UNIX Workshop Series: Quick-Start
Objectives

- Overview – Connecting with ssh
- Command Window Anatomy
- Command Structure
- Command Examples
- Getting Help
- Files and Directories
- Wildcards, Redirection and Pipe
- Create and edit files
Overview
Connecting with ssh

- Open a Terminal program
  - **Mac:** Applications > Utilities > Terminal
    `ssh -Y username@centos.css.udel.edu`
  - **Linux:** In local shell
    `ssh -Y username@centos.css.udel.edu`
  - **Windows:** Start Xming and PuTTY
    Create a saved session for the remote host name centos.css.udel.edu using `username`
Connecting with ssh

- First time you connect

![SSH Terminal and PuTTY Security Alert](image-url)
Unix Basics

- Multi-user
- Case-sensitive
- Bash shell, command-line
- Commands
Command Window Anatomy

Click in the title bar to bring the window to the front and make it active.
Command Window Anatomy

Login banner

Appears as the first line of a login shell.
Command Window Anatomy

Prompts

Appears at the beginning of a line and usually ends in $.
Command Window Anatomy

Place to type commands, which may have options and/or arguments.
Command Window Anatomy

Place for command response, which may be many lines long.
Typed text will appear at the cursor location.
Scroll Bar

Will appear as needed when there are more lines than fit in the window.
Use the mouse to change the window size from the default 80x24.
**Command Structure**

`command [arguments]`

- Commands are made up of the actual `command` and its `arguments`.

`command -options [arguments]`

- The arguments are further broken down into the command `options` which are single letters prefixed by a “-” and other `arguments` that identify data for the command.
Basic Command Examples

[train@centos ~]$ date
Tue Dec 14 14:11:42 EST 2010
[train@centos ~]$ cal
   December 2010
 Su Mo Tu We Th Fr Sa
  1  2  3  4  5  6  7  8  9 10 11
 12 13 14 15 16 17 18
 19 20 21 22 23 24 25
 26 27 28 29 30 31

[train@centos ~]$ hostname
centos.css.udel.edu
[train@centos ~]$ pwd
/home/train
[train@centos ~]$ whoami
train
[train@centos ~]$ ps
  PID TTY      TIME CMD
19189 pts/2   00:00:00 bash
19217 pts/2   00:00:00 ps
[train@centos ~]$ uptime
14:12:01 up 7 days, 17 min, 3 users, load average: 0.00, 0.00, 0.00
Advanced Command Examples

```
[train@centos ~]$ history
1 date
2 cal
3 hostname
4 pwd
5 whoami
6 ps
7 uptime
8 history
[train@centos ~]$ cal 10 2010
 October 2010
 Su Mo Tu We Th Fr Sa
  1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
 31
[train@centos ~]$ uname
Linux
[train@centos ~]$ uname -a
Linux centos.css.udel.edu 2.6.18-194.26.1.el5 #1 SMP Tue Nov 9 12:54:20 EST 2010 x86_64 x86_64 x86_64 GNU/Linux
[train@centos ~]$
```
At the cursor location use

- A to insert a character.
- return to enter the command.
- delete to erase character to left.
- ◀◀ to move to the left or right.
- ▲ to retrieve previous command.
- Tab for command line completion.
Command Input Keys

Summary

Control C or CTRL-c or just C-c
use the control key as a shift

Control U or CTRL-u or just C-u
use the control key as a shift

Control D or CTRL-d or just C-d
use the control key as a shift
Command Output Summary

- Lines longer than the column width are wrapped.
  
  Use the mouse in the resize handle to increase the width of the screen and unwrap the lines.

- Lines are scrolled off the top and the prompt will appear at the bottom of the screen.
  
  Use the mouse in the scroll bar to see lines that have scrolled off the screen.
Getting Help

man \textit{command}

- man is a command that formats and displays on-line manual pages for \textit{command}.

\textit{info} [\textit{subject}]

- info formats and displays online documents that are easily searchable.

\textit{apropos} \textit{keyword}

- apropos searches the whatis database for commands containing the keyword
Getting Help

- man bash

```
train@centos:~$ bash
BASH(1)

NAME
bash - GNU Bourne-Again SHell

SYNOPSIS
bash [options] [file]

COPYRIGHT
Bash is Copyright (C) 1989-2005 by the Free Software Foundation, Inc.

DESCRIPTION
Bash is an sh-compatible command language interpreter that executes
commands read from the standard input or from a file. Bash also incor-
porates useful features from the Korn and C shells (ksh and csh).

Bash is intended to be a conformant implementation of the Shell and
Utilities portion of the IEEE POSIX specification (IEEE Standard
1003.1). Bash can be configured to be POSIX-conformant by default.

OPTIONS
In addition to the single-character shell options documented in the
description of the set builtin command, bash interprets the following
```
Getting Help

- info bash

This text is a brief description of the features that are present in the Bash shell (version 3.2, 28 September 2006).


Bash contains features that appear in other popular shells, and some features that only appear in Bash. Some of the shells that Bash has borrowed concepts from are the Bourne Shell ('sh'), the Korn Shell ('ksh'), and the C-shell ('csh' and its successor, 'tcsh'). The following menu breaks the features up into categories based upon which one of these other shells inspired the feature.

This manual is meant as a brief introduction to features found in Bash. The Bash manual page should be used as the definitive reference on shell behavior.
Getting Help: Info Keys

- **delete**: back up after SPC
- **H**: ‘h’ special link to help.
- **L**: ‘l’ return from help link.
- **Q**: ‘q’ to quit.

To read linearly through all sections
Getting Help: Info Keys

- **tab**: Jump to next link or cross reference.
- **return**: Follow the link at the cursor.
- **esc** + **tab**: Back up to previous link.
- **M**: ‘m’ to go to the menu.
- **L**: ‘l’ return from link.
- **Q**: ‘q’ to quit.
Getting Help

- `apropos bash`

```bash
train@centos:$
```
```bash

disown [builtins] (1) - bash built-in commands, see bash(1)
echo [builtins] (1) - bash built-in commands, see bash(1)
enable [builtins] (1) - bash built-in commands, see bash(1)
eval [builtins] (1) - bash built-in commands, see bash(1)
exec [builtins] (1) - bash built-in commands, see bash(1)
ext [builtins] (1) - bash built-in commands, see bash(1)
export [builtins] (1) - bash built-in commands, see bash(1)
f [builtins] (1) - bash built-in commands, see bash(1)
fg [builtins] (1) - bash built-in commands, see bash(1)
getopts [builtins] (1) - bash built-in commands, see bash(1)
hash [builtins] (1) - bash built-in commands, see bash(1)
help [builtins] (1) - bash built-in commands, see bash(1)
history [builtins] (1) - bash built-in commands, see bash(1)
jobs [builtins] (1) - bash built-in commands, see bash(1)
kill [builtins] (1) - bash built-in commands, see bash(1)
let [builtins] (1) - bash built-in commands, see bash(1)
local [builtins] (1) - bash built-in commands, see bash(1)
logout [builtins] (1) - bash built-in commands, see bash(1)
popd [builtins] (1) - bash built-in commands, see bash(1)
printf [builtins] (1) - bash built-in commands, see bash(1)
push [builtins] (1) - bash built-in commands, see bash(1)
pushd [builtins] (1) - bash built-in commands, see bash(1)
read [builtins] (1) - bash built-in commands, see bash(1)
readonly [builtins] (1) - bash built-in commands, see bash(1)
```
Files

- Data in UNIX is stored in files.

Directories

- Files are organized hierarchically into directories.
- Top-level directory is called “root” represented as a slash (/).
- Every file (and directory) is listed and separated by a slash (/).
The UNIX file system is like an inverted tree.
Working with Files

```
[train@centos ~]$ ls
bessel.c  besselTest  birthtoken.txt  example.c  flowers.txt  lines
[train@centos ~]$ ls -a
  .  .bash_logout  bessel.c  example.c  lines
  ..  .bash_profile  besselTest  flowers.txt  .mozilla
.bash_history  .bashrc  birthtoken.txt  .lesshist  .Xauthority
[train@centos ~]$ ls -la
total 120
drwxr-xr-x  4 train  508 4096 Dec 14 14:30  
\n-drwxr-xr-x 10 root  root  4096 Dec 2 19:27  ..
-rw-------  1 train  student  91 Dec 14 14:11 .bash_history
-rw-r--r--  1 train  student  33 Dec 7 11:11 .bash_logout
-rw-r--r--  1 train  student  176 Dec 7 11:11 .bash_profile
-rw-r--r--  1 train  student  124 Dec 7 11:11 .bashrc
-rw-------  1 train  student  173 Dec 14 14:30 bessel.c
-drwxr-x---  2 train  student  4096 Dec 14 14:30 besselTest
-rw-------  1 train  student  374 Dec 14 14:30 birthtoken.txt
-rw-------  1 train  student  173 Dec 14 14:30 example.c
-rw-------  1 train  student  1391 Dec 14 14:30 flowers.txt
-rw-------  1 train  student  44 Dec 14 14:16 .lesshist
-rw-------  1 train  student  1492 Dec 14 14:30 lines
-drwxr-xr-x  4 train  student  4096 Dec 7 11:11 .mozilla
-rw-------  1 train  student  130 Dec 14 14:11 .Xauthority
[train@centos ~]$
```
Working with Files

```bash
[train@centos ~]$ touch mytest
[train@centos ~]$ ls
bessel.c  besselTest  birthtoken.txt  example.c  flowers.txt  lines  mytest
[train@centos ~]$ cp mytest mytest.new
[train@centos ~]$ ls
bessel.c  besselTest  birthtoken.txt  example.c  flowers.txt  lines  mytest
besselTest  example.c  lines  mytest.new
[train@centos ~]$ rm mytest
[train@centos ~]$ ls
bessel.c  besselTest  birthtoken.txt  example.c  flowers.txt  lines  mytest
besselTest  example.c  lines  mytest.new
[train@centos ~]$ mv mytest.new mytest.old
[train@centos ~]$ ls
bessel.c  besselTest  birthtoken.txt  example.c  flowers.txt  lines  mytest
besselTest  example.c  lines  mytest.old
[train@centos ~]$ rm -i mytest.old
rm: remove regular empty file `mytest.old'? yes
[train@centos ~]$ ls
bessel.c  besselTest  birthtoken.txt  example.c  flowers.txt  lines  mytest
[train@centos ~]$
```
Working with Directories

```
$ pwd
/home/train
$ ls
bessel.c  besselTest  birthtoken.txt  example.c  flowers.txt  lines  mytest
$ mkdir src
$ ls
bessel.c  birthtoken.txt  flowers.txt  mytest
besselTest  example.c  lines  src
$ cd src
$ mkdir C
c$ mkdir TXT
$ ls
C  TXT
$ ls -la
total 32
drwxr-xr-x 4 train student 4096 Dec 14 14:45 drwxr-xr-x 5 train student 4096 Dec 14 14:45 ..
drwxr-xr-x 2 train student 4096 Dec 14 14:45 drwxr-xr-x 2 train student 4096 Dec 14 14:45 C
drwxr-xr-x 2 train student 4096 Dec 14 14:45 TXT
c$ cd
/home/train
c$ pwd
/home/train
c$ 
```
Working with Directories:
Home (~) and Wildcards

```
[train@centos ~]$ cd src
[train@centos src]$ cd C
[train@centos C]$ ls
[train@centos C]$ ls ~
bessel.c birthtoken.txt flowers.txt mytest
besselTest example.c lines src
[train@centos C]$ pwd
/home/train/src/C
[train@centos C]$ cp ~/*.c .
[train@centos C]$ ls
bessel.c example.c
[train@centos C]$ ls ~
bessel.c birthtoken.txt flowers.txt mytest
besselTest example.c lines src
[train@centos C]$ pwd
/home/train/src/C
[train@centos C]$ cd ..
[train@centos src]$ ls
C TXT
[train@centos src]$ cd TXT
[train@centos TXT]$ cp ~/*.txt .
[train@centos TXT]$ ls
birthtoken.txt flowers.txt
[train@centos TXT]$ 
```
Working with Directories

```
$ cd
$ ls
bessel.c  besselTest  birthtoken.txt  example.c  flowers.txt  lines  src
$ rm src
rm: cannot remove `src': Is a directory
$ cd src
$ ls
 C  TXT
$ ls C
bessel.c  example.c
$ ls TXT
birthtoken.txt  flowers.txt
$ rmdir C
rmdir: C: Directory not empty
$ mv C CC
[ train@centos src ]$ ls
 CC  TXT
$ ls CC
bessel.c  example.c
$ rm CC/bessel.c
$ ls CC
example.c
$ [ train@centos src ]$
```
Working with Directories and Files:

Wildcards

```
[train@centos src]$ rm -ri CC
rm: descend into directory 'CC'? y
rm: remove regular file 'CC/example.c'? n
[train@centos src]$ rm -r CC
[train@centos src]$ ls

TXT
[train@centos src]$ rm -r TXT
rm: remove write-protected regular file 'TXT/birthtoken.txt'? y
rm: remove write-protected regular file 'TXT/flowers.txt'? y

[train@centos src]$ ls
[train@centos src]$ pwd
/home/train/src
[train@centos src]$ cd
[train@centos ~]$ ls -lad ??*
-rw------- 1 train student 3622 Jan 7 17:00 .bash_history
-rw-r--r-- 1 train student 33 Dec 7 11:11 .bash_logout
-rw-r--r-- 1 train student 176 Dec 7 11:11 .bash_profile
-rw-r--r-- 1 train student 124 Dec 7 11:11 .bashrc
drwx------ 2 train student 4096 Jan 6 15:03 .gconf
drwx------ 2 train student 4096 Jan 6 15:42 .gconfd
drwx------ 3 train student 4096 Jan 6 14:57 .gnome2
drwx------ 2 train student 4096 Jan 6 14:57 .gnome2_private
-rw------- 1 train student 83 Jan 6 01:15 .lessshst
drwxr-xr-x 5 train student 4096 Jan 2 23:15 .mozilla
-rw------- 1 train student 3838 Jan 7 15:57 .viminfo
-rw------- 1 train student 455 Jan 10 06:28 .Xauthority
[train@centos ~]$ ls *?.
bessel.c example.c
[train@centos ~]$
```
Creating Files: Redirection (STDOUT)

```
[train@centos ~]$ cat > animals-sm.txt
mouse
finch
hamster
[train@centos ~]$ cat > animals-lg.txt
elephant
buffalo
rhinoceros
[train@centos ~]$ more animals*
:::.................
animals-lg.txt
:::.................
elephant
buffalo
rhinoceros
:::.................
animals-sm.txt
:::.................
mouse
finch
hamster
[train@centos ~]$
```

^D (control d for EOF)
Creating Files: Redirection

```
[train@centos ~]$ cat animals-sm.txt animals-lg.txt > animals.txt
[train@centos ~]$ more animals.txt
mouse
finch
hamster
elephant
buffalo
rhinoceros
[train@centos ~]$ sort animals.txt
buffalo
elephant
finch
hamster
mouse
rhinoceros
[train@centos ~]$ sort animals.txt > animals-sorted.txt
[train@centos ~]$ more animals-sorted.txt
buffalo
elephant
finch
hamster
mouse
rhinoceros
[train@centos ~]$
```
Creating Files: Append and Pipe

```
[train@centos ~]$ cat >> animals-lg.txt
cow
horse
[train@centos ~]$ more animals-lg.txt
elephant
buffalo
rhinoceros
cow
horse
[train@centos ~]$ cat animals-lg.txt animals-sm.txt | sort > animals-sorted.txt
[train@centos ~]$ ls
animals-lg.txt  animals.txt  birthtoken.txt  lines
animals-sm.txt  bessel.c  example.c  mytest
animals-sorted.txt  besselTest  flowers.txt  src
[train@centos ~]$ more animals-sorted.txt
buffalo
cow
elephant
finch
hamster
horse
mouse
rhinoceros
```

^D (control d to stop)
Creating Files: Redirection (STDIN) and Pipe

```
[train@centos ~]$ more animals.txt
mouse
finch
hamster
elephant
buffalo
rhinoceros
[train@centos ~]$ sort < animals.txt > animals-orig-sort.txt
[train@centos ~]$ more animals-orig-sort.txt
buffalo
elephant
finch
hamster
mouse
rhinoceros
```

```
[train@centos ~]$ cat flowers.txt birthtoken.txt | grep -i violet | sort
African violet: Such worth is rare.
February: Amethyst: Violet
Violet, blue: Faithfulness.
Violet, white: Modesty.
```

```
[train@centos ~]$ cat flowers.txt birthtoken.txt | grep -i violet | sort | wc -l
4
```

```
[train@centos ~]$  
```
The VIM editor has two modes:

1. **Command**: interprets a letter or sequence of letters as a command.

2. **Insert**: puts anything typed into the file. The ESC key ends insert mode and returns you to command mode.

**Command line** entry at the bottom of the screen appears when the command “:” is typed. VIM starts up in command mode.
VIM Tutor: Getting Started

[train@centos ~]$ vimtutor
Welcome to the VIM Tutor - Version 1.7

Vim is a very powerful editor that has many commands, too many to explain in a tutor such as this. This tutor is designed to describe enough of the commands that you will be able to easily use Vim as an all-purpose editor.

The approximate time required to complete the tutor is 25-30 minutes, depending upon how much time is spent with experimentation.

ATTENTION:
The commands in the lessons will modify the text. Make a copy of this file to practise on (if you started "vimtutor" this is already a copy).

It is important to remember that this tutor is set up to teach by use. That means that you need to execute the commands to learn them properly. If you only read the text, you will forget the commands!

Now, make sure that your Shift-Lock key is NOT depressed and press the j key enough times to move the cursor so that Lesson 1.1 completely fills the screen.

"/tmp/tutorM14320" 970 lines, 33259 characters
Resources

- Unix Tutorial for Beginners
  http://info.ee.surrey.ac.uk/Teaching/Unix/

- VTC (Unix Shell Fundamentals) – need to request an account
  http://www.udel.edu/it/learnit/course/vtccom.html

- VIM Tutor (vimtutor)

- Linux vi and vim editor: Tutorial and advanced features
  http://www.yolinux.com/TUTORIALS/LinuxTutorialAdvanced_vi.html

- Graphical vi-vim Cheat Sheet and Tutorial
  http://www.viemu.com/a_vi_vim_graphical_cheat_sheet_tutorial.html