

Automatisierung von Java Anwendungen (7)

Bean Scripting Framework (BSF), 1

Bean Scripting Framework (BSF), "BSF4Rexx",
"JavaDoc", Java als riesige Funktionsbibliothek
für (Object) Rexx, Wiener Version von BSF4Rexx

Prof. Dr. Rony G. Flatscher

Bean Scripting Framework (BSF)

- Bean Scripting Framework
 - Opensource Projekt der Firma IBM
 - Z.B. in IBM Produkten wie WebSphere (JSP)
 - Herbst 2003
 - Code an Jakarta-Projekt von Apache ausgehändigt
 - Z.B. in "ant", "xerces"
 - BSF-Release 2.4.0: Oktober 2006
 - Rahmenwerk (Framework), um Java-Programmen den Aufruf von Skripten sehr einfach zu ermöglichen
 - Definiert Schnittstellen, um von Skriptprogrammen aus mit Java-Objekten zu interagieren
 - Java Objekte werden in einem Repository auf der Java-Seite gespeichert

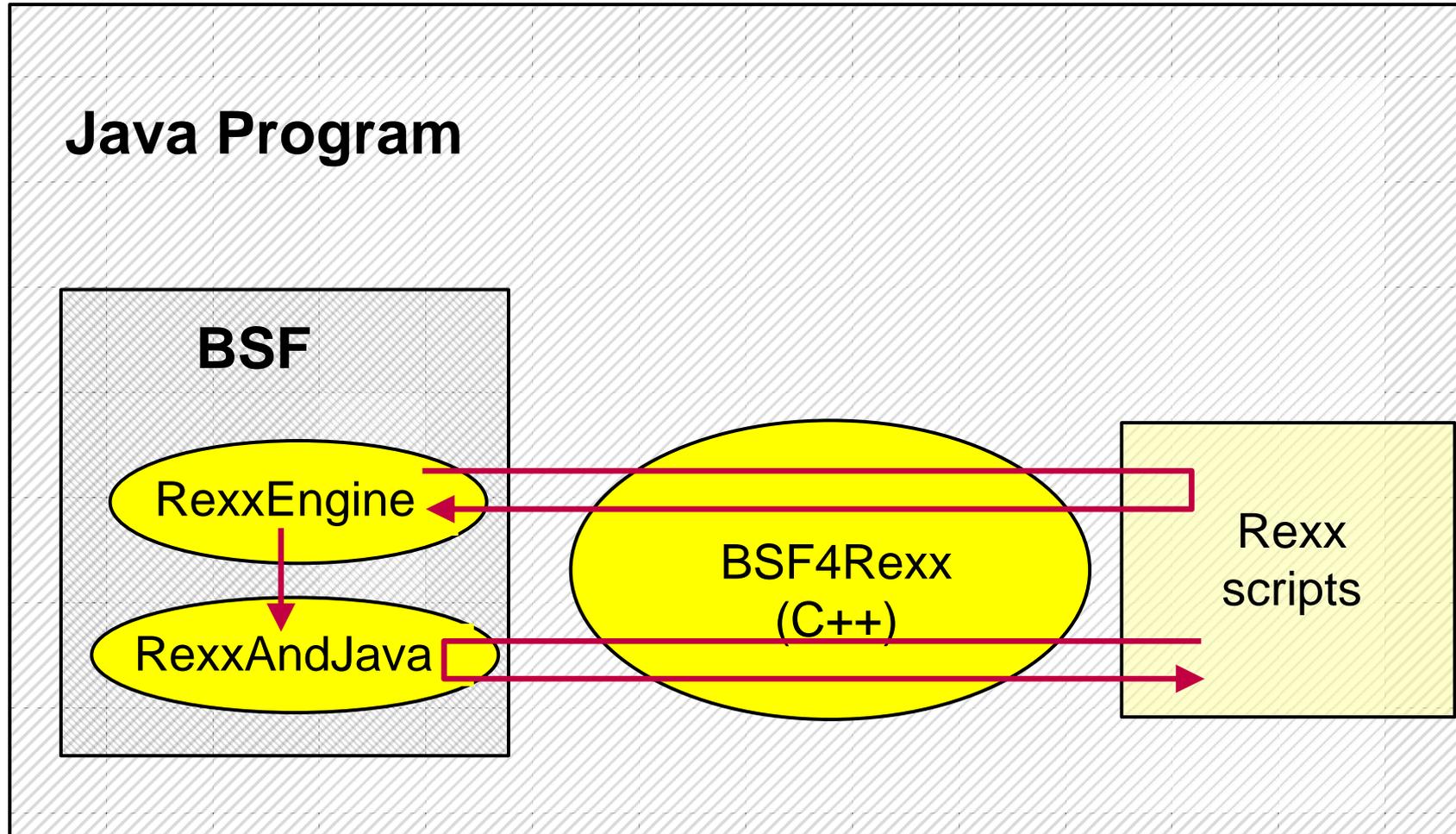
BSF4Rexx

Essener Version, 1

- Fügt Unterstützung von Rexx zu BSF hinzu
 - Ursprünglich im Rahmen eines Seminars als Proof of Concept 2000/01 an der Universität Essen entwickelt
 - Ermöglicht es Java-Programmen, Rexx-Programme aufzurufen
 - Aufgerufene Rexx-Skripte können mit Java-Objekten interagieren
 - Verkapselung von Java Klassen in Object Rexx Klassen
 - **"BSF.CLS"**
 - Object Rexx Klassen, die mit BSF4Rexx interagieren
 - Ermöglicht das Senden von Object Rexx Nachrichten an Java-Objekten
 - Ermöglicht die Benutzung von Java-Feldern ("Arrays") wie wenn es sich um Object Rexx Felder ("Arrays") handeln würde

BSF4Rexx

Essener Version, 2



- Einige Schlussfolgerungen
 - Rexx-Programme müssen über Java aufgerufen werden
 - BSF ermöglicht auch Kommandozeilen-Aufruf über Java
 - `"org.apache.bsf.Main"` ("`com.ibm.bsf.Main`")
 - Argument kann der Name einer Datei sein, in der ein Skriptprogramm gespeichert ist
 - Ermöglicht es damit Java-Programmierern, Rexx und Object Rexx als Skriptsprache zu verwenden
 - Damit können diese einfach zu erlernenden Skriptsprachen dazu benutzt werden, Java-Programme zu automatisieren bzw. fernzusteuern!
 - Bei Interaktion mit Java: strikte Typisierung der Argumente!

BSF4Rexx

Beispiel: Aufruf von (Object) Rexx von Java aus

```
import org.apache.bsf.*; // BSF support
import java.io.*;        // exception handling

public class TestSimpleExec {

    public static void main (String[] args) throws IOException
    {
        try {
            BSFManager mgr = new BSFManager ();
            String rexxCode = "SAY 'Rexx was here!'";

            mgr.exec ("rexx", "debug infos", 0, 0, rexxCode);

        } catch (BSFException e) { e.printStackTrace(); }
    }
}
```

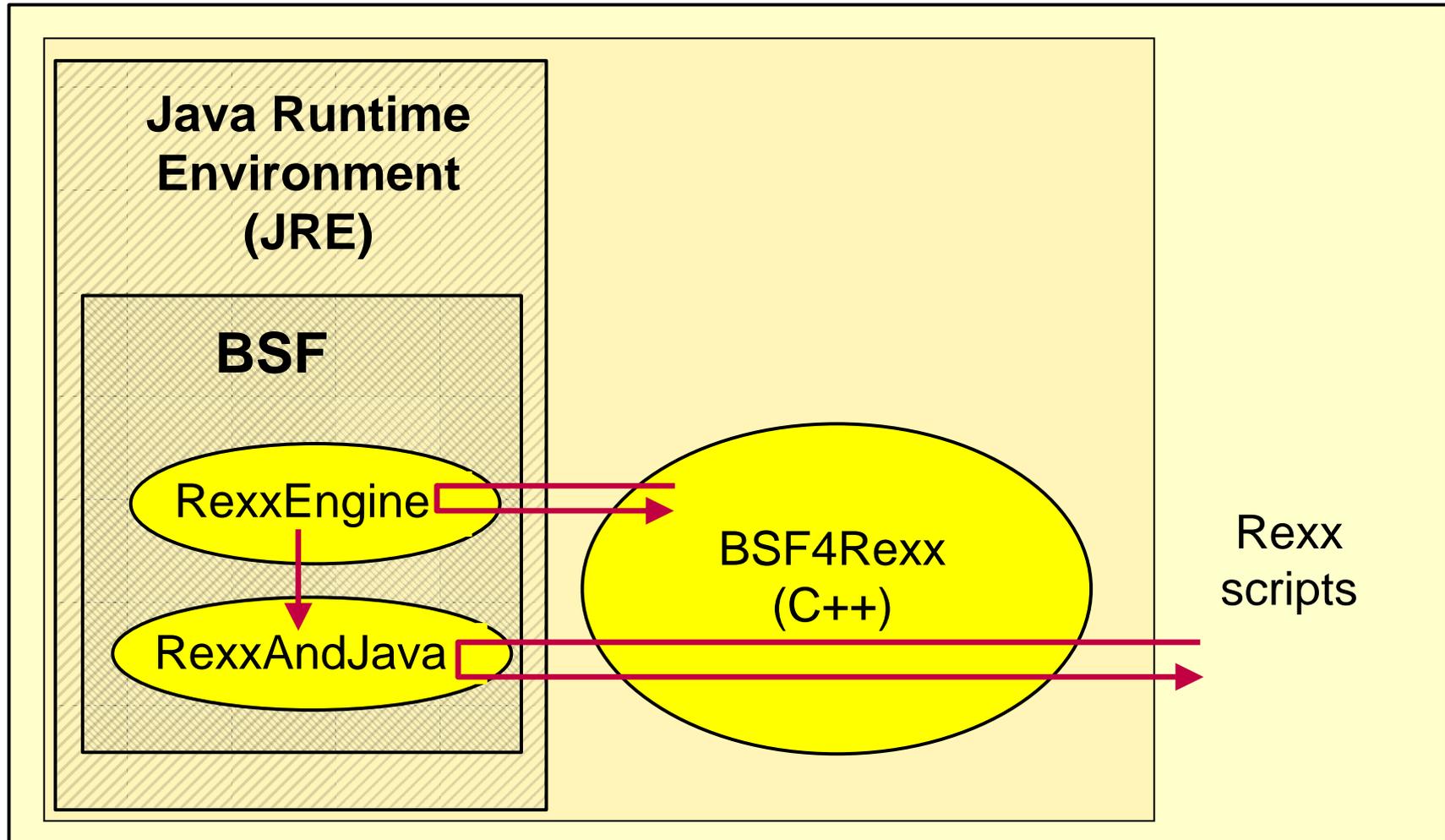
BSF4Rexx

Augsburger Version, 1

- Während der Lehrstuhlvertretung an der Universität Augsburg entstanden
 - 2002-2003
- Vollständige Überarbeitung der Essener Version
 - 100 % kompatibel zur Essener Version
- Fügt unter anderem die Fähigkeit hinzu, bei Bedarf Java von (Object) Rexx aus zu starten
 - Keine Notwendigkeit mehr, ein Java Experte zu sein
 - Lediglich der Umgang mit der Java-HTML Dokumentation notwendig
 - *Jeder* kann das!
<http://java.sun.com/api>
 - Die gesamte Dokumentation der Java-Klassen sind in Form von HTML-Dateien über das WWW verfügbar!!
- Ermöglicht die Benutzung/das Ansteuern von *jeder* Java-Klasse und von *jedem* Java-Objekt!

BSF4Rexx

Augsburger Version, 2



BSF4Rexx

Augsburger Version, 3

- Benutzt Mark Hessling's "RexxTrans"
 - Eine (stark fehler- und funktionsbereinigte) kompilierte DLL von "**BSF4Rexx.cc**" für alle verfügbaren und von RexxTrans unterstützte Rexx Interpreter
 - Fügt folgende externe Rexx-Funktionen für BSF4Rexx hinzu
 - Starten und Niederfahren der JVM
 - "**BSFLoadJava**", "**BSFUnloadJava**"
 - Abfragen aller bzw. aller registrierten externen Rexx-Funktionen
 - "**BSFQueryAllFunctions**", "**BSFQueryRegisteredFunctions**"
 - Abfragen, wie das (Object) Rexx Programm gestartet wurde, via Java oder direkt via Rexx
 - "**BSFInvokedBy**"
 - Abfragen der Version der externen Rexx-Funktionsbibliothek "**BSF4Rexx.cc**"
 - "**BSFVersion**"

- Vollständige Überarbeitung der Java-seitigen Unterstützung von BSF4Rexx
 - Z.B. keine Restriktionen mehr für Java-Array-Objekte
 - Z.B. drei Prioritätsstufen für Java-Adapter-Nachrichten
 - Z.B. Vorregistrieren der wichtigsten Java-Klassen
 - ...

BSF4Rexx

Augsburger Version, 5

- "BSFRegistry"
 - Verwaltung erfolgt auf der Java-Seite
 - Feld der Java-Klasse "**BSFManager**"
 - Ein Verzeichnis jener Java-Objekte, die man von Java und von (Object) Rexx aus ansprechen kann
 - Adressierung erfolgt mit Hilfe einer eindeutigen Zeichenkette, Groß- und Kleinschreibung ist hierbei signifikant!
 - Kann unter anderem zur Koppelung benutzt werden
 - Java- und (Object) Rexx-Programme
 - (Object) Rexx- und (Object) Rexx-Programme
 - Beliebige BSF-Scripting-Engines untereinander oder mit Java

- Frühjahr 2006
 - Entwicklungszeit: 2003-2006
 - Voll kompatibel zur Augsburger Version, Key-Feature:
 - **Keine** Java-Typangaben mehr notwendig!
 - Wenn notwendig dann "strict"-Versionen der entsprechenden Funktionen/Methoden verwenden
- "BSF4Rexx"-Paket
 - Home: <<http://wi.wu-wien.ac.at/rgf/rexx/bsf4rexx/current/>>
"readmeBSF4Rexx.txt", "readmeOOo.txt" lesen !
 - Geplant: <<https://sourceforge.net/projects/bsf4rexx>>

Ein Beispiel

Java Klasse "XyzType", 1

```
public class XyzType    // example class for demonstrating BSF4Rexx
{
    // constructors of this class (same name as class!)
    public XyzType () {           // constructor without arguments
        counter=counter+1;       // increase counter
    }

    public XyzType (String initialValue) { // constructor with argument
        this();                  // invoke (call) constructor without argument
        info=initialValue;      // save initial value
    }

    // keyword "static": class fields (attributes) and class methods
    static public int counter=0; // field: will count # of instances

    // instance fields (attributes) and instance methods
    private String info = null;   // field: no value per default

    public String getInfo () { // accessor (getter) method (function)
        return info;           // return whatever "info" points to
    }

    public void setInfo (String aValue) { // setter method (function)
        info=aValue;           // save received value with "info"
    }
}
```

Ein Beispiel

Java Klasse "XyzType", 2

- Speichern der Datei unter "**XyzType.java**"
 - Achtung auf Groß- und Kleinschreibung!
 - Groß- und Kleinschreibung ist unter Java signifikant!
- Kompilierung mit Hilfe des Java-Compilers, z.B.
`javac XyzType.java`
- Erzeugen der standardisierten HTML Dokumentation
`javadoc XyzType.java`

Ein Beispiel

Java Klasse "XYZType", 3

The screenshot shows a Microsoft Internet Explorer window titled "XYZType - Microsoft Internet Explorer". The browser's menu bar includes "Datei", "Bearbeiten", "Ansicht", "Favoriten", "Extras", and "?". The main content area displays the following information:

Class XYZType

java.lang.Object
|
+-**XYZType**

```
public class XYZType  
extends java.lang.Object
```

Field Summary

static int	counter
------------	-------------------------

Constructor Summary

[XYZType](#) ()

[XYZType](#) (java.lang.String initialValue)

Method Summary

java.lang.String	getInfo ()
void	setInfo (java.lang.String aValue)

The browser's status bar at the bottom shows "Fertig" on the left and "Arbeitsplatz" on the right.

Beispiel "code1-oo.rex"

Ein Object Rexx Programm, 1

```
javaClass = "XYZType" /* determine Java class to use */
say "value of static field 'counter'=" || .bsf~bsf.getStaticValue(javaClass, "counter")
say

o=.BSF~new(javaClass) /* create an instance of "XYZType" */
say "o:" o
say "# 1:" o~getInfo /* get the value via the getter method */
o~setInfo("Hello, from Rexx...")
/* o~bsf.invokeStrict("setInfo", "String", "Hello, from Rexx...") */
say "# 2:" o~getInfo /* get the value via the getter method */
say "value of static field 'counter'=" || o~bsf.getFieldValue("counter")
say

/* create a second Java object */
say "creating another instance of XYZType, this time with an initial value..."
/* create an instance of "XYZType" and supply a string value */
o=.BSF~new(javaClass, "Hi, RexxLA!")
/* tmpClass=.bsf~import(javaClass)
o=.tmpClass~newStrict("String", "Hi, RexxLA!") */
say "o:" o
say "# 3:" o~getInfo /* get the value via the getter method */
say "value of static field 'counter'=" || o~bsf.getFieldValue("counter")

::requires BSF.CLS -- get Java support for ooRexx
```

Beispiel "code1-oo.rex", Aufruf Ein (klassisches) Rexx Programm, 2

- Über Java

```
java com.ibm.bsf.Main -mode exec -lang rex -in code1-oo.rex
```

```
java org.apache.bsf.Main -mode exec -lang rex -in code1-oo.rex
```

- Über Java mit Hilfe einer/s Batch-Datei/Shell-Skripts

```
rexj code1-oo.rex
```

- Oder über Rexx

```
rex code1-oo.rex
```

- Oder über Rexx, wenn Erweiterung ".rex" ausreicht

```
code1-oo.rex
```

Beispiel "code2.rex", Ausgabe Ein Object Rexx Programm, 3

```
value of static field 'counter'=0
```

```
o: XYZType@15f5897
```

```
# 1: The NIL object
```

```
# 2: Hello, from Rexx...
```

```
value of static field 'counter'=1
```

```
creating another instance of XYZType, this time with an initial value...
```

```
o: XYZType@f9f9d8
```

```
# 3: Hi, RexxLA!
```

```
value of static field 'counter'=2
```

"Bean Scripting Framework" (BSF)

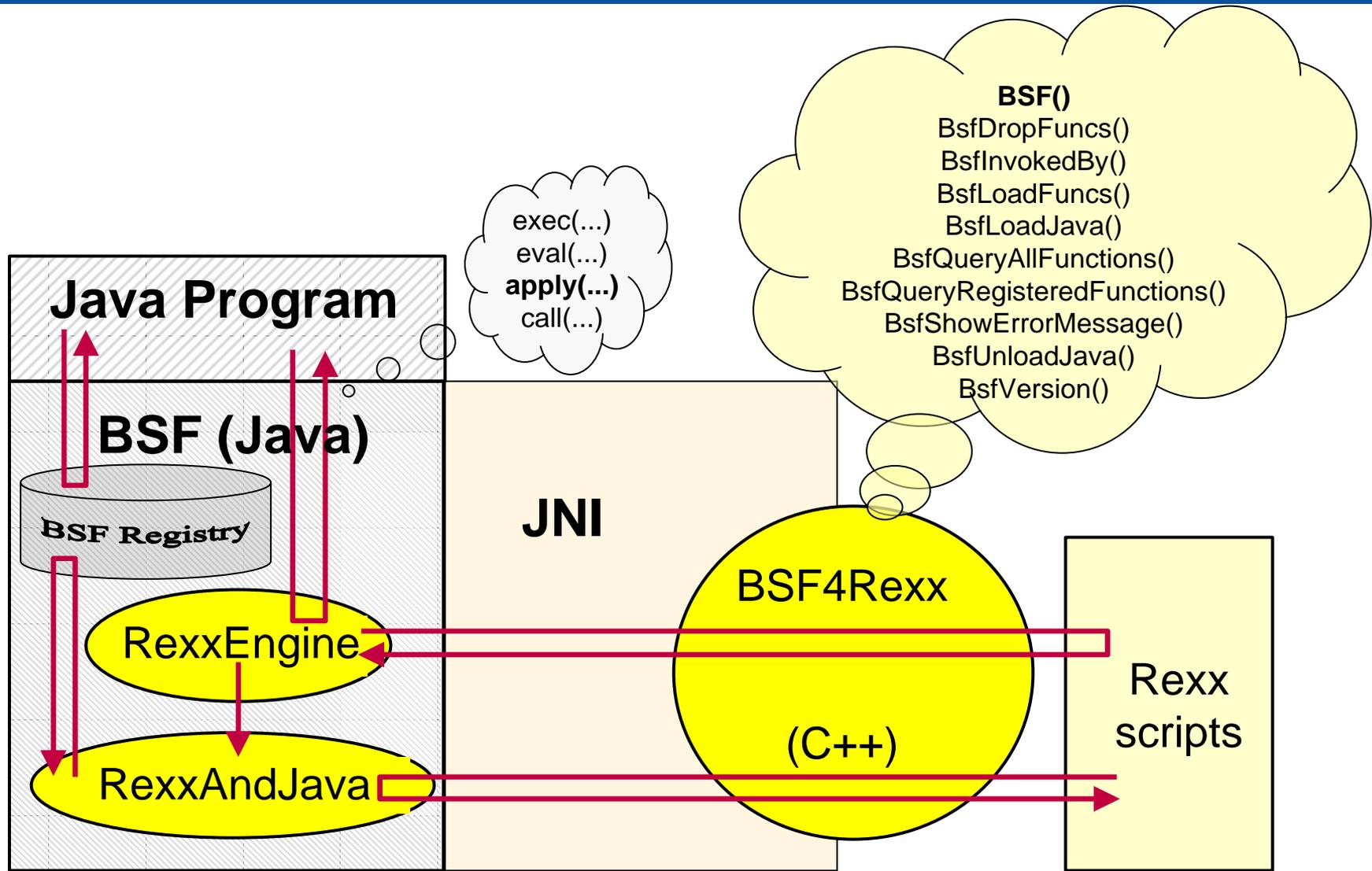
- Bean Scripting Framework
 - Wiederholung
 - Aktueller Stand der "Wiener Version" von BSF4Rexx
 - Detaillierte Auflistung der BSF-Funktionen
 - Detaillierte Auflistung der Methoden der Java Proxy-Klasse ".BSF"
 - Beispiele

- Bean Scripting Framework
 - Ein Java-Rahmenwerk (Framework), mit dessen Hilfe es sehr einfach ist, Skripte in anderen Sprachen aufzurufen
 - Z.B. JavaScript, NetRexx
 - Ursprünglich von IBM als Open-source-Projekt
 - Teil von IBM's WebSphere, um Skripte in Java Server Pages (JSP) einbetten zu können
 - Im Herbst 2003 jakarta.apache.org übertragen
 - Wird z.B. in [ant](#), [xerces](#) eingesetzt

- BSF mit einer Rexx "Engine"
 - Ermöglicht das Benutzen von Rexx über BSF
 - Jedes Java-Programm kann Rexx aufrufen
 - Rexx-Skripte können mit Java Objekten kommunizieren
 - Ermöglicht das Benutzen von Java als riesige externe Rexx-Funktionsbibliothek
 - Alle als öffentlich definierte Methoden und Felder von Java-Objekten bzw. Java-Klassenobjekten können von Rexx aus benutzt werden
 - Wenn notwendig, kann Java auch von Rexx aus gestartet werden
 - Wiener Version
 - Entwickelt 2003-2006
 - Weiterentwicklung der Augsburgener Version

BSF4Rexx Architektur

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



BSF4Rexx (BSFRegistry)

Vor-registrierte Java Objekte

- 1) "Array.class"
- 2) "Class.class"
- 3) "Method.class"
- 4) "Object.class"
- 5) "String.class"
- 6) "System.class"
- 7) "Thread.class"
- 8) "**B**oolean.class"
- 9) "*boolean.class*"
- 10) "**B**yte.class"
- 11) "*byte.class*"
- 12) "**C**haracter.class"
- 13) "*char.class*"
- 14) "**D**ouble.class"
- 15) "*double.class*"
- 16) "**I**nteger.class"
- 17) "*int.class*"
- 18) "**L**ong.class"
- 19) "*long.class*"
- 20) "**F**loat.class"
- 21) "*float.class*"
- 22) "**S**hort.class"
- 23) "*short.class*"
- 24) "**V**oid.class"
- 25) "*void.class*"

- (1) call BSF "addEventListener", beanName, eventSetName, filter, eventText
- (2) call BSF "addEventListenerReturningEventInfos", beanName, eventSetName, filter, eventText, sendBackData
- (3) x=BSF("arrayAt", arrayBeanName, idx0 [, idx1]...)
x=BSF("arrayAt", arrayBeanName, intArrayBean)
- (4) l=BSF("arrayLength", arrayBeanName)
- (5) call BSF "arrayPut", arrayBeanName, newValue, idx0 [, idx1]...
call BSF "arrayPut", arrayBeanName, intArrayBean
- (6) call BSF "arrayPutStrict", arrayBeanName, *typeIndicator*, newValue, idx0 [, idx1]...
call BSF "arrayPutStrict", arrayBeanName, *typeIndicator*, newValue, intArrayBean
- (7) a=BSF("createArray", componentType, dim0 [, dim1]...)
a=BSF("createArray", componentType, intArrayBean)
- (8) w=BSF("wrapArray", arrayObject)
- (9) res= BSF("exit" [, [retVal] [, time2wait in msec]])
- (10) v=BSF("getFieldValue", beanName, fieldName)
v=BSF("getFieldValueStrict", beanName, fieldName)
- (11) p=BSF("getPropertyValue", beanName, propertyName, index | ".NIL")
- (12) s=BSF("getStaticValue", JavaClassName, fieldName)
s=BSF("getStaticValueStrict", JavaClassName, fieldName)
- (13) res=BSF("invoke", beanName, methodName, arg1 [, arg2]...)
res=BSF("invokeStrict", beanName, methodName, *typeIndicator1*, arg1 [, *typeIndicator2*, arg2]...)
- (14) cl=BSF("loadClass", JavaClassName)

- (15) o=BSF("lookupBean", beanName)
- (16) t=BSF("pollEventText" [, timeout in msec])
- (17) call BSF "postEventText", eventText, priority
- (18) o=BSF("registerBean", beanName, beanType, arg1 [...])
- (19) o=BSF("registerBeanStrict", beanName, beanType, *typeIndicator1*, arg1 [, *typeIndicator2*, arg2]...)
- (20) v=BSF("setFieldValue", beanName, fieldName, newValue)
v=BSF("setFieldValueStrict", beanName, fieldName, [*typeIndicator*,] newValue)
- (21) v=BSF("setPropertyValue", beanName, propertyName, index | ".NIL", newValue)
v=BSF("setPropertyValueStrict", beanName, propertyName, index | ".NIL", *typeIndicator*, newValue)
- (22) call BSF "setRexxNullString", newString
- (23) call BSF "sleep", time2sleep in msec
- (24) str=BSF("unregisterBean", beanName)
- (25) v=BSF("version")
- (26) e=BSF("wrapArray", arrayBeanName)
- (27) e=BSF("wrapEnumeration", enumerationBeanName)

BSF4Rexx, Typisierungproblem, 1 "Strict"

- "Wiener" Version, neuer als die "Augsburger"
 - Beta Version
 - <http://wi.wu-wien.ac.at/rgf/rexx/bsf4rexx/>
 - Erlaubt das Auslassen von Typinformationen, die normalerweise für die Benutzung von Java Methoden mit Argumenten bzw. für das Setzen von Werten für Felder und Properties benötigt werden
 - Java ist eine streng typisierte Programmiersprache, Rexx nicht!
 - "strict" ermöglicht trotzdem die ausdrückliche Angabe von Typinformationen
 - Selten benötigt
 - Möglichkeit, dass Java-Methoden denselben Namen und dieselbe Anzahl an Argumenten, aber von unterschiedlichem Typ aufweisen

BSF4Rexx, Typisierungproblem, 2

"Strict"

- "Type indicators" gehen den einzelnen Argumenten voraus, wenn die BSF()-Subfunktionen das Wort "*Strict*" enthalten
 - invokeStrict, setFieldValueStrict, setPropertyValueStrict, bsfRegisterStrict, arrayPutStrict
- "Type indicators" bestehen aus folgenden Zeichenketten
 - "**BO**olean", "**BY**te", "**C**har", "**D**ouble", "**F**loat", "**I**nt", "**L**ong", "**O**bject", "**S**hort", "**S**tring"
 - Nur die fett und in Großbuchstaben dargestellten Teile sind notwendig
 - Java-Typinformationen aus der Java-HTML-Dokumentation
 - "**BO**olean", "**BY**te", "**C**har", "**D**ouble", "**F**loat", "**I**nt", "**L**ong", "**S**hort" sind sogenannte "primitive" Java Datentypen
 - "**S**tring"
 - "**O**bject" bezieht sich auf *jedes* Java-Objekt

– "BSF.CLS"

- Ein Object Rexx Paket/Modul
- Definiert Routinen, Klassen und Methoden, die die prozedurale Schnittstelle von BSF4Rexx von Object Rexx Programmen verstecken
- Verpackt alle BSF()-Subfunktionen in Object Rexx Methoden
- Ermöglicht das Importieren von Java-Klassen
 - Object Rexx "Proxy" Klassen
- Ermöglicht das Anlegen von Object Rexx "Proxy" Objekten, über die mit den entsprechenden Java-Objekten kommuniziert werden kann

Java als Object REXX, 2

BSF.CLS

– "BSF.CLS"

- Unterstützt Java Array-Objekte als Object REXX Array "Proxy" Objekte
 - Ermöglicht die Benutzung von Java-Feldern wie wenn es sich um Object REXX-Felder handeln würde
 - Daher beginnt das erste Element eines Feldes mit dem Index 1 (nicht 0)!
- Benutzt dafür den Object REXX **UNKNOWN**-Mechanismus
 - Ermöglicht eine relativ einfache Implementierung des benötigten "Forward"-Mechanismus, mit dem Object REXX-Nachrichten dazu führen, dass die entsprechenden Java-Methoden aufgerufen werden
- Neben anderen Eigenschaften, werden mit Hilfe des Object REXX Garbage-Collectors automatisch die Java-Objekte aus der "BSFRegistry" entfernt, die nicht mehr benötigt werden

BSF4Rexx, Beispiel

Object Rexx benutzt Java, eine Variante

```
/* Object Rexx version */  
  
say "java.version:" .bsf4rexx ~system.class ~getProperty('java.version')  
  
::requires bsf.cls -- loads the Object Rexx (camouflaging) support
```

Ausgabe z.B.:

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

BSF4Rexx, Beispiel

Object Rexx benutzt Java, eine weitere Variante

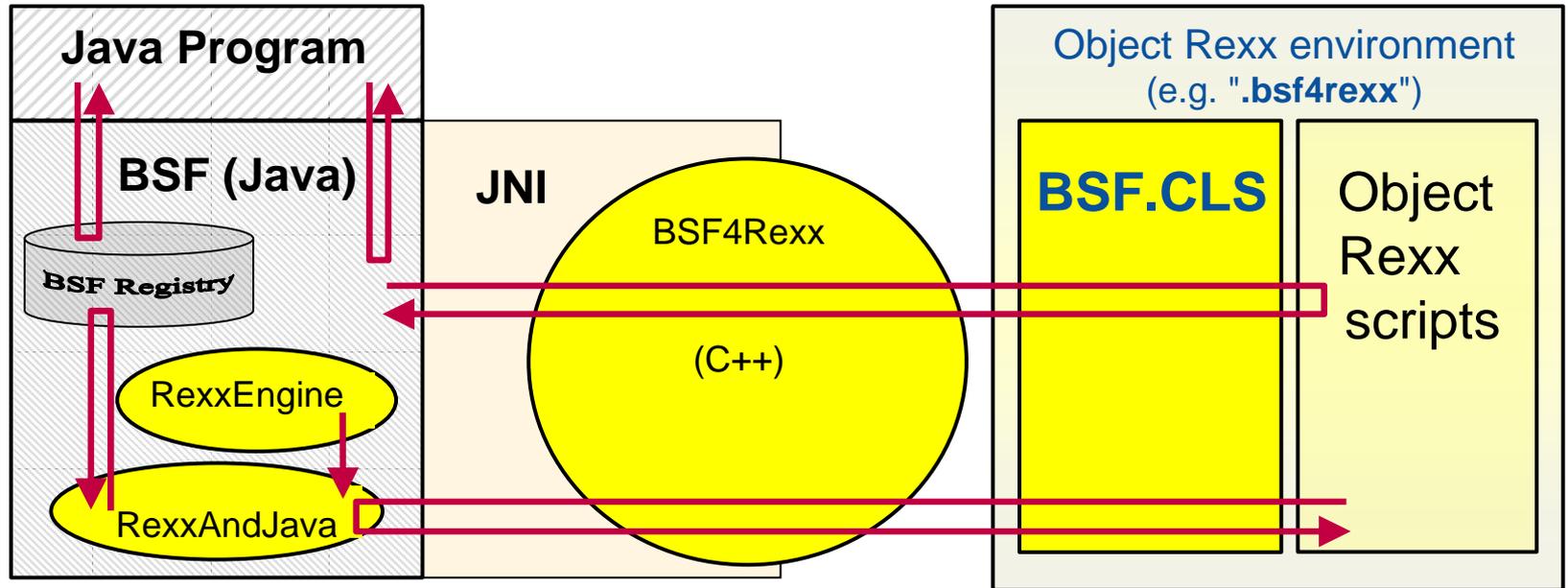
```
/* Object Rexx version */  
  
system=bsf.import("java.lang.System")  
say "java.version:" system~getProperty('java.version')  
  
::requires BSF.CLS -- loads the Object Rexx (camouflaging) support
```

Ausgabe z.B.:

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

Java als Object Rexx, 3

Architektur



BSF4Rexx (BSFRegistry)

Vor-registrierte Java Objekte

- Object Rexx directory ".BSF4Rexx"

- 1) .bsf4rex~Class.class
- 2) .bsf4rex~Object.class
- 3) .bsf4rex~Method.class
- 4) .bsf4rex~Array.class
- 5) .bsf4rex~String.class
- 6) .bsf4rex~System.class
- 7) .bsf4rex~Boolean.class
- 8) *.bsf4rex~boolean*
- 9) .bsf4rex~Byte.class
- 10) *.bsf4rex~byte*
- 11) .bsf4rex~Character.class
- 12) *.bsf4rex~char*
- 13) .bsf4rex~Double.class
- 14) *.bsf4rex~double*
- 15) .bsf4rex~Integer.class
- 16) *.bsf4rex~int*
- 17) .bsf4rex~Long.class
- 18) *.bsf4rex~long*
- 19) .bsf4rex~Float.class
- 20) *.bsf4rex~float*
- 21) .bsf4rex~Short.class
- 22) *.bsf4rex~short*
- 23) .bsf4rex~Void.class
- 24) *.bsf4rex~void*

BSF.CLS, 1

Öffentliche Klassen

- Klasse **BSF**
 - Object Rexx Proxy-Klasse, um Java als Object Rexx erscheinen zu lassen
- Klasse **BSF_ARRAY_REFERENCE**
 - Object Rexx Proxy-Klasse, um Java Array-Objekte als Object Rexx Arrays erscheinen zu lassen
- Klasse **BSF_PROXY**
 - Subklasse von **BSF**
 - Verpackt eine Zeichenkette, die auf einen Eintrag in der "BSFRegistry" verweist in ein Object Rexx "Proxy"-Objekt

BSF.CLS, 2

Öffentliche Routinen

1. box
2. bsf.createArray
3. bsf.getConstant
4. bsf.getEventInfoObject
5. bsf.getStaticValue
6. bsf.getStaticValue *Strict*
7. bsf.import
8. bsf.loadClass
9. bsf.lookupBean
10. bsf.pollEventText
11. bsf.postEventText
12. bsf.unregisterBean
13. bsf.wrap
14. bsf.wrapStaticFields
15. lif
16. Pp
17. unbox

BSF.CLS, 3

Proxy Klasse BSF

- Aufruf von "BSF.CLS" entweder mit **call** oder mit **::requires**

```
call bsf.cls
::requires BSF.CLS
```

- Ermöglicht das Importieren von Java-Klassen und die Interaktion damit so, wie wenn es sich um Object Rexx-Klassen handeln würde

```
.bsf~bsf.import(javaName, rexxName)
.bsf~bsf.import("java.awt.Frame", "javaFrame")
f=.javaFrame~new("hi!")~~show~~ToFront~~setSize(200,100)
```

- Erlaubt das Anlegen/Instantiieren von Java-Objekten

```
.bsf~bsf.import("java.awt.Frame", "javaFrame")
f1=.javaFrame~new("hi!") -- using an imported Java class
F2=.javaFrame~newStrict("String", "hi!") -- using newStrict()
f3=.BSF~new("java.awt.Frame", "hi!") -- using .BSF directly
```

- Proxy-Objekte
 - Object Rexx-Objekte, die Java-Objekte repräsentieren
 - Java-Objekte *müssen* in der [BSFRegistry](#) gespeichert sein!
 - Nachrichten, die für Java-Objekte bestimmt sind und an Object Rexx-Proxy-Objekte gesendet werden, lösen die [UNKNOWN](#) Ausnahme aus
 - [UNKNOWN](#)-Methode leitet die unbekannte Object Rexx-Nachricht an Java weiter
 - Java-Rückgabewerte werden an Object Rexx zurückgegeben
 - Wenn es sich um ein Java-Objekt handelt, wird dafür ein Object Rexx-Proxy-Objekt zurückgegeben!

BSF.CLS, 5

Proxy Klasse BSF

- Procedural BSF()-subfunctions available as (mangled) **instance methods**:
 - (1) bsf.class
 - (2) bsf.addEventListener
 - (3) bsf.addEventListenerReturningEventInfos
 - (4) bsf.dispatch
 - (5) bsf.exit
 - (6) bsf.invoke
 - (7) bsf.invoke*Strict*
 - (8) bsf.getFieldValue
 - (9) bsf.getFieldValue*Strict*
 - (10) bsf.setFieldValue
 - (11) bsf.setFieldValue*Strict*
 - (12) bsf.getPropertyValue
 - (13) bsf.setPropertyValue
 - (14) bsf.setPropertyValue*Strict*
- Procedural BSF()-subfunctions available as methods of .BSF, i.e. **class methods**:
 - (15) bsf.createArray
 - (16) bsf.exit
 - (17) bsf.getStaticValue
 - (18) bsf.getStaticValue*Strict*
 - (19) bsf.import
 - (20) bsf.loadClass
 - (21) bsf.lookupBean
 - (22) bsf.pollEventText
 - (23) bsf.postEventText
 - (24) bsf.setRexxNullString
 - (25) bsf.sleep
 - (26) bsf.wrapArray
 - (27) bsf.wrapEnumeration

BSF.CLS – Einige Bemerkungen, 1

Strenge (strikte) Typisierung

- Strenge Typisierung
 - Wenn notwendig, benutzen Sie die "strict"-Versionen der entsprechenden Methoden
 - Wenn Typinformationen für das Instantiieren von Java-Objekten benötigt werden (also statt der Klassenmethode "new" eine Klassenmethode "newStrict")

- Importieren der entsprechenden Java-Klasse

```
.bsf~bsf.import("javaName", "rexxName")
```

- Verwenden Sie anschließend die Klassenmethode "newStrict"

- Die Klassenmethode "newStrict" wird beim Importieren dynamisch erstellt

```
.bsf~bsf.import("java.awt.Frame", "javaFrame")
```

```
f1=.javaFrame~newStrict("String", "Hi there!")
```

```
-- or:
```

```
f2=.javaFrame~new("Hi there!")
```

BSF.CLS – Einige Bemerkungen, 2

Erzeugen und Benutzen von Java Feldern (Arrays)

```
-- create a two-dimensional (5x10) Java Array of type String
arr=.bsf~bsf.createArray(.bsf4rex~string.class, 5, 10)

arr[1,1]="First Element in Java array."      -- place an element
arr~put("Last Element in Java array.", 5, 10) -- place another one

do i over arr      -- loop over elements in array
  say i
end

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

Ausgabe:

```
First Element in Java array.
Last Element in Java array.
```

URLs

- **ooRexx**

- <http://www.ooRexx.org>
 - Homepage "Open Object Rexx (ooRexx)"
- http://wi.wu-wien.ac.at/rgf/rexx/misc/ecoop06/ECOOP2006_RDL_Workshop_Flatscher_Paper.pdf
 - Übersichtsartikel über die Geschichte und Konzepte von ooRexx

- **BSF4Rexx**

- <http://wi.wu-wien.ac.at/rgf/rexx/bsf4rexx/current/>
 - Homepage von BSF4Rexx
- http://wi.wu-wien.ac.at/rgf/rexx/orx15/2004_orx15_bsf-orx-layer.pdf (2004)
- http://wi.wu-wien.ac.at/rgf/rexx/orx14/orx14_bsf4rexx-av.pdf (2003)
- http://wi.wu-wien.ac.at/rgf/rexx/orx12/JavaBeanScriptingWithRexx_orx12.pdf (2001)
- <http://www.informit.com/articles/article.asp?p=418864&rl=1> (2005)
- <http://wi.wu-wien.ac.at/rgf/rexx/bsf4rexx/docs.apache-rexxla/>

- **Sun's Java Online Dokumentation**

- <http://java.sun.com/reference/api/>
 - Online Dokumentation über alle Java Klassen (in HTML)

- **"vim" bzw. "gvim"**

- <http://www.vim.org/>  <http://wi.wu-wien.ac.at/rgf/rexx/misc/vim/> (vim 7.0 <= 7.1, 2007-12)
 - Freier, opensource Editor, für viele Betriebssysteme verfügbar; ab Version 7.0 mit ooRexx Syntax-Highlighting

- **Apache BSF Homepage**

- <http://jakarta.apache.org/bsf/>

Überblick, Aufgabenstellungen

- Installieren Sie BSF4Rexx
 - Studieren Sie **alle** Beispiele
 - Erstellen von zwei einfachen Beispielen
 - Aufruf von Java von Object Rexx Skript aus
 - Recherchieren Sie die online-Hilfe zur **JRE**
 - Recherchieren Sie interessante, opensource Java-Klassen/-Anwendungen, die Sie von ooRexx aus steuern/benutzen möchten