Modul University Vienna

Introduction To BSF4ooRexx850

ooRexx/Java Language Bindings

Easily exploiting Java from ooRexx on all operating system platforms

37th International Rexx Symposium
May 3rd – May 6th 2026, Barcelona

Prof. Rony G. Flatscher

Overview

- Some information on Java and an example of using ooRexx to exploit it
- Some important things to know about Java
- Introducing the ooRexx package (program) BSF.CLS
  - Camouflages Java as ooRexx
  - Makes it possible to simply send ooRexx messages to Java (class) objects
  - Provides some important utility features
- Download links
- Roundup

Prof. Rony G. Flatscher

Java

- Programming language with the following notable features
  - Compiles to machine instructions (*"bytecode"*) of an *artificial processor*
  - Needs a "Java virtual machine (JVM)" to execute the bytecode
    - JVMs are available for all important operating systems and hardware architectures
    - *Hence, a Java class or a Java program, once compiled can be run everywhere!*
  - Distributed with a (huge) "Java runtime environment (JRE)"
    - A *huge Java class library* that offers everything that an application may possibly need
      - E.g. Socket classes for Internet programming, GUI classes for graphical user interfaces, ...
    - Uncountable third party Java class libraries, most available as open-source (e.g. ASF)
  - Most important programs get programmed with Java (even Android applications!)
  - Many professional applications that are not programmed in Java offer Java APIs
    - E.g. SAP, OpenOffice/LibreOffice, ...
- Hence Java is truly a programmer's "treasure trove" for all operating systems!

BSF4ooRexx850

- External Rexx function package
  - Allows to interact with the Java runtime environment (JRE)
    - Exploit functionality of Java classes
    - Exploit functionality of Java objects
  - ooRexx 5.0 or later, Java 8 or later
  - Package "BSF.CLS"
    - Camouflages Java as ooRexx (Java appears to be dynamic and message based)
    - Supplies class BSF and public routines
- "Everything that is available in Java becomes directly available to ooRexx !"
  - Java: "write once, run everywhere!"
    - Windows, MacOS, Linux, ...

BSF4ooRexx: An Example, 1

- The following example
  - Uses the ::requires directive to load the ooRexx-Java bridge

```
::requires "BSF.CLS"
```

    - Directives get processed in the setup phase, right before the program starts
  - Creates an instance of the Java class named java.awt.Dimension and interacts with it via ooRexx messages that denote the method names to run
    - Studying the documentation of the Java class java.awt.Dimension one can see which Java methods are available for use
  - Displays the string that the message toString returns
  - Changes the values for the width and height fields
  - Displays the string that the message toString returns

Prof. Rony G. Flatscher

BSF4ooRexx: An Example, 2

```
dim=.bsf~new("java.awt.Dimension", 100, 200)     -- create with width and height
say dim~toString                                 -- show string value

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

Output:

```
java.awt.Dimension[width=100,height=200]
```


Package: java.awt

Downloading Java (Usually Free and Open-source)

- JRE versus JDK
  - JRE: "**J**ava **R**untime **E**nvironment", no compiler
  - JDK: "**J**ava **D**evelopment **K**it", compiler & tools
- Java/OpenJDK 8 LTS ("long term support")
  - Released spring 2014, supported until 2030 (Temurin, IBM JDK), 2031 (Liberica)
- Java/OpenJDK 25 LTS ("long term support", "modular Java")
  - Released fall 2025, supported at least until 2033 (Azul, Oracle), 2034 (Liberica)
- Suggestion: download OpenJDK *with JavaFX* support, e.g.
  - Scroll down to see all versions pick the *JavaFX* installation package
    - **Full JDK:** <https://bell-sw.com/pages/downloads/> ("Liberica", 2026-04-23)
    - **JDK FX:** <https://www.azul.com/downloads/> ("Azul", 2026-04-23)

Things to Know About Java, 1

- Strictly typed language
  - Primitive types
    - boolean, byte, char, short, int, long, float, double
  - Object-oriented types
    - Any Java class, e.g.
      - `java.awt.Dimension`, `java.lang.String`, `java.lang.System`, `...`
    - Wrapper classes for primitive types
      - `java.lang.Boolean`, `java.lang.Byte`, `java.lang.Character`, `java.lang.Short`, `java.lang.Integer`, `java.lang.Long`, `java.lang.Float`, `java.lang.Double`
      - "boxing": wraps up a primitive value into a wrapper object
      - "unboxing": retrieves a primitive value from its wrapper object

Things to Know About Java, 2

- Case sensitive
  - Upper- and lowercase significant!
- Classes organized in packages
  - Package names may be compound
    - E.g. "java.lang"
  - Fully "qualified class name" includes package name
    - e.g. "java.lang.String"
  - "Unqualified class name"
    - e.g. "String"

Prof. Rony G. Flatscher

Things to Know About Java, 3

- A Java class may consist of
  - Fields (comparable to ooRexx attributes) and
  - Methods (comparable to ooRexx methods)
- Fields and methods
  - Static fields and static methods
    - Sometimes dubbed "class fields" and "class methods"
    - Available to the class object *and* its instances
  - Otherwise "instance methods"
    - Only available to instances of a Java class

Prof. Rony G. Flatscher

Things to Know About Java, 4

- A Java class, its fields and methods may be
  - "public"
    - These can be accessed by the "world" (everyone)
  - "private"
    - Only accessible within the Java class
  - "protected"
    - Only accessible within Java classes of the same package and subclasses
  - None of the above modifiers given ("package private")
    - Only accessible within Java classes of the same package, but to noone else

Prof. Rony G. Flatscher

Things to Know About Java, 5

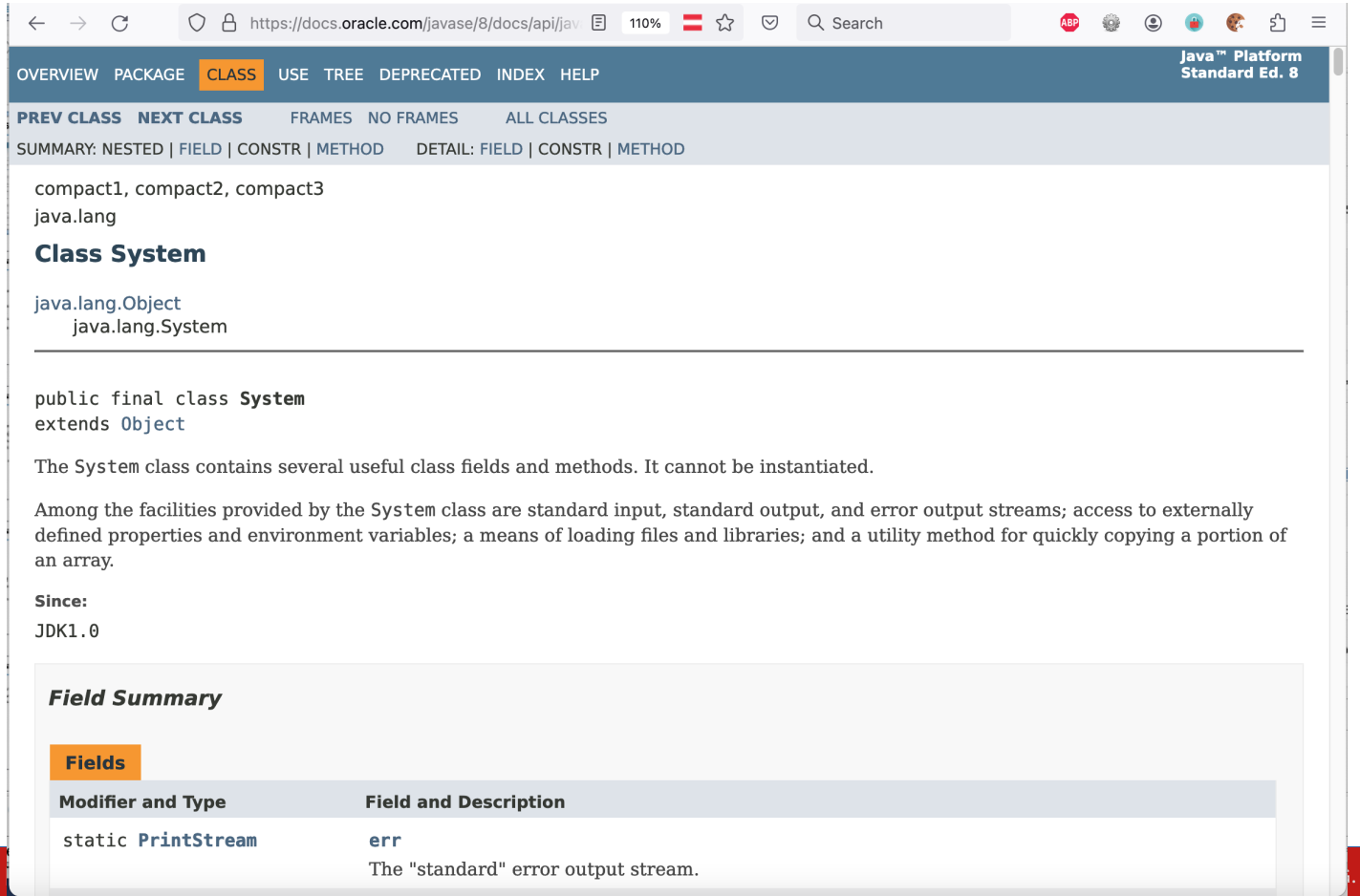
- Excellent documentation ("JavaDoc")
  - Extensive set of interlinked HTML documents
    - Created right from the comments in Java sources
  - Can be studied on the Internet, search e.g. with

```
javadoc 8 java.awt.Dimension
javadoc 8 Dimension
javadoc 25 java.awt.Dimension
javadoc 25 Dimension
```

- Documentation can be downloaded to local computer, e.g.
  - Java/JDK 8 LTS ("long term support"):
    - <https://www.oracle.com/java/technologies/javase-jdk8-doc-downloads.html> (2026-04-23)
  - Java/JDK 25 LTS ("long term support"):
    - <https://www.oracle.com/java/technologies/javase-jdk25-doc-downloads.html> (2026-04-23)

A Javadoc Example (JDK8LTS), 1

Search keywords:
Javadoc 8 System



A Javadoc Example (JDK8LTS), 2

Method Summary

All Methods | **Static Methods** | **Concrete Methods** | **Deprecated Methods**

Modifier and Type	Method and Description
static void	**arraycopy**(**Object** src, int srcPos, **Object** dest, int destPos, int length) Copies an array from the specified source array, beginning at the specified position, to the specified position of the destination array.
static **String**	**clearProperty**(**String** key) Removes the system property indicated by the specified key.
static **Console**	**console**() Returns the unique **Console** object associated with the current Java virtual machine, if any.
static long	**currentTimeMillis**() Returns the current time in milliseconds.
static void	**exit**(int status) Terminates the currently running Java Virtual Machine.
static void	**gc**() Runs the garbage collector.
static **Map**<**String**,**String**>	**getenv**() Returns an unmodifiable string map view of the current system environment.
static **String**	**getenv**(**String** name) Gets the value of the specified environment variable.
static **Properties**	**getProperties**() Determines the current system properties.
static **String**	**getProperty**(**String** key)

BSF.CLS: *Camouflages Java* as ooRexx

- ooRexx proxy class "**BSF**"
  - Allows to create Java objects
  - Requires the fully qualified Java class name
- Invoking Java methods
  - Just send the name of the method as a message to the Java object
    - Supply the arguments as documented, if any
      - Type conversions between ooRexx and Java are done automatically by BSF4ooRexx, if necessary
      - Return values are automatically converted by BSF4ooRexx, if necessary

BSF.CLS: Creating Java Objects

- ooRexx proxy class "**BSF**"
  - Allows to create Java objects
  - Needs at least fully qualified Java class name
- Possible arguments for creating Java objects
  - Can be found by studying the "*Constructor*" section in the Javadocs
  - Supply the arguments as documented after the fully qualified Java class name argument
    - Type conversions ("marshalling") between ooRexx and Java are done automatically by BSF4ooRexx, if necessary

BSF.CLS: Creating Java Objects, Example

```
-- see Javadocs: search Internet with "javadoc java.awt.Color"
red=.bsf~new("java.awt.Color",255,0,0)      -- create color red
say "red:" red~toString -- toString will show the RGB values

myColor=.bsf~new("java.awt.Color",100,200,3) -- create an individual color
say "myColor:" myColor~toString
brighter=myColor~brighter   -- get a brighter color
say "brighter:" brighter~toString

::requires "BSF.CLS"   -- get ooRexx-Java bridge
```

Output (maybe):

```
red: java.awt.Color[r=255,g=0,b=0]
myColor: java.awt.Color[r=100,g=200,b=3]
brighter: java.awt.Color[r=142,g=255,b=4]
```


Prof. Rony G. Flatscher

BSF.CLS: *Camouflages Java* as ooRexx

- Allows to load any Java class
  - **bsf.loadClass(JavaClassName)**
    - Java class name
      - Use of the exact case is mandatory !
      - Java class name must be fully qualified !
- Allows accessing static (class) methods and fields (attributes)
  - Example uses java.lang.System's static getProperty() method to query the Java version from ooRexx

Prof. Rony G. Flatscher

BSF.CLS: *Loading* a Java Class, Example

```
-- see Javadocs: search Internet with "javadoc java.lang.System"
clz=bsf.loadClass("java.lang.System")   -- loads the Java class
say "java.version:" clz~getProperty("java.version")

::requires "BSF.CLS"   -- get ooRexx-Java bridge
```

Output (maybe):

```
java.version: 1.8.0_462
```

BSF.CLS: *Camouflages Java* as ooRexx

- Allows to import any Java class
  - **bsf.import(JavaClassName)**
    - Java class name
      - Use of the exact case is mandatory !
      - Java class name must be fully qualified !
- Imported Java class can be treated as if it were an ooRexx class
  - Allows to use the ooRexx "**new**"-method to create instances of the imported Java class
    - Possible arguments for creating Java objects can be found by studying the "Constructor" section in the Javadocs

BSF.CLS: *Importing* a Java Class, Example

```
-- see Javadocs: search Internet with "javadoc java.awt.Color"
clzColor=bsf.importClass("java.awt.Color")  -- import Java class

red=clzColor~red         -- get static field for red color
say "red:" red~toString -- toString will show the RGB values

myColor=clzColor~new(100,200,3) -- create an individual color
say "myColor:" myColor~toString
brighter=myColor~brighter   -- get a brighter color
say "brighter:" brighter~toString

::requires "BSF.CLS"    -- get ooRexx-Java bridge
```

Output (maybe):

```
red: java.awt.Color[r=255,g=0,b=0]
myColor: java.awt.Color[r=100,g=200,b=3]
brighter: java.awt.Color[r=142,g=255,b=4]
```

BSF.CLS: *Camouflages Java* as ooRexx

- Accessing, setting Java fields
  - ooRexx treats public fields as ooRexx attributes
  - Java "get" and "set" pattern methods for Java fields honored by BSF4ooRexx
    - Just use the field name following "get" and "set" only
  - Static fields can be accessed via the
    - Java class object or
    - Any of its instances

BSF.CLS: *Java Fields As ooRexx Attributes*

```
-- see Javadocs: search Internet with "javadoc java.awt.Dimension"
dim=.bsf~new("java.awt.Dimension", 100, 200)
say dim~toString
dim~height=321       -- treat field height as if it was an ooRexx attribute
dim~width =1024      -- treat field width as if it was an ooRexx attribute
say dim~toString

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

Output:

```
java.awt.Dimension[width=100,height=200]
java.awt.Dimension[width=1024,height=321]
```

- About respecting case
  - Case of fully qualified Java class name
    - Always significant!
- Case of fields and method names insignificant!
  - Eases coding considerably

BSF.CLS: Creating Java Arrays, 1

- Java arrays
  - Strictly typed
  - Fixed capacity
  - Indices start with value "**0**"
- Public routine "**bsf.createJavaArray(...)**"
  - Arguments
    - First argument gives the Java type
      - Fully qualified Java class name or Java class object
    - Each further argument is an integer value, denoting the maximum elements in that dimension

BSF.CLS: Creating Java Arrays, 2

- Public routine "**bsf.createJavaArray(...)**"
  - Resulting Java array can be used as if it was an ooRexx array object!
    - Indices start at "**1**" as with ooRexx arrays!
    - Possesses the fundamental *ooRexx array methods* like "**AT**", "**[]**", "**PUT**", "**[]=**", "**supplier**", and "**makeArray**"
    - Can be therefore used in ooRexx "**DO ... OVER**" and "**DO WITH ... OVER**" loops

Prof. Rony G. Flatscher

BSF.CLS: Creating a Java Array

```
-- 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 o over arr      -- loop over elements in array (makearray)
   say o
end
say

do with index i item o over arr  -- loop over elements in array (supplier)
   say i":" o
end

::requires BSF.CLS -- loads Java support
```

Output:

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

1,1: First Element in Java array.
5,10: Last Element in Java array.
```

BSF4ooRexx: BSFCreateRexxProxy, 1

- RexxProxy
  - A *Java object* that proxies an ooRexx object
  - Allows Java to send messages to ooRexx objects
  - Any method invocations on the Java object will be forwarded as an ooRexx message to the proxied ooRexx object
    - All arguments supplied to the Java method are forwarded in the same sequence with the ooRexx message
    - BSF4ooRexx always appends an additional argument, "**slotDir**" (an ooRexx directory object) to the ooRexx message, which will contain information about the Java method invocation

BSF4ooRexx: BSFCreateRexxProxy, 2

- RexxProxy
  - **BSFCreateRexxProxy(rexxObj [, userData])**
    - Creates and returns a Java object that proxies "**rexxObj**"
    - If "**userData**" (any Rexx object) supplied, then it will be added to the "**slotDir**" directory
  - **BSFCreateRexxProxy(rexxObj [, [userData], jiClz[, ...]])**
    - "**jiClz**" can be one or more Java interface classes the returned RexxProxy can be used for!
  - **BSFCreateRexxProxy(rexxObj [, [userData], jaClz[, arg[,...]])**
    - "**jaClz**" is an abstract Java class, "**arg**" can be one or more arguments for creating an instance of it

BSF4ooRexx: RexxProxy, 1

```
rexxObj=.myClass~new
rexxObj~hello
say "---"
rp=BSFCreateRexxProxy(rexxObj)    -- create a Java RexxProxy object
rp~sendMessage0("hello")          -- send via Java

::requires BSF.CLS    -- get Java support

::class myClass
::method hello
  say "hello from" pp(self)
```

Output:

```
hello from [a MYCLASS]
---
hello from [a MYCLASS]
```


Prof. Rony G. Flatscher

BSF4ooRexx: RexxProxy, 2

```
rexxObj=.myClass~new
rexxObj~hello
say "---"
userData="This is some Rexx string."      -- sent only if invoked via Java
rp=BSFCreateRexxProxy(rexxObj,userData)   -- create a Java RexxProxy object
rp~sendMessage0("hello")          -- send via Java

::requires BSF.CLS   -- get Java support

::class myClass
::method hello
  use arg slotDir    -- available only, if called from Java
  if slotDir~isA(.directory) then
     say "hello from" pp(self) "userData:" pp(slotDir~userData)
  else
     say "hello from" pp(self)
```

Output:

```
hello from [a MYCLASS]
---
hello from [a MYCLASS] userData: [This is some Rexx string.]
```


Prof. Rony G. Flatscher

Links (As of: 2026-04-23)

- ooRexx

  https://sourceforge.net/projects/oorexx/files/oorexx/5.2.0/

- Java/OpenJDK
  - Try to get the installation package that contains *JavaFX*
  - From Amazon, Azul, IBM, Microsoft, Oracle, …
- BSF4ooRexx850 (external ooRexx function and class package)

  https://sourceforge.net/projects/bsf4oorexx/files/GA/BSF4ooRexx-850.20240304-GA/
  - Run and study samples that start with “1-” in BSF4ooRexx850/samples
- Slides introducing the ooRexx-Java bridge
  - https://wi.wu.ac.at/rgf/wu/lehre/autojava/material/foils/

BSF4ooRexx: Roundup, 1/2

- External Rexx function package
  - BSF4ooRexx version *850* needs at least Java *8* or later, and ooRexx *5.0* or later
  - Allows interacting with Java classes and objects
- "**BSF.CLS**"
  - Camouflages Java as ooRexx
  - Allows easy creation of Java objects
    - Java class name *must be fully qualified and in exact case*
  - Allows sending ooRexx messages to Java objects
    - No strict casing, no strict typing necessary!

BSF4ooRexx: Roundup, 2/2

- BSFCreateRexxProxy()
  - Wraps up an ooRexx object in a Java object
  - Allows to send messages to ooRexx from Java
  - Very powerful if used with Java interface classes or Java abstract classes
    - Java abstract methods can be implemented in ooRexx!