "FilterName"
props1(0).Value = "Text - txt - csv (StarCalc)"
props1(1).Name = "FilterOptions"
props1(1).Value = "9,34,ANSI,1,1/2/ 2/2/ 3/1/ 4/1/"
'chargement du fichier source .csv dans la première feuille
oDoc = StarDesktop.loadComponentFromURL(CsvURL, "_blank", 0, props1())
'sauvegarde vers la cible au format .ods
LeDocURL = ConvertToURL("C:\chemin\vers\Monclasseur.ods")
oDoc.storeAsURL(LeDocURL, props2())
```

Le document créé ne comporte qu'une feuille, nommée d'après le fichier CSV source.

- Dans l'exemple, le document créé est affiché. Pour le masquer :
 - ajoutez l'option Hidden (valeur True) dans props1()
 - ajoutez oDoc.close(True) en fin de traitement.

Un document existant de même nom est écrasé sans avertissement.

Gestion des contenus – Instructions natives

Mise en œuvre

- Obtenir un n° d'identifiant interne sur le fichier (FreeFile),
- Ouvrir le fichier (Open),
- Écrire le fichier (Print, Put ou Write) ou le lire (Get, Line Input# ou Input#),
- Fermer le fichier (Close)

Accès aux contenus des fichiers par leur handle (identifiant interne).

Close	Ferme un fichier précédemment ouvert par Open.
Eof()	Détermine si le pointeur de fichier a atteint la fin d'un fichier.
FileAttr()	Renvoie le mode d'accès ou le handle d'un fichier ouvert par Open.
FreeFile	Renvoie un n° de handle disponible avant l'ouverture d'un fichier.
Get	Lit un enregistrement dans un fichier et l'insère dans une variable.
Input	Lit les données d'un fichier séquentiel ouvert.

Line Input	Lit une ligne d'un fichier.
Loc()	Renvoie la position actuelle dans un fichier ouvert.
LoF()	Renvoie la taille d'un fichier ouvert, en octets.
Open	Ouvre un canal de données.
Print	Écrit des données dans un fichier séquentiel (ligne non délimitée).
Put	Écrit un enregistrement dans un fichier.
Reset	Ferme tous les fichiers ouverts et force l'écriture sur le disque dur du contenu de toutes les mémoires tampon des fichiers.
Seek()	Renvoie la position de la prochaine écriture ou lecture dans un fichier ouvert avec l'instruction Open For Random.
Write	Écrit des données dans un fichier séquentiel (ligne, délimitée).

☞ **Print ou Write ?**
Print enregistre le texte tel quel. Write inscrit le texte avec des délimiteurs qui caractérisent le type de l'information (texte : ", date et booléen : #). Ces délimiteurs sont ignorés lors de leur lecture ultérieure par Input

Modes d'ouverture (fichiers texte)

Écriture séquentielle Open For Output **Lecture** séquentielle Open For Input

☞ **Autres types d'accès (accès direct ou binaire), utilisez les API « flux ».**

Exemple de lecture séquentielle (fichier texte)

Lire un fichier repéré par son adresse FichierURL.

```
Dim Handle As Integer, Ligne As String
Handle = FreeFile
Open FichierURL For Input As #Handle
'lecture ligne à ligne
Do While Not Eof(Handle)
'lecture de toute la ligne courante
Line Input #Handle, Ligne
Loop
Close #Handle
```

Exemple d'écriture séquentielle (fichier texte)

Écrire un fichier repéré par son adresse FichierURL.

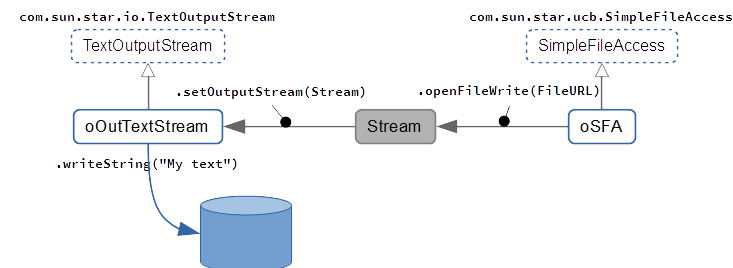
```
Dim Handle As Integer
Handle = FreeFile 'obtenir un numéro de fichier ouvert
Open FichierURL For Output As #Handle
'écriture ligne à ligne
Print #Handle, "Du texte."
Print #Handle, "Encore..."
Print #Handle, "Pour finir."
Close #Handle
```

Gestion des contenus – API « flux »

Fait appel aux services SimpleFileAccess et Stream de l'API LibreOffice.

Principe

Exemple : écriture d'un fichier texte (voir le codage dans l'exemple plus loin).



Mise en œuvre

1. **Créer l'objet** d'accès aux fichiers,
oSFA = createUNOService ("com.sun.star.ub.SimpleFileAccess")
2. **Connecter** le flux correspondant au traitement (selon type d'accès),
3. **Écrire** le fichier ou le **lire** (selon type de fichier),
4. Si écriture, **purger** le flux (.flush),
5. **Fermer** le fichier (.closeXxx).

Accès aux contenus des fichiers

Service SimpleFileAccess (SFA)

openFileRead	Ouvre un fichier en lecture.
openFileWrite	Ouvre un fichier en écriture.
openFileReadWrite	Ouvre un fichier en lecture et écriture.

Services Flux (InputStream, OutputStream et Stream)

Ce sont les services « actifs ».

Correspondances entre les méthodes du service SFA et les flux :

Méthodes SFA	services Flux
openFileRead	com.sun.star.io.InputStream
	com.sun.star.io.TextInputStream
openFileWrite	com.sun.star.io.OutputStream
	com.sun.star.io.TextOutputStream
openFileReadWrite	com.sun.star.io.Stream

Service Stream

getInputStream	Retourne la partie InputStream du flux lecture/écriture. ☞ La fermeture de ce flux entraîne celle du flux OutputStream.
getOutputStream	Retourne la partie OutputStream du flux lecture/écriture. ☞ La fermeture de ce flux entraîne celle du flux InputStream.

Service InputStream

readBytes	Lit le nombre spécifié d'octets.
readSomeBytes	Lit le nombre d'octets disponibles, avec le maximum spécifié.
skipBytes	Saute le nombre spécifié d'octets (valeur positive).
available	Indique le nombre d'octets qui peuvent être lus ou sautés sans blocage.
closeInput	Ferme le flux.

Service TextInputStream

Hérite de InputStream.

readLine	Lit le texte jusqu'à rencontrer un saut de ligne (CR, LF, ou CRLF) ou bien EOF et renvoie la chaîne correspondante (sans CR ou LF). Les caractères lus sont convertis selon l'encodage spécifié par setEncoding. Si EOF a été atteint avant l'exécution de la méthode, retourne une chaîne vide.
readString	Lit le texte jusqu'à rencontrer un des délimiteurs ou EOF. Retourne la chaîne correspondante. ☞ CRLF n'est pas le délimiteur par défaut ! Si aucun délimiteur n'est précisé ou aucun trouvé, le flux est donc lu jusqu'à EOF. Les caractères lus sont convertis selon l'encodage spécifié par setEncoding. Si EOF a été atteint avant l'exécution de la méthode, retourne une chaîne vide.
isEOF	Retourne le statut de EOF. ☞ Ce statut ne peut pas être détecté en tentant de lire une chaîne vide parce cette réponse peut être valide pour readLine() (lorsque la ligne est vide) et readString() (lorsque deux délimiteurs se suivent).
setEncoding	Positionne l'encodage des caractères (défaut UTF-8). Les noms utilisés sont ceux qui se trouvent sur le document : http://www.iana.org/assignments/character-sets (col. Name). Les jeux de caractères supportés dépendent de l'implémentation.

Service OutputStream

writeBytes	Écrit toute une séquence dans le flux (appel bloquant).
flush	Vide les tampons du flux.
closeOutput	Appelé pour signaler que toutes les données ont été écrites.

Service TextOutputStream

Hérite de OutputStream.

writeString	Écrit une chaîne dans le flux en utilisant l'encodage défini par setEncoding. Les sauts de ligne et délimiteurs éventuellement nécessaires doivent être ajoutés manuellement à la chaîne.
setEncoding	Positionne l'encodage des caractères (défaut UTF-8). Les noms utilisés sont ceux qui se trouvent sur le document : http://www.iana.org/assignments/character-sets (col. Name). Les jeux de caractères supportés dépendent de l'implémentation.

Exemple : Lire un fichier texte

```
Dim oSFA As Object, oInText As Object
Dim FichierURL As String, Ligne As String

oSFA = createUNOService("com.sun.star.ub.SimpleFileAccess")
FichierURL = ConvertToURL("C:\chemin\vers\MonFichier.txt")
oInText = createUNOService("com.sun.star.io.TextInputStream")
oInText.setInputStream(oSFA.openFileRead(FichierURL))
Ligne = oInText.readLine()
oInText.closeInput()
```

Exemple : Écrire un fichier texte

```
Dim oSFA As Object, oOutText As Object
Dim FichierURL As String

oSFA = createUNOService("com.sun.star.ub.SimpleFileAccess")
FichierURL = ConvertToURL("C:\chemin\vers\MonFichier.txt")
oOutText = createUNOService("com.sun.star.io.TextOutputStream")
oOutText.setOutputStream(oSFA.openFileWrite(FichierURL))
'écriture (les délimiteurs de ligne doivent être spécifiés)
'[ici CRLF (Windows)]
oOutText.WriteString("Hello World" & Chr(13) & Chr(10))
oOutText.WriteString("Ligne 2" & Chr(13) & Chr(10))
'vider les tampons et fermer
oOutText.flush
oOutText.closeOutput()
```

Crédits

Auteur : Jean-François Nifenecker – jean-francois.nifenecker@laposte.net

Nous sommes comme des nains assis sur des épaules de géants. Si nous voyons plus de choses et plus lointaines qu'eux, ce n'est pas à cause de la perspicacité de notre vue, ni de notre grandeur, c'est parce que nous sommes élevés par eux. (Bernard de Chartres [attr.])

Historique

Version	Date	Commentaires
1.01	01/10/2017	Première version
1.14	02/12/2018	Corrections mineures

Licence

Cet aide-mémoire est placé sous licence

Creative Commons BY-SA v3 (fr).

Informations

<https://creativecommons.org/licenses/by-sa/3.0/fr/>

