Institut für Betriebswirtschaftslehre und Wirtschaftsinformatik

Automatisierung von Java Anwendungen (7)

Bean Scripting Framework (BSF), 1

Bean Scripting Framework (BSF), "BSF4ooRexx", "JavaDoc", Java als riesige Funktionsbibliothek für ooRexx

Prof. Dr. Rony G. Flatscher

Wirtschaftsuniversität Wien ■ Augasse 2-6 ■ A-1090 Wien

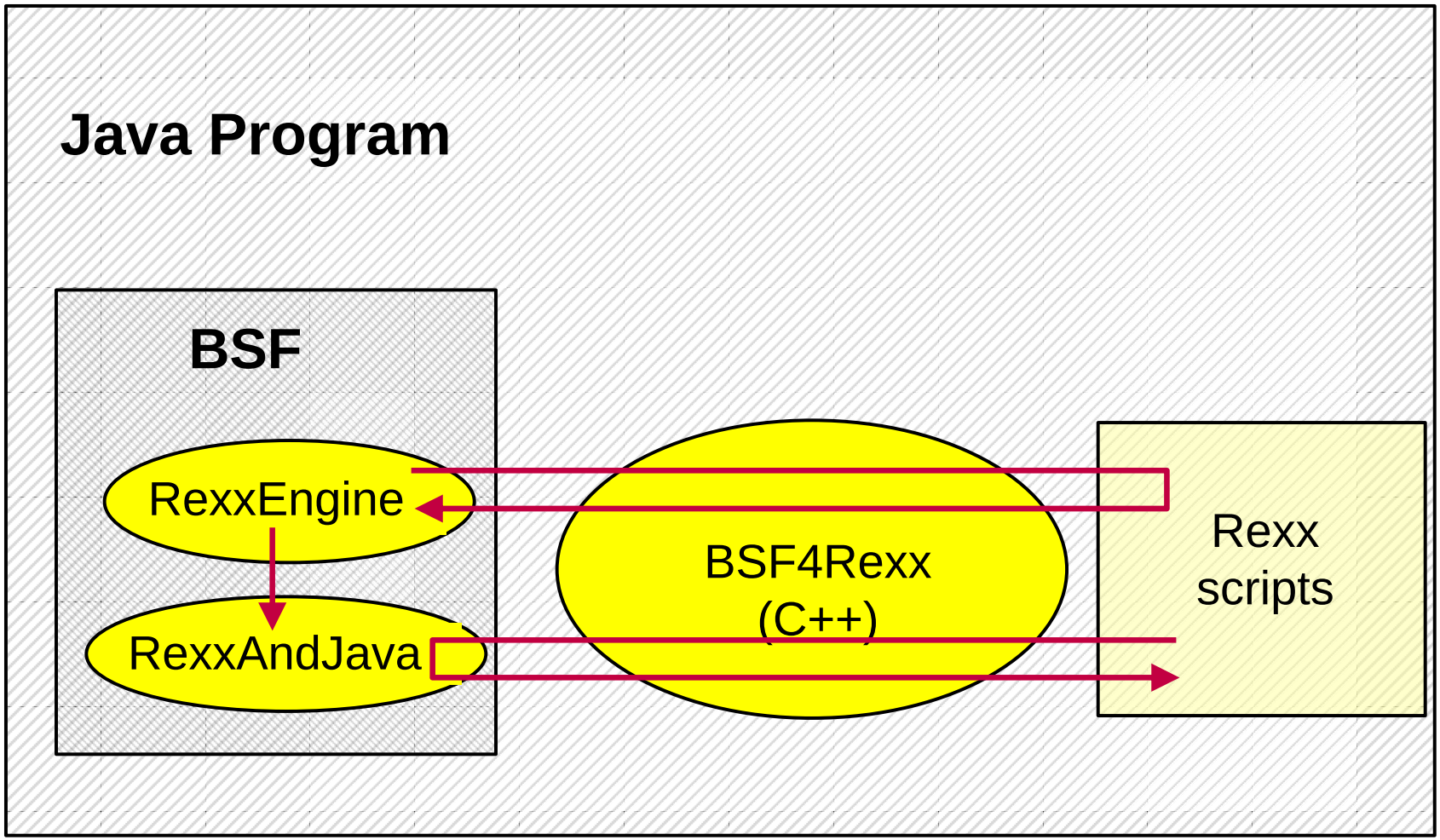
Bean Scripting Framework (BSF)

- Bean Scripting Framework
  - Opensource Projekt der Firma IBM
    - Z.B. in IBM Produkten wie WebSphere (Java Server Pages, 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 (2001), 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 ooRexx Klassen
    - "**`BSF.CLS`**"
      - ooRexx Klassen, die mit BSF4Rexx interagieren
      - Ermöglicht das Senden von ooRexx Nachrichten an Java-Objekten
      - Ermöglicht die Benutzung von Java-Feldern ("Arrays") wie wenn es sich um ooRexx Felder ("Arrays") handeln würde

BSF4Rexx
Essener Version (2001), 2



BSF4Rexx
Essener Version (2001), 3

- 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 ooRexx 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

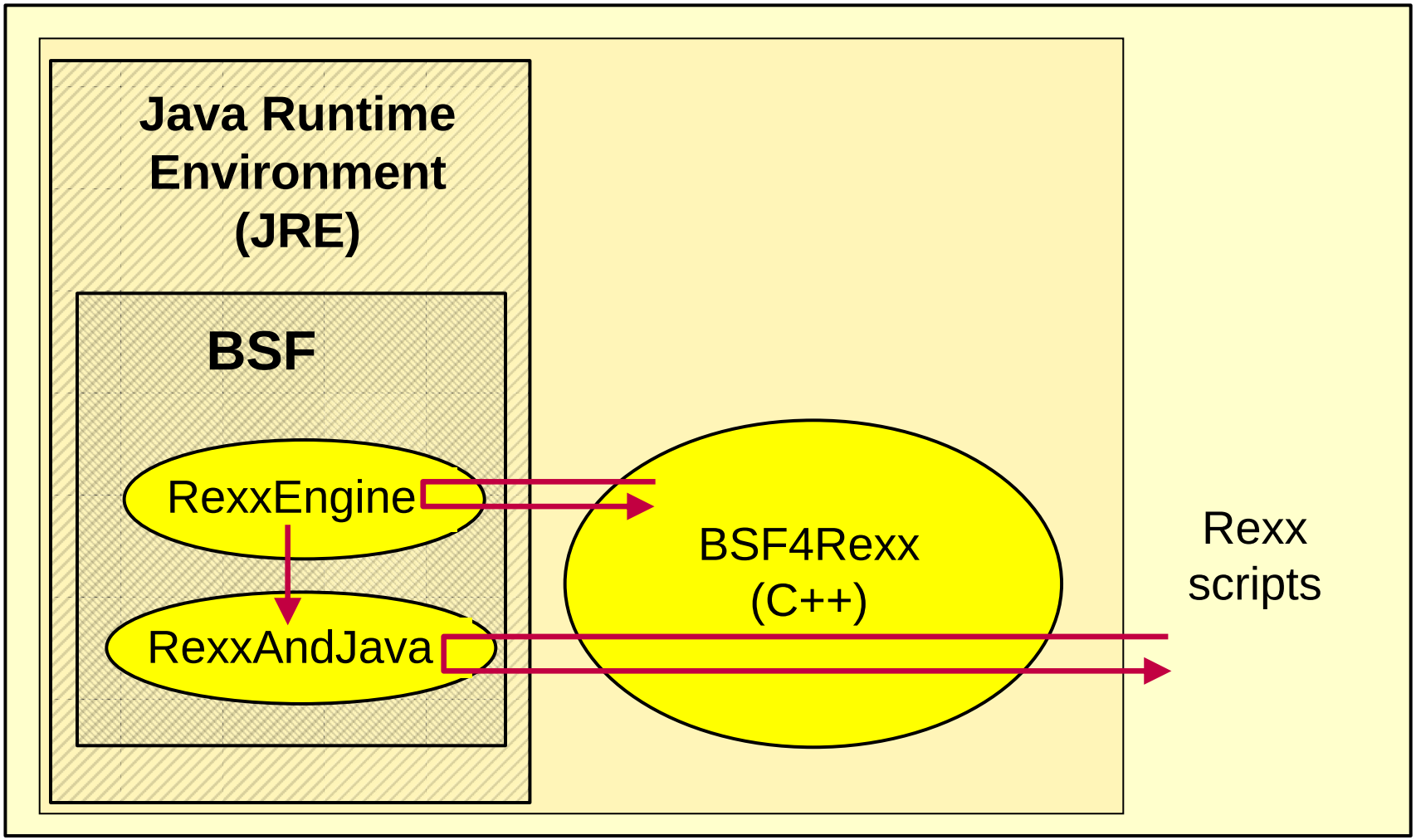
```
import org.apache.bsf.*; // BSF support
import java.io.*;        // exception handling

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 (2003), 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 (2003), 2



BSF4Rexx
Augsburger Version (2003), 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 externene Rexx-Funktionsbibliothek "**BSF4Rexx.cc**"
      - "**BSFVersion**"

BSF4Rexx
Augsburger Version (2003), 4

- 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 (2003), 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

BSF4Rexx
Wiener Version (2006)

- 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>

BSF4ooRexx (2009, 2010)

- May 2010
  - Originated in the fall 2009
- Needs ooRexx 4.0.1 (May 2010)
  - Uses new ooRexx APIs
- "BSF4ooRexx" package
  - Home: <http://wi.wu-wien.ac.at/rgf/rexx/bsf4oorexx/current/> "readmeBSF4ooRexx.txt", "readmeOOo.txt" lesen !
  - Adds
    - Realtime event handling
    - Proxying of ooRexx objects
    - ...

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 withou 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 (zeitlos)
Java Klasse "XyzType", 2

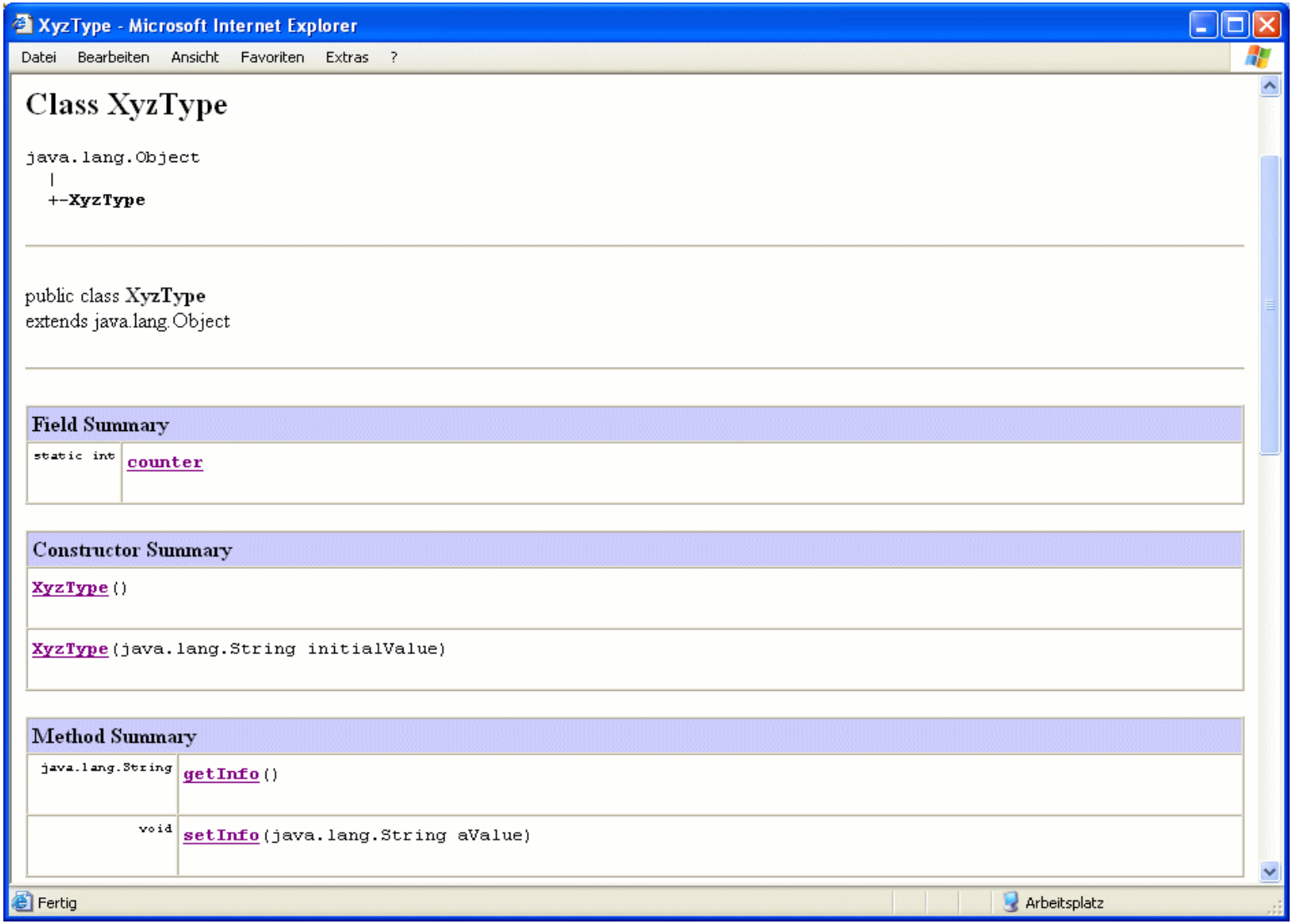
- Speichern der Datei unter "**`XyzType.java`**"
  - Achtung auf Groß- und Kleinschreibung!
    - Groß- und Kleinschreibung ist unter Java signifikant!
- Kompilierung mit Hilfe des Java-Compilers, z.B.

```
javac XyzType.java
```

- Erzeugen der standardisierten HTML Dokumentation

```
javadoc XyzType.java
```

Ein Beispiel (zeitlos)
Java Klasse "XyzType", 3



Beispiel "code1-oo.rex" (zeitlos)
Ein ooRexx-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 ooRexx-Programm, 2

- Über Java

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

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

  ```
  rexxj2 code1-oo.rex
  ```

- Oder über Rexx

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

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

  ```
  code1-oo.rex
  ```

Beispiel "code2.rex", Ausgabe
Ein ooRexx-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 BSF4ooRexx
    - Benötigt ooRexx 4.0.1 (Ausgabe Mai 2010)
    - Detaillierte Auflistung der BSF-Funktionen
    - Detaillierte Auflistung der Methoden der Java Proxy-Klasse "**.BSF**"
  - Beispiele

BSF

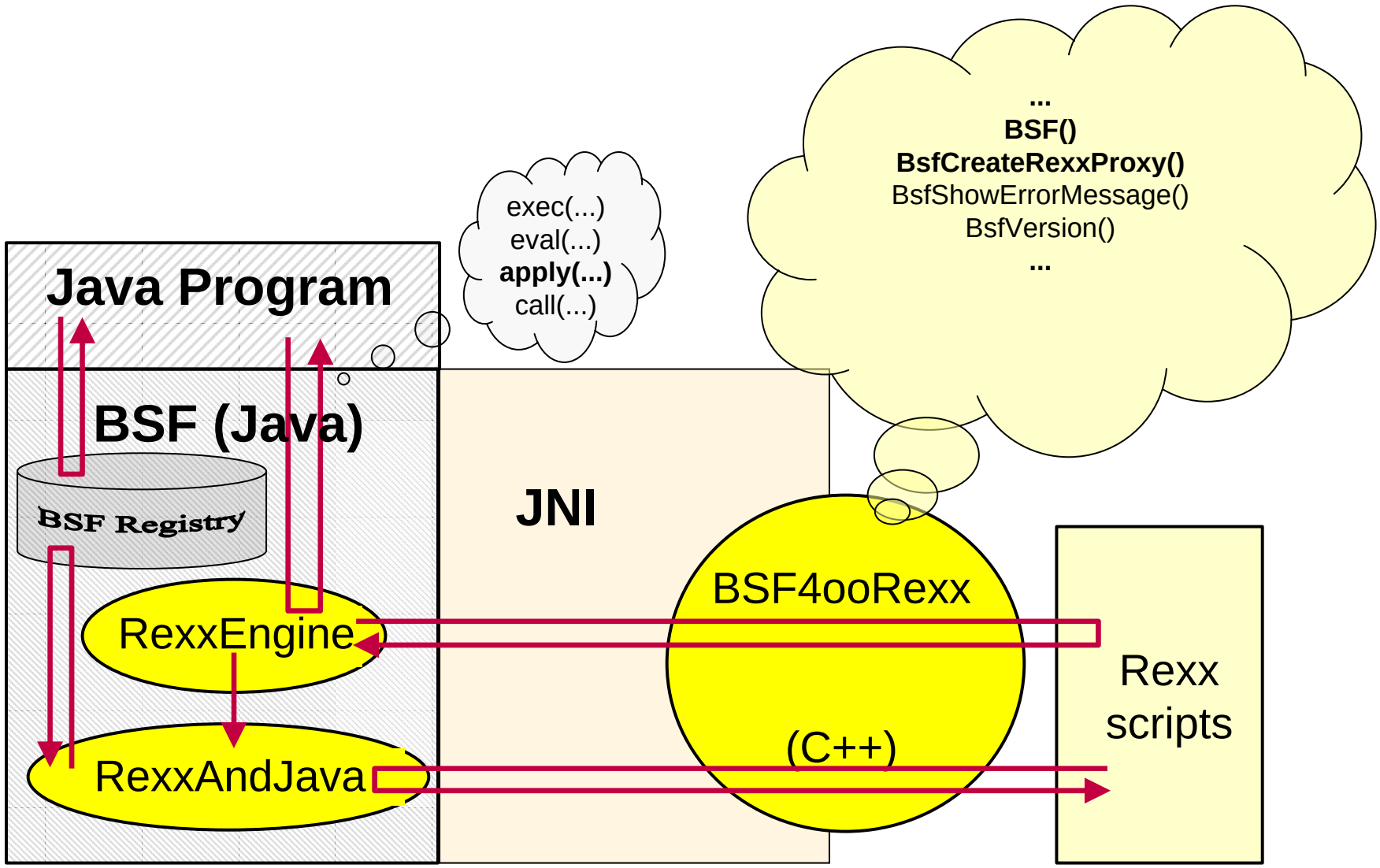
- Bean Scripting Framework
  - Ein Java-Rahmenwerk (Framework), mit dessen Hilfe es sehr einfach ist, Skripte in anderen Sprachen aufzurufen
    - Z.B. JavaScript, NetRexx
  - Ursprünglich von IBM als Open-source-Projekt
    - Teil von IBM's WebSphere, um Skripte in Java Server Pages (JSP) einbetten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

BSF4ooRexx (Stand: 2010)

- 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 2003-2010
    - Weiterentwicklung der Augsburger Version

BSF4ooRexx Architektur

http://wi.wu-wien.ac.at/rgf/rexx/bsf4oorexx/current



BSF4ooRexx (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) "**b**oolean.class"*
10) "**B**yte.class"
*11) "**b**yte.class"*
12) "**C**haracter.class"
*13) "**c**har.class"*
14) "**D**ouble.class"
*15) "**d**ouble.class"*
16) "**I**nteger.class"
*17) "**i**nt.class"*
18) "**L**ong.class"
*19) "**l**ong.class"*
20) "**F**loat.class"
*21) "**f**loat.class"*
22) "**S**hort.class"
*23) "**s**hort.class"*
24) "**V**oid.class"
*25) "**v**oid.class"*

BSF4ooRexx
BSF()-Subfunktionen, 1

(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 "arrayPut*Strict*", arrayBeanName, *typeIndicator*, newValue, idx0 [, idx1]...
call BSF "arrayPut*Strict*", 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("getFieldValue*Strict*", beanName, fieldName)

(11) p=BSF("getPropertyValue", beanName, propertyName, index | ".NIL")

(12) s=BSF("getStaticValue", JavaClassName, fieldName)
s=BSF("getStaticValue*Strict*", JavaClassName, fieldName)

(13) res=BSF("invoke", beanName, methodName, arg1 [, arg2]...)
res=BSF("invoke*Strict*", beanName, methodName, *typeIndicator1*, arg1 [, *typeIndicator2*, arg2]...)

(14) cl=BSF("loadClass", JavaClassName)

BSF4ooRexx
BSF()- Subfunktionen, 2

(15) o=BSF("lookupBean", beanName)

(16) t=BSF("pollEventText" [, timeout in msecs])

(17) call BSF "postEventText", eventText, priority

(18) o=BSF("registerBean", beanName, beanType, arg1 [,...])

(19) o=BSF("registerBean*Strict*", beanName, beanType, *typeIndicator1*, arg1 [, *typeIndicator2*, arg2]...)

(20) v=BSF("setFieldValue", beanName, fieldName, newValue)
v=BSF("setFieldValue*Strict*", beanName, fieldName, [*typeIndicator,*] newValue)

(21) v=BSF("setPropertyValue", beanName, propertyName, index | ".NIL", newValue)
v=BSF("setPropertyValue*Strict*", beanName, propertyName, index | ".NIL", *typeIndicator*, newValue)

(22) call BSF "setRexxNullString", newString

(23) call BSF "sleep", time2sleep in msecs

(24) str=BSF("unregisterBean", beanName)

(25) v=BSF("version")

(26) e=BSF("wrapArray", arrayBeanName)

(27) e=BSF("wrapEnumeration", enumerationBeanName)

BSF4ooRexx, Typisierungsproblem, 1 "Strict"

- "strict"-Subversionen
  - Erlauben 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

BSF4ooRexx, Typisierungsproblem, 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 ", "**SH**ort", "**ST**ring"
    - Nur die fett und in Großbuchstaben dargestellten Teile sind notwendig
  - Java-Typinformationen aus der Java-HTML-Dokumentation
    - "BOolean", "BYte", "Char", "Double", "Float", "Int", "Long", "SHort" sind sogenannte "primitive" Java Datentypen
    - "STring"
    - "Object" bezieht sich auf *jedes* Java-Objekt

Java verkleidet als ooRexx, 1
BSF.CLS

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

Java verkleidet als ooRexx, 2
BSF.CLS

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

BSF4ooRexx, Beispiel
ooRexx benutzt Java

```
/* ooRexx version */

-- some Java classes are so important, they get preloaded by BSF.CLS
say "java.version:" .java.lang.System ~getProperty('java.version')

::requires bsf.cls -- loads the ooRexx (camouflaging) support
```

Ausgabe z.B.:

```
java.version: 1.6.0_18
```

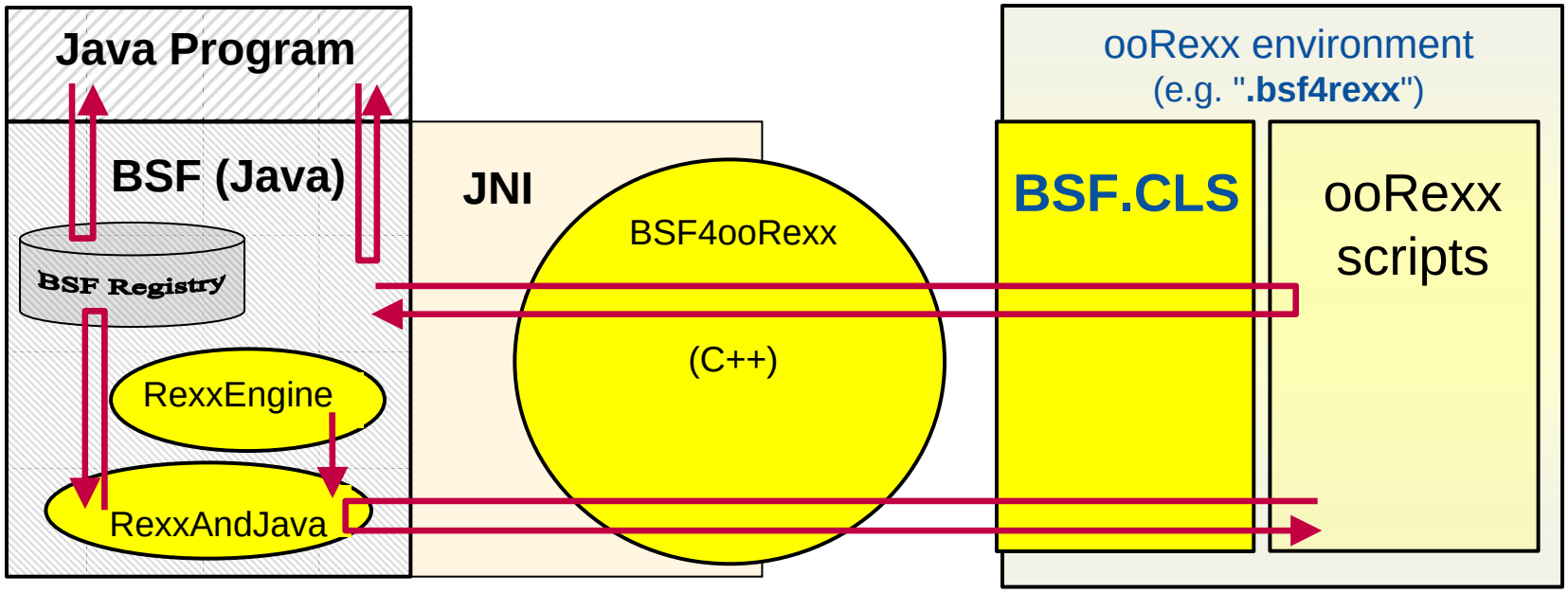
BSF4ooRexx, Beispiel
ooRexx benutzt Java, eine weitere Variante

```
/* ooRexx version */

clz=bsf.loadClass("java.lang.System") -- load a Java class
say "java.version:" clz~getProperty('java.version')

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

Ausgabe z.B.:

```
java.version: 1.6.0_18
```

Java als ooRexx, 3
Architektur



BSF4ooRexx (BSFRegistry) Vor-registrierte Java Objekte

- ooRexx directory "**.BSF4Rexx**"

1) .bsf4rexx~Class.class
2) .bsf4rexx~Object.class
3) .bsf4rexx~Method.class
4) .bsf4rexx~Array.class
5) .bsf4rexx~String.class
6) .bsf4rexx~System.class
7) .bsf4rexx~Boolean.class
8) *.bsf4rexx~**boolean***
9) .bsf4rexx~Byte.class
10) *.bsf4rexx~**byte***
11) .bsf4rexx~Character.class
12) *.bsf4rexx~**char***
13) .bsf4rexx~Double.class
14) *.bsf4rexx~**double***
15) .bsf4rexx~Integer.class
16) *.bsf4rexx~**int***
17) .bsf4rexx~Long.class
18) *.bsf4rexx~**long***
19) .bsf4rexx~Float.class
20) *.bsf4rexx~**float***
21) .bsf4rexx~Short.class
22) *.bsf4rexx~**short***
23) .bsf4rexx~Void.class
24) *.bsf4rexx~**void***

BSF.CLS, 1
Öffentliche Klassen

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

BSF.CLS, 2
Öffentliche Routinen

1. box
2. bsf.createJavaArray
3. bsf.createJavaArrayOf
4. bsf.getConstant
5. bsf.getEventInfoObject
6. bsf.getStaticValue
7. bsf.getStaticValue*Strict*
8. bsf.importClass
9. bsf.loadClass
10. bsf.lookupBean
11. bsf.pollEventText
12. bsf.postEventText
13. bsf.unregisterBean
14. bsf.wrap
15. bsf.wrapStaticFields
16. iif
17. pp
18. 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 ooRexx-Klassen handeln würde

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

- Erlaubt das Anlegen/Instantiieren von Java-Objekten

```
.bsf~bsf.importClass("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
```

BSF.CLS, 4
Proxy Klasse BSF

- Proxy-Objekte
  - ooRexx-Objekte, die Java-Objekte repräsentieren
    - Java-Objekte *müssen* in der BSFRegistry gespeichert sein!
  - Nachrichten, die für Java-Objekte bestimmt sind und an ooRexx-Proxy-Objekte gesendet werden, lösen die UNKNOWN Ausnahme aus
    - UNKNOWN-Methode leitet die unbekannte ooRexx-Nachricht an Java weiter
    - Java-Rückgabewerte werden an ooRexx zurückgegeben
      - Wenn es sich um ein Java-Objekt handelt, wird dafür ein ooRexx-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.createJavaArray
(16) bsf.createJavaArrayOf
(17) bsf.exit
(18) bsf.getStaticValue
(19) bsf.getStaticValue*Strict*
(20) bsf.import
(21) bsf.loadClass
(22) bsf.lookupBean
(23) bsf.pollEventText
(24) bsf.postEventText
(25) bsf.setRexxNullString
(26) bsf.sleep
(27) bsf.wrapArray
(28) 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.importClass("javaName", "rexxName")
```

    - Verwenden Sie anschließend die Klassenmethode "**newStrict**"
      - Die Klassenmethode "newStrict" wird beim Importieren dynamisch erstellt

```
.bsf~bsf.importClass("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.createJavaArray("java.lang.String", 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 ooRexx (camouflaging) support
```

Ausgabe:

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

URLs

- **ooRexx**
  - <http://www.ooRexx.org>
    - Homepage "Open ooRexx (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
- **BSF4ooRexx**
  - http://wi.wu-wien.ac.at/rgf/rexx/bsf4oorexx/current/
    - Homepage von BSF4ooRexx
  - 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, *zunächst:* "J2SE 1.4.2", später die aktuellste Version**
  - http://java.sun.com/reference/api/
    - Online Dokumentation über alle Java Klassen (in HTML)
- **"vim" bzw. "gvim"**
  - http://www.vim.org/
    - Freier, opensource Editor, für viele Betriebssysteme verfügbar; seit Version 7.1 mit ooRexx Syntax-Highlighting
- **Apache BSF Homepage**
  - http://jakarta.apache.org/bsf/

Tasks

- Install ooRexx 4.0.1

  http://www.oorexx.org/download.html

- Install BSF4ooRexx
  - BSF4ooRexx_install.zip
  - readmeBSF4ooRexx.txt
- Examples
  - Run all examples of the "samples" directory, but not its subdirectories!
  - Create two small nutshell examples