



# API OpenOffice.org for JAVA

Aide à la programmation / Yann LERJEN 2006



## Table des matières

<a href="#">1.Introduction</a>	4
<a href="#">Corrections et améliorations</a>	4
<a href="#">Pré requis</a>	4
<a href="#">2.Le module OpenOffice</a>	5
<a href="#">Lancement de xDesktop</a>	5
<a href="#">Fermeture de xDesktop</a>	6
<a href="#">3.Action sur un document OOo Writer</a>	7
<a href="#">Charger un modèle existant</a>	7
<a href="#">Ouvrir un document existant</a>	7
<a href="#">Sauvegarder le document</a>	8
<a href="#">Exporter le document en PDF</a>	9
<a href="#">Impression</a>	10
<a href="#">Fermer le document</a>	11
<a href="#">4.Action sur le contenu du document</a>	12
<a href="#">Rechercher une chaîne de caractère</a>	12
<a href="#">Insérer un fichier existant dans notre document</a>	12
<a href="#">Modifier les marges de la page</a>	12
<a href="#">Accès aux champs de propriétés</a>	13
<a href="#">Modifier l'arrière-plan du document</a>	13
<a href="#">5.Curseur et texte</a>	15
<a href="#">Caractères spéciaux</a>	15
<a href="#">Création d'un curseur dans un bookmark</a>	15
<a href="#">Création d'un curseur au début du texte</a>	15
<a href="#">Récupération du curseur visible</a>	15
<a href="#">Déplacement d'un curseur</a>	15
<a href="#">Ajout de texte dans un document</a>	16
<a href="#">Formatage du texte (police, paragraphe,...)</a>	16
<a href="#">Insertion de saut de paragraphe</a>	16
<a href="#">Insertion de saut de page</a>	17
<a href="#">Insertion d'une image au curseur</a>	17
<a href="#">6.Actions sur les Bookmarks</a>	19
<a href="#">Insertion de texte dans un bookmark existant</a>	19
<a href="#">7.Actions sur les champs de texte</a>	20
<a href="#">Création de champ</a>	20
<a href="#">Suppression de champ</a>	20
<a href="#">Cacher un champ</a>	21
<a href="#">8.Publipostage (Mailmerge)</a>	22
<a href="#">Déclaration des variables communes</a>	22
<a href="#">Fonction Main() de mon .class</a>	22
<a href="#">Fonction MailMerger()</a>	23
<a href="#">Fonction setupConnection()</a>	23
<a href="#">Fonction OpenWriter()</a>	25
<a href="#">Fonction doMerge()</a>	25

<a href="#">Fonction saveAsTemplate()</a> .....	28
<a href="#">9.Liens utiles</a> .....	29

## 1. Introduction

Ayant dû travailler durant plusieurs mois sur un développement pour OpenOffice en Java, j'ai acquis certaines compétences sur les différentes fonctionnalités de l'api OOO. Ce document a été créé dans le seul but de faciliter la programmation en Java pour OpenOffice.org, il n'est donc pas un cours de programmation et ne doit pas être interprété comme tel. Il n'est pas non-plus un exemple à suivre pour tous développements avec l'api OOO, mais est simplement une aide contenant des exemples sur les différentes manipulations possibles avec JAVA.

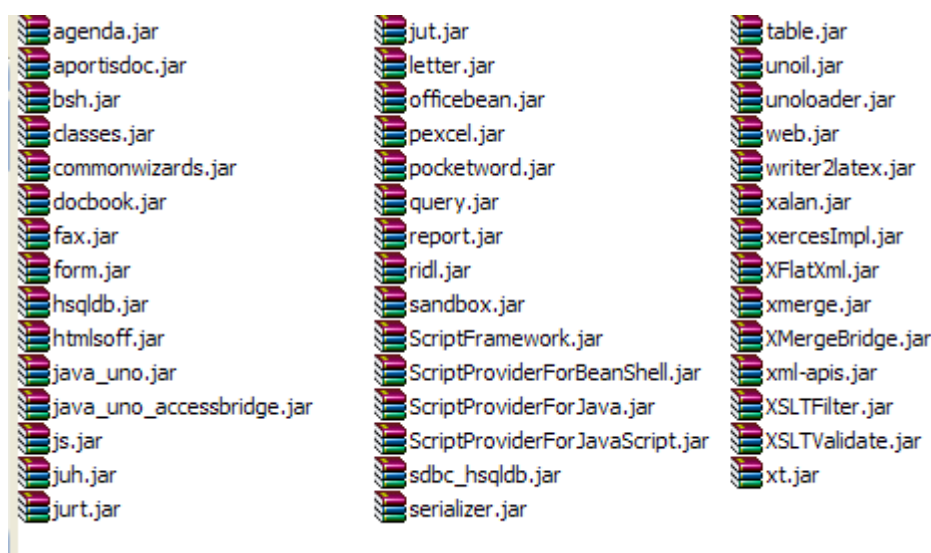
Les morceaux de codes ci-dessous sont décrits de manière à ce qu'une personne ayant des connaissances en Java puisse les comprendre.

### Corrections et améliorations

Afin que ce document puisse être utilisé et compris par tous, il faudra certainement y faire quelques modifications. La meilleure façon d'y contribuer serait de relever des critiques constructives lors de sa lecture. J'accepte volontiers toutes propositions d'améliorations ou de corrections devant être implémentées à ce document.

### Pré requis

- Connaissance de programmation en Java
- La suite OpenOffice 2.0 ou supérieur installée (Je n'ai pas testé sur des versions antérieures, il se peut donc que certaines choses ne fonctionnent pas)
- SDK for OpenOffice installé et \*.jar suivant disponible:



## 2. Le module OpenOffice

Afin d'accéder à notre document OpenOffice depuis l'api, nous devons utiliser la méthode « getDesktop » qui initialise l'UNO (Universal Network Object) puis appelle le « Service Manager » et les objets du bureau OOo. Ceux-ci nous permettront de « communiquer » avec OpenOffice et donc avec les documents que nous voulons traiter.

### Lancement de xDesktop

```
public static com.sun.star.frame.XDesktop getDesktop()
{
    com.sun.star.frame.XDesktop xDesktop = null;
    com.sun.star.lang.XMultiComponentFactory xMCF = null;

    try
    {
        com.sun.star.uno.XComponentContext xContext = null;

        // APPELLE remote office component context POUR L'OUVERTURE D'OOo
        xContext = com.sun.star.comp.helper.Bootstrap.bootstrap();

        // APPELLE Remote office service manager POUR LE MANAGEMENT D'OOo
        xMCF = xContext.getServiceManager();
        if( xMCF != null )
        {
            System.out.println("Connected to a running office ...");
            Object oDesktop = xMCF.createInstanceWithContext(
                "com.sun.star.frame.Desktop", xContext);
            xDesktop = (com.sun.star.frame.XDesktop)
                UnoRuntime.queryInterface(
                    com.sun.star.frame.XDesktop.class, oDesktop);
        }
        else
        {
            System.out.println( "Can't create a desktop. No connection,
                no remote office service manager available!" );
        }
    }
}
```

```
}  
catch ( Exception e)  
{  
    e.printStackTrace(System.err);  
    System.exit(1);  
}  
return xDesktop;  
}
```

## Fermeture de xDesktop

```
xDesktop.terminate();
```

## 3.Action sur un document OOO Writer

Pour toutes les actions qui sont exécutées sur le document, il nous faut d'abord avoir une connexion créée avec l'UNO, c'est à dire le xDesktop. Cette partie n'est pas réécrite ici, car elle est déjà détaillée dans le chapitre précédent.

### Charger un modèle existant

```
//LIEN VERS NOTRE FONCTION D'OUVERTURE DE FICHER
XComponent Document = newDocComponentFromTemplate(TemplateURL);

//FONCTION D'OUVERTURE DE FICHER
public static XComponent newDocComponentFromTemplate(String
TemplateUrl) throws java.lang.Exception
{
    // POUR L'OBJET xDesktop, NOUS AVONS BESOIN DU xComponentLoader
    xComponentLoader =
        (XcomponentLoader)UnoRuntime.queryInterface(
            XComponentLoader.class,    xDesktop);
    // DEFINI LES PROPRIETE DE CHARGEMENT DU DOCUMENT
    // L'ATTRIBUT "AsTemplate" COMMUNIQUE AU SERVICE QU'IL DOIT CREER
    UN NOUVEAU
    // DOCUMENT DEPUIS LA PAGE DONNEE
    PropertyValue[] loadProps = new PropertyValue[1];
    loadProps[0] = new PropertyValue();
    loadProps[0].Name = "AsTemplate";
    loadProps[0].Value = new Boolean(true);

    // CHARGEMENT D'UN NOUVEAU DOCUMENT
    return xComponentLoader.loadComponentFromURL(loadUrl, "_blank", 0,
    loadProps);
}
```

### Ouvrir un document existant

```
// POUR L'OBJET xDesktop, NOUS AVONS BESOIN DU xComponentLoader
xComponentLoader = (XcomponentLoader)UnoRuntime.queryInterface(
    XComponentLoader.class, xDesktop);
```

```
// DEFINI LES PROPRIETE DE CHARGEMENT DU DOCUMENT
PropertyValue[] loadProps = new PropertyValue[3];
loadProps[0] = new PropertyValue();
loadProps[0].Name = "AsTemplate";
loadProps[0].Value = new Boolean(false);
loadProps[1] = new PropertyValue();
loadProps[1].Name = "ReadOnly";
loadProps[1].Value = new Boolean(false);
loadProps[2] = new PropertyValue();
loadProps[2].Name = "Hidden";
loadProps[2].Value = new Boolean(true);
try
{
    // CHARGEMENT D'UN NOUVEAU DOCUMENT
    Document = xComponentLoader.loadComponentFromURL(DocumentURL,
        "_blank", 0, loadProps);
}
catch (IOException e)
{
    e.printStackTrace();
}
catch (IllegalArgumentException e)
{
    e.printStackTrace();
}
```

## Sauvegarder le document

```
// Query for XStorable interface of object
XStorable xStorable = (XStorable) UnoRuntime.queryInterface(
    XStorable.class, doc);
// Set up the Overwrite and FilterName properties
PropertyValue[] propertyvalue = new PropertyValue[2];
propertyvalue[0] = new PropertyValue();
propertyvalue[0].Name = "Overwrite";
```

```
propertyvalue[0].Value = new Boolean(true);
//
propertyvalue[1] = new PropertyValue();
propertyvalue[1].Name = "FilterName";
propertyvalue[1].Value = "swriter: StarOffice XML (Writer)";
// Enregistre le document
String StoredFile = "FileName+URL";
xStorable.storeAsURL(StoredFile, propertyvalue);
```

## Exporter le document en PDF

```
public static void export_pdf(XComponent xTemplateComponent, String
    Document_Name)
{
    xDoc = (XTextDocument)
        UnoRuntime.queryInterface(XTextDocument.class,
            xTemplateComponent);

    //CREATION DE L'ELEMENT xStorable QUI PERMET L'EXPORT
    XStorable xStorable = (XStorable)UnoRuntime.queryInterface(
        XStorable.class, xDoc );

    //CREATION DES PARAMETRES D'EXPORT
    PropertyValue[] propertyvalue = new PropertyValue[4];
    //Paramètre l'export de fichier pour pdf
    propertyvalue[0] = new PropertyValue();
    propertyvalue[0].Name = "FilterName";
    propertyvalue[0].Value = "writer_pdf_Export";
    //Paramètre les pages prises en compte
    propertyvalue[1] = new PropertyValue();
    propertyvalue[1].Name = "Pages";
    propertyvalue[1].Value = "All";
    //Si le fichier existe déjà dans le répertoire: Ecraser.
    propertyvalue[2] = new PropertyValue();
    propertyvalue[2].Name = "Overwrite";
    propertyvalue[2].Value = new Boolean(true);
    //Evite la compression de fichier
```

```
propertyvalue[3] = new PropertyValue();
propertyvalue[3].Name = "Quality";
propertyvalue[3].Value = "100";

//ESSAYE L'EXPORT
try
{
    String pdf_url = "file:///c:/temp/" + Document_Name +
        ".pdf";
    xStorable.storeToURL(pdf_url, propertyvalue);
}
catch( IOException exportererror )
{
    exportererror.printStackTrace();
}
}
```

## Impression

```
public static void Print_Document(XComponent Document,String
PagesToPrint)
{
    //appelle l'interface XPrintable du document
    XPrintable xPrintable = (Xprintable)UnoRuntime.queryInterface(
        XPrintable.class, Document);
    //Choix de l'imprimante
    PropertyValue[] printerDesc = new PropertyValue[1];
    printerDesc[0] = new PropertyValue();
    printerDesc[0].Name = "Name";
    printerDesc[0].Value = "Lexmark T640";

    //Affecte les propriétés de l'imprimante à la config d'impression
    xPrintable.setPrinter(printerDesc);

    // Créer un tableau d'options
    PropertyValue[] printOpts = new PropertyValue[3];
    // Choix des pages à imprimer
```

```
printOpts[0].Name = "Pages";
printOpts[0] = new PropertyValue();
printOpts[0].Value = PagesToPrint; //Ex: 3-5,7,8
// Nombre d'exemplaire à imprimer
printOpts[1] = new PropertyValue();
printOpts[1].Name = "CopyCount";
printOpts[1].Value = (short)NbrCopie;
// Attendre la fin de l'impression avant de continuer
printOpts[2] = new PropertyValue();
printOpts[2].Name = "Wait";
printOpts[2].Value = true;
try
{
    xPrintable.print(printOpts); //lance l'impression
}
catch (IllegalArgumentException e)
{
    System.err.println("Erreur lors de l'impression!!!");
}
}
```

## Fermer le document

```
//"document" correspond à notre Xcomponent ouvert ci-dessus dans
"Charger un modèle existant"
document.dispose();
```

## 4. Action sur le contenu du document

### Rechercher une chaîne de caractère

```
com.sun.star.uno.XInterface xSearchInterface = null;
com.sun.star.text.XTextRange xSearchTextRange = null;

XTextDocument TextDocument = (XTextDocument)
    UnoRuntime.queryInterface(XTextDocument.class, xComponent);

xSearchInterface = (com.sun.star.uno.XInterface)FindFirst(
    TextDocument, "texte à rechercher" );
```

### Insérer un fichier existant dans notre document

```
//Avant d'effectuer ce bout de code il faut avoir créé un curseur
"XCursor"

XDocumentInsertable xdi =(XDocumentInsertable)
    UnoRuntime.queryInterface(XDocumentInsertable.class, xCursor);

xdi.insertDocumentFromURL(CorpsURL, loadProps);
```

### Modifier les marges de la page

```
// Récupération des propriétés du textCursor
XPropertySet xTextCursorProps =
(XPropertySet)UnoRuntime.queryInterface(XPropertySet.class, mxDocCursor);

// Page Style name
String pageStyleName=
xTextCursorProps.getPropertyValue("PageStyleName").toString();

// Récupération de l'interface StyleFamiliesSupplier du document
XStyleFamiliesSupplier xSupplier =
(XstyleFamiliesSupplier)UnoRuntime.queryInterface(XStyleFamiliesSupplier
.class, mxDoc);

// On utilise l'interface StyleFamiliesSupplier pour récupérer le style
actuel
XNameAccess xFamilies =
(XNameAccess)UnoRuntime.queryInterface(XNameAccess.class,
xSupplier.getStyleFamilies());

// Acces à la famille du 'PageStyles'
XNameContainer xFamily
=(XNameContainer)UnoRuntime.queryInterface(XNameContainer.class,
```

```
xFamilies.getByName("PageStyles"));

// Insertion du nouveau style cree dans la famille du PageStyles
XStyle xStyle= (Xstyle)UnoRuntime.queryInterface(XStyle.class,
xFamily.getByName(pageStyleName));

// Modification des propriétés du TextCursor
XPropertySet xStyleProps =
(XpropertySet)UnoRuntime.queryInterface(XPropertySet.class, xStyle);
xStyleProps.setPropertyValue("LeftMargin",120);
xStyleProps.setPropertyValue("RightMargin",120);
```

## Accès aux champs de propriétés

```
//Créer l'accès aux propriétés du document
XDocumentInfoSupplier infoSupp =
(XDocumentInfoSupplier)UnoRuntime.queryInterface(XDocumentInfoSupplier.c
lass, xDocument);

//Correspond aux champs éditables spécifiques au User
XDocumentInfo docInfo = infoSupp.getDocumentInfo();

//Correspond aux propriétés éditées spécifiques au document
XPropertySet docinfoProps =
(XpropertySet)UnoRuntime.queryInterface(XPropertySet.class, docInfo);

try
{
    //Rempli le champ commentaire de l'onglet description
    docinfoProps.setPropertyValue("Description",TextToInsert);
    //Renomme et remplit le champ numéro 0 de l'onglet Utilisateur
    docInfo.setUserFieldName((short)0, "Verso");
    docInfo.setUserFieldValue((short)0, "Nom du verso");
}
catch (Exception e)
{
    e.printStackTrace();
}
```

## Modifier l'arrière-plan du document<sup>1</sup>

```
// create a supplier to get the Style family collection
XStyleFamiliesSupplier xSupplier = ( XStyleFamiliesSupplier )
UnoRuntime.queryInterface(XStyleFamiliesSupplier.class, xTextDocument );
```

<sup>1</sup> Snippet tiré de : <http://codesnippets.services.openoffice.org/Writer/Writer.ChangeDocumentBackgroundColor.snip>

```
// get the NameAccess interface from the Style family collection
XNameAccess xNameAccess = xSupplier.getStyleFamilies();

XNameContainer xPageStyleCollection = (XNameContainer)
UnoRuntime.queryInterface(XNameContainer.class, xNameAccess.getByName(
"PageStyles" ));

// create a PropertySet to set the properties for the new Pagestyle
XPropertySet xPropertySet = (XPropertySet)
UnoRuntime.queryInterface(XPropertySet.class,
xPageStyleCollection.getByName("Standard" ));

// setBackgroundColor
xPropertySet.setPropertyValue("BackColor",new Integer( (int)255 ) );
```

## 5. Curseur et texte

### Caractères spéciaux

```
(char)9 // Tabulation
(char)13 // Retour à la ligne
```

### Création d'un curseur dans un bookmark

```
XTextCursor xCursor = xBookmarkRange.getText().createTextCursorByRange(
    xBookmarkRange);
```

### Création d'un curseur au début du texte

```
XTextDocument xTextDocument =
(XTextDocument)UnoRuntime.queryInterface(XTextDocument.class,
xComponent);
//Créé un TextRange au début du document
XTextRange xStart = xTextDocument.getText().getStart();
XTextCursor xCursor = xStart.getText().createTextCursorByRange(xStart);
```

### Récupération du curseur visible

```
//Appelle l'interface xModel du composant
XModel xModel = (XModel)UnoRuntime.queryInterface(XModel.class,
xComponent);
// Le modèle renvoie son controller
XController xController = xModel.getCurrentController();
// le controller nous donne le TextViewCursor
// Appelle l'interface du Viewcursor
XTextViewCursorSupplier xViewCursorSupplier =
    (XTextViewCursorSupplier)UnoRuntime.queryInterface(
        XTextViewCursorSupplier.class, xController);
// Prend la position du curseur
XTextViewCursor xViewCursor = xViewCursorSupplier.getViewCursor();
```

### Déplacement d'un curseur

```
//Le boolean placé en argument et à true pour un déplacement en
sélection et à false pour un déplacement sans sélection.
xCursor.gotoStart(false); //Déplacement en début de document
```

```
xCursor.gotoEnd(false); //Déplacement en fin de document
```

## Ajout de texte dans un document

```
//Voici comment ajouter un texte suivi d'un retour à la ligne  
xCursor.setString("Ce texte a été inséré par mon curseur!" + (char)13);
```

## Formatage du texte (police, paragraphe,...)

```
com.sun.star.beans.XPropertySet xTCPS =  
    (com.sun.star.beans.XPropertySet)UnoRuntime.queryInterface(  
        com.sun.star.beans.XPropertySet.class, xCursor);  
//Formatage du texte  
    // Texte normal  
xTCPS.setPropertyValue("CharWeight", new  
    Float(com.sun.star.awt.FontWeight.NORMAL));  
    // Texte en gras  
xTCPS.setPropertyValue("CharWeight", new  
    Float(com.sun.star.awt.FontWeight.BOLD));  
//Formatage du paragraphe  
    //Paragraphe Centré:  
xTCPS.setPropertyValue("ParaAdjust", ParagraphAdjust.CENTER);  
    //Style de paragraphe par défaut:  
xTCPS.setPropertyValue("ParaAdjust", ParagraphAdjust.getDefault());  
    //Taille du texte (ici 10px)  
xTCPS.setPropertyValue("CharHeight", new Float(10));  
    //Style de texte  
xTCPS.setPropertyValue("CharFontName", new String("Arial"));
```

## Insertion de saut de paragraphe

```
protected void InsertParagraphBreak(XText xText, XTextCursor  
xTextCursor) throws Exception  
{  
    // Initialise le curseur  
    XPropertySet xCursorProps = (XPropertySet)  
        UnoRuntime.queryInterface(XPropertySet.class, xTextCursor);  
  
    // Insère un saut de paragraphe  
    xText.insertControlCharacter(xTextCursor,  
        ControlCharacter.PARAGRAPH_BREAK, false);  
}
```

## Insertion de saut de page

```
protected void InsertPageBreak(XText xText, XTextCursor xTextCursor)
throws Exception
{
    // Initialise le curseur
    XPropertySet xCursorProps = (XPropertySet)
        UnoRuntime.queryInterface(XPropertySet.class, xTextCursor);

    // Insère un saut de page
    xCursorProps.setPropertyValue("BreakType", BreakType.PAGE_AFTER);
}
```

## Insertion d'une image au curseur

```
try
{
    XTextDocument xTextDocument = (XTextDocument)UnoRuntime.queryInterface(
        XTextDocument.class, xComponent);

    //Créé un TextRange au début du document
    XTextRange xStart = xTextDocument.getText().getStart();
    XTextCursor xCursor = xStart.getText().createTextCursorByRange(xStart);

    XText xText = xCursor.getText();
    XTextContent xImage = null;

    XMultiServiceFactory xMSF = (XMultiServiceFactory) UnoRuntime.
        queryInterface(XMultiServiceFactory.class, xComponent);
    xImage = (XTextContent) UnoRuntime.queryInterface(
        XtextContent.class, xMSF.createInstance(
            "com.sun.star.text.TextGraphicObject"));

    //Propriété d'insertion
    XPropertySet xProps = (XPropertySet) UnoRuntime.queryInterface(
        XPropertySet.class, xImage);
    //Détermine l'ancrage à la page, au texte ou au caractère
    xProps.setPropertyValue(
        "AnchorType", com.sun.star.text.TextContentAnchorType.AT_PAGE);
}
```

```
//Détermine l'URL concernant l'image
xProps.setPropertyValue("GraphicURL", "file:///C:/image.gif");
//Détermine si l'objet sera imprimé aussi lors de l'impression du doc.
xProps.setPropertyValue("Print", true);
//Largeur de l'image
xProps.setPropertyValue("Width", (int)17000 );
//Hauteur de l'image
xProps.setPropertyValue("Height", (int)11000);
//Alignement vertical
xProps.setPropertyValue("VertOrient", VertOrientation.CENTER);
//Adaptation du texte par rapport à l'image
xProps.setPropertyValue(
    "TextWrap", com.sun.star.text.WrapTextMode.THROUGHT);
//En arrière plan
xProps.setPropertyValue("Opaque", false);

// Insère le graphique là où se trouve le curseur
xText.insertTextContent(xCursor, xImage, false);
}
catch (Exception e)
{
System.out.println("Failed to insert Graphic");
e.printStackTrace();
}
```

## 6.Actions sur les Bookmarks

### Insertion de texte dans un bookmark existant

```
try
{
    // APPELLE L'INTERFACE XBookmarksSupplier
    // POUR L'ACCES AUX COMPOSANTS DU DOCUMENT
    XBookmarksSupplier xBookmarksSupplier =
        (XbookmarksSupplier)UnoRuntime.queryInterface(
            XBookmarksSupplier.class, xTemplateComponent);
    // ACCES AUX BOOKMARKS DU DOCUMENT
    xNamedBookmarks = xBookmarksSupplier.getBookmarks();
    // RECHERCHE DU BOOKMARK PAR SON CHAMPS
    Object bookmark = xNamedBookmarks.getByName(BookmarkName);
    // LA COMMANDE getAnchor() NOUS PERMET DE CONNAITRE LA POSITION DU
    // BOOKMARK
    xBookmarkContent = (XtextContent)UnoRuntime.queryInterface(
        XtextContent.class, bookmark);
    xBookmarkRange = xBookmarkContent.getAnchor();
    // REMPLACE LE BOOKMARK TROUVE PAR UNE STRING OU LE CONTENU D'UNE
    // VARIABLE
    xBookmarkRange.setString(BookmarkContent);
}
catch (Exception e)
{
    System.err.println("Le champ " + BookmarkName +
        " a créé l'erreur suivante:");
    e.printStackTrace();
}
```

## 7.Actions sur les champs de texte

### Création de champ<sup>2</sup>

```
XDependentTextField myField = (XDependentTextField)
    UnoRuntime.queryInterface(XDependentTextField.class,
        xTextDocumentFactory.createInstance("com.sun.star.text.TextField
            .User"));
XPropertySet xPropertySet =
    (XpropertySet)UnoRuntime.queryInterface(XPropertySet.class,
        xTextDocumentFactory.createInstance("com.sun.star.text.
            FieldMaster.User"));
xPropertySet.setPropertyValue("Name", "myField");
xPropertySet.setPropertyValue("Value", new Integer(1));
myField.attachTextFieldMaster(xPropertySet);
```

### Suppression de champ<sup>3</sup>

```
XTextFieldsSupplier xTFS = (XTextFieldsSupplier)
    UnoRuntime.queryInterface(XTextFieldsSupplier.class, yourTextDocument);
if (xTFS != null)
{XNameAccess xEA = xTFS.getTextFieldMasters();
    if (xEA.hasByName(yourFieldName))
    {
        try
        {
            Object xTFL = xEA.getByName(yourFieldName);
            XComponent xC =
                (Xcomponent)UnoRuntime.queryInterface(XComponent.class,
                    xTFL);
            xC.dispose();
        }
        catch (java.lang.Exception ex)
        {ex.printStackTrace();}
    }
}
```

<sup>2</sup> Snippet tiré de : <http://codesnippets.services.openoffice.org/Writer/Writer.RemovingUserTextField.snip>

<sup>3</sup> Snippet tiré de : <http://codesnippets.services.openoffice.org/Writer/Writer.RemovingUserTextField.snip>

## Cacher un champ<sup>4</sup>

```
public void hideField(XComponent component, String fieldName )
{
    // Get Access to the TextFields in the document
    XTextFieldsSupplier xTextFieldsSupplier =
        (XtextFieldsSupplier)UnoRuntime.queryInterface(
            XTextFieldsSupplier.class, component);
    XEnumerationAccess xEnumeratedFields =
        xTextFieldsSupplier.getTextFields();
    XEnumeration enumeration = xEnumeratedFields.createEnumeration();

    boolean changed = false;
    // Loop through the TextFields and search for the right field
    while (enumeration.hasMoreElements() && !changed)
    {
        Object field = enumeration.nextElement();
        XDependentTextField dependentTextField =
            (XdependentTextField)UnoRuntime.queryInterface(
                XDependentTextField.class, field);
        XPropertySet propertySet = dependentTextField.getTextFieldMaster();
        String name = (String) propertySet.getPropertyValue("Name");
        if (name.equals(fieldName))
        {
            XPropertySet fieldProperties =
                (XpropertySet)UnoRuntime.queryInterface(
                    XPropertySet.class, field);
            fieldProperties.setPropertyValue("IsVisible", Boolean.FALSE);
            changed = true;
        }
    }
}
```

---

4 Snippet tiré de : <http://codesnippets.services.openoffice.org/Writer/Writer.HidingUserfield.snip>

## 8. Publipostage (Mailmerge)

Afin de mieux comprendre le publipostage à travers ce code qui est assez long, je l'ai divisé en plusieurs fonctions. Afin que le code fonctionne, toutes les parties de ce chapitre doivent être intégrées dans le programme.

Ce publipostage va chercher les informations dans un fichier à plat ayant l'extension **.ods** (donc calc) et le charge dans un modèle Writer contenant les champs de donnée. Il est néanmoins possible de faire les mêmes manipulation avec un **.csv** ou autres source de données.

### Déclaration des variables communes

```
// L'URL du dossier avec les tables sources de base
// flat:file: correspond à un fichier à plat
private String mDataSourceDir = "sdbc:flat:file:///C:\\temp\\";
// Le nom de la source de données qui sera utilisée pour le MailMerge
// ATTENTION, CE NOM DOIT-ÊTRE ECRIT SANS EXTENSION!!!
private static String mDataSourceName = "MA_BASE_ODS";
// Le nom de la table dans la source de donnée qui sera utilisé pour le
MailMerge
private static String mTableName = "Feuille1";
// L'URL où le modèle pour le MailMerge est stocké
private static String mFileURL = "file:///C:/temp/mmresult/Model.odt";
// L'URL où seront sauvés les fichiers de sortie
private static String mFileOutputURL = "file:///C:/temp/mmresult";

// Ces objets sont utilisés pour créer ou accéder aux API OpenOffice
private XMultiComponentFactory mxMCF;
private XMultiServiceFactory mxMSF;
private XComponentContext mxComponentContext;
private XComponentLoader mxComponentLoader;
```

### Fonction Main() de mon .class

```
public static void main(String args[])
{
    XComponent Document = null;
    MailMerger mm = new MailMerger(
        mFileURL, mDataSourceName, mTableName);
    mm.setupConnection();
}
```

```
mm.doMerge();  
}
```

## Fonction MailMerger()

```
public MailMerger(String file, String db, String table)  
{  
    mFileURL = file;  
    mDataSourceName = db;  
    mTableName = table;  
}
```

## Fonction setupConnection()

```
public void setupConnection()  
{  
    try  
    {  
        com.sun.star.uno.XComponentContext xContext = null;  
        com.sun.star.frame.XDesktop xDesktop = null;  
  
        //APPELLE remote office component context POUR L'OUVERTURE D'OOo  
        xContext = com.sun.star.comp.helper.Bootstrap.bootstrap();  
  
        //APPELLE Remote office service manager POUR LE MANAGEMENT D'OOo  
        mxMCF = xContext.getServiceManager();  
        if( mxMCF != null )  
        {  
            System.out.println("Connected to a running office ...");  
            Object oDesktop = mxMCF.createInstanceWithContext(  
                "com.sun.star.frame.Desktop", xContext);  
            xDesktop = (com.sun.star.frame.XDesktop)  
                UnoRuntime.queryInterface(  
                    com.sun.star.frame.XDesktop.class, oDesktop);  
        }  
    }  
}
```

```
else
{
    System.out.println(
        "Can't create a desktop. No connection, no remote
        office servicemanager available!" );
}
//Query for the XPropertySet interface.
XPropertySet xpropertysetMultiComponentFactory = (XPropertySet)
    UnoRuntime.queryInterface(XPropertySet.class, mxMCF);
//Get the default context from the office server.
Object objectDefaultContext =
    xpropertysetMultiComponentFactory.getPropertyValue(
        "DefaultContext");
// Query for the interface XComponentContext.
mxComponentContext = (XComponentContext)UnoRuntime.queryInterface(
    XComponentContext.class, objectDefaultContext);
mxComponentLoader = (XComponentLoader)UnoRuntime.queryInterface(
    XComponentLoader.class,
mxMCF.createInstanceWithContext("com.sun.star.frame.Desktop",
    mxComponentContext));
//Query for an XMultiServiceFactory instance from the global
//service manager
if (mxMSF == null)
{
    mxMSF = (XmultiServiceFactory)UnoRuntime.queryInterface(
        XmultiServiceFactory.class,
        mxComponentContext.getServiceManager());
}
}
catch (Exception exception)
{
    System.err.println(exception);
}
}
```

## Fonction OpenWriter()

```
public XTextDocument openWriter()
{
XTextDocument oDoc = null;
XComponent aDoc = null;
try
{
    PropertyValue[] loadProps = new PropertyValue[0];
    aDoc = mxComponentLoader.loadComponentFromURL(
        "private:factory/swriter", "_blank", 0, loadProps);
    oDoc = (XTextDocument) UnoRuntime.queryInterface(
        XTextDocument.class, aDoc);
}
catch (Exception e)
{
    System.err.println("Error opening new document" + e);
}
return oDoc;
}
```

## Fonction doMerge()

```
public void doMerge()
{
Object mmservice = null;
try
{
    // Create an instance of the MailMerge service
    mmservice = mxMCF.createInstanceWithContext(
        "com.sun.star.text.MailMerge", mxComponentContext);
}
catch (Exception e)
{
    System.err.println("Error getting MailMerge service: " + e);
return;
}
```

```
}

// Get the XPropertySet interface of the mmSERVICE object
XPropertySet oObjProps = (XPropertySet)UnoRuntime.queryInterface(
    XPropertySet.class, mmSERVICE);

try
{
    /*=====
    = SEULE 1 DES FONCTIONS SUIVANTES DOIT-ETRE ECRITE DANS LE CODE =
    =====*/

    //CI DESSOUS: IMPRESSION DIRECT SUR PRINTER...
    oObjProps.setPropertyValue("DataSourceName", mDataSourceName);
    oObjProps.setPropertyValue("Command", mTableName);
    oObjProps.setPropertyValue("CommandType", new
        Integer(com.sun.star.sdb.CommandType.TABLE));
    oObjProps.setPropertyValue("OutputType", new
        Short(com.sun.star.text.MailMergeType.PRINTER));
    oObjProps.setPropertyValue("DocumentURL", mFileURL);

    //CI DESSOUS: IMPRESSION DANS UN FICHER COMMUN AVEC UN NOM DONNÉ...
    oObjProps.setPropertyValue("DataSourceName", mDataSourceName);
    oObjProps.setPropertyValue("Command", mTableName);
    oObjProps.setPropertyValue("CommandType", new
        Integer(com.sun.star.sdb.CommandType.TABLE));
    oObjProps.setPropertyValue("OutputType", new
        Short(com.sun.star.text.MailMergeType.FILE));
    oObjProps.setPropertyValue("OutputURL", mFileOutputURL);
    oObjProps.setPropertyValue("DocumentURL", mFileURL);
    oObjProps.setPropertyValue("FileNamePrefix", "Mon_Fichier");
    oObjProps.setPropertyValue("SaveAsSingleFile", new Boolean(true));
}
```

```
//CI DESSOUS: IMPRESSION DANS DES FICHIERS SÉPARÉS AYANT LE NOM D'UNE
COLONNE SUIVI D'UNE INDENTATION AUTOMATIQUE.
oObjProps.setPropertyValue("DataSourceName", mDataSourceName);
oObjProps.setPropertyValue("Command", mTableName);
oObjProps.setPropertyValue("CommandType", new
    Integer(com.sun.star.sdb.CommandType.TABLE));
oObjProps.setPropertyValue("OutputType", new
    Short(com.sun.star.text.MailMergeType.FILE));
oObjProps.setPropertyValue("OutputURL", mFileOutputURL);
oObjProps.setPropertyValue("DocumentURL", mFileURL);
oObjProps.setPropertyValue("FileNameFromColumn", new Boolean(true));
oObjProps.setPropertyValue("FileNamePrefix", "Nom_de_la_colonne_");
oObjProps.setPropertyValue("SaveAsSingleFile", new Boolean(false));

}
catch (Exception e)
{
    System.err.println("Error setting MailMerge properties: " + e);
    return;
}
// Appelle l'interface XJob et execute le publipostage
XJob job = (XJob) UnoRuntime.queryInterface(XJob.class, mmService);

try
{
    job.execute(new NamedValue[0]);
    System.out.println("----- <OK> -----");
}
catch (com.sun.star.lang.IllegalArgumentException iae)
{
    System.err.println("Caught IllegalArgumentException: " + iae);
}
catch (com.sun.star.uno.Exception e)
{
    System.err.println("Caught UNO Exception: " + e);
}
```

```
}  
}
```

## Fonction saveAsTemplate()

```
public void saveAsTemplate(XTextDocument doc)  
{  
  try  
  {  
    // Query for XStorable interface of object  
    XStorable xStorable = (XStorable) UnoRuntime.queryInterface(  
      XStorable.class, doc);  
    // Set up the Overwrite and FilterName properties  
    PropertyValue[] propertyvalue = new PropertyValue[2];  
    propertyvalue[0] = new PropertyValue();  
    propertyvalue[0].Name = "Overwrite";  
    propertyvalue[0].Value = new Boolean(true);  
    propertyvalue[1] = new PropertyValue();  
    propertyvalue[1].Name = "FilterName";  
    propertyvalue[1].Value = "swriter: StarOffice XML (Writer)";  
    // Store the document  
    xStorable.storeAsURL(mFileURL, propertyvalue);  
  }  
  catch (Exception e)  
  {  
    System.err.println("Error while saving: " + e);  
  }  
}
```

## ***9.Liens utiles***

---

Le site officiel pour le téléchargement et les informations d'OpenOffice.

<http://www.openoffice.org/>

Le guide du développeur, bible indispensable pour le développement d'application avec les API java d'OpenOffice.

<http://api.openoffice.org/docs/DevelopersGuide/DevelopersGuide.xhtml>

Le site officiel des API d'OpenOffice.org

<http://api.openoffice.org/>

Le forum francophone pour l'utilisation et l'aide d'OpenOffice.org et de ses API.

<http://www.forum-openoffice.org/forum/>

Le forum anglophone pour l'utilisation et l'aide d'OpenOffice.org et de ses API.

<http://www.ooforum.org/>