Rexx Symposium 1998

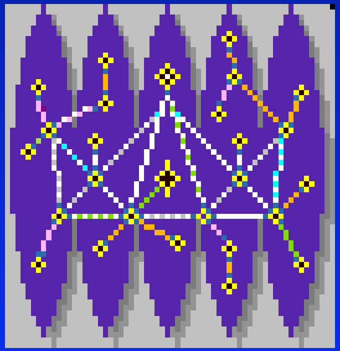
NetRexx News and Overview

http://www2.hursley.ibm.com/netrexx

Mike Cowlishaw

IBM Fellow

mfc@uk.ibm.com



Overview

- Brief introduction to NetRexx
- NetRexx 1.1 enhancements
- NetRexx News
- Questions?

What is NetRexx?

- A complete *alternative* to the Java language, for writing classes for the Java virtual machine
- Based on the simple syntax of Rexx, with Rexx decimal arithmetic
- Fully exploits the Java object model, exception handling, and binary arithmetic
- Automates type selection and declaration
- Simplified by removal of historical quirks

NetRexx Java implementation

- Current implementation first ***translates*** NetRexx to accessible Java source code
- Is written in NetRexx, so runs on any Java platform
- Any class written in Java can be used
  - GUI, TCP/IP, I/O, DataBase, *etc.*
- Anything you could write in Java can be written in NetRexx

. . . and it's free

NetRexx programs

toast.nrx

```
/* This wishes you good health. */
say 'Cheers!'
```

Control constructs

```
if  answer='yes'  then say 'OK!'
                  else say 'shucks'

loop  i=0  for  mystring.length
   say  i':'  mystring[i]
   end i
```

also `do..end` *for simple grouping,*
and `select..when..otherwise..end`

Arithmetic

- Preferred arithmetic is from ANSI Rexx
- Decimal, just one type of number
  - follows human rules (2 * 1.20 is 2.40)
  - gives exact results when expected (***e.g.***, for 0.1, 0.3)
  - no overflow at binary boundaries
  - arbitrary precision

```
numeric digits 300
say 1/7
```

numeric digits 300

```
0.14285714285714285714285714285714
2857142857142857142857142857142857
1428571428571428571428571428571428
5714285714285714285714285714285714
2857142857142857142857142857142857
1428571428571428571428571428571428
5714285714285714285714285714285714
2857142857142857142857142857142857
142857142857142857142857142857
```

Binary classes and methods

- The **binary** keyword instructs the compiler to use native (binary) arithmetic types and operations

  (boolean, byte, short, int, long, float, *etc.*)

- Achieves the full speed of the Java Virtual Machine

- No performance penalty for using NetRexx instead of Java

Explicit typing

- Casting/conversions use the ***blank*** (concatenation) ***operator***
- Consistently extends to method arguments

```
number=int 7*y     -- number is an int
number2=int        -- variable declaration

method size(x=int, y=int, depth=int 3)
```

Other features from Rexx

- Case-insensitivity
- Parse
- Trace (methods, all, results)

```
2 *=*   number=1/7
  >v> number "0.142857143"
3 *=*   parse number before '.' after
  >v> before "0"
  >v> after "142857143"
4 *=*   say after'.'before
  >>> "142857143.0"
```

Exceptions

- Semantics from Java
- Generalized and simplified syntax (extends existing control constructs)

```
say 'Please enter a number:'
number=ask    -- read a line
do
  say 'reciprocal is:' 1/number
catch Exception
  say 'Sorry, could not divide'-
      '"'number'" into 1'
end
```

NetRexx 1.1 enhancements

- All documented in *The NetRexx Supplement*
  - see: http://www2.hursley.ibm.com/netrexx

- Array initializers
  Adapter classes
  Deprecation
  Hexadecimal and binary numbers
  JavaBean properties
  Minor and Dependent (Inner) classes
  Miscellaneous improvements

NetRexx 1.1 enhancements (2)

- Array initializers
  - define the type and content of an array

```
x=[1, 2, 3, 4, 5]
```

  - may be multidimensional: [[1, 2], [3, 4]]
- Adapter classes
  - fill in event handler methods for Java 1.1 event model

```
class  Macavity   adapter   implements MouseListener
```

  - see the Scribble example on the NetRexx WWW site

NetRexx 1.1 enhancements (3)

- The **deprecated** keyword
  - Indicates that a class, method or property is deprecated: a better alternative is available

- Hexadecimal and binary numbers
  - specifies an integer in hexadecimal or binary notation
  - generalizes Java notation to self-defining signed numbers

```
0x08  =>  8
0x8F  =>  143
2x8F  =>  -113

0b1000  => 8
```

NetRexx 1.1 enhancements (4)

- JavaBean (indirect) properties
  - Properties (instance variables) that are private, accessed indirectly through conventionally-named methods

```
properties indirect
    filling=Color.red
```

generates (or checks):

```
method getFilling returns java.awt.Color
     return filling
method setFilling($1=java.awt.Color)
     filling=$1
```

NetRexx 1.1 enhancements (5)

- Minor and Dependent classes

```
class Foo
  x=Bar()
  y=Foo.Bar null
  z='Hello'
  x.Counter

class Foo.Bar dependent extends AnOther
  method Counter
    say  parent.z
```

See also the 'Buttons' example...

NetRexx 1.1 enhancements (6)

- Miscellaneous enhancements:
  - Binary methods
  - Shared classes, properties, and methods
  - Transient properties (not saved when an object is made persistent)
  - String.class (returns java.lang.Class object)
  - sourceline
  - new Options:
    - comments
    - explicit
    - sourcedir
    - symbols
    - trace1, trace2

NetRexx News

- Visual editors and builders now appearing
  - Dion Gillard's *Visual NetRexx*
  - Wingsoft's *NetRexx Interactive Development Environment*
  - Martin Lafaix's *devPad*
- Example applications and code
  - Freely available code
  - Documentation generators
  - Pipes for NetRexx
  - MaxBase and RxFile
  - *etc.*
- Documentation, tutorial, *etc.*

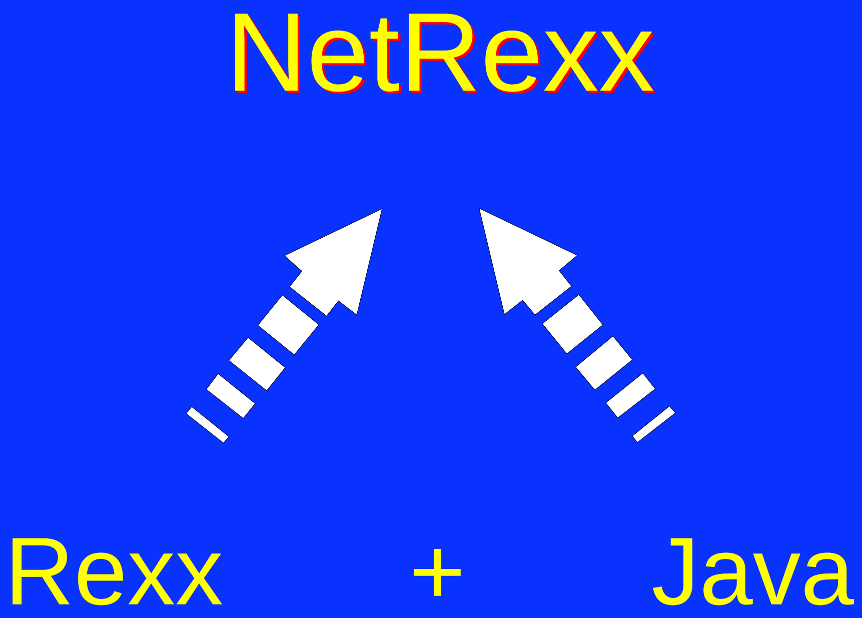
NetRexx News from IBM

- NetRexx released with VM/ESA Version 2 R3.0
  - *same binaries as every other platform*
- Redbook: *Creating Java Applications Using NetRexx*
  - *recommended!*
- Java Development Kit and platform plans
- Server-side scripting
- VisualAge for Java NetRexx prototype

Summary

- A blend of Rexx and Java
  - scripting ***and*** application development
  - a truly general-purpose language
- Both decimal and binary arithmetic
- High productivity and simplicity
  - Java source for a typical class has 35% more tokens than NetRexx
- Designed for ***users***, not compilers.

http://www2.hursley.ibm.com/netrexx/



NetRexx

Rexx + Java

Strong typing doesn’t need extra typing