> psh

a fancy POSIX-like shell

The Team

Contributors:

- Aditya R
- Krishna Kumar
- Alayna Monteiro
- Sumithra Suresh
- Siddhartha Rao

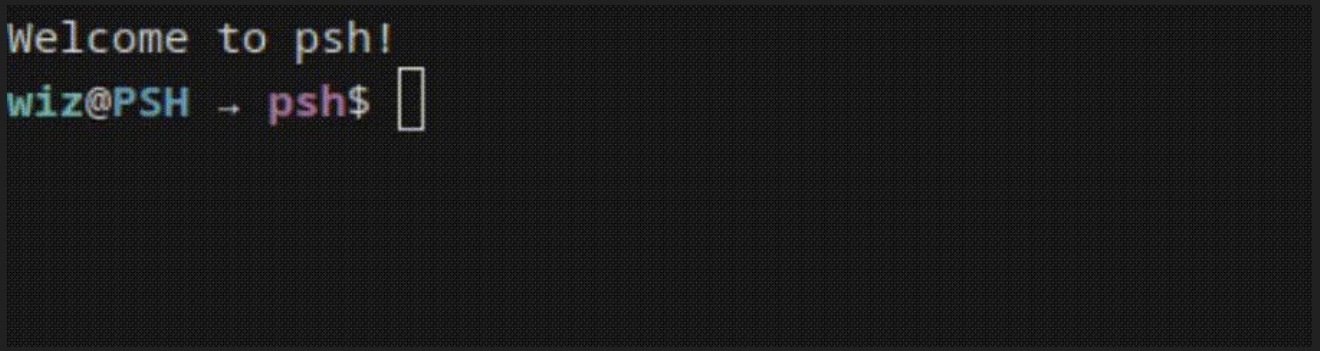
Mentors:

- Nathan Paul
- Tejas R
- Anupam G
- Navneet Nayak

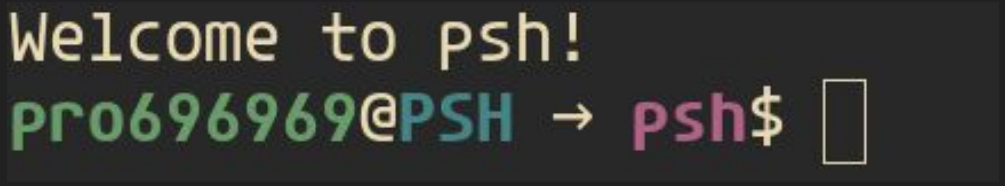
homebrew-ec-foss/psh

What’s a shell?

- A program helping users to interact with the OS using cli
- Eg: Bash, Zsh, Dash

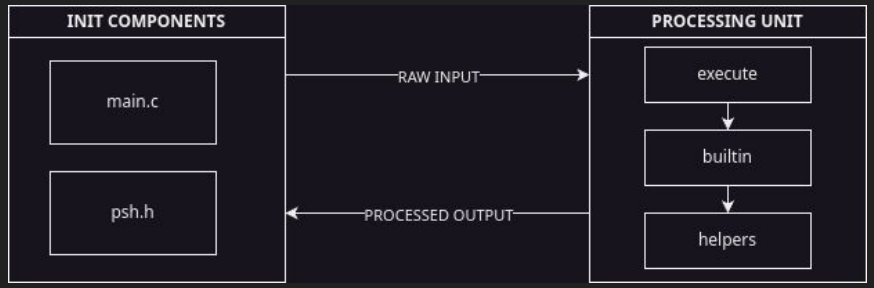


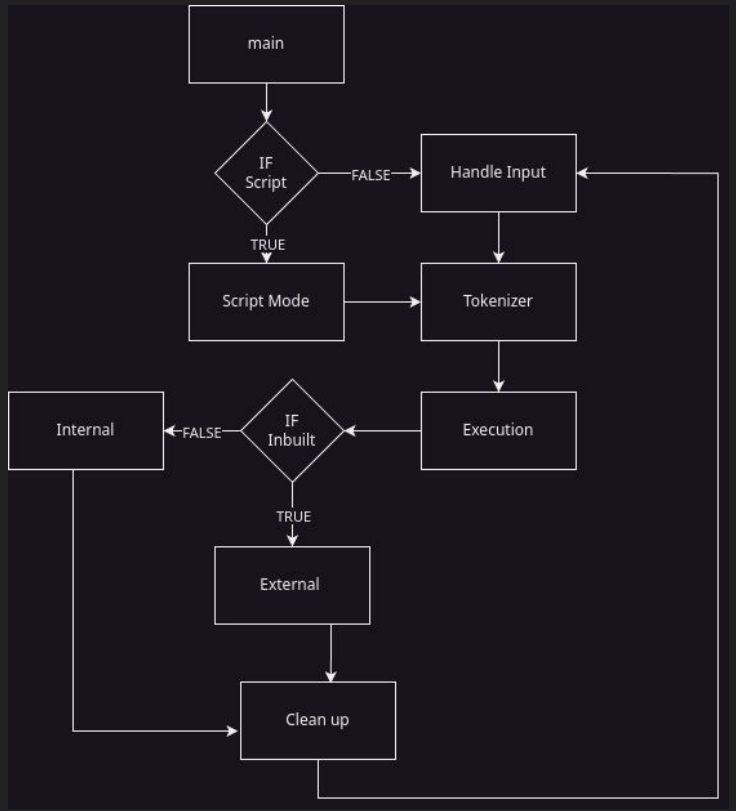
What’s psh ?



- `psh` aims to be a POSIX-like compliant shell.
- Building it helps in understanding and learning how shells work under the hood

Architecture





- Script Mode check
- Handle Input
- Tokenizer
- Execution
- Execution Paths:
  - Internal commands
  - External commands
- Cleanup

Key features of psh:

- Shell scripting capabilities with **.sh** files
- Handling **signal-interrupts** internally
- **Syntax highlighting** and basic **colour** theming
- Browse through the previous commands executed using ⬆ **arrow** keys
- **Autocomplete** commands

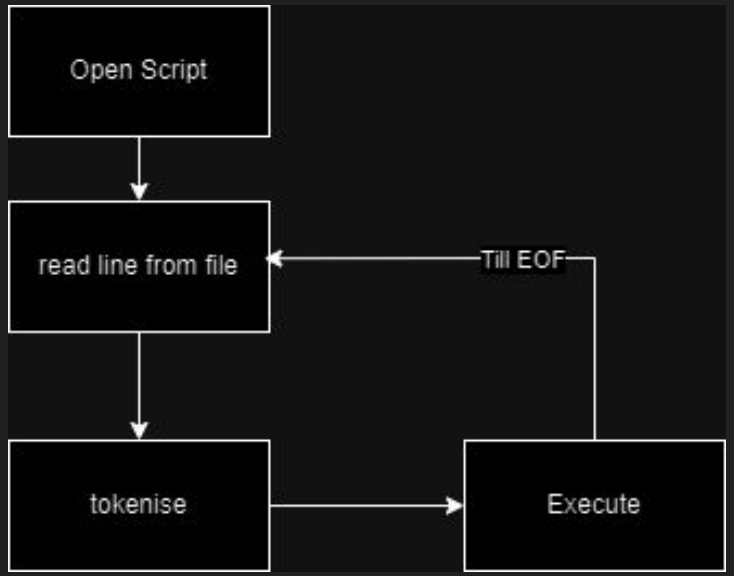
Demo

Script mode

Script is interpreted and executed by the shell.

Parsing is, reading the script to interpret its command and involves executing this commands sequentially.

```
1 # script.sh
2 ls
3 a=10
4 echo hii
5 echo hello there
6 echo $a
```



Arrow keys

Used ‘`termios`’ to enable raw mode and disable canonical mode allowing immediate character-by-character input processing.

- ↑: displays the previous command
- →: Moves the cursor to the right
- ←: Moves the cursor to the left
- ↓: Cycles through previous commands

Syntax highlighting

- Incomplete commands appear yellow.
- Complete commands appear green.

Colors

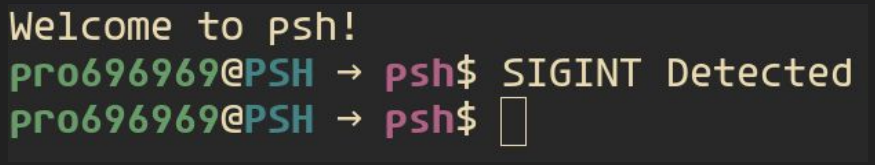
- PS1 shell variable defines the text printed before the cursor.
- pshrc stores the PS1 variable.

Multiple sessions

- Multiple sessions can be run at the same time.
- Each session has its own unique session ID and temporary session file.

Signals and keybinds

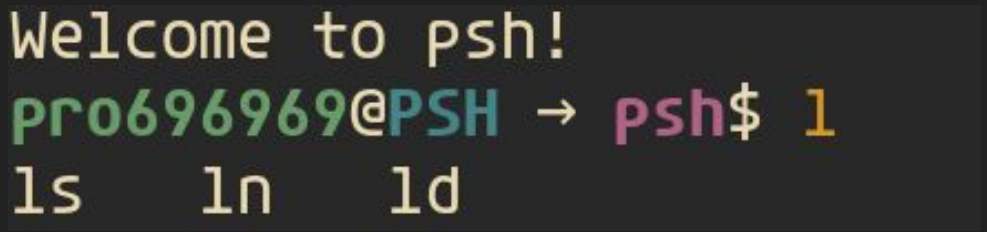
- Ctrl + C : SIGINT (Interrupt Signal)
- Ctrl + D : EOF
- Ctrl + L : Clears the screen



Autocomplete

- When tab is pressed it shows 3 closest suggestions for the entered command
- Implemented using fuzzy search algorithm

Eg:



Other basic features :

- Built in commands such as cd, pwd, exit, fc, echo, alias, export, read, type ...
- Run any external commands
- For loops
- Globbing and expansion(? and *)
- Works across multiple sessions

Key Learnings

- Acquired an understanding of POSIX (Portable Operating System Interface) standards and ensuring compatibility across systems.
- Gained experience with system calls and library functions.
- Handling input and parsing commands.
- Handling dynamic memory allocation and deallocation in C programming.

Conclusion

- Successfully developed a POSIX-like compliant shell over the course of 5 weeks.
- Implemented core features of a shell.
- Added user-friendly features like autocomplete, syntax highlighting and script mode to improve usability.
- Feel free to contribute!

Thank You!!