Job Control
Stanley Yao
Computer Science Department
University of Arizona

Process .vs. Job
- Multiple processes in the same system
- Multiple processes in the same shell
- Job: commands on a single command line which may include several processes

- PID: unique to the system
- Job number: unique to the shell

Process .vs. Job
- Multiple processes in the same system
- Multiple processes in the same shell
- Job: commands on a single command line which may include several processes

- PID: unique to the system
- Job number: unique to the shell

Forefront .vs. Background
- Foreground job
  - Has control over the terminal (input & output)
  - Shell waits for it to finish
- Background job:
  - Has no control over the terminal (input & output)
  - Blocks if reading from the terminal
  - By default, can print to the terminal (intermingle with the output from other jobs)
  - "stty tostop" to change the above default behavior
  - Shell doesn’t wait for it to finish (run in parallel)

Foreground .vs. Background
- Foreground job
  - Has control over the terminal (input & output)
  - Shell waits for it to finish
- Background job:
  - Has no control over the terminal (input & output)
  - Blocks if reading from the terminal
  - By default, can print to the terminal (intermingle with the output from other jobs)
  - "stty tostop" to change the above default behavior
  - Shell doesn’t wait for it to finish (run in parallel)

Operations
- Run a job in background
  - command &
- Suspend a running job
  - ^Z
- Put a job to the background
  - bg [%n]
- Bring a job to the foreground
  - fg [%n]
- List jobs
  - jobs
- List processes
  - ps

Operations
- Run a job in background
  - command &
- Suspend a running job
  - ^Z
- Put a job to the background
  - bg [%n]
- Bring a job to the foreground
  - fg [%n]
- List jobs
  - jobs
- List processes
  - ps

Operations (cont.)
- Kill a process
  - kill [-9] PID
- Kill a job
  - kill [-9] %n
- Kill by pattern match
  - kill %?sleep

Operations (cont.)
- Kill a process
  - kill [-9] PID
- Kill a job
  - kill [-9] %n
- Kill by pattern match
  - kill %?sleep

Status Notification
- A background job J1 exits, while another foreground job J2 is running. C shell will print the message about J1’s exit when J2 exits.
- By setting the shell variable notify, we can force the C shell to print the message about J1’s exit right away.

Status Notification
- A background job J1 exits, while another foreground job J2 is running. C shell will print the message about J1’s exit when J2 exits.
- By setting the shell variable notify, we can force the C shell to print the message about J1’s exit right away.
Directory Change

- You create jobs while you are in different directories, so that different jobs have different “current working directory”.
- When you bring a job, whose CWD is different from your current CWD, to the foreground, the shell’s CWD changes to the job’s CWD during the job is running. After the jobs finishes, the shell CWD return to the original one.

Application

- Edit your program in emacs. ^Z to suspend emacs. Compile and run your program in the shell. “fg” back into emacs and modify your program again.
- When running an X application, usually run it in background so that we can do other stuff in the terminal.

Acknowledgement