

Document management with the OpenOffice.org API

- A short history of OOO
- Compatibility between OOO and MSO
- UNO: Unified Network Objects
- The Java API/Code Examples
- Topics not covered
- Conclusions

- A short history of OOO
- Compatibility between OOO and MSO
- UNO: Unified Network Objects
- The Java API/Code Examples
- Topics not covered
- Conclusions

- mid 1980's: StarDivision is founded
- summer of 1999: Sun Microsystems acquires StarDivision
- June 2000: StarOffice 5.2 released under GPL
- October 2000: First release of OOo
- September 2001: OOo 1.0 SDK released
- September 2003: OOo 1.1 released
- June 2005: Sun and MS join hands in supporting each others document formats
- November 2005: OOo 2.0 released
Current version: OOo 2.0.2

- OOo supports these external file formats:
 - Microsoft Word 6.0/95/97/2000/XP) (.doc, .dot)
 - Microsoft Word 2003 XML (.xml)
 - Microsoft Excel 97/2000/XP (.xls, .xlw, .xlt)
 - Microsoft Excel 2003 XML (.xml)
 - Microsoft PowerPoint 97/2000/XP (.ppt, .pps)
 - Microsoft PowerPoint 97/2000/XP Template

 - PDF (export only)

- A short history of OOO
- **Compatibility between OOO and MSO**
- UNO: Unified Network Objects
- The Java API/Code Examples
- Topics not covered
- Conclusions

WARNING:

OpenOffice.org can open Microsoft Office files. The reverse is not true: at this time, Microsoft Office can not open OpenOffice.org formats.

If you need to send files to someone using Microsoft Office, save your file first in the native OpenOffice.org format, then save it to one of the many supported Microsoft Office formats.

By doing this, you ensure that even if the filter can not translate perfectly, you have your original in its native format.

- Some caveats

- Cells copied into Word from Excel are opened as text table
- WordArt objects are converted to FontWork
- Vector graphics get imported almost literally
- Links from linked texts are lost
- Macro languages of MSO and OOo are not interchangeable
- Formula arguments in Spreadsheets may differ in type
- Impress does not support three-color gradients, double and triple borders, or round-dotted borders

<http://documentation.openoffice.org/manuals/index.html>

- A short history of OoO
- Compatibility between OoO and MSO
- **UNO: Unified Network Objects**
- The Java API/Code Examples
- Topics not covered
- Conclusions

- Environment for network objects across programming languages and platforms
- Specified in an abstract meta language, called UNOIDL
- Uses a factory called the **service manager**
- Provides bridges to send method calls and receive return values between processes and between objects written in different implementation languages
- Most objects of OpenOffice.org are able to communicate in a UNO environment

- OOo API is a language independent approach to specify the functionality of OOo
 - Access the functionality of OOo
 - Extend the functionality by own solutions and new features
- Available for
 - Java
 - C++
 - OOo Basic
 - MS Automation Bridge
 - Common Language Infrastructure (.Net)
 - Python

- OOo API consists of
 - Data types
 - e.g. void, boolean, double, any
 - Interfaces
 - single-inheritance and multiple-inheritance
 - Services
 - high-level multiple-inheritance interfaces
 - Structs
 - "class without methods"
 - Modules
 - similar to namespaces in C++ or packages in Java
 - Exceptions

- API drawbacks:
 - No one-to-one relationship between a certain service specification and a real object
 - It is impossible to comprehend fully from the API reference what a given instance of an object in OOo is capable of
 - There are several entry points where object implementations are actually available
 - Mix of old-style and new-style implementations
- Consequently, it is sometimes confusing to look up a needed functionality in the API reference so refer to the Development Guide

- Basic workflow
 - Start OOo either locally or remotely
 - Connect to OOo instance
 - Request static UnoRuntime environment through ServiceManager
 - Request XComponent by querying the static UnoRuntime environment for the needed interface (old style) or by setting Properties (new style)
 - Get access to text, pages, shapes etc. by querying the UnoRuntime environment for the needed interfaces (old style) or setting Properties (new style)
 - Save/print/convert document and close it
- Closing the last document will shut OOo down

- A short history of OOO
- Compatibility between OOO and MSO
- UNO: Unified Network Objects
- **The Java API/Code Examples**
- Topics not covered
- Conclusions

- Creating a local connection

```
XComponentContext xRemoteContext = Bootstrap.bootstrap();
XMultiComponentFactory xRemoteServiceManager =
    xRemoteContext.getServiceManager();
Object desktop =
    xRemoteServiceManager.createInstanceWithContext(
        "com.sun.star.frame.Desktop", xRemoteContext);
```

- ## Creating a remote connection

```

XComponentContext xLocalContext = Bootstrap.createInitialComponentContext(
    null);
XMultiComponentFactory xLocalServiceManager =
    xLocalContext.getServiceManager();
Object urlResolver = xLocalServiceManager.createInstanceWithContext(
    "com.sun.star.bridge.UnoUrlResolver", xLocalContext);
XUnoUrlResolver xUrlResolver = (XUnoUrlResolver)
    UnoRuntime.queryInterface(XUnoUrlResolver.class, urlResolver);
Object initialObject = xUrlResolver.resolve(
    "uno:socket,host=localhost,port=2002;urp;StarOffice.ServiceManager");
XPropertySet xPropertySet =
    (XPropertySet)UnoRuntime.queryInterface(XPropertySet.class,
    initialObject);
Object context = xPropertySet.getPropertyValue("DefaultContext");
XComponentContext xRemoteContext =
    (XComponentContext)UnoRuntime.queryInterface(XComponentContext.class,
    context);
XMultiComponentFactory mxRemoteServiceManager =
    xRemoteContext.getServiceManager();
Object desktop = mxRemoteServiceManager.createInstanceWithContext(
    "com.sun.star.frame.Desktop", null);
    
```

- OOo server is started with

```
soffice -accept=socket,host=0,port=2002;urp;
```

or

```
soffice -accept=socket,host=0,port=2002;urp; -invisible
```

- Connection is closed when not used anymore, remote counterpart crashes or the connection fails. Use XEventListener to detect closing connection.

- Documents are created or loaded using

```
XComponentLoader xComponentLoader =
(XComponentLoader)UnoRuntime.queryInterface(XComponentLoad
er.class, desktop);
PropertyValue[] pPropValues = new PropertyValue[0];
XComponent document =
xComponentLoader.loadComponentFromURL(templateURL,
"_blank", 0, pPropValues);
```

- Here templateURL is either URL to file or one of

```
private:factory/swriter
private:factory/scalc
private:factory/sdraw
private:factory/simpress
```

- Documents can be stored by
 - querying XStorable interface
 - calling store() or storeAsURL(String aURL)
- Documents can be exported by
 - querying XStorable interface
 - specifying the FilterName Property
 - e.g. "MS Word 97", "writer_pdf_Export"
 - calling store() or storeAsURL(String aURL)
- Documents can be printed by
 - querying XPrintable class
 - setting Properties like Name, PageSize, etc.
 - calling print()

DEMO

- Text Documents
 - Text
 - e.g. characters, words, lines, paragraphs
 - Service Manager (document internal)
 - e.g. text tables, text fields, drawing shapes, text frames
 - Draw Page
 - for images so text can wrap around them
 - Text Content Suppliers
 - provide access to texts in Text and Service Manager
 - Objects for styling and numbering

- XText provides access to text Strings

```
XText xText = document.getText();  
xText.setString("Hello World!");
```

- Move around using `getStart()` and `getEnd()`
- More flexible moving around with `XTextCursor`, `XWordCursor`, `XSentenceCursor` and `XParagraphCursor`
- Using one of the cursor types provides access to font, colors, size, weight, shadow, adjustment, tab stops etc

- Other features include
 - search and replace
 - tables
 - text fields (e.g. date, author, page number, cross) reference
 - bookmarks
 - indexes
 - references
 - headers and footers
 - images
 - drawings
 - styles (e.g. numbers, bullets, outline numbering)

DEMO

- Spreadsheet Documents
 - Spreadsheets Container
 - contains the sheets (duh)
 - Service Manager (document internal)
 - e.g. shape objects, page headers
 - Draw Page
 - for images so sheet content can wrap around them
 - Content Properties
 - Objects for styling
 - provide access to cells, pages and number formats

- XSpreadsheets provides access to sheets
- XSpreadsheet provides access to cells

```
XSpreadsheetDocument xDocument =
(XSpreadsheetDocument)UnoRuntime.queryInterface(XSpreadshe
etDocument.class, document);
XSpreadsheets xSheets = xDocument.getSheets();
XSpreadsheet xSheet = xSheets.getByName("Sheet1");
```

- Get access to cells via e.g. `getCellByPosition(x,y)` or `getCellRangeByPosition(x0, y0, x1, x2)`
- Set value or formula on individual cell, group of cells, columns or rows
- Cells contents can be formatted e.g. font, color etc

- Other features include
 - search and replace
 - images
 - drawings
 - styles (e.g. numbers, bullets, outline numbering)
 - sorting
 - filtering
 - import from database
 - data validation
 - charts
 - calculations

DEMO

- Draw/Impress Documents
 - Draw Pages
 - contain shapes, text, tables, charts etc.
 - Master pages
 - defines formatting, numbers etc
 - Layer managers
 - Hands out pages (Impress only)
 - Presentation capacities (Impress only)

- XDrawPagesSupplier/XPresentationSupplier
- XDrawPage/XPresentation

```
XDrawPagesSupplier xDrawPagesSupplier =
(XDrawPagesSupplier)UnoRuntime.queryInterface(XDrawPagesSupplier.class,
document);
XDrawPages xDrawPages = xDrawPagesSupplier.getDrawPages();
Object drawPage = xDrawPages.getByIndex(0);
XDrawPage xDrawPage = (XDrawPage)UnoRuntime.queryInterface(XDrawPage.class,
drawPage);
```

- Create an XShape object and start drawing
- Full control over shape, size, fill, color, lines, layers
- Shapes can only be accessed by their index, not by their name

- Other features include
 - moving, scaling, rotating, shearing
 - transforming
 - grouping, combining, binding
 - text properties like font and color
 - slide and shape animation (impress)
 - starting and stopping presentation
 - zooming

DEMO

- A short history of OOO
- Compatibility between OOO and MSO
- UNO: Unified Network Objects
- The Java API/Code Examples
- **Topics not covered**
- Conclusions

- Charts
- OOO Basic
- Database access
- Forms
- Universal Content Broker
- Configuration management
- JavaBean
- Accessibility
- Scripting Framework
- Custom extensions to OOO

- A short history of OOO
- Compatibility between OOO and MSO
- UNO: Unified Network Objects
- The Java API/Code Examples
- Topics not covered
- **Conclusions**

- Pros
 - OOo API is very, very powerfull
 - Almost complete access to all OOo features
 - Capable of opening, modifying and storing MSO docs
- Cons
 - Steep learning curve
 - Undocumented features are hard to grasp
 - Not 100% compatible with MSO so be carefull with opening, modifying and then saving MSO documents
 - Heavy on system resources

- Useful URLs

- <http://openoffice.org>
- <http://api.openoffice.org/>
 - api.openoffice.org/DevelopersGuide/DevelopersGuide.html
- <http://www.oooforum.org/>
- <http://documentation.openoffice.org>
 - documentation.openoffice.org/manuals/index.html
- <http://www.amis.nl>
- <http://technology.amis.nl/blog>

wouter.van.reeven@amis.nl

Q & A