Windows-Automatisation 4

Object Rexx (“ooRexx”) vs. MS Visual Basic Script ("VBScript")

Business Programming 1 | **Business Programming 2**

Basics, Parsing	Commands, APIs	Window-Automatisation, Web-Scripting	Security, Debugging	Graphical User Interfaces (GUI), Sockets, ...

Prof. Rony G. Flatscher

MS Visual BASIC (Script/Application) ....

- Extremely popular on Windows for OLE programming
- Different closely related versions
  - VBS a.k.a. VBScript (Visual BASIC script)
  - VBA (Visual BASIC for Applications)
- Searching the Internet with the keywords "*vba ole someApp/Method*"
  - Usually yields many hits with VBScript/VBA code snippets
  - Therefore important to know how to translate VBScript/VBA <---> ooRexx
- A comment on VB.NET
  - Visual BASIC for .NET
  - Resembles VBScript and VBA, but is not compatible

ooRexx vs. VBScript, 1

ooRexx

- Message operator

  `~` (Tilde)

- Continuation character

  `,` (comma) or `-` (dash)

- String concatenation

  ` ` (space between strings)

  `||` (2 vertical bars)

  var1"some string"*var2* (abuttal)

- Defining variables

  Just denote the name in an assignment

VBScript

- "Message" (dereference) operator

  `.` (dot)

- Continuation character

  `_` (underline)

- String concatenation

  `&` (ampersand)

- Defining variables

  `DIM` var_names

Prof. Rony G. Flatscher

ooRexx vs. VBScript, 2

ooRexx

- Line comment

  `--` (2 dashes)

- Block (multi-line) comments

  `/* ... */`

  - May span multiple lines
  - May be nested

VBScript

- Line comment

  `'` (apostroph)

  `REM`

  - Abbreviation for REMark
  - No statement before it allowed

  `: REM`

  - If following a statement in the same line, must be preceeded by a column surrounded by a space

ooRexx vs. VBScript, 3

ooRexx

- Calling a procedure

```
CALL proc1 a1, a2, a3
```

- Calling a function

```
a=proc1(a1, a2, a3)
```

or:

```
CALL proc1 a1, a2, a3
a=result
```

VBScript

- Calling a procedure

```
CALL proc1(a1, a2, a3)
```

or:

```
proc1 a1, a2, a3
```

- Calling a function

```
a=proc1(a1, a2, a3)
```


Prof. Rony G. Flatscher

ooRexx vs. VBScript, 4

ooRexx

- Calling a function

```
a=proc1( , , a3)
```

or:

```
CALL proc1 , , a3
a=result
```

VBScript

- Calling a function

```
a=proc1(a1, a2, a3)
```

- Calling a function using named arguments, e.g.

```
a=proc1( a3 := "Das 3. Argument!" )
```

ooRexx vs. VBScript, 5

ooRexx

- Defining a procedure

```
proc1: procedure
  use arg a1, a2, a3
  say "a1="a1 "a2="a2 "a3="a3
  return
```

or:

```
::routine proc1
  use arg a1, a2, a3
  say "a1="a1 "a2="a2 "a3="a3
```

VBScript

- Defining a procedure

```
Sub proc1(a1, a2, a3)
   MsgBox "a1=" & a1 & " a2=" & a2 _
          " a3=" & a3
End Sub
```

ooRexx vs. VBScript, 6

ooRexx

- Defining a function

```
proc1: procedure
  use arg a1, a2, a3
  return a1 || a2 || a3
```

or:

```
::routine proc1
  use arg a1, a2, a3
  return a1 || a2 || a3
```

VBScript

- Defining a function

```
Func proc1(a1, a2, a3)
   proc1=a1 & a2 & a3
End Func
```

ooRexx vs. VBScript, 7

ooRexx

- Address structure/object:

```
MyLabel~Height  = 2000
MyLabel~Width   = 2000
MyLabel~Caption = "This is MyLabel"
```

- or (to save a little bit of typing):

```
m=MyLabel
m~Height  = 2000
m~Width   = 2000
m~Caption = "This is MyLabel"
```

VBScript

- Address structure/object: WITH statement

```
With MyLabel
   .Height  = 2000
   .Width   = 2000
   .Caption = "This is MyLabel"
End With
```

- without WITH statement

```
MyLabel.Height  = 2000
MyLabel.Width   = 2000
MyLabel.Caption = "This is MyLabel"
```

ooRexx vs. VBScript, 8

```
' VBScript: "counter.vbs"
dim MyVar
Set MyVar = createObject("Some.Counter")
wscript.echo "Counter: " & MyVar.counter
call wscript.echo( "Counter: " & MyVar.increment )
```

```
-- REXX: "counter.rex"
MyVar = .OLEObject~new("Some.Counter")
wscript~echo( "Counter:" MyVar~counter )
wscript~echo( "Counter:" MyVar~increment )
```