CS0330: SHELL 2 GEAR UP

Project Overview

C Shell getting 2 major features

- Multiple Jobs
  - Run either in background or foreground
- Signals
  - Mask signals for different processes
  - Manage processes, update status

Topics to Review

- Signals
    - Masking signals
- Reaping
- Jobs
- Processes
- Jobs vs. Processes
- Job Management

Roadmap: Set up

- Run ‘cs0330_install shell_2’
  - Copy your Makefile and sh.c from your *shell_1* directory into your *shell_2* directory.
- Stencil/support code: jobs.c and jobs.h
  - Handles the jobs list

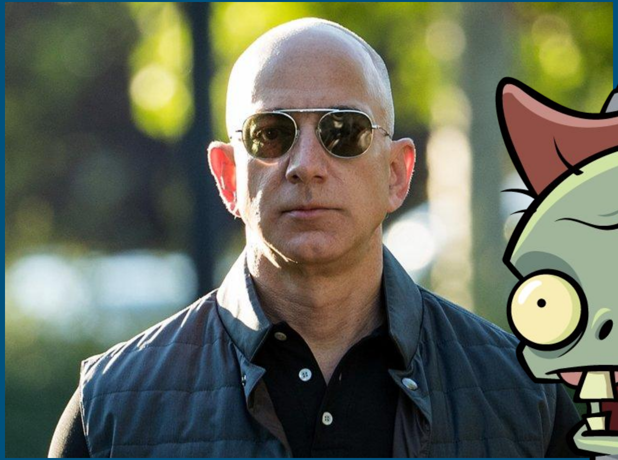
Roadmap: Parsing

- Parsing support: new commands, job management.
  - New commands: bg, fg, jobs
  - Does your parsing strategy need to change?
- We suggest that you start by parsing the command line for new commands and put off implementing them until later

Roadmap: Signals

- Understand how you are going to use signals in this project.
  - How should your shell respond to signals?
  - How should a foreground job respond to those signals?
  - How should a background job respond?
- Tcsetpgrp: used to ensure that the correct process group receives signals
- Once you understand those questions, write your signal handling code!

Roadmap: Reaping



- Understand reaping.
  - What is reaping?
    - Cleaning up zombie processes
  - Why do processes need to be reaped?
  - How *do* you reap, exactly?
    - waitpid()
    - Job status: WIFEXITED, WIFSIGNALED , WIFSTOPPED, WIFCONTINUED

Roadmap: fg and bg

- What’s the difference between running a **background** job and running a **foreground** job?
- Managing jobs will involve maintaining a list of active jobs, reaping processes that have terminated, and **moving jobs between foreground and background**

Demo/Testers

Demo:
cs0330_shell_2_demo OR cs0330_shell_2_noprompt_demo

Run all traces until the first failure:

```
cs0330_shell_2_test -s <33noprompt>
```

Run a single trace:

```
cs0330_shell_2_test -s <33noprompt> -t trace<#>.txt
```

For a list of other useful flags: cs0330_shell_2_test -h

**Make sure to run cs0330_cleanup_shell after every time you work on Shell 2 on department machines!

Demo

Tips

- Give yourself enough time to debug and **start early**!
- Finishing the signals lab as early as possible will give you a good start to understanding signals.
- man pages are you friends. Use them. Love them. Read them!
  - waitpid
  - tcsetpgrp
  - setpgid
  - etc.
- Skimming over the functions in jobs.c will be helpful to ensure you have an idea of support functions available for job management.

More Tips

- As always, factor your code well
  - It makes it easier to debug and easier to read!
- Do the project linearly according to the handout sections
  - Signal management
  - Adding multiple jobs
  - Reaping
  - fg and bg

Additional Resources

- Don't be afraid to try things out!
    - In our demo
    - In your normal shell (e.g., bash)
- Look at the traces and their outputs in */course/cs0330/pub/shell_2/*.
    - There is README in */course/cs0330/pub/shell_2/traces* giving a summary of what each trace does.

What You’ll Learn

- How a shell works, including:
    - Managing multiple processes and jobs
    - Managing foreground and background jobs
    - Responding to signals
- You will learn to use your normal shell more effectively
- You will gain a deeper understanding of how your operating system and processes communicate
- You will gain a deeper understand of how processes work

Questions?