# CE350 Lecture4

## The Bourne Again Shell (bash)

by İlker Korkmaz and Kaya Oğuz

## Contents

- This lecture mainly gives the basis on configuring, running, and using the bash.
- The contents of this lecture are based on
    - Chapter 13 of the textbook (UNIX Shells by Example)
    - and also some parts of **http://tldp.org/LDP/abs/html/** document

## What bash provides:

- interactivity as a shell
- features of built-in commands and command-line shortcuts within a shell
    - such as history, aliases, file and command completion, command-line editing, ...

## bash

- born on January 10, 1988
- fathered by Brian Fox
- adopted and enhanced by Chet Ramey
- freely available under GNU public license
- first version was 0.99
    - your version ?
        - $ bash –version
        - $ echo $BASH_VERSION

## startup for bash

- System boots -> then **init** process runs
- The init spawns a **getty** process -> then **/bin/login** is executed
- The login program starts up the shell, **/bin/bash**, with the last entry in **passwd** file.
- The bash looks for the system file, **/etc/profile**, and executes its commands -> then bash executes **.bash_profile**, initialization file for the user's directory, -> then bash usually runs **.bashrc**, and finally **$ prompt** appears on the screen
- Then the bash waits for commands...

## The initialization files

- **.bash_profile** sets the user's **aliases** and **functions** and then sets user-specific **environment variables** and startup **scripts**.
- if no .bash_profile -> then try **.bash_login**
- if no .bash_login -> then try **.profile**
- **/etc/profile** file is a systemwide initialization file set up by admin to perform tasks when the user logs on.
- **/etc/bashrc** includes systemwide aliases and functions (The primary prompt PS1 is often set here).

# Some extra

- **~/.bash-logout** to record the logout and to perform housekeeping tasks
- **.inputrc** is also read when bash starts up. It contains variables to customize keystroke behavior and settings to be used on performing command-line editing by vi, emacs, ...
- To prevent startup files being executed:
    - $ bash –noprofile
    - $ bash -p

## Setting bash options with the built-in commands

- $ set -o
    - allows to customize the shell environment
    - either “on”/“off”, and set in BASH_ENV file
- $ shopt -*argument*

## The prompts

- Normally, the prompts are defined in /etc/bashrc or .bash_profile.
- Primary prompt
  - dollar sign ( $ )
  - $ PS1= ...
    - (*dissect the table on page 769*)
- Secondary prompt (the PS2 variable)
  - sign: >
  - appears when a command is not completed or more input is expected.

## The search path

- to print the PATH
  - $ echo $PATH
- usually the dot ( . ) is not in the search path,
  - $ runScript *does not work*
  - $ ./runScript *works*
- to set the PATH
  - $ PATH=$HOME:/usr/bin:/usr/local/bin:
  - $ export PATH
  - OR -> $ export PATH=$HOME:/usr/bin:

## The order of processing commands

- The commands in the shell are executed according to their types in the following order:
    - aliases
    - keywords (*if, function, while*, ...)
    - functions
    - built-in commands
    - executables and scripts
- built-in commands and functions are defined within the shell, whereas scripts and executable programs are stored on disk.
- To understand the command type:   $ type *command*

## Multiple commands at a line

- at the same line: use ;
  - $ ls; pwd; date; whoami
- command grouping:
  - $ ( ls; pwd; date; whoami ) > outFile
- conditional execution of commands:
  - use && (AND) , || (OR) ; according to <u>**exit**</u> status
  - if a command is successful it has 0 exit staus.
  - $ cc prg1.c -o prg && ./prg (*if compiles, run*)
  - $ cc prg1.c -o prg || less prg1.c (*remember short circuit evaluation*)
- To execute a command in the backgorund: use &
  - $ ./myServerProgram &
  - $ kill -9 $! (variable ! evaluates to the PID of the job most recently put in the background)

## Job control

- Control the jobs (running processes) via signals:
  - to terminate: Ctrl-C OR Ctrl-\
  - to suspend: Ctrl-Z
  - some commands to control the jobs:
    - bg (put background), fg (foreground), jobs, kill, stop,...(dissect the table on page 782)

## Command-line shortcuts

- command and filename completion: use [***tab***] key
  - $ da[tab]   #expands to date command
  - $ ca[tab] [tab]  #lists all commands starting with ca
- history file: use the arrow keys on the keyboard to move within the history list
- use ! (bang) to re-execute a command
  - !! (re-executes the previous command)
  - !-N (re-executes the Nth command back from present)

## Aliases

- An alias is a bash user-defined abbreviation for a command.
- Aliases are normally set in .bashrc file.
- to manage an alias (*nickname*)

| | |
|---|---|
| • to list: | $ alias |
| • to create: | $ alias les=less |
| • to use/call: | $ les |
| • to turn off temporarily: | $ \les |
| • to delete: | $ unalias les |

## Metacharacters

- Metacharacters (wildcards) are special characters used to represent something other than themselves.
- \ & ; $ ? * [*abc*] [!*abc*] (*commands*) {*commands*} ...
- to be dissected: Table 13.10 on page 801.

## Filename substitution

- The process of expanding the metacharacter into filenames is called ***filename substitution***, or ***globbing***.
- *  ?  [*abc*]  [*a-z*]  [!*a-z*]  \  ...
- to be dissected: Table 13.11 on page 802.
- Furthermore, to illustrate the samples on some metacharacters, asterisk ( * ), question mark ( ? ), square brackets ( [ ] ), braces ( { } ), escape ( \ ), tilde ( ~ ), hyphen ( - ), read Sections 13.9.1-13.9.6 on pages 802-807.