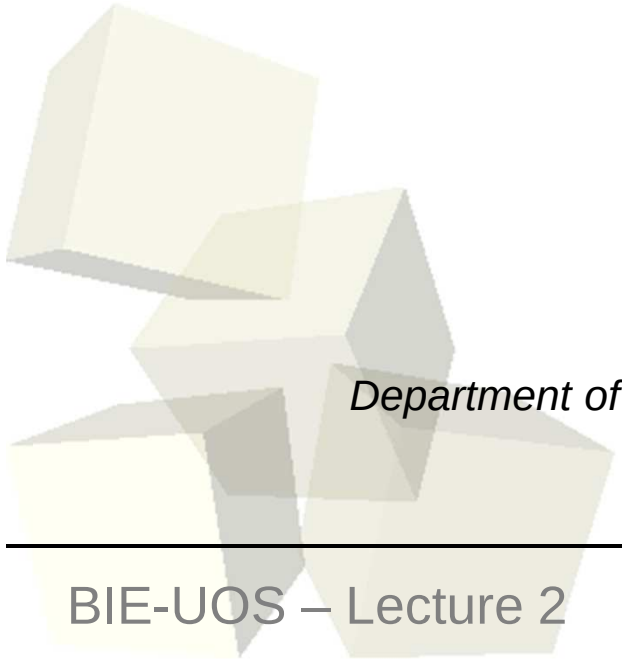
Lecture 2

Unix: CLI – CLI Parse Order, Special Characters, Command Execution, Shell Variables.



Department of Computer Systems FIT, Czech Technical University in Prague

Motivation

Variables

A=Hello

A="Number of logged-in users: `finger | tail +2 | wc –l` "

Commands

ls –la /

echo $A

finger | tail +2 | wc –l

:(){ [$1 -gt $2] && echo $1 || echo $2; }; : 1 3; : 2 1

Metacharacters

- Characters that represent something other than its literal self (depending on the context).

  #, " ", ´ ´, ` `, \, $, |, {}, (), [], …

Command Line Parse Order

1. Turning off the special meaning of characters

\	(backslash/escape) turns off the special meaning of the following character.
´ … ´	(single quotes) turn off the special meaning of all characters
" … "	(double quotes) turn off the special meaning of characters except: \ turn off the spec. meaning ` cmd` command substitution $ $NAME - variable substitution $((expr)) - arithmetic expansion (except sh) $(cmd) - command substitution (except sh)

– Shell remove these characters during command line parsing.

Command Line Parse Order

2. **Comment removing** # (hash)
3. **Command line splitting into simple commands and definition of I/Os**

- **Simple command**
  - Sequence of optional variable assignment followed by blank-separated words
  - Command name followed by options and arguments

- **Pipeline**
  - sequence of one or more commands separated by the character |

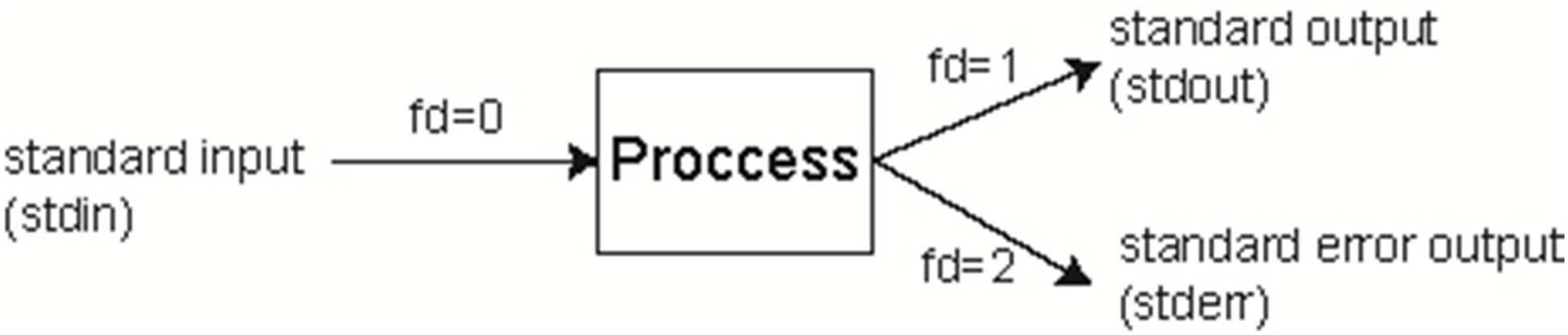
- **List**
  - sequence of one or more pipelines separated by one of the operators ; & && or ||, and optionally terminated by one of ; & <newline>

- **Compound command**

  (list) { list ; } ((expression)) [[expression]] for while until if case

Command Line Parse Order

Input/output Redirection



- **File descriptor** is an abstract indicator for accessing a file (0,1,2,…).
- Every process has 3 standard POSIX file descriptors by default:
  0 – standard command input (stdin)
  1 – standard command output (stdout)
  2 – standard command error output (stderr)
- New process inherits file descriptors from his parent process by default.
- User can redefine/redirect every file descriptor by special characters.

Command Line Parse Order

cmd < file	(less than) executes cmd, using file as the source of the standard input
cmd > file	(greater then) executes cmd, placing the standard output in file
cmd >> file	executes cmd, appends the standard output to the end of the file

Command Line Parse Order

cmd 2> file	executes cmd, placing the standard error output in file
cmd >&*n*	executes cmd, placing the standard output in file defined by file descriptor *n*
cmd *m*>&*n*	executes cmd, placing the output defined by file descriptor *m* in file defined by file descriptor *n*
cmd > file 2>&1	executes cmd, placing the standard output and standard error output in

- IO redirection is evaluated from left to right..

Command Line Parse Order

- **Exit status/return code** is a small number passed from a child process to a parent process when it has finished executing.

  (0 = succes, 1,2,3,…,255 = error)

cmd &	(ampersand) The shell executes the command in the background in a subshell. The shell does not wait for the command to finish.
cmd1 ; cmd2	(semicolon) Commands are executed sequentially; the shell waits for each command to terminate in turn.
(cmd1 ; cmd2)	(brackets) Commands are executed sequentially by the subshell.

Command Line Parse Order

cmd1 | cmd2	(pipe/pipeline) The standard output of cmd1 is connected via a pipe to the standard input of cmd2. Commands are executed in parallel. The return status of a pipeline is the exit status of the last command.
cmd1 && cmd2	cmd2 is executed if, and only if, cmd1 returns an exit status of zero.
cmd1 || cmd2	cmd2 is executed if and only if cmdd1 returns a non-zero exit status.

Command Line Parse Order

4. Replacement of

- Aliases
- File name Abbreviation ~ (tilde)

~	Home directory of the current user (except of sh)
~username	Home directory of the given user (except of sh)

- Command substitutions

` cmd `	Shell performs the expansion by executing cmd and replacing the command substitution with the standard output of the cmd.
$(cmd)	Different syntax. (except of sh)

- Arithmetic expressions $((expr))
- Variables ($1, $HOME,…)

Command Line Parse Order

5. Word splitting

newline space TAB	Default word separators (it can be change by shell variables IFS)

Command Line Parse Order

6. File Name Substitution

*	(asterisk) matches zero or more characters, except the leading . (period) of a hidden file.
?	(question mark) matches exactly one character except the leading . (period) of a hidden file.
[abc] [a-z]	(square brackets) matches one character in the set or in the range
[!abc] [!a-z] [^ijk] [^m-z]	matches one character not in the set or in the range ([^…] except of sh)

Command Line Parse Order

7. **Options and arguments setting**
8. **Variable setting or command execution**

- **Command** eval string
  - The command line is scanned twice by the shell.
  - First, the shell interprets the command line when it passes to the eval command.
  - Then shell interprets it a second time as a result of executing the eval command

- **Example:**

```
unset B  ;  A='$B'  ;  B=xzy  ;  echo $A

unset B  ;  A='$B'  ;  B=xzy  ;  eval echo $A

echo “Enter command: "  ;   read A  ;  eval $A
```

Command Execution

- **Which program will be executed?**
  - **Absolute/relative path to command**
  - **Shell function**
  - **Shell built-in command**
  - **Only program name**
    - Shell variable PATH contains list of directories.
    - The shell searches through each of these directories, one by one, until it finds a directory where the executable program exists.

Shell Variables

variable=value	Creation of local variable
export variable[=value]	Creation of environment/global variable
$variable	Display the content of variable
set [variable]	List all variables
env	List only the environment/global variables

- **Some built-in shell variables**

$PS1	defines prompt
$PAGER	defines program for displaying of man. pages
$PATH	list of directories for command searching
$MANPATH	list of directories for manual pages searching

Shell Variables

- **Some built-in shell variables**

$HOME	home directory
$PWD	current working directory
$0	program name
$1,…,$9	command line arguments
$#	number of command line arguments
$$	process ID of current shell
$?	exit status of the last executed command