CS 551Project #1 Design Document.

MinixCustomShell

Group 8

Date: 27th Feb 2017

Primary Author(s): Chethan Bhaskar,

Sudhindra Manohar Bellur Vasudeva,

Shashank Ravindranath

CONTENTS

ABSTRACT

This document describes the design details of MinixCustom Shell.

INTRODUCTION

MinixCustomShell is a Shell invoked by "ash shell" provided by Minix. This shell supports all commands supported by ash shell (when their paths are appropriately indicated in PROFILE file) with support to execute a list of commands in a particular format. It fires an alarm after 5 second of execution of the command. It also supports parallel and sequence execution of commands. It also has a PROFILE file where you can customize the home folder, command prompt and PATH to export, alarm.

FEATURES SUPPORTED

MinixCustomShell supports the following features:

1. Read the prompt sign, HOME directory and PATH from PROFILE file.
2. Read the commands from Standard input (Keyboard) and output the result on Standard Output (Monitor).
3. Executes all commands present in /bin and /usr/bin [tested but can execute any commands when appropriate PATH is specified in PROFILE file].
4. Handles interrupts signals CTRL + C. MinixCustomShell asks the user if he/she wants to exit before exiting the process and going back to Minix shell.
5. MinixCustomShell also exits when user types command such as "exit" or "quit"

6. MinixCustomShell fires an alarm after 5 seconds of execution of command and asks for the user that if he/she wants to exit from this shell to Minix shell. This alarm firing can be configured via PROFILE file.
7. MinixCustomShell supports the execution of a list of commands in the following format:
   supports a command line with parentheses and the sequence and parallel execution operators (";" and "&"). A pair of parentheses encloses a command line which can be treated as a command. cmd&(cmd1;cmd2) is a valid command line specifying that cmd will be executed in parallel with (cmd1;cmd2). cmd;(cmd1&cmd2) is also a valid command, this will execute as cmd first, then (cmd1&cm2). Further cmd1 and cmd2 will execute in parallel.
8. MinixCustomShell doesn't Supports pipe "|", "<" and ">" operation.

FEATURE DESIGN DETAILS

Each of the features of MinixCustomShell is explained in detail here.

PROFILE FILE

PROFILE file is used to customize the prompt symbol, change the home directory of MinixCustomShell and export existing utilities from the specified path. When the MinixCustomShell is invoked by the default ash shell of Minix, the MinixCustomShell will see if PROFILE file is present or not. If it is present then it will read the prompt symbol, home directory and replaces it with the default values. PROFILE can also be used to customize alarm. It also, reads the PATH specified in the PROFILE file and exports that path. The new prompt will be preceded by the current directory path. Ex: if home=/root/CustomShell/ and prompt=$ then prompt string on invoking shell will be /root/CustomShell$.

SIGNAL AND ACTION

MinixCustomShell handles SIGINT signal. The SIGINT is for catching the CTRL-C interrupt. Initially a setjump method is called to save the state in a buffer. Once the interrupt occurred the confirmation is asked from the user and a longjump is performed.

ALARM AND ACTION

MinixCustomShell handles alarm, which fires after 5sec of command execution. For this event, will handle by using our own custom function. Initially a setjump method is called to save the state in a buffer. Once the alarm fired the confirmation is asked from the user and a longjump is performed.

COMMANDS AND IMPLEMENTATION DETAILS

MinixCustomShell can take a list of commands as input. The list of command scan begins with or without opening the braces "(". Commands are de-limited by delaminating ";".
Eg. Command1&(command2;command3)
Here command1 and (command2;command3) will execute in parallel. Further command2 and command3 will execute in sequence.

IMPLEMENTATION DETAILS:

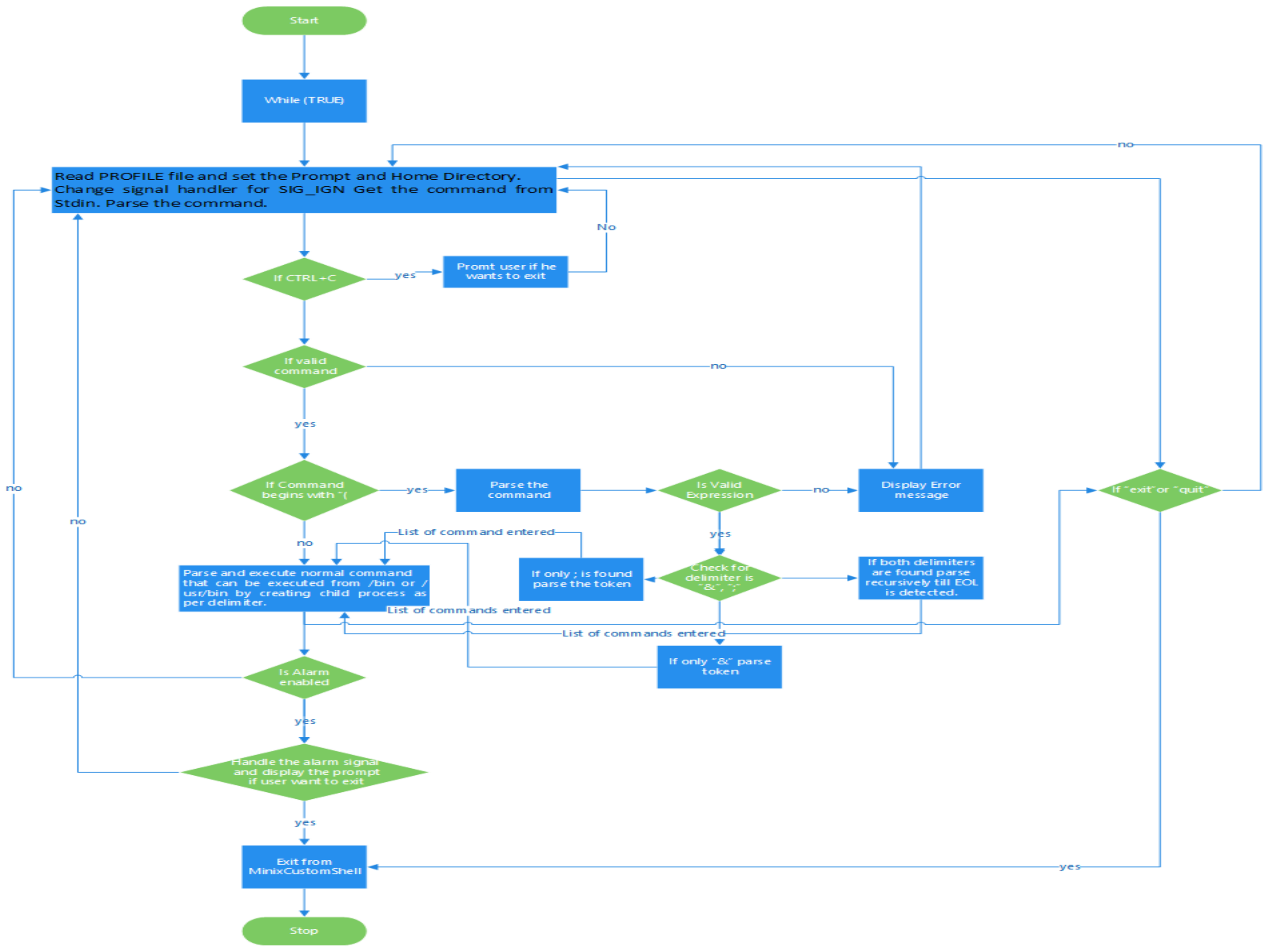
- Read PROFILE file and get the $ symbol.
- Check the command entered is valid command by comparing the number opening and closing braces if braces are given in command. If not valid show message as "Invalid Input - Paranthesis is not proper\n". Else continue to execute command. If braces are not given in command it will continue to execute.
- If it has braces then check for "("then, handle the braces and parse the entered command.
- After parsing Process the command to check if it has any delimiter and assign the delimiter.
- Execute the command is called.
  - If the entered command has "<",">" and "|" these. If yes, the show "Command not supported".
  - If the command entered is "exit","quit" or "cd .." then take the required action for the commands.
  - If command doesn't have any delimiter execute normally by creating a Child process.

- If command has delimiter, then parse for the command and find which is first delimiter. If it is "&" then tokenize the string and take first command before & and again tokenize the command after the delimiter. Execute both the commands by creating two separate Child process. When they are finished, it will display the result on MinixCustomShell.
    - Example: (command1&command2) – Execute both in parallel by forking two different child process and display result as and when it is finished.
- If command has ";" at first. Execute command one by one by creating once Child process to execute the first one and continue with same process to execute other commands. After executing all the commands exit from process created. Display the result on MinixCustomShell.
    - Example: (command1; command2)

- If it has the combination of commands, then Parse the command recursively until the \n or \r is found. In commands, whichever is occurs first in the command, based on that decide whether to execute in parallel or sequential.
    - Example: (command1;(command2&command3)) – command1 will be executed first. (Command2&command) executes next. In this Command2 and command3 will be executed in parallel by creating two different child process. Once both comman2 and command3 is executed both child process is exited and it will display command.

MinixCustomShell also supports all normal commands when they are given in the format that it supports by ASH shell.

FLOW CHART


Start
While (TRUE)
no
Read PROFILE file and set the Prompt and Home Directory. Change signal handler for SIG_IGN Get the command from Stdin. Parse the command.
No
If CTRL+C
yes
Promt user if he wants to exit
If valid command
no
yes
no
no
If Command begins with "(
yes
Parse the command
Is Valid Expression
no
Display Error message
If "exit" or "quit"
yes
no
List of command entered
Parse and execute normal command that can be executed from /bin or /usr/bin by creating child process as per delimiter.
If only ; is found parse the token
Check for delimiter is "&", ";"
If both delimiters are found parse recursively till EOL is detected.
List of commands entered
List of commands entered
If only "&" parse token
Is Alarm enabled
yes
Handle the alarm signal and display the prompt if user want to exit
yes
Exit from MinixCustomShell
yes
Stop

EXCEPTION HANDLING

MinixCustomShell handles error cases gracefully. To list a few it handles below error cases.

1. When PROFILE file is not present then MinixCustomShell throws an error and assumes current directory as the home directory, $ as the prompt,uses existing PATH variable value and proceeds.
2. For non-existing commands like abcd, treyttru, MinixCustomShell handles them gracefully by showing a Custom message in the shell and displays the prompt again.
3. When opening and closing braces are not proper then our MinixCustomShell handles that also by showing custom message for user.
4. If one command in the list of commands is not defined then, it gracefully handles that by displaying custom for that particular command and executes for rest of the commands.
5. If the user gives any command without any options, then that will display the display how it should be used.
6. If user has given double space between the commands, then it shows Custom message to handle it.
7. If the command entered is in upper case it shows custom message.
8. If user enters “exit” or “quit” command it shows the PID of the shell.
9. If forking of process is failed, then it displays the custom message by showing the PID.

EXAMPLE OF EXCEPTION HANDLING

```
# ./customshell
Shell start PID 253
/usr/src/Minix-Custom-Shell-Prompt$ (ls -l;pwd
        Invalid Input - Paranthesis is not proper
/usr/src/Minix-Custom-Shell-Prompt$ (ls -l;(pwd&ls)
        Invalid Input - Paranthesis is not proper
/usr/src/Minix-Custom-Shell-Prompt$ dsafsdg
        Command not found in Custom shell.
/usr/src/Minix-Custom-Shell-Prompt$ abc&pwd
        Command not found in Custom shell.
        /usr/src/Minix-Custom-Shell-Prompt
```

```
/usr/src/Minix-Custom-Shell-Prompt$ grep
        --this display how the command needs to be used.

/usr/src/Minix-Custom-Shell-Prompt$ ^C – CTRL+C
        Are you sure?[Y/N]
/usr/src/Minix-Custom-Shell-Prompt$ LS
        Command not found in Custom shell.
/usr/src/Minix-Custom-Shell-Prompt$
        --Alarm fired
        Are you sure?[Y/N]
/usr/src/Minix-Custom-Shell-Prompt$ exit
        Shell Exit: PID 253
/usr/src/Minix-Custom-Shell-Prompt$ quit
        Shell Exit: PID 253
```