



# UNIX for Programmers and Users

**“UNIX for Programmers and Users”**

**Third Edition, Prentice-Hall, GRAHAM GLASS, KING ABLES**

# Starting with Unix

- WARNING:
  - There are differences between the textbook and what you will see experience using Ubuntu. The examples in lectures might also be different. Use the online help to get exact syntax and functionality for another Unix system that you might be using. The differences are mechanical, and not conceptual.

# Starting with Unix

- **Logging In**
  - In order to use a UNIX system, you must first **log in with a suitable username**.
- **A username is a unique name** that distinguishes you from the other users of the system.
- **Your username and initial password** are assigned to you by **the system administrator**.
- UNIX first asks you for your username by prompting you **with the line “login:”** and then asks for **your password**.

# Starting with Unix

- When you enter your password, the letters that you type are not displayed on your terminal for security reasons.
- UNIX is case sensitive, so make sure that the case of letters is matched exactly to those of your password.
- Depending on how your system is set up, you should then see either a \$, a % or another prompt. That is your default shell prompting you to provide some course of action.
- Here's an example login :

UNIX® System V Release 4.0

login : glass

Password : --> What is typed here is secret and doesn't show.

Last login: Sun Feb 15 18:33:26 from dialin

\$ \_

# Shells

- The \$ or % prompt that you see when you first log in is displayed by a special kind of program called a shell.
- A Shell is a program that acts as a middleman between you and the raw UNIX operating system.
- It lets you run programs, build pipelines of processes, save output to files, and run more than one program at the same time.
- A shell executes all of the commands that you enter.
- The four most popular shells are:
  - the Bourne shell (sh)
  - the Korn shell (ksh)
  - the C shell (csh)
  - the Bash Shell (bash)

# Shells

- All of these shells share a similar set of core functionality, together with some specialized properties.
- The Korn shell is a superset of the Bourne shell, and thus users typically choose either the C shell or the Korn shell to work with.
- The Bash shell, or Bourne Again Shell, is an attempt to combine the best features from Korn and C shells.
- Bash is becoming the most popular shell, because it is freely available and comes as a standard shell on Linux systems.

# Running a utility

- To run a utility, simply enter its name at the prompt and press the Enter key.
- One utility that every system has is called `date`, which displays the current date and time:

`$ date` --> run the date utility.

Thu, Aug 18, 2016 7:11:59 AM

`$ _`

# Running shell

- Running shell

ka 9: `echo $SHELL`

`/usr/bin/bash`



# clear

- This utility clears your screen.

# man: online help

- All UNIX systems have a utility called **man** ( short for **manual page** ) that **puts this information** at your fingertips.
- The manual pages are **on-line copies of the original UNIX documentation**. They contain information about **utilities, system calls, file formats, and shells**.

# Organization of the manual pages

- The typical division of topics in manual pages ([sections](#)) is as follows:
  1. Commands and Application Programs.
  2. System Calls
  3. Library Functions
  4. Special Files
  5. File Formats
  6. Games
  7. Miscellaneous
  8. System Administration Utilities

# Using man

Here's an example of man in action:

```
$ man chmod      ---> select the first manual entry.
CHMOD(1V)        USER  COMMANDS          CHMOD(1V)
NAME
    chmod - change the permissions mode of a file
SYNOPSIS
    chmod C -fR V mode filename ...
                                --> the description of chmod goes here.
SEE ALSO
    csh(1), ls(1V), sh(1), chmod(2V), chown(8)
```

# Terminating a Process: Control-c

- There are often times when you **run a program** and then wish **to stop it** before it's finished.
- The standard way to execute this action in UNIX is to press the keyboard sequence **Control-C**.
- Most processes **are immediately killed** and your shell prompt is returned.
- Here's an example of the use of Control-C:

```
$ man chmod
```

```
CHMOD(1V)  USER  COMMANDS  CHMOD(1V)  
NAME
```

```
chmod - change the permissions mode of a file
```

```
^C
```

```
$ _
```

# Pausing Output to Terminal: Control-s/Control-q

- If the output of a process starts to rapidly scroll up the screen, you may **pause** it by **pressing Control-S**.
- **To resume the output**, you may either **press Control-s again** or **press Control-q**.
- This sequence of control characters is sometimes called **XON/XOFF** protocol.
- Here's an example of its use:

```
$ man chmod
```

```
...
```

```
^s
```

---> suspend terminal output.

```
^q
```

---> resume terminal output.

```
...
```

```
$ _
```

# End of Input: Control-d

- You must tell the utility when **the input from the keyboard is finished**.
- To do so, **press Control-D** on a line of its own after the last line of input. **Control-D** signifies the end of input.
- For example, the **mail utility** allows you to send mail from the keyboard to a named user:

\$ **mail tim** --> send mail to my friend tim.

Hi Tim, --> input is entered from the keyboard.

I hope you get this piece of mail. How about building a country  
one of these days?

- with best wishes from Graham

**^D** --> tell the terminal that there's no more input.

\$ \_

# Setting/Changing password

- After you [first login to a UNIX system](#), it's a good idea [to change your initial password](#). The password protects all your private information.
- You might be forced to change your password by the administrator.
- Remember however, that [superuser can access everything](#).
- Passwords should generally [be at least six letters long](#) and should [not be words from a dictionary or proper nouns](#). Some system administrators will put restrictions on the life span of passwords, so you have to exchange them periodically.
- The best is to use a mixed expression of letters, numbers and other characters, like [“GWK#145W%”](#).
- If you [forget your password](#), the only thing to do is to [contact your system administrator](#) and [ask for a new password](#).



# Setting/Changing Password: passwd

- `passwd` allows you to change your password.
- You are `prompted for your old password` and then twice for the new one.
- The new password may be stored in an encrypted form in the password file `“/etc/passwd”` or in a `“shadow”` file ( for more security), depending on your version of UNIX.

- An example of changing a password:

\$ `passwd`

Current password : `penguin`

New password( ? For help ) : `GWK145W` --> invisible

New password( again ) : `GWK145W` --> invisible

Password changed for glass

\$ \_

- Note that you `wouldn't normally be able to see the passwords`, as UNIX turns off the keyboard `echo` when you enter them.

# Logging out

- To leave the UNIX system, press the keyboard sequence **Control-D** at your shell prompt.
- This command tells your login shell that there is no more input for it to process, causing it to disconnect you from the UNIX system.
- Most systems then display a “login:” prompt and wait for another user to log in.
- \$ ^D
- UNIX® System V Release 4.0
- login : --> wait for another user to log in.
- If you connect to a remote server through ssh connection, you will not see the new login prompt; instead, you will be disconnected.

# User Home Directory

- Every UNIX process has a location in the directory hierarchy, termed its current working directory.
- When you log into a UNIX system, your shell starts off in a particular directory called your home directory.
- In general, every user has a different home directory, which often begins with the prefix “/home”.
- For example, my author’s home directory is called “/home/glass”.
- The system administrator assigns these home-directory values.

# Printing Working Directory: pwd

- To display **your shell's current working directory**, use the **pwd** utility, which works like this:

```
$pwd
```

```
/home/glass
```

```
$ _
```

# Directory Hierarchy, Home Directory and Login Shell

- Here's a diagram that indicates the location of user's login Korn shell in the directory hierarchy:

