# Alternative Linux Shell in C: IMC Shell

# Documentation

## Authors:

## Lisa-Maria Hollaus

## Selin R. Meşeli

## Due Date: 02.02.2025, 23:59

**“IMC Shell: IMCSH”**

**Authors: Hollaus, Meseli**

This project develops an **Alternative Linux shell in C named IMCSH**, which interacts with users through standard input (stdin). The shell processes user commands and displays the output via standard output (stdout). As direct execution of standard Linux commands is not permitted, the project implements its own set of commands to handle various operations.

**Table of Contents**

* [IMC](http://localhost:63342/markdownPreview/1598062234/markdown-preview-index-516810009.html?_ijt=t86ti3tehp37555pq0q0350hti#final-assignment-simple-imc-messaging-protocol) Shell: IMCSH - [Authors: Hollaus, Meseli](http://localhost:63342/markdownPreview/1598062234/markdown-preview-index-516810009.html?_ijt=t86ti3tehp37555pq0q0350hti#authors-hollaus-meseli)
  + [Requirements](http://localhost:63342/markdownPreview/1598062234/markdown-preview-index-516810009.html?_ijt=t86ti3tehp37555pq0q0350hti#requirements)
  + [Usage](http://localhost:63342/markdownPreview/1598062234/markdown-preview-index-516810009.html?_ijt=t86ti3tehp37555pq0q0350hti#usage):
    - [Preparations:](http://localhost:63342/markdownPreview/1598062234/markdown-preview-index-516810009.html?_ijt=t86ti3tehp37555pq0q0350hti#preparations)
    - [How](http://localhost:63342/markdownPreview/1598062234/markdown-preview-index-516810009.html?_ijt=t86ti3tehp37555pq0q0350hti#start-the-daemon) to Start the Project
    - Terminal
* Basic Commands
* Features and Functionality
  + [Implementation Notes](http://localhost:63342/markdownPreview/1598062234/markdown-preview-index-516810009.html?_ijt=t86ti3tehp37555pq0q0350hti#implementation-notes)
  + [Project](http://localhost:63342/markdownPreview/1598062234/markdown-preview-index-516810009.html?_ijt=t86ti3tehp37555pq0q0350hti#architecture-overview) Structure
  + [Limitations](http://localhost:63342/markdownPreview/1598062234/markdown-preview-index-516810009.html?_ijt=t86ti3tehp37555pq0q0350hti#known-limitations)
  + Future Enhancements
  + Conclusion

**Requirements**

* **Ubuntu 13.2.0**
* We developed our projects on Ubuntu, since we are Windows users. Users with different operating systems (e.g. MacOS, Linux) should be able to also run it with no problem.

**Usage**

**Preparations:**

* If not installed yet, a C Compiler needs to be installed.

🡪 **Linux: sudo apt-get install gcc make**

**🡪 MacOS: xcode-select –install**

**🡪 Windows:** Use WSL (Windows Subsystem for Linux)

* Once project files are there, the preferred terminal needs to be opened to navigate to the project folder **(cd + path/to/project/directory)**
* To use Makefile to compile the project, the user should run “make” in the terminal.

🡪 with no errors, imcsh.c should be created. If errors exist, user should fix them and re-run “make” again

* If not installed yet, a c Compiler needs to be installed (mentioned the alternatives for MacOS and Linux, we used Ubuntu for our Windows OS computers)

**How to Start with the Project**

* Once project files are there, the preferred terminal needs to be opened to navigate to the project folder **(cd + path/to/project/directory)**
* The user should use the command “make clean” to clean any previous buildings
* By using the command “make”, the program can be compiled.
* If no errors occur so far, which is the expected outcome, imcsh.c file should get running by the command “./imcsh, which activates the custom-made terminal

**Terminal**

Once the custom-made terminal is activated, “user@host>” will be seen in the terminal as the starting point. From there on, basic commands implemented can be entered.

**Basic Commands**

* **Execute a command to list all running processes with detailed information:**

user@host> exec ps -uax

* **Run a command to list files in the current directory in long format, and redirect the output the the given file name:**

user@host> exec ls -l > directory\_output.txt

* **Display version information:**

user@host> globalusage

* **Call the globalusage() function and redirect the output to the given file:**

user@host> globalusage > usage.txt

* **Quit the shell:**

user@host> quit

* **Execute a command to pause for input seconds and run in the background:**

user@host> exec sleep 3 &

**Features and Functionality**

**Executing Commands (**exec**)**

* Runs standard Linux commands
* Uses execvp() to execute commands directly

**Background Execution(**&**)**

* Appends & to a command to run it I the background
* The process ID (PID) is displayed, IMCSH does not wait for it to finish

**Output Redirection (**>**)**

* Redirects command output to a file
* Uses dup2() to modify standard output
* Opens the file in append mode (“a”) to prevent overwriting

**Built-In Commands**

* globalusage: Displays shell version and authors
* quit: Checks for background processes before exiting

**Managing Background Processes**

* Background processes are tracked
* Exiting the shell prompts the user to confirm quitting if processes are still running

**Implementation Notes**

* Uses fork() and execvp() to handle execution
* Uses waitpid() with WNIHANG to check background process completion
* Implements process tracking for proper management of background execution
* Uses dup2() for output redirection without overwriting files
* Handles user confirmation before quitting if background processes are running

**Project Structure (After having tried come basic commands)**

├── Documentation.docx

├── Makefile

├── README.md

├── directory\_output.txt

├── functions.c

├── functions.h

├── imcsh.c

├── usage.txt

**Limitations**

* No command history with arrowed keys (Up arrow does not recall the previous commands)
* Does not support multiple redirections (>>)
* Limited background process management

**Future Enhancements**

* Command history support could be implemented for some more ease
* Adding tab-completion for command names

**Conclusion**

IMCSH is a simplified Linux shell designed for our educational project. It provides fundamental shell functionalities to execute command, handling background processes and redirecting the output. With enhancements, it could get to a more expanded implementation with richer features being included.