OpenOffice.org

Conference 2008 Beijing

世界开源大会

OpenOffice.org Conference 2008 Beijing

The OpenOffice.org Scripting Framework: Adding a Scripting Language

Prof. Dr. Rony G. Flatscher

WU (Wirtschaftsuniversität Wien)

Austria, Europe

Agenda

- OOo scripting framework
  - Overview
  - Dispatching scripts/macros
  - Example
    - Intermixing OOo Basic with ooRexx and vice versa
- Apache Software Foundation's BSF
  - Overview
- Adding a scripting language to OOo
- Outlook

OOo Scripting Framework, 1

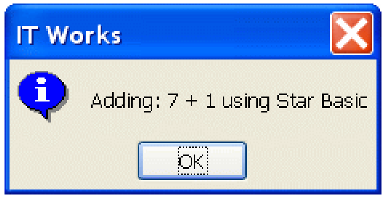
- Module "scripting"
  - Since OOo 2.0
  - Implemented in Java
    - BeanShell
    - JavaScript (Mozilla "Rhino")
    - Java
  - Allows to
    - Maintain scripts
      - Create, edit, remove scripts
      - Supports the OOo locations: user, share, application

OOo Scripting Framework, 2

- Allows to (continued)
  - Dispatch scripts
    - Arguments (IN, OUT, IN/OUT)
    - Returns script's return value, if any
  - Extend OOo with new scripting engines
    - Need for interfacing with Java
      - Scripting languages implemented in Java
        - Extremely easy to interface
      - Scripting languages implemented e.g. in C++
        - Need to use JNI (Sun's Java Native Interface)
        - To ease coding use ASF's BSF 2.4

Example, 1

- Simple OOo Basic script, adds two arguments
- Function in an OOo Basic library ("application")
- Code (cf. Frysak)


IT Works

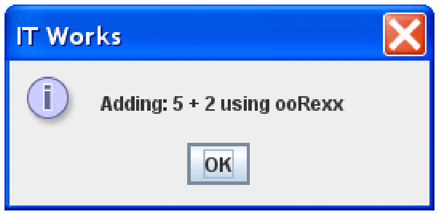
Adding: 7 + 1 using Star Basic

```
Function addition(arg1, arg2 as Integer) as Integer

  ' view that we are currently using Star Basic
  MsgBox("Adding: " & arg1 & " + " & arg2 & " using Star Basic", 64, "IT Works")

  ' return calculation
  ' to calculate make sure the parameters are Integers
  addition = CInt(arg1) + CInt(arg2)

End Function
```

Example, 2

- Simple ooRexx script, adds two arguments
- Stand-alone program, located in "user"
- Code (cf. Frysak)

```
-- a small test macro to test the x_RunMacro.rex macro
parse arg arg1, arg2

info = "Adding:" arg1 "+" arg2 "using ooRexx"
.bsf.dialog~messageBox(info, "IT Works", "information")

return arg1+arg2

::requires BSF.CLS
```

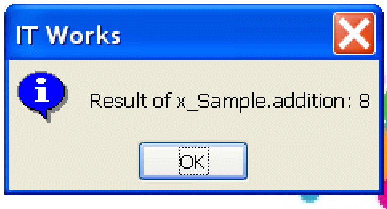


Adding: 5 + 2 using ooRexx

Example, 3

- Invoking OOo Basic and ooRexx scripts (Basic)

```
Sub RunMacro

' create the Dispatcher service
oDisp = createUnoService("com.sun.star.frame.DispatchHelper")

' prepare parameters as array
Dim a(1) As New com.sun.star.beans.PropertyValue
a(0).Name = "arg1" : a(0).Value = 7
a(1).Name = "arg2" : a(1).Value = 1

' macro URL to addition function above
sMacroURL = "vnd.sun.star.script:BakkMacros_Basic.x_Sample.addition?" & _
            "language=Basic&location=application"

' call addition function
r = oDisp.executeDispatch(StarDesktop, sMacroURL, "", 0, a())

' view result
MsgBox("Result of x_Sample.addition: " & r.result, 64, "IT Works")

' ... continued on next page ...
```

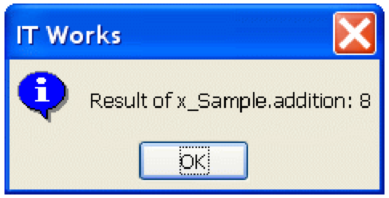

IT Works

Result of x_Sample.addition: 8

Example, 4

- Invoking OOo Basic and ooRexx scripts (Basic)

```
'... continued from previous page ...

' macro URL to x_Sample.rex
sMacroURL = "vnd.sun.star.script:BakkMacros.x_Sample.rex?" & _
            "language=ooRexx&location=user:uno_packages/BakkMacros.oxt"

' call x_Sample.rex and use the same parameters again
r = oDisp.executeDispatch(StarDesktop, sMacroURL, "", 0, a())

' show result
MsgBox("Result of x_Sample.addition: " & r.result, 64, "IT Works")

End Sub
```

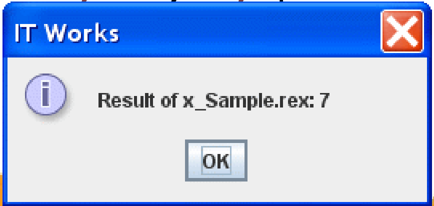

IT Works

Result of x_Sample.addition: 8

Example, 5

- Invoking OOo Basic and ooRexx scripts (ooRexx)

```
x_ScriptContext = uno.getScriptContext()          -- get the script's context
x_Context = x_ScriptContext~getComponentContext -- get component context
x_Desktop = x_ScriptContext~getDesktop          -- get desktop (an XDesktop)

-- create DispatchHelper service and query its interface
x_MSF = x_Context~getServiceManager~XMultiServiceFactory
x_DispHlp = x_MSF~createInstance("com.sun.star.frame.DispH")~XDispatchHelper

x_DispatchProvider = x_Desktop~XDispatchProvider -- get dispatch provider interface

-- prepare parameters
parameters = uno.CreateArray(.UNO~PropertyValue, 2)
parameters[1] = uno.createProperty("arg1", 5)
parameters[2] = uno.createProperty("arg2", 2)

-- define ooRexx dispatch target
MacroURL = "vnd.sun.star.script:BakkMacros.x_Sample.rex?" || -
           "language=ooRexx&location=user"
-- make dispatch call
r = x_DispHlp~executeDispatch(x_DispatchProvider, MacroURL, "", 0, parameters)
msg = "Result of x_Sample.rex:" r~result
.bsf.dialog~messageBox(msg, "IT Works", "information")
-- ... continued on next page ...
```


IT Works

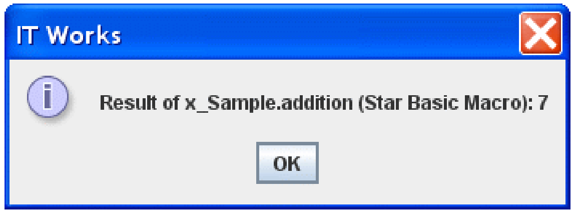
Result of x_Sample.rex: 7

OK

Example, 6

- Invoking OOo Basic and ooRexx scripts (ooRexx)

```
-- ... continued from previous page ...


-- define Star Basic dispatch target
MacroURL = "vnd.sun.star.script:BakkMacros.x_Sample.addition?" || -
           "language=Basic&location=application"

-- make dispatch call
r = x_DispHlp~executeDispatch(x_DispatchProvider, MacroURL, "", 0, parameters)
msg = "Result of x_Sample.addition (Star Basic Macro):" r~result
.bsf.dialog~messageBox(msg, "IT Works", "information")

::requires UNO.CLS
```



Result of x_Sample.addition (Star Basic Macro): 7

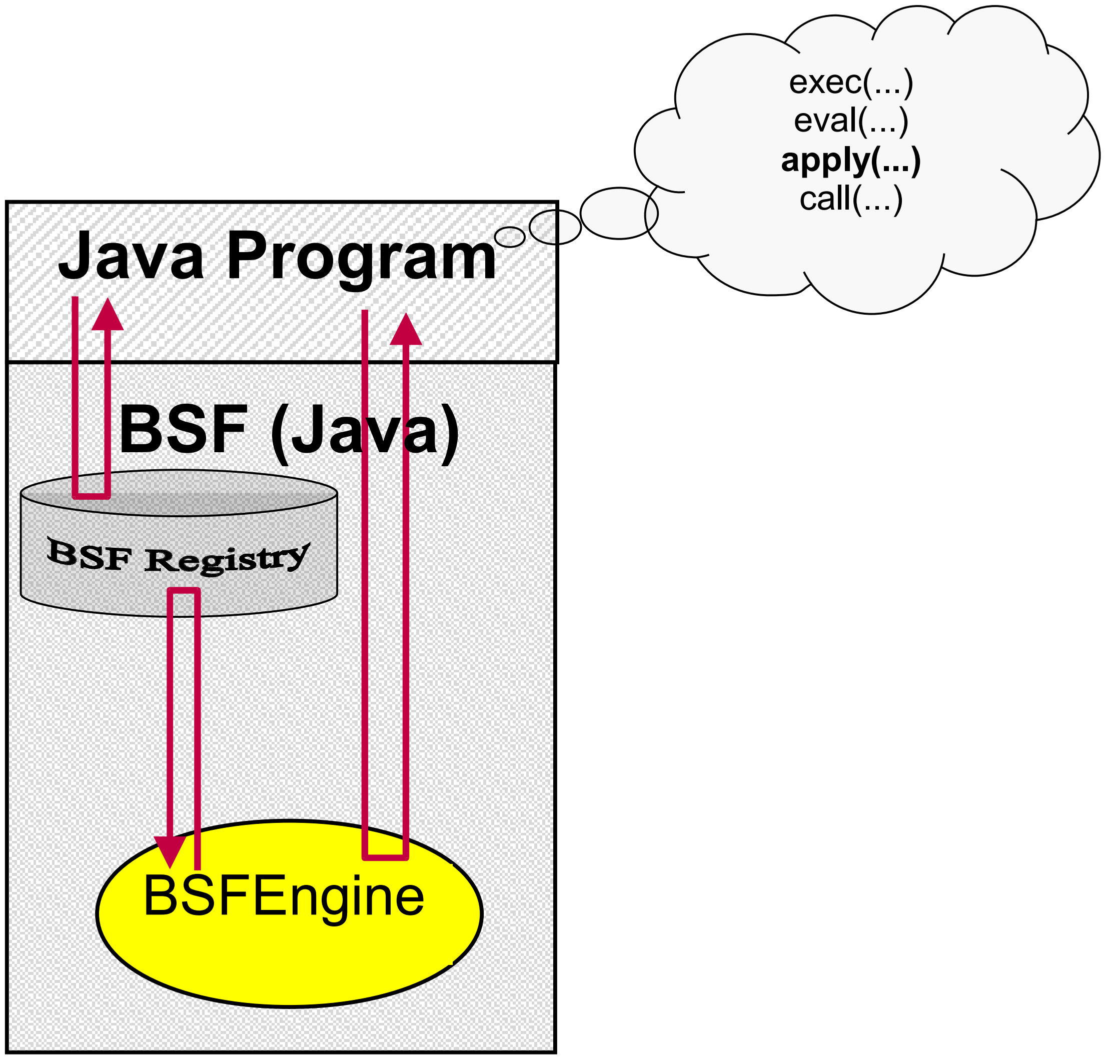
Bean Scripting Framework

- Developed at IBM as an opensource project
  - Purpose: deploying scripting languages in JSPs
- Donated to the Apache Software Foundation
  - Part of ASF's Jakarta project
    - http://jakarta.apache.org/bsf
  - Used in many other projects within ASF
    - e.g. ant, xerces, etc.
- A Java framework
  - Eases access to scripting languages from Java
  - Eases (reflective) interaction with Java from scripts

Bean Scripting Framework

- Quite a few BSF scripting engines available
  - Usually implemented in Java, e.g.
    - Groovy, Groovy Monkey
    - Jacl (TcL)
    - JavaScript (Rhino)
    - JLog (PROLOG)
    - JRuby (Ruby)
    - Jython (Python)
    - ObjectScript
    - …
  - Can be *easily* added to OOo via BSF!

BSF Architecture



BSF, Executing a Script, 1

- Executing a JavaScript script

```
import org.apache.bsf.*;   // import BSF support

public class TestSimpleExecJavaScript {

  public static void main (String[] args) throws java.io.IOException
  {
    try
    {
      BSFManager mgr    = new BSFManager ();
      BSFEngine  engine = mgr.loadScriptingEngine("javascript");
      String     code   = "java.lang.System.out.println(\"JavaScript was here!\")";
          // invoke the JavaScript script
      engine.exec ("javascript", 0, 0, code);
    }
    catch (BSFException e)
    {
      e.printStackTrace();
    }
  }
}
```

BSF, Executing a Script, 2

- Executing an ooRexx script

```
import org.apache.bsf.*;   // import BSF support

public class TestSimpleExecRexx {

  public static void main (String[] args) throws java.io.IOException
  {
    try
    {
      BSFManager mgr    = new BSFManager ();
      BSFEngine  engine = mgr.loadScriptingEngine("rexx");
      String     code   = "SAY 'Rexx was here!'";
          // invoke the Rexx script
      engine.exec ("rexx", 0, 0, code);
    }
    catch (BSFException e)
    {
      e.printStackTrace();
    }
  }
}
```

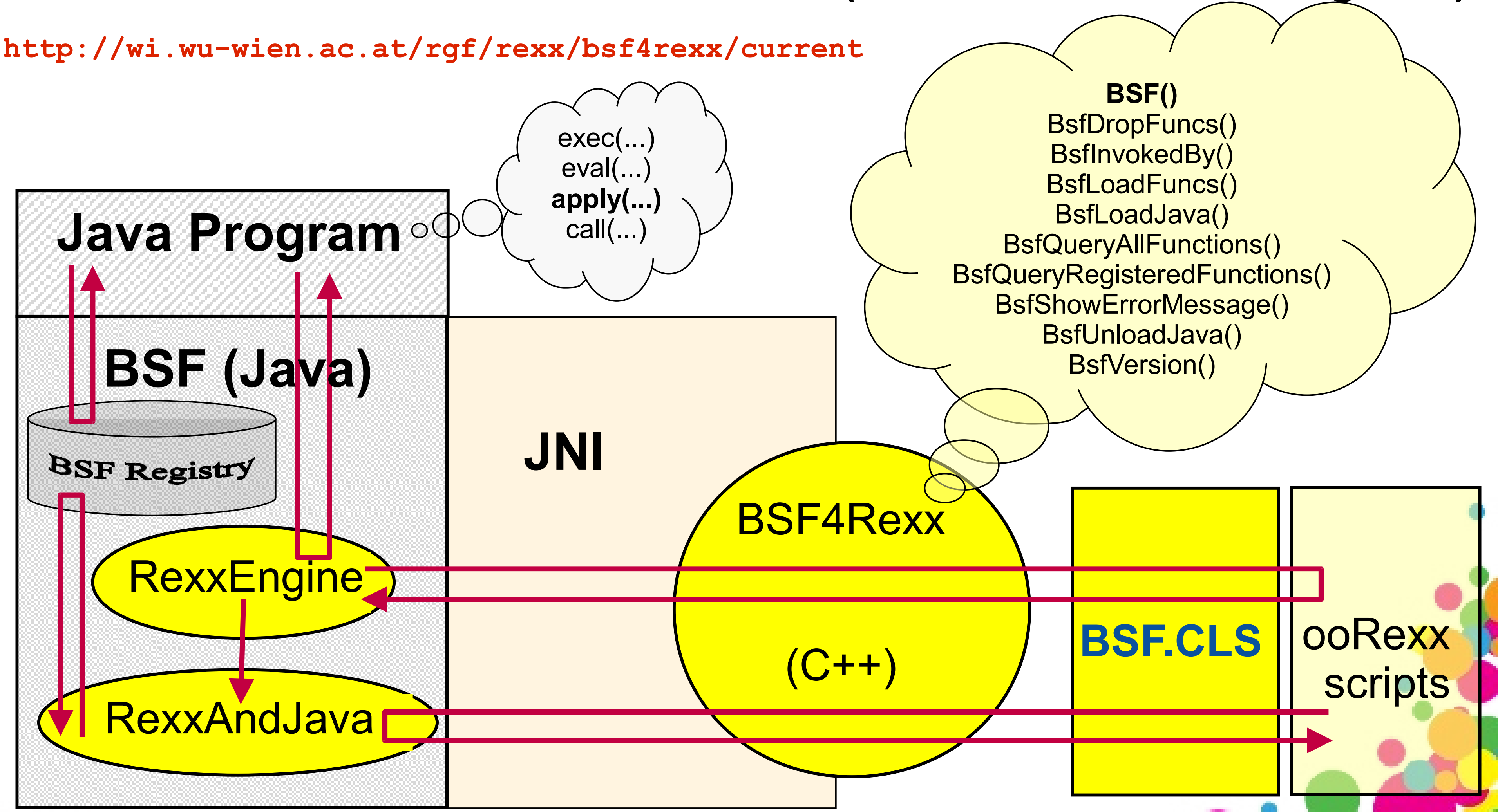
BSF4Rexx, 1

- Example: ooRexx (Open Object Rexx)
  - A free, dynamic and opensource scripting language, perfect for EUD (end-user development)
    - http://www.ooRexx.org
  - Implemented in C++, *not* Java!
  - Creating a BSF engine for ooRexx
    - JNI (Sun's Java Native Interface)
    - Add (reflective) support on the Java side, e.g.
      - Loading Java classes, creating instances, dispatching messages
      - Creating event adapters on the fly (Java bytecode) ...
    - If usable via BSF, it becomes easy to deploy it
      - ➔ ooRexx (a non-Java language) can use all of Java!

BSF4Rexx, 2

- Architecture of "BSF4Rexx" (ooRexx BSF engine)

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



BSF4Rexx, 3

- "RexxAndJava"
  - Support for non-Java programs to
    - Load Java classes, create Java instances, dispatch messages, marshall arguments and return values, etc.
  - Can be used by any other BSF engine!
    - Hence not restricted to ooRexx!
  - *If you have a non-Java language that you wish to add to OOo, use this existing infrastructure for interfacing with Java (to ease your life considerably ☺) !*
    - If you have any questions, please approach me or use the mailing list "dev@api.openoffice.org" or the newsgroup "news:comp.lang.rexx"

Adding a New Engine, 1

- Outlining the process to add a new engine
  - OOo scripting framework (Java)
  - Using BSF 2.4 (alternatively: BSF 3.0 / JSR-223)
- Reference implementation using BSF
  - "BSF4Rexx"
    - http://wi.wu-wien.ac.at/rgf/rexx/bsf4rexx/current/
    - BSF4Rexx-apache-bsf-source.jar
  - Used OOo's BeanShell implementation as a template, cf.
    - com/sun/star/script/framework/provider/beanshell/

Adding a New Engine, 2

- Module "scripting"
  - http://framework.openoffice.org/scripting/
    - Homepage of the OOo scripting framework
    - Specifications
  - svn co svn://svn.services.openoffice.org/ooo/trunk/scripting scripting
    - Checking out the entire OOo scripting module
  - scripting/java/com/sun/star/script/framework/provider
    - Root for OOo script engines
    - Utility/helper programs for script engines
    - "beanshell", "java", "javascript" script engines

Adding a New Engine, 3

- Module "scripting"
  - Java 1.5 or higher (since OOo 3.0, October 2008)
  - Define a name for your engine, e.g. "ABC"
    - Create a directory of "abc" (lowercase!) in "provider"
    - Define an extension for your language, e.g. "A"
  - Use "beanshell" implementation as a blueprint
    - Copy all "beanshell/*" files to "abc/"
    - Rename "template.bsh" to "template.A"
      - Change template script code to your language

Adding a New Engine, 4

- Module "scripting" (continued)
  - Rename "ScriptEditorForBeanShell.java" to "ScriptEditorForABC.java"
    - Adapt all occurrences of "BeanShell" and "bsh" in this program to match your engine's names (i.e. "ABC", "A") and functionality
  - Rename "ScriptProviderForBeanShell.java" to "ScriptProviderForABC.java"
    - Adapt all occurrences of "BeanShell" and "bsh" in this program to match your engine's names (i.e. "ABC", "A") and functionality

Adding a New Engine, 5

- Module "scripting" (continued)
  - Define a name for your engine, e.g. "ABC"
    - Create a directory of "abc" (lowercase!) in "provider"
    - Define an extension for your language, e.g. "A"
  - Use "beanshell" implementation as a blueprint
    - Copy all "beanshell" files to "abc"
    - Rename "template.bsh" to "template.A"
      - Change template script code to your language
    - Rename "ScriptEditorForBeanShell.java" to "ScriptEditorForABC.java"
    - Rename "ScriptProviderForBeanShell.java" to "ScriptProviderForABC.java"

Adding a New Engine, 6

- Module "scripting" (continued)
  - Create a manifest file, e.g.

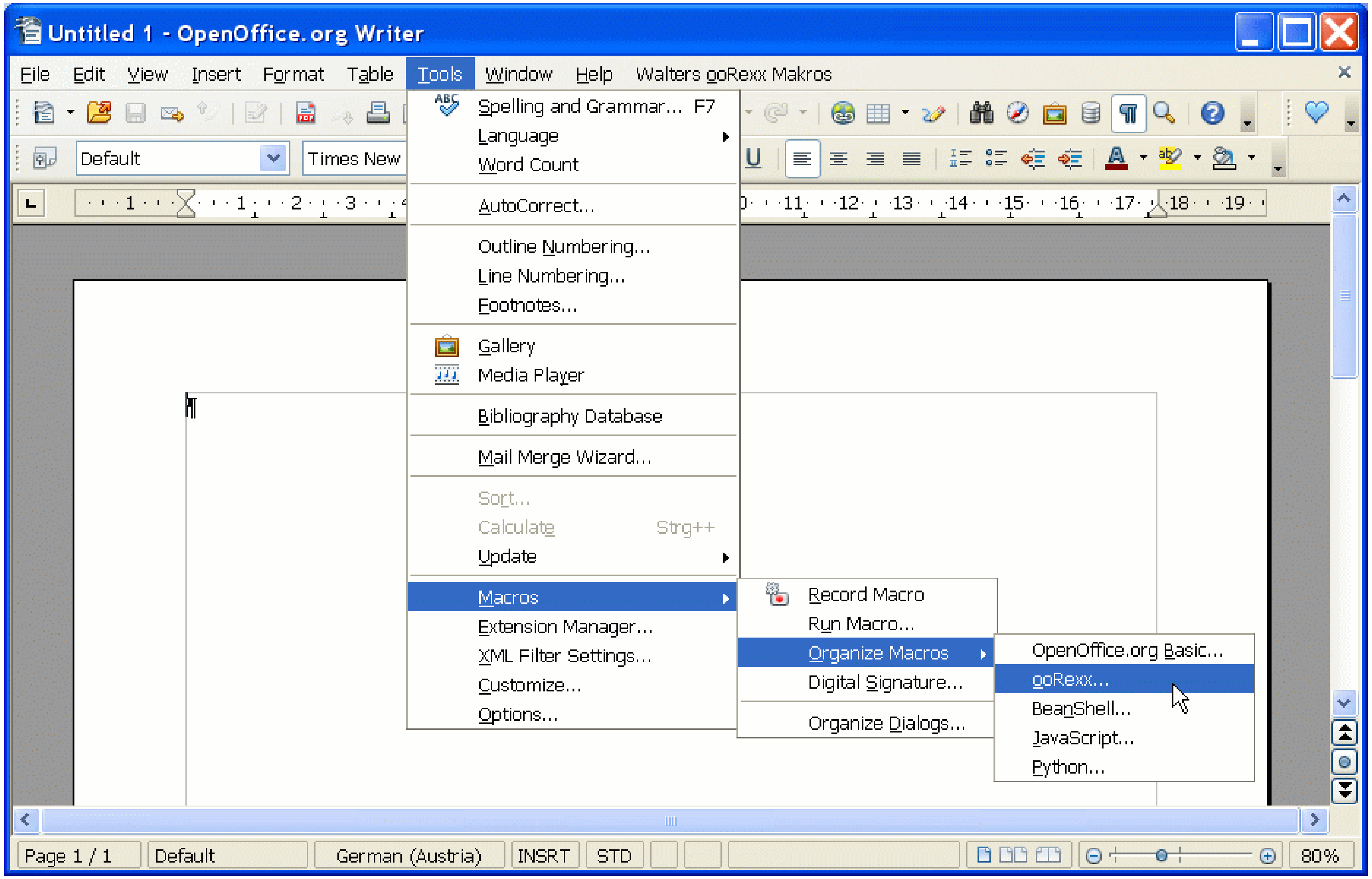
```
Manifest-Version: 1.0
RegistrationClassName:
com.sun.star.script.framework.provider.abc.Scri
 ptProviderForABC
Created-By: YourName
Specification-Title: ABC to/from UNO Bridge
Specification-Version: 0.999
Specification-Vendor: YourName
Implementation-Title: org.abc.uno
Implementation-Version: 0.999
Implementation-Vendor: YourName
```

Adding a New Engine, 7

- Create a Java archive
  - Name: "ScriptProviderForABC.jar"
  - Entries
    - META-INF/MANIFEST.MF
    - com/sun/star/script/framework/provider/abc/*
  - Add additional resources as needed
    - If using additional Java archives add them to the manifest
    - Cf. OOo documentation on packaging and deploying
- Use the OOo package/extensions manager to deploy

OOo with an Added Scripting Language

OpenOffice.org Conference 2008 Beijing



BSF 3.0/JSR-223, 1

- JSR-223
  - Defined the Java scripting framework
    - Package "javax.script"
  - Introduced with Sun's Java 6
  - *Only available for Java 6 or higher!*
- Apache's BSF 3.0
  - Opensource Implementation of JSR-223
    - Implements the package "javax.script"
    - Available for Java 1.4 or higher!
  - Part of ASF's Harmony ("Apache's Java")

BSF 3.0/JSR-223, 2

- OOo Scripting engine using JSR-223
  - Supply BSF 3.0 with your OOo engine
    - OOo engine can run on pre Java 6 installations!
  - On Java 6 or higher, the Java 6 scripting framework will be used instead of BSF 3.0
    - As "javax.script" is part of the Java runtime environment, it will get picked up before any other package!
  - OOo engine may *in addition* use BSF 2.4
    - Taking advantage of BSF 2.4 support
    - E.g. taking advantage of "RexxAndJava" for non-Java languages

Roundup & Outlook, 1

- Adding a new scripting language to OpenOffice.org is **easy**!
- OOo scripting framework is implemented in Java
  - Use Java frameworks/infrastructures to ease implementation
  - ASF's BSF 2.4 a good choice
    - Reference implementation available ("BSF4Rexx")
    - Take advantage of "RexxAndJava", if necessary
  - ASF's BSF 3.0 / JSR-223 a good choice
    - Deploy with the BSF 3.0 package for pre Java 6

Roundup & Outlook, 2

- Now that you have added a new scripting language
  - Make it easy to use UNO using it !
    - Add support modules to reduce the coding needs, e.g.
      - Ease querying interfaces
      - Ease interacting with properties
      - Ease reflection of UNO objects
      - ...
  - This evening's presentation "Creating/Devising Specific OpenOffice.org Support for Dynamic Scripting Languages" concentrates on this issue

OpenOffice.org
Conference 2008 Beijing

Thanks!

凝聚全球力量 绽放开源梦想

www.OOobeijing2008.com