Software Development on Linux Systems

4002-XXX-XX

By

Cody Van De Mark

Today

- Exam/Practical Review

Review

- Topics:
  - Gratis vs Libre
  - Open Source Licenses
  - Creative Commons Types
  - Open Source Business Approaches
  - Package installation
  - Code compiling
  - Bash scripting
  - Version Control
  - Man Pages

Review

- Gratis vs Libre
  - Libre - Free as in Freedom
    - No restrictions; Free to use, modify and distribute
  - Gratis - Free as in Beer
    - No price; Not free to use, modify and distribute

Review

- Open Source Licenses
  - GPL - Distribute modified code as long as derivative is compatible with the original license; No linking to proprietary software
  - LGPL - Distribute modified source code as long as derivative is compatible with the original license; Can **link** with proprietary software, but all changes must be provided under original license
  - Apache - Distribute with proprietary software as long as open source code is attributed; No implication you wrote the code or that it was endorsed by the foundation that made this license

Review

- Open Source Licenses
  - BSD - Distribute with proprietary software as long as open source code is attributed; No implication of open source developers, organization or software endorsement
  - CCDL - Distribute modified code if license is the same; Can distribute with proprietary software if the open source code is attributed; Licenses each file individually
  - Eclipse - Distribute modified code if license is compatible with original; Distribute with proprietary software as long as open source code is included with license; Grants patent rights code under a patent

Review

- Open Source Licenses
  - MIT - Distribute modified software as long as the open source code is included with its license; Distribute with software of any license as long as the open source code is included with its license
  - MPL - Distribute modified code as long as it uses the original license; Distribute with proprietary software if the open source code is attributed with its license; Distribute under any license as long as the original files are preserved and each file is marked as the original license;Not allowed to distribute executables without the source code and license

Review

- Creative Commons Licenses
  - Attribution - Must credit author/licensor as license requires
  - Noncommercial - Must be used only for noncommercial purposes
  - No Derivatives - Must use verbatim; No modifications or by-products
  - Share Alike - Must distribute derivatives under exact same license

Review

- Business Approaches
  - Donations - Requires non-profit status and can have inconsistent income
  - Merchandise - Selling items to consumers to promote your project and increase income; Maintains non-profit status
  - Freemium/Open Core - Offering a free version and a paid proprietary version
  - Partner/Referral - Coexistence through a mutual agreement with another party
  - Advertising - Offsetting cost through promotion of another product; Maintains non-profit status

Review

- Business Approaches
  - Service Provider - Offering a service leveraging your open source software
  - Support Provider - Offering consulting/aid contracts for users of certain software
  - License Exemption - Offering special permission to a buyer to use software outside of the license terms, but within the terms you specify for them
  - Business Catalyst - When your software acts as a stimulant for your business approach

Review

- Package Installation
  - Fedora
    - su -c 'yum install packageName'

      AND

    - su -c 'yum groupinstall “Group Name”'
  - Ubuntu
    - sudo apt-get install packageName

Review

- Compiling
  - C - gcc file.c -o executableName
  - C++ - g++ file.cpp -o executableName
  - Java - javac file.java
  - C# - gmcs file.cs
  - C++ with libraries - g++ file.cpp -o executableName -lLibName

Review

- Execution
  - C - ./executableName
  - C++ - ./executableName
  - Java - java executableName
  - Python - python file.py
    or ./file.py if the script has **#!/usr/bin/env python**
  - Perl - perl file.pl
    of ./file.pl if the script has **#!/usr/bin/env perl**
  - C# - ./file.exe

Review

- Bash Scripting
  - #!/bin/bash - Required header for bash scripts
  - Variables - varName="completed";
    echo $varName;
  - Array - declare -A arrayName;
    arrayName['juice']="grape";
    arrayName['wine']="white";

    @ means at all parameters
    ! means get array keys, not values

Review

- Bash Scripting
  - Command Execution - `ls -l`;

    varName=`ls -l`;
  - Operators -

-eq	: equal to (num)
-ne	: not equal to (num)
-lt	: less than (num)
-le	: less than equal to (num)
-gt	: greater than (num)
-ge	: greater than equal to (num)
=	: equal to (string)
!=	: not equal to (string)
-z	: is empty (string)
-n	: is not empty (string)

Review

- Bash Scripting
  - If statements -

```
if [ $varName = “hi” ]
then
    echo $varName;
elif [ -z $varName ]
then
    echo “Empty string”;
else
    echo “String was not hi”;
fi
```

Review

- Bash Scripting
  - If Statements -

```
if  [  $varName -eq 0  ]
then
      echo $varName;
elif [ $varName -gt 10  ]
then
      echo $varName;
else
      echo “variable was not greater than 10”;
fi
```

Review

- Bash Scripting
  - Arguments - Taken in as ${1}, ${2}, ${3}, etc from the command line

    Example: ./test.sh --no-output 1
  - Processing Arguments

```
if [ “${1}” = “--no-output” ]
then
      output=”false”;
elif [ “${1}” -eq 1 ]
      recompile=”true”;
fi
```

Review

- Bash Scripting
  - Loops -

```
for item in “${arrayName[@]}”
do
    echo $item;
done
```

```
for item in “${!arrayName[@]}”
do
    echo $item;
    echo ${arrayName[$item]};
done
```

Review

- Bash Scripting
  - Loops - while [$varName -lt 5];
    do
        let varName=varName+1;
    done

    until [$varName -eq 0];
    do
        let varName=varName-1;
    done

Review

- Bash Scripting
  - chmod - chmod 755 file.py file.pl file.sh

    644 : read access (for others)
    755 : read/execute access (for others)
  - uname - uname -m
    prints the processor architecture

    i686 : 32 bit
    i386 : older 32 bit
    x86_64 : 64 bit
  - Execution - ./fileName.sh

Review

- In the exam, you will have to write a script to compile all of the files given to you in their language
- You will have to execute them inside the script with execution back ticks and store the results of each in an array
- You will need to modify the permissions of files
- You will need to loop through the array and output the results of each
- You will also need an if statement to gather the architecture

Review

- You really should practice the compiling, executing and running from a script before the practical
- You may use man pages during the exam, but it may not necessarily be helpful
- Remember space sensitivity in variables, if statements and loops
- Remember you can set a variable to a command with back ticks
  IE: varName=`ls -l`;

Review

- Version Control
  - In the exam, you can use git or bzr
  - You will need to initialize a new branch, set up who you are, add your files and commit them
  - You will also need to merge files through version control

Version Control

- Bzr
  - Whoami - bzr whoami 'first lname <me@rit.edu>'
  - Initialize branch - bzr init branch-name
  - Add files - bzr add file.sh file.pl file.py
  - Commit - bzr commit -m 'new message'
  - Merge - bzr merge ../path/to/other/directory

Version Control

- Git
  - Config - git config --global user.name “first last”
    git config --global user.email “me@rit.edu”
  - Initialize branch - git init branchName
  - Add files - git add file.sh file.pl file.py
  - Commit - git commit -m 'new message'
  - Merge
    - git remote add otherBranchName ../path/to/other/branch
      git fetch otherBranchName
      git merge otherBranchName/master

Review

- Version Control
  - You really should practice version control before the practice, specifically merging
  - Again, you may use man pages, but they may not be useful to you

Review

- Man Pages
  - Title Header

    .TH [name] [section #] [center footer] [left footer] [center header]

    Example:

    .TH foo 1 “January 2012” “1.1 Linux Release” “User Manuals”

  - Section Header

    .SH [section name]

    Example:

    .SH NAME

Review

- Man Pages
  - Subsection

    .SS [section name]

    Example:

    .SH OPTIONS
    .SS “File Options”

  - Item Name

    .IP [item name]

    Example:

    .IP --debug

Review

- Man Pages
  - Bold Text
    .b [word or words]

    Example:

    .B This is bold text
  - Italic Text
    .I [word or words]

    Example:

    .I This is italic text

Review

- Man Page Example

.TH practical 1 “January 2012” “1.2 Linux Version” “User Manuals”

.SH NAME
practical \- Application to compile all the practical code automatically and print the execution results

.SH OPTIONS
.IP --debug
Prints debug information about the program
.IP –no-output
Compiles all the files, but will not display any of the output at the end

.SH “SEE ALSO”
.B gcc(1), g++(1), javac(1), gmcs(1), python(1), perl(1)

Review

- Man Pages
  - You will need to build a short man page during the exam to describe the script you wrote
  - You should practice this before the exam