
OpenOffice.org
Conference 2008 Beijing
世界开源大会

OpenOffice.org Conference 2008 Beijing

# Creating/Devising Specific OpenOffice.org Support for Dynamic Scripting Languages

Prof. Dr. Rony G. Flatscher

WU (Wirtschaftsuniversität Wien)

Austria, Europe

# Agenda


- Static versus Dynamic Languages
  - Overview
  - Example languages
    - Java as a static language, OOo scripting framework
    - ooRexx as a dynamic language, OOo scripting engine
- OOo (OpenOffice.org)
  - UNO (Universal Network Objects)
  - OOo modules
- Roundup

# Static Languages



- Informal definition
  - Type of values is predetermined
    - Constant and variables accept only values of predetermined type, otherwise an error gets raised
  - Allows compiler to pedantically check for errors (misuse of types)
- Examples: Java, C++
- OOo and static languages
  - "Java bridge" to interact with OOo objects
  - OOo's scripting framework is implemented in Java

# Dynamic Languages


- Informal definition
  - Type of values is not predetermined
    - Constant and variables accept any values at runtime
- Examples: OOo Basic, ooRexx, Python
- OOo and dynamic languages
  - OOo Basic as the standard scripting language
  - Python built-in
- Adding new scripting languages
  - Need to use Java for the OOo Scripting Framework

# Matching Dynamic Languages with Static Java


- Apache's BSF 2.4
  - http://jakarta.apache.org/bsf
  - Java framework to deploy scripting languages
    - Helper and utility classes, including dynamic (bytecode) creation of event adapters
    - BSF engines for many scripting languages
      - e.g. Jacl (TcL), JRuby, Jython, ooRexx, …
- Apache's BSF 3.0/JSR-223 ("Java 6 scripting")
  - Framework to dispatch and interact with scripting languages, part of Java 6 and higher
  - Needs BSF 3.0/JSR-223 engines

# ooRexx, 1


- Open Object Rexx (oorexx)
  - Free opensource scripting language
    - http://www.ooRexx.org
  - Dynamic language
    - Interpreter based
  - Originally the OO successor to IBM's REXX
    - IBM handed source over to the non-profit Rexx language association (http://www.RexxLA.org)
    - Opensourced in 2004 by RexxLA
    - Support for many platforms
    - Major upgrade (version 4.0) upcoming

# ooRexx, 2


- Open Object Rexx (oorexx)
  - Perfect for EUD (end-user development)
    - Easy syntax (almost like pseudo-code)
    - Easy to learn and to understand
    - Powerful OO-paradigm implementation
  - *Very* successfully used for teaching BA students
    - Programming
    - OO concepts
    - Scripting of Windows and Windows applications
      - ooRexx supplies the ActiveX scripting engine interfaces
    - Scripting of Java and Java applications
      - Platform independent scripting of OpenOffice.org !

# What is needed for OOo?


- OOo Scripting engine
  - Java framework!
  - ➔ Scripting language must be able to interact with Java!
- UNO (Universal Network Objects)
  - Component and interaction technology of OOo
  - There is a Java bridge to UNO
  - ➔ Scripting language must be able to use the UNO Java bridge!

# Interfacing with Java, 1



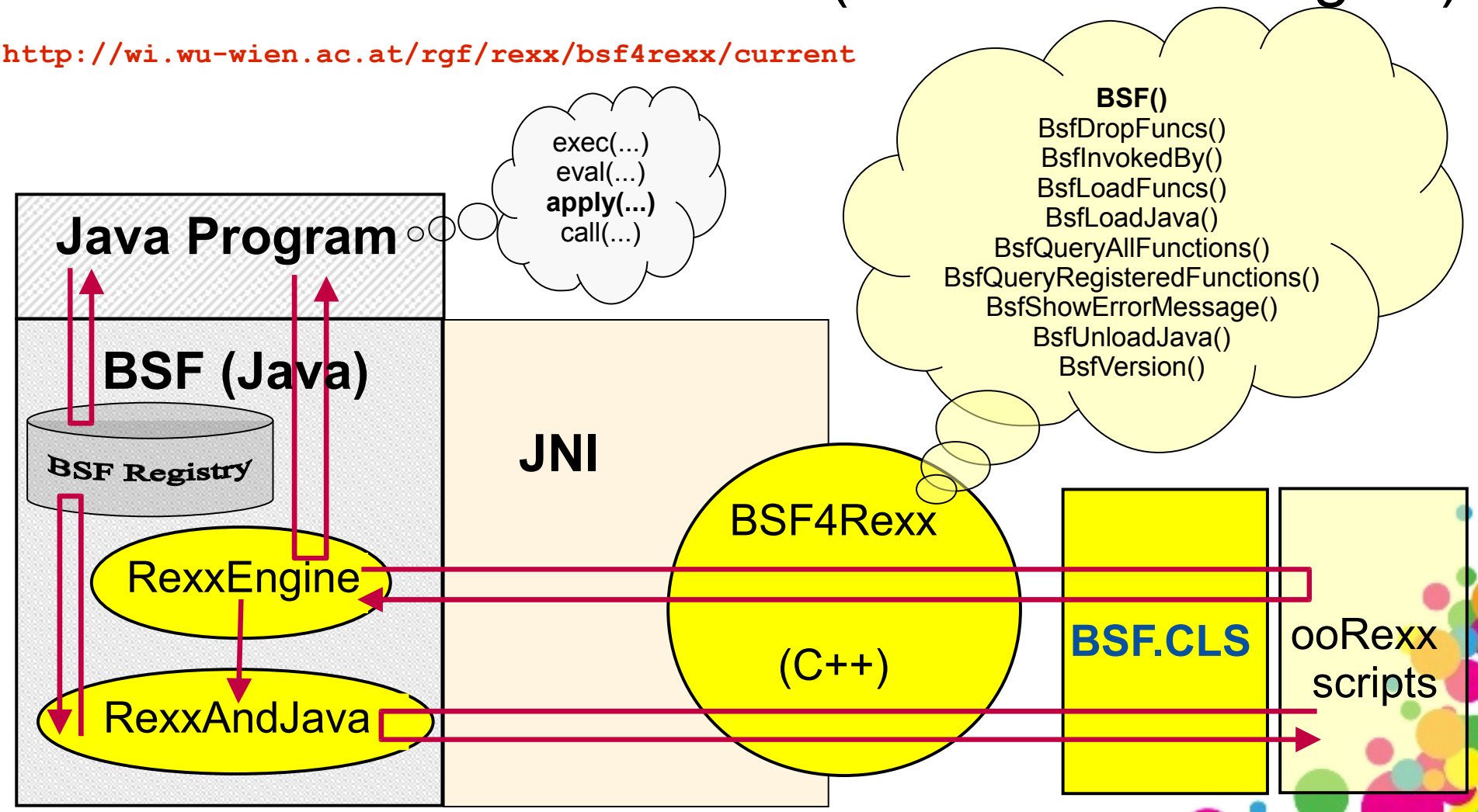
- "BSF4Rexx"
  - http://wi.wu-wien.ac.at/rgf/rexx/bsf4rexx/current/
  - BSF 2.4 scripting engine for ooRexx
    - JNI (Sun's Java Native Interface)
    - Adds (reflective) support on the Java side, e.g.
      - Loading Java classes, creating instances, dispatching messages
      - Creating event adapters on the fly (Java bytecode) ...
    - Easy to interface Java with ooRexx and vice versa (!)
      - → ooRexx (a non-Java language) can use all of Java!
  - BSF.CLS
    - An ooRexx module (program) to camouflage Java as ooRexx!

# Interfacing with Java, 2

OpenOffice.org Conference 2008 Beijing

- Architecture of "BSF4Rexx" (ooRexx BSF engine)

http://wi.wu-wien.ac.at/rgf/rexx/bsf4rexx/current



# Interfacing with Java, 3a


- Querying the version of the Java in use

```
say "java.version:" .bsf4rexx ~system.class ~getProperty('java.version')

::requires BSF.CLS /* loads the Object Rexx (camouflaging) support */
```

**Possible Output:**

```
java.version: 1.6.0_02
```

# Interfacing with Java, 3b


- Behind the curtain
  - ooRexx uppercases everything outside of quotes
    - Therefore ooRexx is not case sensitive, as everything gets uppercased anyway
    - However, Java is case sensitive!
      - ➔ The ooRexx Java support must take this into account!
  - ooRexx example after uppercasing

```
SAY "java.version:" .BSF4REXX ~SYSTEM.CLASS ~GETPROPERTY('java.version')

::REQUIRES BSF.CLS /* loads the Object Rexx (camouflaging) support */
```

# Interfacing with Java, 4


- BSF4Rexx allows to
  - Import Java classes that behave like ooRexx classes
    - "Camouflaging Java as ooRexx"
  - Create Java objects
  - Send ooRexx messages to Java objects
  - Get/set Java fields as if they were ooRexx attributes
  - Create dynamically Java event adapters
  - Interact with a specific Java thread from different ooRexx threads

# Interfacing with Java, 4



- BSF4Rexx allows to
  - Ease creation of (strictly typed) Java array objects
    - Supplies the most important Java class objects representing the primitive Java datatypes in .bsfrexx
  - Interact with Java arrays as if they were ooRexx arrays, e.g.
    - Indices are 1-based for ooRexx programs!
    - Using the ooRexx "DO ... OVER" loop allows to iterate over all Java array elements
  - ...

# Interfacing with OOo, 1



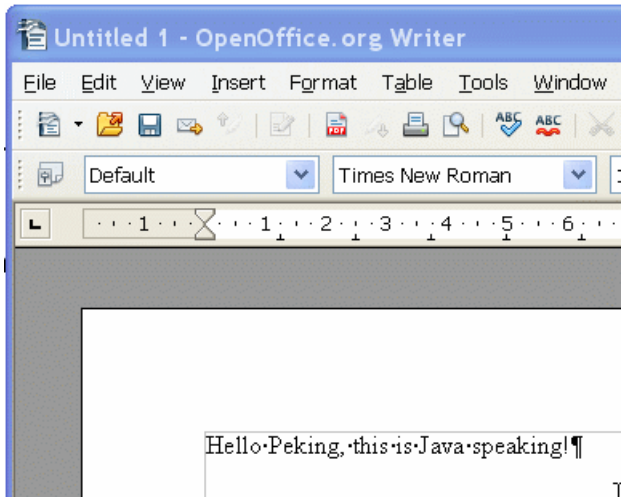
- UNO (Universal Network Objects)
  - CORBA-like component model
  - IDL (Interface Description Language)
    - Used to create a type library available at runtime
  - Client-/Server-model using TCP/IP by default
    - Server side of OOo may reside on another computer!
  - UNO service objects
    - Containers for services, interfaces and properties
    - Recurrent need to "queryInterface" service objects in order to become able to use the interface's functionality and attributes!

# UNO Java Example, 1a


```
import com.sun.star.beans.PropertyValue;
import com.sun.star.comp.helper.Bootstrap;
import com.sun.star.frame.XComponentLoader;
import com.sun.star.frame.XDesktop;
import com.sun.star.lang.XComponent;
import com.sun.star.lang.XMultiComponentFactory;
import com.sun.star.text.XText;
import com.sun.star.text.XTextDocument;
import com.sun.star.uno.UnoRuntime;
import com.sun.star.uno.XComponentContext;

class CreateTextDocument {
   public static void main (String args[]) {
      try {
         XComponentContext xContext=Bootstrap.bootstrap();  // bootstrap UNO
         XMultiComponentFactory xMCF=xContext.getServiceManager();
         if (xMCF != null) {
           Object oDesktop=xMCF.createInstanceWithContext("com.sun.star.frame.Desktop", xContext);
           XDesktop xDesktop=(XDesktop) UnoRuntime.queryInterface(XDesktop.class, oDesktop);

           XComponentLoader xComponentLoader=(XComponentLoader)
               UnoRuntime.queryInterface(XComponentLoader.class, xDesktop);

           String url="private:factory/swriter";              // define a text document
           PropertyValue noProps[]=new PropertyValue[0];      // no properties
           XComponent xWriterComponent=xComponentLoader.loadComponentFromURL(
                                                  url, "_blank", 0, noProps);

// Continued ...
```

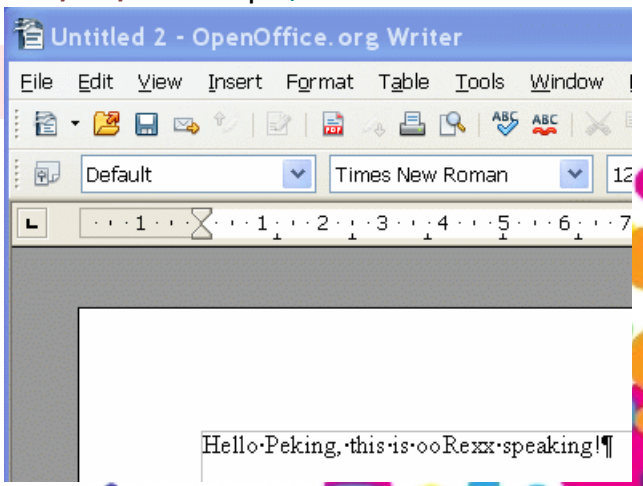
# UNO Java Example, 1b

```
// ... continued

            XTextDocument xTextDocument=(XTextDocument)
                        UnoRuntime.queryInterface(XTextDocument.class, xWriterComponent);

            XText xText=xTextDocument.getText();
            xText.setString("Hello Peking, this is Java speaking!");
          }
        }
        catch( Exception e) {
           e.printStackTrace(System.err);
           System.exit(1);
        }
        System.exit(0);
    }
}
```



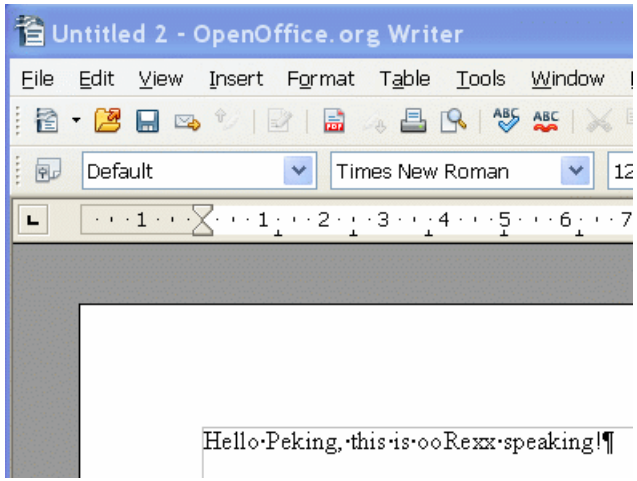
# UNO Java:ooRexx, 1:1


```
UnoRuntime=bsf.loadClass("com.sun.star.uno.UnoRuntime")

clz=bsf.loadClass("com.sun.star.comp.helper.Bootstrap")
xContext=clz~bootstrap                  -- bootstrap UNO
xMCF=xContext~getServiceManager
if xMCF<>.nil then
do
  oDesktop=xMCF~createInstanceWithContext("com.sun.star.frame.Desktop", xContext)
  clz=bsf.loadClass("com.sun.star.frame.XDesktop")
  xDesktop=UnoRuntime~queryInterface(clz, oDesktop)

  clz=bsf.loadClass("com.sun.star.frame.XComponentLoader")
  xComponentLoader=UnoRuntime~queryInterface(clz, xDesktop)

  url="private:factory/swriter"         -- define a text document
  clz=bsf.loadClass("com.sun.star.beans.PropertyValue")
  noProps=uno.createArray(clz,0)        -- no properties
  xWriterComponent=xComponentLoader~loadComponentFromURL(url, "_blank", 0, noProps)

  clz=bsf.loadClass("com.sun.star.text.XTextDocument")
  xTextDocument=UnoRuntime~queryInterface(clz, xWriterComponent)

  xText=xTextDocument~getText()
  xText~setString("Hello Peking, this is ooRexx speaking!")
end

::requires BSF.CLS   /* get the Java support for ooRexx  */
```



# UNO.CLS (ooRexx)


```
oDesktop=UNO.createDesktop()          -- get Desktop object (bootstraps)
xComponentLoader=oDesktop ~com.sun.star.frame.XDesktop ~XComponentLoader

url = "private:factory/swriter"       -- define a text document
xWriterComponent=xComponentLoader~loadComponentFromURL(url,"_blank", 0,.UNO~noProps)

xText=xWriterComponent~XTextDocument~getText()
xText~setString("Hello Peking, this is ooRexx speaking!")

::requires UNO.CLS   /* get the UNO support for ooRexx (uses BSF.CLS itself) */
```



# Query UNO Interfaces


- *Make it as easy as possible for the users of your language to query UNO interfaces from UNO service objects!*
  - Advice: do *not* forgo the need to query those UNO interfaces though
    - Possible e.g. in OOo Basic
    - Ramification
      - It is extremely hard and cumbersome for the non-OOo-expert to find the UNO interfaces from which functionality is being used
      - Makes it practically impossible to transcribe OOo Basic scripts to e.g. C++, Java or any other language that needs to explicitly query UNO interfaces

# Properties, 1



- Optional members of UNO Services
- Peculiarities
  - If getting or setting a PropertyValue one is always forced to explicitly query the respective interface, e.g.
    - com.sun.star.beans.XFastPropertySet
    - com.sun.star.beans.XHierarchicalPropertyState
    - com.sun.star.beans.XMultiPropertySet
    - com.sun.star.beans.XPropertySet
      - getPropertyValue(String aName)
      - setPropertyValue(String aName, any aValue)
    - ...

# Properties, 2


- Peculiarities (continued)
  - If property values are of primitive types, they need to be boxed!
  - Creating a property (struct) usually involves
    1. Creation of a com.sun.star.beans.PropertyValue instance
    2. Assigning a value for the property's Name
    3. Assigning a (boxed) value for the property's Value

# Properties, 3

OpenOffice.org Conference 2008 Beijing

- Possible resolutions
  - Allow getting and setting property values implicitly
    - Remove requirement to explicitly query the com.sun.star.beans.XPropertySet interface from the UNO service, if its methods getPropertyValue(aName) and setPropertyValue(aName, aValue) are about to be used
    - If setting a property value, then autobox aValue !

```
lbl = factory~createInstance("com.sun.star.awt.UnoControlFixedTextModel")
x_lbl = lbl~com.sun.star.beans.XPropertySet
x_lbl~setPropertyValue("TabIndex",  box("short", 1))
x_lbl~setPropertyValue("Name",      "Label1")
x_lbl~setPropertyValue("PositionX", box("int", 8))
```

```
-- No "queryInterface()", no "box(...)"ing:

lbl = factory~createInstance("com.sun.star.awt.UnoControlFixedTextModel")
lbl~setPropertyValue("TabIndex",  1)
lbl~setPropertyValue("Name",      "Label1")
lbl~setPropertyValue("PositionX", 8)
```

# Properties, 4

- Possible resolutions (continued)
  - Supply an easy way to create PropertyValues

```
props = uno.createArray(.UNO~propertyValue,3)
props[1] = .UNO~PropertyValue~new
props[1]~Name = "user"
props[1]~Value = "stefan"
props[2] = .UNO~PropertyValue~new
props[2]~Name = "password"
props[2]~Value = "apple"
props[3] = .UNO~PropertyValue~new
props[3]~Name = "JavaDriverClass"
props[3]~Value = "com.mysql.jdbc.Driver"
```

    - E.g. in form of a routine:
      - prop=uno.createProperty(aName[, aValue=.nil])

```
props = uno.createArray(.UNO~propertyValue,3)
props[1] = uno.createProperty("user"           , "stefan")
props[2] = uno.createProperty("password"       , "apple")
props[3] = uno.createProperty("JavaDriverClass", "com.mysql.jdbc.Driver")
```

# Add Conveniences, 1



- *Create convenience routines, methods, constants*, e.g.
  - Bootstrapping
    - Allow to easily bootstrap an Office
    - Allow to easily bootstrap and create a Desktop object
  - Working with table cells
    - Allow for flexible numeric-only or alphanumeric addressing of cells
  - Make it easy to supply an empty array of type com.sun.star.beans.PropertyValue
    - Needed quite often, e.g. creating documents
  - ...

# Add Conveniences, 2


- The ooRexx UNO.CLS module can be used as an example to research potential conveniences
  - Created over three years, observing the problems end-user programmers have with OOo programming
  - Over 35 convenience routines
  - Supply UNO class objects and constants in the .UNO collection, e.g. the "UNO null" value (the field value com.sun.star.uno.Any.VOID)
    - E.g. a string argument that has no value, needs to submit this value instead (**not** .nil/null) !
      - Peculiarity which is not widely known
  - … and much more!

# "Human-Centric" Error messages


- *Make error messages as helpful as possible!*
  - Script language users are very likely "*end-user developers*"
    - Probably not UNO/OOo experts !
  - Example
    - Property name cannot be found, possible causes
      - Addressing wrong service (does not possess the property)
      - Optional property, does not exist at this time
      - Misspelled name
    - Give a list of alphabetically, case-independently ordered properties that are defined for the service object!

# Uncover as Much Documentation as Possible


- Script programmers may need to know the definition of UNO objects in hand
  - For end-user developers it is very difficult to get to the relevant information
  - Expert users *may* know about UNO IDL and the locations where these are documented
- Make it easy to get at the definition of UNO objects
  - E.g. use the Java class of the BSF4Rexx package org.oorexx.uno.RgfReflectUNO

# Example


```
xContext = UNO.connect()
xMcf = xContext~getServiceManager

clzName="com.sun.star.frame.Desktop"
say "uno.getDefinition(clzName):" ppd(uno.getDefinition(clzName))
say "---"

o = xMcf~createInstanceWithContext(clzName,xContext)
say "o~uno.getDefinition:       " ppd(o~uno.getDefinition)
say "---"

::requires UNO.CLS
```

---

```
E:\OOoCon>showDefs.rex
uno.getDefinition(clzName): UNO_SERVICE|com.sun.star.frame.Desktop|
        com.sun.star.frame.Frame|UNO_SERVICE||com.sun.star.frame.Desktop
        com.sun.star.frame.XDesktop|UNO_INTERFACE||com.sun.star.frame.Desktop
        com.sun.star.frame.XComponentLoader|UNO_INTERFACE||com.sun.star.frame.Desktop
        com.sun.star.document.XEventBroadcaster|UNO_INTERFACE||com.sun.star.frame.Desktop
---
o~uno.getDefinition:        UNO_SERVICE|com.sun.star.frame.Desktop|com.sun.star.comp.framework.Desktop
        com.sun.star.frame.Frame|UNO_SERVICE||com.sun.star.frame.Desktop
        com.sun.star.frame.XDesktop|UNO_INTERFACE||com.sun.star.frame.Desktop
        com.sun.star.frame.XComponentLoader|UNO_INTERFACE||com.sun.star.frame.Desktop
        com.sun.star.document.XEventBroadcaster|UNO_INTERFACE||com.sun.star.frame.Desktop
---
```

# Roundup, 1


- Creating specific UNO/OOo support will ease programming OOo considerably!
- At least create specific support for easying
  - Querying interfaces of UNO services
  - Creating property value structs
  - Getting/setting property values
- Consider to create as user-friendly error messages as possible
  - Makes the programmer much more productive!

# Roundup, 2



- Consider to
  - Create convenience methods, routines, constants
  - Allow the programmer to easily get at the UNO IDL definitions in a ledgible, compact form
    - Take advantage of the org.oorexx.uno.RgfReflectUNO Java class, which is part of the BSF4Rexx package
    - String is built such, that it can be easily parsed
    - Makes the programmer much more productive
  - Further resources
    - Sources and reference PDF-files at: **http://wi.wu-wien.ac.at/rgf/rexx/bsf4rexx/current**

OpenOffice.org
Conference 2008 Beijing

# Thanks!

凝聚全球力量 绽放开源梦想

www.OOabeijing2008.com