ooRexx on MacOSX: 2007 Update

REXX on the Mac

René Vincent Jansen
RexxLA Tampa, Florida
2007
Mac Package announced at Austin

(All examples made and tested on this version, running under Max OSX 10.4.9 on Intel and Power PC)
ooRexx 3.1.1 Downloads

as of May 1st, 2007

281 downloads:
139 * i386
142 * PowerPC
Specific Mac OSX problems

- System V IPC Shared Memory API implementation
- Have to modify /etc/rc sysctl calls
- This was stopping distribution of ooRexx for MacOSX last symposium, but has since been debugged
- Problem is hopefully gone in ooRexx 4.0
- This is using sockets instead of shared memory for IPC
sysctl -w kern.sysv.shmmax=4194304
kern.sysv.shmmin=1
kern.sysv.shmmni=32
kern.sysv.shmseg=8
kern.sysv.shmall=1024

While this is still the only solution, it is ugly and not suitable for Mac user base

It was also needed for PostgreSQL, but they seem to have relaxed the numbers.
Changes for installer package

Need to softlink to /usr/bin and /usr/lib
  Single most often asked question was:
  “I ran the installer and Rexx cannot be found” - so admonishing the user to put /
  opt/ooRexx/bin on the $PATH does not really work

  And: some packages run only languages that can be found on the standard path

Need to provide a single binary package for both PPC and Intel
ooRexx 3.1.2 OSX Status

Builds successfully from the source distribution

Intel Package (including BSF4Rexx) is working

Includes one patch over 3.1.2

PPC Package will be ready early June 2007

(By the way, the PPC is water cooled)
BSF4Rexx4Mac

Working

And included & working in 3.1.1 installer

Since about November of 2006

Cross-platform GUI for ooRexx - Complete Rexx-Java integration

Totally integrates Java and Object Rexx

Due to different Java VM implementations at the moment only runnable from Java, using the included *rexxj* command.

Special support added for Java classes that have originated in NetRexx - automatic, castless Rexx-NetRexx integration.
util = .bsf~bsf.import("org.netrexx.midms.util.Dmsutil")
Datum = .bsf~bsf.import("org.netrexx.midms.util.Datum")

util~printBanner('Hello NetRexx')

Hello NetRexx
started by rvjansen on Mac OS X 10.4.9 i386 - Java version 1.5.0_07
current directory: /Users/rvjansen

d=datum~new
say "Timestamp :" Datum~getTimestamp
say Datum~getLogTimestamp
say d~getDayOfWeekInMonth
say d~isWeekend

Timestamp :20070403213840409
2007-04-03 21:38:40.409
Wednesday 1 in April
Is Weekend: N
jdbc = .bsf~bsf.import("java.lang.Class")
driverMgr = .bsf~bsf.import("java.sql.DriverManager")

-- instantiate jdbc driver
jdbc~forname('org.postgresql.Driver')~newinstance

-- make the dbms connection and open a statement
stmt = driverMgr~getConnection('jdbc:postgresql')~createStatement

-- specify query and execute to get result set
rs = stmt~executeQuery('select * from obj_x_obj')
do while rs~next
  say rs~getString("OBJ_OID")
end

::requires 'bsf.cls'
Most important applications on MacOSX are scriptable, (think Windows automation), using AppleScript, an English-like scripting language. There are various ways to start an Applescript command using Rexx.

Some examples follow.
Sending an IM Query Users

AppleScript

There is just one command for Skype API, but it is a very powerful command, because you can send any the command strings as specified in Skype API protocol documentation to control Skype or request information.

Examples

Full Skype API

Using Rexx & OSASCRIPSCRIPT
userid = 'venetiamaduro'
message = 'hello'

cmdline = "osascript -e 'tell application ""Skype""' ",
"-e 'send command "MESSAGE userid" message"' script name "msg", 
"-e 'end tell'"

cmdline
Get a list of your Skype Contacts

And have them available in ooRexx variables
Plan to integrate using Address

- It looks like it would be useful to integrate with OS X to enable the address environment statement for some applications
  - address iTunes
  - address iPhoto
  - address Skype
- variables could be added to the pool for specific use of these scripts
Writing ooRexx instead of Applescript

Mac OS X Automator

Using Rexx to start Applescript is fun, but \textit{way nicer} of course is to write ooRexxx in order to avoid writing AppleScript.
Automator

It is like a 21st century CMS Pipelines!

Assisted creation and sequencing of library actions

Actions can be written in ooRexx for MacOSX
Necessary Preparations

- Run an ooRexx version that contains the -e startup argument
  - Which will be in the new ooRexx Mac OSX installer package
- Add Symbolic links to the default /etc/* hierarchy
- Add Rexx as a shell scripting language to automator configuration
The -e startup argument to Rexx

- Some scripting languages and Unix shells have this traditionally
- Mostly used for quick oneliners in shell scripts
- Feeds a program from the command line into the interpreter
- Rexx -e “say x2d(ff)”
  - replaces the “interpret args(0)” script (just as well, ‘say’ is a program)
  - need to do creative quoting as the shell has to be taken into account
Symbolic links

- The executables and library locations follow the suggested Linux file system layout for optional packages /opt/*
- This is not in standard user path environment variable
- Automator expects the shells and scripting languages in the standard /etc/* hierarchy
- The solution is to craft a script that writes a set of `ln -s` commands
  - Looking at the moment how to run this from the installer package
Script to add symbolic links

```
"cd /opt/ooRexx/lib/ooRexx"
"ls | rxqueue"

do queued()
    parse pull line
    do i=1 to line~words()
        say "sudo ln -s /opt/ooRexx/lib/ooRexx/"line~word(i) "/usr/lib/"line~word(i)
    end
end

"cd /opt/ooRexx/bin"
"ls | rxqueue"

do queued()
    parse pull line
    do i=1 to line~words()
        say "sudo ln -s /opt/ooRexx/bin/"line~word(i) "/usr/bin/"line~word(i)
    end
end
```
Add to Shells.plist

- Rexx needs to be added to a file that Automator reads on startup
- It is defined as every other shell or interpreter

This file is located in

/System/Library/Automator/Run Shell Script.action/
Contents/Resources/Shells.plist

Remember to copy it using “sudo cp” as it needs root authority.
Add this set of lines and restart Automator (if it was already started)

```
<key>/usr/bin/rexx</key>
<dict>
  <key>args</key>
  <array>
    <string>-e</string>
    <string>%</string>
    <string>--</string>
  </array>
  <key>script</key>
  <array>
    <string>do while lines()&#10;say linein()&#10;end</string>
    <string>parse arg arg1 arg2 arg3</string>
  </array>
</dict>
```
Demo of some features
Thank You!

Get In Touch

rvjansen@xs4all.nl
rene.vincent.jansen@gmail.com