BUILDING TESTING DEBUGGING PACKAGING

BUILDING OOREXX

René Vincent Jansen

27th International Rexx Language Symposium, Tampa 2016

AGENDA

- Getting the code
- Building
- Testing
- Debugging
- Packaging

GETTING THE CODE

GETTING THE CODE FROM SOURCEFORGE

- need: subversion (svn) client
- need: cmake
- need: make (or nmake on windows)
- need: ncurses
- https://sourceforge.net/projects/oorexx/
  - here you can find where to point svn to:
  - svn checkout svn://svn.code.sf.net/p/oorexx/code-0/main/trunk oorexx-code-0

CMAKE

CMAKE

- modern form of autotools
  - a way to adapt C/C++ project builds to different platforms
- performs out-of-source builds
- make a build directory and cd into it
  - cmake path-to-source -options
- The whole build procedure (all platforms) is in the file **CmakeLists.txt**

CMAKE EXAMPLE

```
mkdir -p ../build

cd ../build

cmake -DBUILD_DEB=1 -DOS_DIST=ubuntu1604 -DCMAKE_BUILD_TYPE=RELEASE
$WORKSPACE

make clean          # make sure rexx picks up the current build date

make
```

MAKE

MAKE

- cmake generates makefiles
- make is a build tool that (re)builds programs if the source is newer
- you can tell it about dependencies
- oldest and most standard build tool
- gnu make is nearly everywhere

RASPBERRY PI NOTES

Building on the Raspberry Pi

Raspbian Wheezy

The build needs cmake, at least GNU G++ 4.8.2 and the ncurses development library
The cmake in the raspbian wheeze distribution is too old; it needs to be built from source. Download and untar the 3.5.2 source package; then run ./bootstrap && make && make install - this will take care of make.
The C++ compiler on Wheezy is 4.6.3, it is too old and has severe bugs in template handling. From ooRexx 5.00 on, templates are required.
`sudo apt-get install gcc-4.8-base sudo apt-get install g++-4.8`
Finally, for a build the ncurses development header files are required. They can be installed like this:
`sudo apt-get install libcurses5-dev`
After this, do a standard cmake out-of source build

Raspbian Jessie

`sudo apt-get install cmake`

LINUX ON THE MAINFRAME

After provisioning the virtual machine image:

```
sudo zypper install cmake
sudo zypper install ncurses-devel
```

and do a standard out-of-source cmake build.

Note that before a sudo make install, processes started using the rexx executable from the bin build directory do not disappear and need to be dispatched with kill -9. After installing rexx, this problem goes away. Note that in some virtual images there are problems involving the firewall and the rxapi daemon.

WINDOWS

Prerequisites

- Subversion client (svn) from e.g. https://sourceforge.net/projects/win32svn/
- Cmake 3.2.3 from http://cmake.org
- MS Visual Studio Express: MS Visual Community 2013 from https://www.visualstudio.com/en-us/products/visual-studio-community-vs.aspx

Probably down the line you will have to install NSIS, Xalan and Xerces, but the above is enough to build a local copy and run it. The ooRexx documentation is a different issue and needs other tools.

Environment variables

This set of environment variables is suggested; match this to your local environment

```
set TEST_DIR=C:\Users\rvjansen\oorexxtest
set SRC_DRV=C:
set BLD_DIR=\Users\rvjansen\oorexxbuild
set REXX_BUILD_HOME=%SRC_DRV%%BLD_DIR%
set REXX_HOME=%SRC_DRV%%BLD_DIR%
call "C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\vcvarsall.bat" x64
set INCLUDE=%INCLUDE%;c:\Program Files (x86)\Microsoft SDKs\Windows\v7.1A
\include;
path c:\NSIS;%REXX_BUILD_HOME%\bin;%PATH%;c:\Xalan\bin;c:\Xerces\bin;%TEST_DIR%;
%TEST_DIR%\framework;
```

WINDOWS (CONTINUED)

Check out the code from the Subversion repository
Checked out trunk to \Users\rvjansen\oorexx with:

```
svn co http://svn.code.sf.net/p/oorexx/code-0/main/trunk .
```

(if you want to commit stuff from here, you need the svn+ssh notation and your SF password)

Configure the build with cmake
Then switch to the \Users\rvjansen\oorexxbuild directory and issued:

```
cmake ..\oorexx -G "NMake Makefiles”
```

This tells cmake to generate the makefiles and not the default Visual Studio project. Important is to clean out that directory every time something goes wrong, because cmake seems easy to confuse. Most important here is that the compiler is happy in finding the include files and libraries it needs.

Run the build
Afterwards, in that same directory,

```
nmake
```

This builds the system in the bin directory of the oorexxbuild directory. It is runnable in that state.

TEST

TEST

- the ooRexx source comes with its
  - own testing tool (ooRexxUnit)
  - own testing suite
- run tests:
  - **rexx ./testOORexx.rex -s -X native_API -x socketClass**

TEST SUITE

- ooRexxUnit is modeled on JUnit (and now xUnit)
- will finish a test suite and gives results afterwards
- this is useful because you have immediate insight in which classes pass and which classes fail
- after a source code update, all supported platforms should be tested immediately

DEBUG

DEBUGGING

- make sure to build without -DCBUILD_TYPE=Release
- a build without this is non-optimized and has symbols for debugging
- you can use gdb to set breakpoints
- you can also add print statements

JENKINS

JENKINS

Automate your software builds
Distributed Master/Slave Model
Compatibility with existing systems/protocols
Build, deploy, test, report
Plugins for various environments

JENKINS TO Z/OS

- We run Jenkins from a Tomcat instance … anywhere
- In this case an existing build server on Linux
- Jenkins is an easy tool
  - Master
  - Slaves
  - Credentials
  - Jobs

JENKINS' MAIN DASHBOARD

Jenkins

search

René Vincent Jansen | log out

Jenkins ›

ENABLE AUTO REFRESH

- New Item
- People
- Build History
- Manage Jenkins
- Credentials
- My Views

add description

Build Queue (1)

ooRexx-Linux-Mint17

Build Executor Status

IBMZ
1 Idle

ams-01
1 Idle
2 Idle
3 Idle
4 Idle

ams-02
1 Idle

ams-03 (offline)

ams-04
1 Idle

ide-01 (offline)

ubuntu16
1 Idle

Alle +

S	W	Name ↓	Last Success	Last Failure	Last Duration
		netrexx-commit	2 mo 2 days - #3	N/A	24 sec
		netrexx-commit-Z	2 mo 1 day - #10	N/A	17 sec
		ooRexx-Linux-Mint17	26 days - #24	1 mo 29 days - #12	2 min 51 sec
		ooRexx-macOS-build	9 days 16 hr - #81	N/A	20 sec
		ooRexx-macOS-test	9 days 16 hr - #54	9 days 16 hr - #53	3 min 58 sec
		ooRexx-OpenSUSE-Tumbleweed-X86_64-build	9 days 4 hr - #48	19 days - #46	4 min 51 sec
		ooRexx-OpenSUSE-Tumbleweed-X86_64-test	1 mo 18 days - #15	9 days 4 hr - #19	3 min 9 sec
		ooRexx-Raspbian-Jessie-build	9 days 16 hr - #59	1 mo 3 days - #51	9 min 56 sec
		ooRexx-Raspbian-Jessie-test	9 days 16 hr - #66	9 days 23 hr - #65	5 min 1 sec
		ooRexx-Raspbian-Wheezy-build	9 days 16 hr - #47	2 mo 0 days - #21	45 sec
		ooRexx-Raspbian-Wheezy-test	9 days 16 hr - #13	9 days 22 hr - #12	7 min 28 sec
		oorexx-ubuntu16-build	9 days 15 hr - #27	N/A	2 min 21 sec
		oorexx-ubuntu16-test	9 days 15 hr - #28	9 days 22 hr - #27	2 min 50 sec
		ooRexx-Z-build	9 days 16 hr - #46	1 mo 3 days - #36	13 sec
		ooRexx-Z-test	9 days 16 hr - #28	9 days 23 hr - #27	3 min 1 sec

Icon: S M L

Legend | RSS for all | RSS for failures | RSS for just latest builds

CONFIGURE A SLAVE LPAR

localhost

Jenkins

search

Rene Jansen | log out

Jenkins › Nodes › raspi

- Back to List
- Status
- Delete Slave
- Configure
- Build History
- Load Statistics
- Log

Build Executor Status

Name	raspi
Description	raspi2
# of executors	1
Remote root directory	/home/pi
Labels	
Usage	Utilize this node as much as possible
Launch method	Launch slave agents on Unix machines via SSH
Host	10.0.0.53
Credentials	pi/****** Add
Advanced...	
Availability	Keep this slave on-line as much as possible

Node Properties

- [ ] Environment variables
- [ ] Tool Locations

CONFIGURE A JOB

Project name: netrexx

Description

[Escaped HTML] Preview

- [ ] Discard Old Builds
- [ ] This build is parameterized
- [ ] Disable Build (No new builds will be executed until the project is re-enabled.)
- [ ] Execute concurrent builds if necessary
- [ ] Restrict where this project can be run

Advanced Project Options

Advanced...

Source Code Management

- ( ) None
- ( ) CVS
- ( ) CVS Projectset
- (•) Subversion

Modules

Repository URL: https://svn.kenai.com/svn/netrexx~netrexxc-repo/netrexxc/trunk

Credentials: rvjansen/****** (kenai userid) Add

SPECIFY WHAT THE JOB RUNS

Build

Execute shell

Command

```
make
```

See the list of available environment variables

Delete

Add build step

Post-build Actions

Add post-build action

Save Apply

PUBLISH THE PACKAGES

- Install the "**Publish over SSH**" plugin
- Use credentials from Jenkins, not from slave machine

ANY QUESTIONS?
THANK YOU!