

C-Shell

Shreeya Singh

Description

This project implements a basic shell in C, which supports built-in commands like `cd`, `echo`, `pwd`, `history` and `exit`.

- The shell displays a prompt with the format `<username@system_name:curr_dir>`, and handles commands like a regular shell.
- The directory from which the shell is invoked will be the home directory of the shell and will be indicated by `~`.

How to Compile and Run

To compile the project, use the provided `makefile` by running the command `make` in the terminal. Then, you can run the shell using `./cshell`.

Run:

```
make && ./cshell
```

Files and Directories

- `main.c` and `main.h`: Contains the main logic of the shell, including the command execution loop.
- `display.c` and `display.h`: Implement the display functions for the shell prompt.
- `execute.c` and `execute.h`: Handles the execution of commands like `cd`, `echo`, `pwd`, `history` and `exit`.
- `input.c` and `input.h`: Handle reading user input and tokenizing it.

- `history.c` and `history.h`: Provide a command history feature that can store up to 20 commands across sessions.
- `builtin-commands/`: Directory containing the implementation of built-in commands (`cd.c`, `echo.c`, `pwd.c`, and `func.h` for header files).
- `history.txt`: File used to store command history across sessions (automatically created if not present).
- `makefile`: Makefile for building the project.

Built-in Commands

CD

The `cd` command allows the user to change the current working directory. It supports the following options:

- `cd`: Go to the home directory.
- `cd ..`: Move up one level in the directory hierarchy.
- `cd -`: Move to the previous directory.
- `cd ~`: Go to the home directory.
- `cd .`: Stays in the current directory.
- `cd [directory]`: Attempt to change to the specified directory.

ECHO

The `echo` command displays lines of text or strings passed as arguments. It handles spaces and tabs between words.

PWD

The `pwd` command prints the absolute pathname of the current working directory.

HISTORY

The `history` command displays the command history. By default, it shows the last 10 commands, but you can specify a different count. It also stores up to 20 commands across sessions, overwriting the oldest commands if the limit is exceeded. Identical consecutive commands are not stored.

Handling Repeated Commands and Persistence of History

Error Handling

The code uses `perror.h` to handle errors appropriately. If a command cannot be executed or returns an error, it is handled accordingly.

Also has customised error messages for invalid commands in RED color.

Modular Structure

The code is organized into separate modules, each responsible for a specific functionality. This ensures a well-structured and modular codebase.