Maîtriser le menu contextuel dans un OfficeBean

Diffusé Par
Le Projet Documentation d'OpenOffice.org

Version 1.0

Aurélie Schröder –Université de Genève
Division informatique
AurelieSch@netcourrier.com

Table des matières

1 Introduction

Le but de ce document est d'expliquer comment contrôler le menu contextuel sur une application utilisant le traitement de texte d'OpenOffice.org. Ce document sera traité avec la technologie JAVA mais il est tout à fait envisageable de traduire ce code dans un autre langage (C++ , delphi ou Visual Basic)

Pour atteindre notre but, nous tenterons de réunir deux exemples du SDK : SimpleBean et ContextMenuInterceptor. SimpleBean permet d'apprendre à utiliser les beans de OfficeBean et ContextMenuInterceptor permet de modifier le menu contextuel dans OpenOffice.

Ce document explique quelles sont les modifications à apporter à chacun des exemples et pourquoi.

Cette recherche a été faite dans le cadre d'une étude de faisabilité du contrôle de l'interface du traitement de texte de OpenOffice.org. Le but était de donner une interface la plus succincte possible pour que les utilisateurs soient guidés dans leur utilisation.

La documentation concernant SimpleBean est au chapitre 16 du SDK. Le chapitre de ContextMenuInterceptor a été égaré et est en cours de re-construction. Nous ne rentrerons pas dans le détail de l'API OpenOffice.org et laissons le lecteur se référer au SDK pour le détail des fonctions utilisées

Tout le code utilisé ci-dessous vient du SDK de OpenOffice.org (chapitre OfficeBean pour BasicOfficeBean et Office Development pour le ContextMenuInterceptor).

2 Utiliser SimpleBean

SimpleBean sert à se connecter à OpenOffice.org par l'intermédiaire d'un Bean : BasicOfficeBean. Un bean est un composant que nous pouvons ré-utiliser ainsi que ses méthodes et ses propriétés. L'exemple contient une autre fonctionnalité, non exploitée ici : la possibilité de modifier la visibilité des barres de menu.

2.1 Connexion à OpenOffice par BasicOfficeBean

Avant toute chose, il est indispensable de comprendre comment un bean se connecte à OpenOffice.org.

Il faut tout d'abord créer un ServiceFactory pour entamer le dialogue entre le bean et OpenOffice.org, dont on se sert comme serveur. Pour mieux maîtriser le paramétrage de OpenOffice.org en mode serveur, veuillez vous référer au chapitre First Step du SDK.

```
// Etablir une connexion par une demande au ServiceFactory
 XmultiComponentFactory compfactory;
 compfactory = mConnection.getComponentContext().getServiceManager();
 mServiceFactory = (XmultiServiceFactory)UnoRuntime.queryInterface(
XMultiServiceFactory.class, compfactory);
```

Le code vient du SDK de OpenOffice.org, Developer's Guide, chapitre OfficeBean. Ce code est partiel, il n'est là que pour souligner les phases importantes de la connexion.

Les interfaces UnoRuntime.queryInterface[1] facilitent la création d'objet, le service factory dans notre cas.

Ensuite, il faut initialiser une Frame pour accueillir l'objet de OpenOffice.org. Cette Frame est construite à partir du ServiceFactory initialisé ci-dessus.

```
XWindow window = (XWindow) UnoRuntime.queryInterface( XWindow.class, mWindow.getUNOWindowPeer());
object  = mServiceFactory.createInstance( "com.sun.star.frame.Task");
if ( object == null )
 object  = mServiceFactory.createInstance( "com.sun.star.frame.Frame");
mFrame = (XFrame)UnoRuntime.queryInterface( XFrame.class, object );
mFrame.initialize(window);
```

Note : Retenons que toute Frame a un controller. Cela nous resservira plus tard.

Après la connexion, on chargera l'objet d'OpenOffice.org que l'on veut utiliser (Writer, Draw, Presentation, Calcul...) grâce à son URL. Pour Writer, cette URL a la forme :

1 Cf chapitre « First steps/Working with objects » du SDK

« **private:factory/swriter** » (scalc, sdraw, simpress).

Le code de chargement de l'objet de OpenOffice.org :

```
// Chargement du document
XComponent xComponent = xLoader.loadComponentFromURL( url, mFrame.getName(), FrameSearchFlag.ALL, aArgs );
```

url précise l'outil que l'on souhaite instancier dans OpenOffice. Le document peut être un document existant ou non.

mFrame précise sur quel conteneur Writer va être déposé

Arg donne les valeurs des propriétés liées à cet objet. On gardera les valeurs par défaut données par l'exemple.

Pour de plus amples précisions sur l'utilisation de loadComponentFromURL se référer au SDK ou à la doc de l'API : com.sun.star.Frame.XComponentLoader.

Si SimpleBean est une classe fille de BasicOfficeBean et que simpleBean est une instance de SimpleBean, alors nous pouvons appeler une instance de Writer (basée sur le modèle par défaut) de OpenOffice.org comme ci-dessous :

```
simpleBean.load("private:factory/swriter");
```

2.2 ContextMenuInterceptor

La classe ContextMenuInterceptor a été créée dans le SDK pour ajouter une fonctionnalité au menu contextuel de base. Nous ne détaillerons pas ici son fonctionnement, mais uniquement sa manière de se connecter au BasicOfficeBean.

Dans le SDK, il est expliqué au chapitre 6 « Office Development » qu'il faut lancer OpenOffice.org avec une option qui permet de rendre le port ouvert pour le piloter depuis Java.

Regardons comment la connexion se fait dans cet exemple :

```
private OfficeConnect(String sHost, String sPort){
 try
 {
  String sConnectString = "uno:socket,host=" + sHost + ",port=" + sPort +  ";urp;StarOffice.ServiceManager";
  com.sun.star.lang.XMultiServiceFactory xLocalServiceManager =
com.sun.star.comp.helper.Bootstrap.createSimpleServiceManager();
  com.sun.star.bridge.XUnoUrlResolver xURLResolver = (com.sun.star.bridge.XUnoUrlResolver)      UnoRuntime.queryInterface
(com.sun.star.bridge.XUnoUrlResolver.class,                                    xLocalServiceManager.createInstance
                                                        ("com.sun.star.bridge.UnoUrlResolver"));
  mxServiceManager = (com.sun.star.lang.XMultiServiceFactory) UnoRuntime.queryInterface(
            com.sun.star.lang.XMultiServiceFactory.class,
```

```
                xURLResolver.resolve(sConnectString));
 }
 catch (com.sun.star.uno.RuntimeException exUNO){
   System.out.println("connection failed" + exUNO);
 }
 catch (com.sun.star.uno.Exception exRun){
   System.out.println("connection failed" + exRun);
 }
 catch (java.lang.Exception exJava) {
   System.out.println("connection failed" + exJava);
 }
}
```

La connexion est similaire à celle de OfficeBean : il faut créer un ServiceFactory pour ensuite faire la connexion avec la Frame de ContextMenuInterceptor.

```
OfficeConnect aConnect = OfficeConnect.createConnection("localhost", "8100");
com.sun.star.frame.XDesktop xDesktop = (com.sun.star.frame.XDesktop)
aConnect.createRemoteInstance(com.sun.star.frame.XDesktop.class,
                    "com.sun.star.frame.Desktop");
com.sun.star.frame.XFrame xFrame = xDesktop.getCurrentFrame();
```

createConnection appelle OfficeConnect décrit ci-dessus.

Voici le code de CreateRemoteInstance qui utilise le ServiceFactory pour créer la connexion :

```
public Object createRemoteInstance(Class aType, String sServiceSpecifier)
{
 Object aResult = null;
 try
 {
   aResult = UnoRuntime.queryInterface(aType,mxServiceManager.createInstance(sServiceSpecifier));
 }
 catch (com.sun.star.uno.Exception ex) {
   System.out.println("Couldn't create Service of type " + sServiceSpecifier + ": " + ex);
   System.exit(0);
 }
 return aResult;
}
```

la Frame créée sur BasicOfficeBean et dans ContextMenuInterceptor sont donc de même nature.

3 Les modifications à apporter pour atteindre notre but

Il suffit d'affecter à ContextMenuInterceptor la Frame de BasicOfficeBean pour se connecter sur son menu contextuel et faire les modification souhaitées :

Code du load de BasicOfficeBean modifié :

```
/**
 * Loads a document referenced by a URL.
 *
 * @param url The document's URL string.
 * @exception java.io.IOException if the document loading process has
 *                      failed.
 */
public synchronized void load( String url ) throws java.io.IOException
{
 [...] //aucun changement au code initial

// Connexion à ContextMenuInterceptor pour contrôler le click droit.
 ContextMenuInterceptor mContext = new ContextMenuInterceptor(mFrame.getController() ) ;


}
```

Dans le code de ContextMenuInterceptor, on modifie le constructeur de telle manière qu'il puisse recevoir le contrôleur comme argument. Nous devons rajouter un constructeur vide pour que le code marche.

En réalité, on utilise le controller de mFrame de BasicOfficeBean, car c'est le dernier élément de Frame que ContextMenuInterceptor utilise pour se connecter, cela évite de recommencer à construire un desktop, une Frame etc...

Voici le code modifié dans ContextMenuInterceptor :

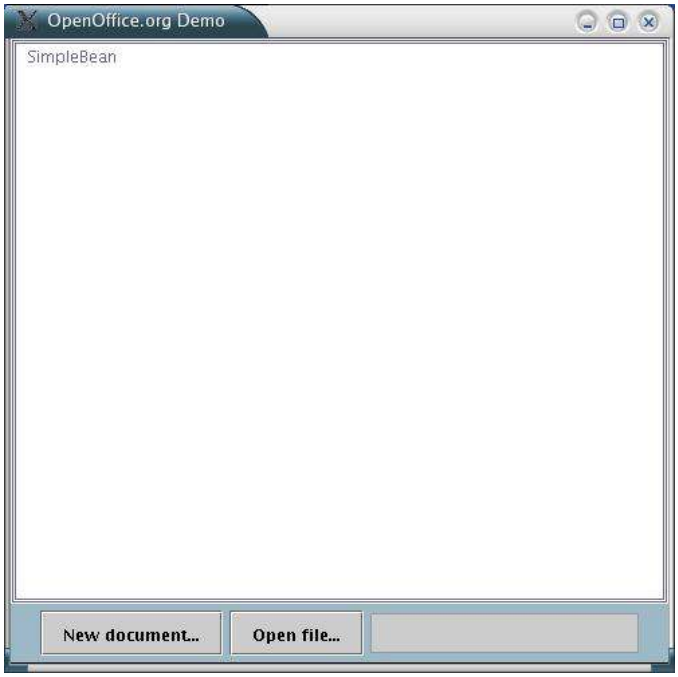
```
public ContextMenuInterceptor()
{}


public ContextMenuInterceptor(XController aXController )
{
 try {
  com.sun.star.frame.XController xController = aXController ;
  if ( xController != null ) {
   com.sun.star.ui.XContextMenuInterception xContextMenuInterception =
            ( com.sun.star.ui.XContextMenuInterception ) UnoRuntime.
            queryInterface(com.sun.star.ui.XContextMenuInterception.class,
```

```
                xController ) ;
    if ( xContextMenuInterception != null ) {
      ContextMenuInterceptor aContextMenuInterceptor = new ContextMenuInterceptor() ;
      com.sun.star.ui.XContextMenuInterceptor xContextMenuInterceptor =
                  ( com.sun.star.ui.XContextMenuInterceptor )
                  UnoRuntime.queryInterface(
                  com.sun.star.ui.XContextMenuInterceptor.class,
                  aContextMenuInterceptor ) ;
      xContextMenuInterception.registerContextMenuInterceptor(xContextMenuInterceptor ) ;
    }
   }
 }
 catch ( java.lang.Throwable ex ) {
   // catch java exceptions ? do something useful
   System.out.println( " Sample caught exception! " + ex ) ;
   System.exit( 1 ) ;
 }
}
```

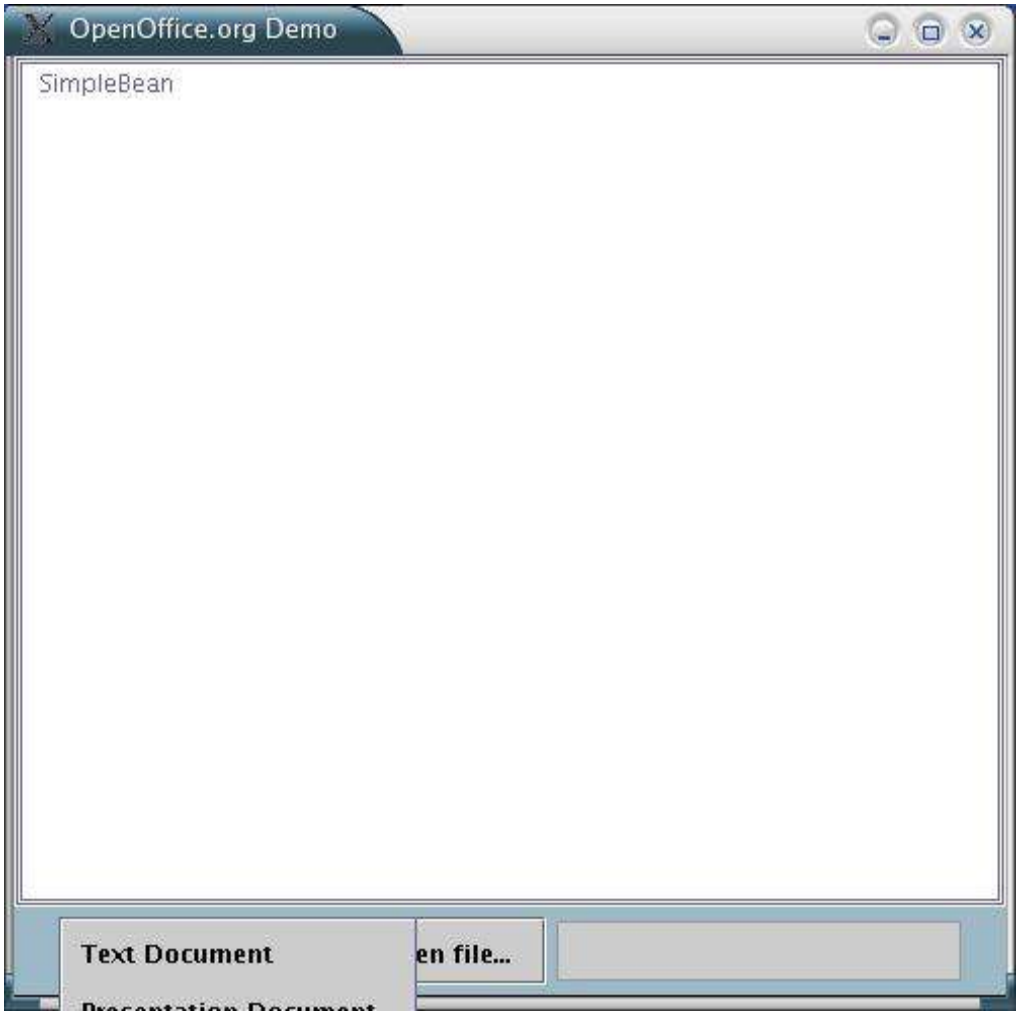
4 Mise en oeuvre

Dans le cadre de notre développement de cette interface simplifiée, nous allons donner un aperçu de la mise en oeuvre.

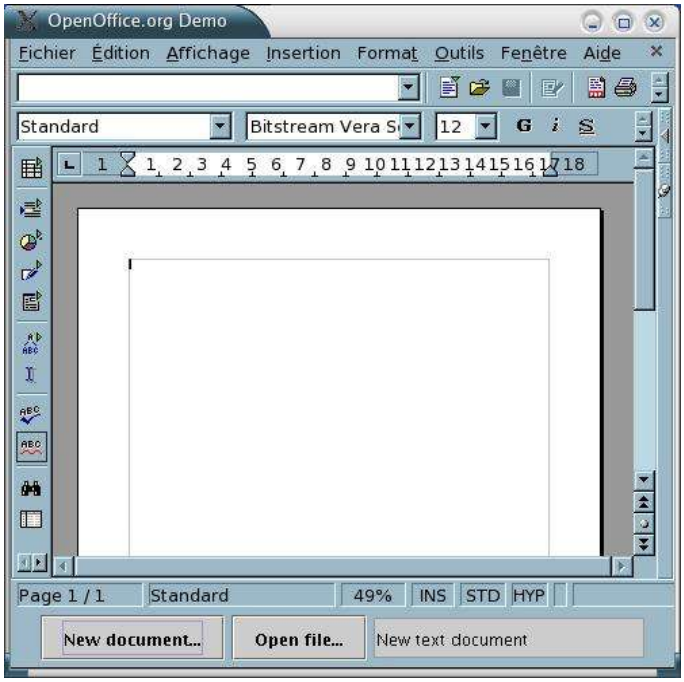
A l'exécution du code, une fenêtre applicative s'ouvre pour donner place à une boîte de dialogue qui permet de choisir si l'on veut ouvrir un nouveau document ou un document existant.

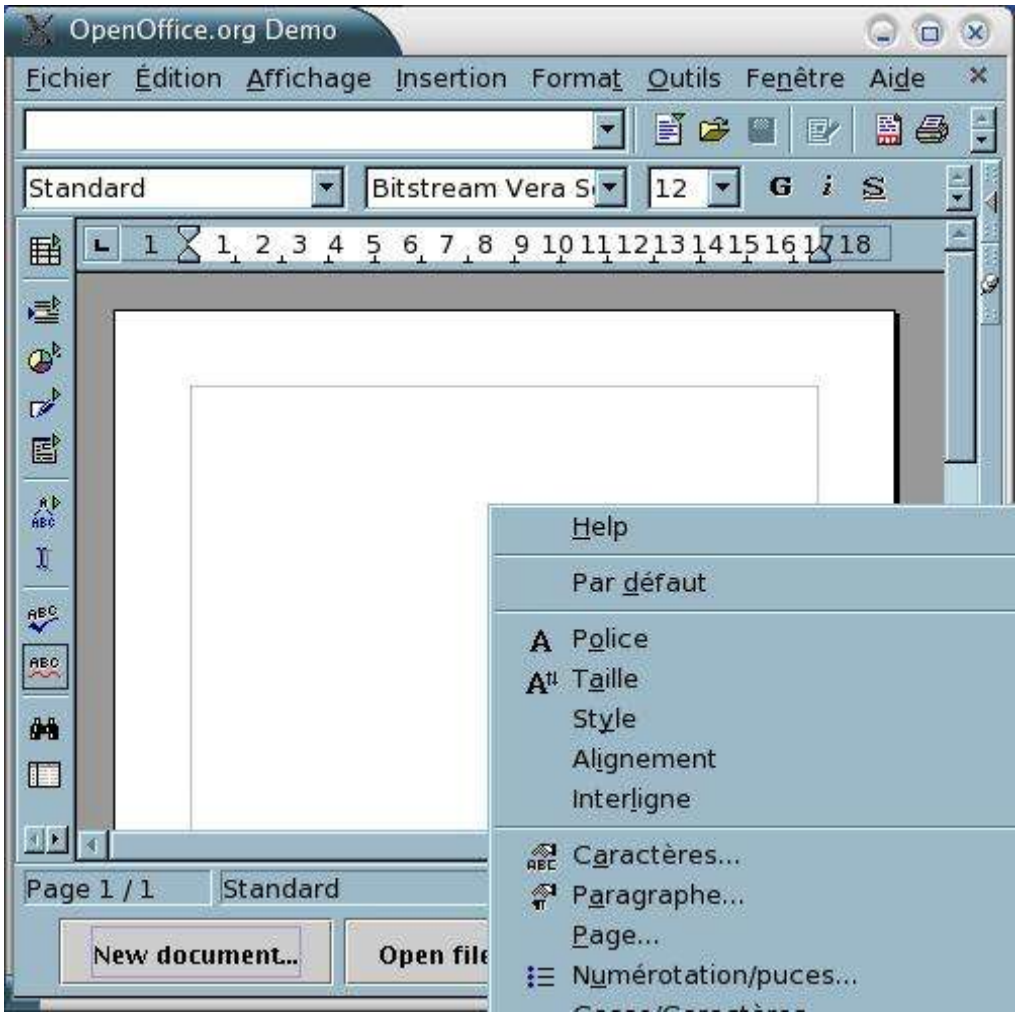


Prenons l'exemple d'un nouveau document :



Une fois le document ouvert (quelque soit sa nature)





le processus de modifications du menu contextuel se met en place. Si on clique sur le bouton droit, le menu sera modifié en fonction du code que le programmeur aura écrit dans la méthode notifyContextMenuExecute : disparition du menu, ajout de nouveaux raccourcis (ici le raccourci Help), suppression de certains raccourcis. Cf documentation SDK pour plus de détails.

5 Conclusion

Ce que nous enseigne cet exemple est l'utilisation du ServiceFactory et du port pour connecter le client au serveur et des Frames pour maîtriser l'interface graphique de OpenOffice.

Le ServiceFactory permet de connecter l'élément client (le bean) au serveur. Au lancement de OpenOffice en tant que serveur, il faut juste préciser quel port est ouvert pour cette connexion.

Le deuxième enseignement est le moyen de maîtriser l'interface de OpenOffice. Le programmeur doit définir les Frames par rapport au ServiceFactory défini précédemment pour pouvoir réceptionner l'objet OpenOffice désiré. Cette Frame nous donne alors accès à des propriétés que peuvent exploiter différents modules : gestion des menus (ajout, suppression, modification), gestion du menu contextuel, gestion du type de document (writer, calc, draw, Presentation...) etc...

Enfin,.

Une fois la connexion réalisée, l'objet « OpenOffice » devient très malléable et configurable. On peut brancher des autres composants en utilisant les propriétés de la Frame initiale. Cela permet essentiellement de contrôler l'environnement de l'utilisateur, soit dans un but de l'aider en lui rajoutant des raccourcis soit pour le guider en lui donnant un outil très simple à utiliser.

Une seule limitation, si une fois dans Writer vous utilisez la voie normale pour ouvrir une nouvelle page, les propriétés rajoutées dans SimpleViewer (la classe qui instancie le bean OfficeBean) seront perdues.

6 Annexe : code de la fonction `notifyContextMenuExecute :`

Le code est clairement expliqué dans les commentaires.

```
 /**
 *
 * @param aEvent
 * @return
 * @throws java.lang.RuntimeException
 */
public ContextMenuInterceptorAction notifyContextMenuExecute(
        com.sun.star.ui.ContextMenuExecuteEvent aEvent ) throws RuntimeException {

    try {

        // Récupère le contenu de menu contextuel et demande au serviceFactory
        // de créer des sous-menus, des entrées de menus et des séparateurs.
        com.sun.star.container.XIndexContainer xContextMenu = aEvent.ActionTriggerContainer;
        com.sun.star.lang.XMultiServiceFactory xMenuElementFactory =
            (com.sun.star.lang.XMultiServiceFactory)UnoRuntime.queryInterface(
            com.sun.star.lang.XMultiServiceFactory.class, xContextMenu );
        if ( xMenuElementFactory != null ) {
            // Crée une entrée de menu racine et son sous-menu.
            com.sun.star.beans.XPropertySet xRootMenuEntry =
                (XPropertySet)UnoRuntime.queryInterface(
                    com.sun.star.beans.XPropertySet.class,
                    xMenuElementFactory.createInstance( "com.sun.star.ui.ActionTrigger" ));

            // Crée une ligne de séparation pour notre nouveau sous-menu « help »
            com.sun.star.beans.XPropertySet xSeparator =
                (com.sun.star.beans.XPropertySet)UnoRuntime.queryInterface(
                    com.sun.star.beans.XPropertySet.class,
                    xMenuElementFactory.createInstance( "com.sun.star.ui.ActionTriggerSeparator" ));

            Short aSeparatorType = new Short( ActionTriggerSeparatorType.LINE );
            xSeparator.setPropertyValue( "SeparatorType", (Object)aSeparatorType );

            // Demande le sous menu  pour que le conteneur d'index obtienne l'accès
            com.sun.star.container.XIndexContainer xSubMenuContainer =
                (com.sun.star.container.XIndexContainer)UnoRuntime.queryInterface(
                    com.sun.star.container.XIndexContainer.class,
                        xMenuElementFactory.createInstance(
```

```
            "com.sun.star.ui.ActionTriggerContainer" ));

    // Initialise l'entrée du menu racine
    xRootMenuEntry.setPropertyValue( "Text", new String( "Help" ));
    xRootMenuEntry.setPropertyValue( "CommandURL", new String( "slot:5410" ));
    xRootMenuEntry.setPropertyValue( "HelpURL", new String( "5410" ));
    xRootMenuEntry.setPropertyValue( "SubContainer", (Object)xSubMenuContainer );

    // Crée des entrées de menu pour le nouveau sous-menu

    // Initialise l'aide/le contenu de l'entrée du menu
    XPropertySet xMenuEntry = (XPropertySet)UnoRuntime.queryInterface(
                        XPropertySet.class, xMenuElementFactory.createInstance(
                            "com.sun.star.ui.ActionTrigger" ));

    xMenuEntry.setPropertyValue( "Text", new String( "Content" ));
    xMenuEntry.setPropertyValue( "CommandURL", new String( "slot:5401" ));
    xMenuEntry.setPropertyValue( "HelpURL", new String( "5401" ));

    // Insère une entrée de menu pour le sous-menu
    xSubMenuContainer.insertByIndex( 0, (Object)xMenuEntry );

    // initialise l'aide/l'agent de l'aide
    xMenuEntry = (com.sun.star.beans.XPropertySet)UnoRuntime.queryInterface(
                com.sun.star.beans.XPropertySet.class,
                    xMenuElementFactory.createInstance(
                        "com.sun.star.ui.ActionTrigger" ));
    xMenuEntry.setPropertyValue( "Text", new String( "Help Agent" ));
    xMenuEntry.setPropertyValue( "CommandURL", new String( "slot:5962" ));
    xMenuEntry.setPropertyValue( "HelpURL", new String( "5962" ));

    // Insère l'entrée du menu pour le sous-menu
    xSubMenuContainer.insertByIndex( 1, (Object)xMenuEntry );

    // Initialise l'aide et les tips
    xMenuEntry = (com.sun.star.beans.XPropertySet)UnoRuntime.queryInterface(
                com.sun.star.beans.XPropertySet.class,
                    xMenuElementFactory.createInstance(
                        "com.sun.star.ui.ActionTrigger" ));
    xMenuEntry.setPropertyValue( "Text", new String( "Tips" ));
    xMenuEntry.setPropertyValue( "CommandURL", new String( "slot:5404" ));
    xMenuEntry.setPropertyValue( "HelpURL", new String( "5404" ));
```

```
            // Insère les entrées du sous-menu.
            xSubMenuContainer.insertByIndex( 2, (Object)xMenuEntry );

            // Ajoute un séparateur dans le menu contextuel donné
            xContextMenu.insertByIndex( 0, (Object)xSeparator );

            // Ajoute un nouveau sous menu dans le menu de contexte donné
            xContextMenu.insertByIndex( 0, (Object)xRootMenuEntry );

            // Le contrôleur devrait exécuter le menu contextuel modifié and arrêter de notifier les autres intercepteurs.
            return com.sun.star.ui.ContextMenuInterceptorAction.EXECUTE_MODIFIED;
        }
    }
    catch ( com.sun.star.beans.UnknownPropertyException ex ) {
        // faire quelque chose qui puisse aider
        // nous utilisons une propriété inconnue
    }
    catch ( com.sun.star.lang.IndexOutOfBoundsException ex ) {
        // faire quelque chose qui puisse aider
        // Nous utilisations un index invalide pour accéder au conteneur
    }
    catch ( com.sun.star.uno.Exception ex ) {
        // Quelque chose d'étrange s'est produite !
    }
    catch ( java.lang.Throwable ex ) {
        // catch java exceptions ? do something useful
    }

    return com.sun.star.ui.ContextMenuInterceptorAction.IGNORED;
}
```

7 Crédits

Auteur: Aurélie Schröder – Université de Genève – AurelieSch@netcourrier.com

Remerciements: Laurent Godard pour son soutien et ses conseils toujours avisés.

Intégré par : Sophie Gautier

Dernière modification: 2003/11/06

Contacts : Projet Documentation OpenOffice.org http://fr.openoffice.org

Traduction :

8 Licence