Assignment 1: Stallion Shell



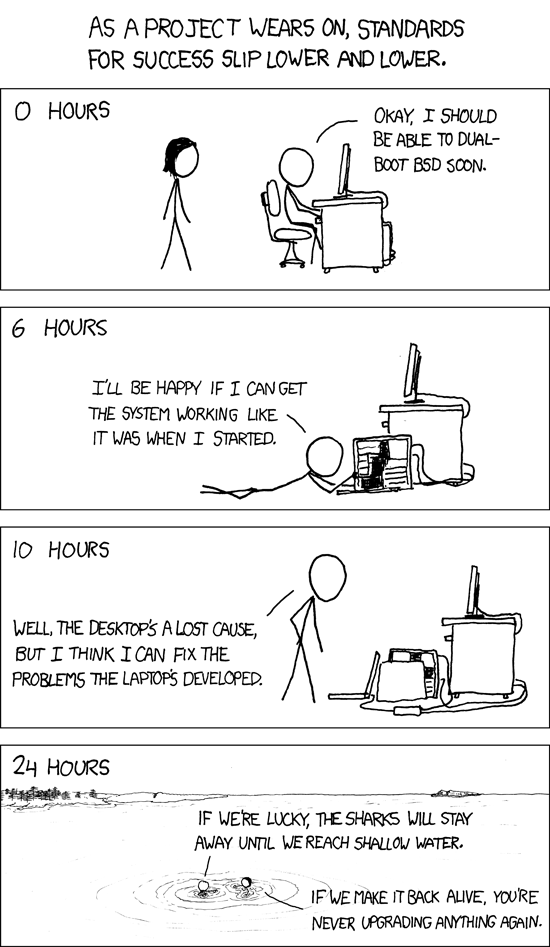
COP4610

Taylor Brockhoeft

David Perez

Design

Development Journal



9/2: Taylor created GIT repository with README for the project. Initial meeting to brainstorm ideas and delegate responsibilities.

9/3-9/8: Project began very well. Had two meetings or so in this time period and Taylor began most of the initial work on the prompt and setting up the backbone for the program, completing the prompt and the I/O on 9/7.

9/9: Part of the shell built-in utilities (mostly Exit and an early partial version of cd) were completed by Taylor.

9/10: Parser begun by Taylor, and David began looking into assembling the path strings in order to execute program binaries. Progress then stalled for a few days, as we both focused on our other classes and assignments for the rest of that week.

9/12-9/14: Mostly worked apart on our separate responsibilities. Taylor rewrote and completed the parser, and David got a decent working attempt at building the strings with the paths and executing simple binaries.

9/15: Taylor rewrote parsing function and restructured part of the codebase, as there were existing issues due to commands not null terminating.

9/16: David began looking into background execution and thinking about how to restructure processing the commands to allow easier attempts at implementing piping. Taylor began I/O redirection, but still had issues toward the end of the day.

9/17: Attempt at restructuring processing had to be shelved, glaring issues with getting even the most basic functionality to work after attempting to change things.

Division of Labor

Taylor: Prompt, Tokenizing input, Cleanup, Built-in utilities, IOACCT, IO Redirect, Pipelining, Readme.

David: Report, Background Processes, Processing Binary Commands, Readme.

Missing Functionality

Pipelining: Currently not implemented. Issues with processCommands() function and combining pipes with with the current implementation proved rather disastrous.

Io redirect: Currently not implemented in final product, a semi working variant is written in the code, however it's not fully working. Had issues creating the file on the system. Tried manually to create file on system to see if we could get it to write to that file, and each time the file would be empty. Not quite sure why not, however. If we could touch on this during recitation that would be really nice.

Ioacct: Currently not implemented in final product, a semi working variant is written in the code, however it's not fully working. Had issues creating the file on the system. Tried manually to create file on system to see if we could get it to write to that file, and each time the file would be empty. Not quite sure why not, however. Ended up removing the code to activate io redirection in the tokenizer() function.

Bugs/Issues (fixed ones specified)

* Issue: calling functions after bad or unknown commands(i.e. la or valgrind if it's not installed)
  + - User has to exit shell twice. I'm pretty sure this is because it is attempting to create a process and forking it. It wont find the command, so it wont execute it. The first time you enter exit, you leave the forked instance of the shell, since noting was able to execute and exit. Then, when you type exit again, then it will exit the program.
* Issue(fixed): the shell does not work any more. Creates multiple child processes of the shell.
  + - Resolved on 9/12. Was a simple while loop issue.
* Issue: When trying to run a program that doesn't exist, the shell doesn't take in input properly. 9-16-14
* Issue(fixed): with tokenize() and parsing
  + - Made us to decide to make certain variables global
* Issue(fixed): with null termination args
  + - Issue resolved by appending 'NULL' to end of imput
* Issue(think it's fixed now): Must exit multiple times from shell.
  + - Occasionally, for what we thing are a majority of issues with forked process, the user may need to exit the shell multiple times. We believe it's been fixed today. (9-17-14)
* Issue: Occasionally commands that require a param get hung up, such as wc without a specified file
  + - type “wc” and you'll see what I'm talking about.
    - Type “wc shell.c” and it works fine
    - not sure why this is an issue
* Issue(fixed): Had an issue today(9-17-14) where the prompt wouldn't print out after executing a command. This was caused by the process being forked and then waiting for new input before going back to the main while loop and printing the prompt.
* Issue(fixed): In fixing the issue above, we created a new issue. When we call clear or ls or any other unix command after calling cd, which is built in. We have an issue where it wont take in any more input. However, cd will work indefinitely. We think that there are issues with CD closing. Our assumptions we're correct and we were able to quickly squish that bug.

Suggestions for Assignment Write-up

David: Overall, the assignment is very well written and structured for students to learn some of the initial concepts of the class. The project is a decent introduction/refresher on C, so there really isn’t much room for improvement. Perhaps there should be a mini project or homework assignment to get a bit more practice with C before throwing us into the assignment. The features and requirements of the project are enough to keep us busy throughout the two week period in which we have to complete the assignment.

Taylor: I agree with the mini project concept. For myself, this is my first time working in C. Most people coming into this class are not as familiar with C as a language, rather they're more familiar with C++. I'm not trying to put down the C language, as they're are many things this project has made me aware of within the language. It would be nice to have a simple assignment that displayed all the little niche things that C is constrained by, such as initializing counters before for loops, Accessing and structuring 2D arrays in C, and even a quick refresher on dynamic memory allocation in C.