Lowest Common Denominator Coding
With vi(1) and sh(1)

Michael Dexter

Portland Linux/Unix Group
July 3rd, 2014
Lowest Common Denominator

vi - screen-oriented (visual) display editor

This utility shall be provided on systems that both support the User Portability Utilities option and define the POSIX2_CHAR_TERM symbol. On other systems it is optional.

On virtually all non-"embedded*" POSIX systems

http://pubs.opengroup.org/onlinepubs/9699919799/utilities/vi.html

* See: Internet of SH*T Things
Lowest Common Denominator

/bin/sh – runs the world

... boots most systems

... glues the system together
Lowest Common Denominator

# which python
# python: Command not found.

Always Available

“The Ruby Way or the Highway” will not get you far with clients
vi Crash Course

ESC – Change Mode
  :q – Quit
  i – Insert (Mode)
  :w – Write (Often!)
  :wq – Write and Quit
  :q! – Quit Without Saving
  x – Delete Character
  dd – Delete Line
  u – Undo (One Level)
vi Crash Course
ESC is your friend

Consider one of those USB pedals...

Windows 8 “Wish we were a phone” severely reduced ESC key sizes :(

:set verbose showmode
Tells you your mode and more
Slightly More vi

J – Join line below
r – Replace character
R – Replace multiple characters
0 or ^ – Go to the beginning of a line
$ – Go to the end of a line
1G – Go to the beginning of the file
G – Go to the end of the file
A – Append to the end of a line
/ – search
Super Fancy

w – Move 1 word ahead
dw – Delete word
d4d – Delete 4 lines

:<number> – Go to line <number>

:%s/foo/bar/g
Replace all foo's with bar's (globally)
Living la vida vi

That's 90% of the commands I use!

Get the mug or smart phone app

Run through vitutor/vimtutor

OSCON: Damien Conway!
The Better™ Way

There are surely 10~100 more ways to do everything you will see

“Better” is 100% subjective
sh Scripting 101

# vi myscript.sh
<i> echo Hello World <ESC>
:wq
# sh myscript.sh
Hello World

(Note the script(1) command)
sh Scripting 101

It *is* the shell

# vi myscript.sh
<i> ls
<date <ESC>
:wq
# sh myscript.sh
<i>ls output>
<i>date output>
Variables

# sh
# foo=beer ; echo $foo
beer

Quote strings "foo" if "Hello World"
Note single vs. double quotes
DON'T FORGET TO CLOSE THEM
ELSE: PAIN, SUFFERING
Variables

Two commands in one line:

```
# foo=beer ; echo $foo
```

One command in two lines:
(within a script)

```
echo This is one very very very very very very long command to type
```
Environment Variables

# set
< list of environment variables >
...

PAGER=more
...

# export foo=bar
# echo $foo
bar
Comments and Debug Output

#!/bin/sh
# myscript.sh (c) 2014 Michael Dexter
# < license >
# This script doesn't do much
echo Let us run date
date
date
echo We got THIS far
exit # stop here while we debug
Comments and Debug Output

LINEAR IS YOUR FRIEND

Limited debugging:

# sh -xmyscript.sh
+ echo Hello World
Hello World
Comments and Debug Output

Some people, when confronted with a problem, think…

“L know, I’ll use multithreading”.

Nothhw tpe yawrve o oblems.
The Unseen: Exit Status Codes

# sh
# foo=beer ; echo $foo
beer
# echo $?  
0

0 = Success
1 or greater = Fail
The Unseen: Exit Status Codes

```
# cat foo
cat: foo: No such file or directory
# echo $?  
1
```

Fail!
The Unseen: Exit Status Codes

# cat foo ; ls

```
cat: foo: No such file or directory
```

# <ls output>

(The script continued)
The Unseen: Exit Status Codes

# . foo ; ls
.: foo: cannot open foo: No such file...
#
# echo $?  
2

(The script failed and stopped before ls)
(Be sure of what behavior you want)
The Unseen: Exit Status Codes

# sh -e ...

Exit on any error
The Unseen: Exit Status Codes

# date
# echo $?
0
# date && echo Success
Thu Jul  3 16:39:50 PDT 2014
# Success

(Report Success)
The Unseen: Exit Status Codes

# date
# echo $? 
0
# date || echo Fail
Thu Jul  3 16:39:50 PDT 2014
#

(Report Failure)
The Unseen: Exit Status Codes

“[“, Pronounced “test”

# foo=bar
# [ $foo = bar ]
# echo $?
0

# [ $foo = flabbble ]
# echo $?
1
The Unseen: Exit Status Codes

if [ foo = bar ]; then
    echo Beer can be found at bars
elif [ foo = pub ]; then
    echo Beer can be found at pubs
else
    echo Beer not found ; exit 1
fi
Readability Through Indenting

if [ foo = bar ]; then
  < conditional command >
elif [ foo = pub ]; then
  < conditional command >
else
  if [ cat = dog ]; then
    < nested conditional cmd ... >
  fi
fi
The Unseen: Exit Status Codes

case $foo in
  bar)
    echo Beer can be found at bars
  ;;
  pub)
    echo Beer can be found at pubs
esac
The Unseen: Exit Status Codes

All are variations on the same theme

Write LOTS of tests

The extra few seconds required can save you hours of debugging
for vm_found in "$host_vmdir"/*; do
    start_vm $vm_found
done

Caution: Found files with spaces will fail
Verify that over 10 arguments are safe
Functions

start_vm()
{
    <do stuff on variable $foo>
}

start_vm $1
More Hidden Things

`$?` – Most Recent Exit Status
`$1 $2 $3` – Arguments to the Command
`$*` – All Arguments to the Command
`$0` – Name of the Command
`$#` – Number of Arguments
# More Hidden Things

```bash
[ $# -gt 0 ] && shift 1
Remove command name from args.
```
Sub Shells

# thedate=$( date )
# echo $thedate
Thu Jul 3 17:16:11 PDT 2014

Else with back ticks (cannot be nested):

# thedate=`date`
Extra Credit: sed

```
sed -i " -e "s/vm_device="/vm_device="${md_device}"/"
${host_vmdir}/${vm_name}/${vm_name}.conf

Replace vm_device="" with /dev/ada0 in
/usr/local/vmrc/vm/vm0/vm0/conf
```
Extra Credit: regex and friends

Simple built-in string manipulation:

```
vm_foundname="${vm_found##*/}"  # Strip path

"/usr/local/vmrc/vm/vm0" becomes "vm0"
```
Actual Project
github.com/michaeldexter/vmrc
(massive commit pending)
This was dogfooded with PC-BSD
demo || Thanks!
(get it?)