Basic Shell Under Linux

Overview

The basic shell under Linux project is a simple shell program built in C for Linux, designed to execute both internal and external commands, with support for background processes. This shell includes basic commands such as cd, exit, and jobs, while also handling external commands like ls, cat, and echo using fork and exec system calls.

Key Features

1. Internal Commands:

- exit: Terminates the shell, ensuring that all background processes are cleaned up before exiting.
- cd: Changes the current working directory. Usage: cd <directory>, cd .. for moving to the parent directory.
- jobs: Displays a list of all background processes with their PIDs and commands.

2. External Commands:

- Any command not recognized as internal is treated as an external command, executed using exec system calls.
- Supports standard Linux commands like ls, echo, cat, and more.

3. Background Processes:

- Commands ending with & are executed in the background.
- A maximum of 4 background processes can run simultaneously.
- If more than 4 background processes are attempted, an error message is shown.

4. Process Management:

- Uses fork to create child processes and exec to run external commands.
- Background processes are managed and cleaned up in real-time.
- Foreground processes are reaped with waitpid to prevent zombies.

5. Error Handling:

- Invalid commands or system call failures generate appropriate error messages with the command name and errno.

6. User Experience:

- The shell runs in an infinite loop, displaying a prompt run_shell$ for input.
- Empty commands (pressing Enter) are handled gracefully by re-displaying the prompt.

Command Examples

- cd directory_name: Changes the current directory.
- exit: Exits the shell after cleaning up background processes.
- jobs: Lists currently running background processes.
- ls -l: Executes ls in long format.
- sleep 10 &: Runs sleep for 10 seconds in the background.

Error Messages

- run_shell: invalid command: Displayed for invalid commands.
- run_shell: too many background commands running: Shown when more than 4 background commands are running.
- run_shell: <system_call> failed, errno is <errno>: Shown when a system call fails.

Implementation Details

- Internal Command Handling: Directly processes internal commands (cd, exit, jobs) without creating child processes.
- Background Processes: Commands ending with & are run using fork and exec.
- System Calls: Uses chdir for changing directories, fork, exec, and waitpid for process management.
- Signal Handling: Continuously checks and cleans up completed background processes.

Installation and Execution

1. Compilation: Run "make" to compile the shell into an executable called run_shell.
2. Cleanup: Run "make clean" to remove executables and object files.
3. Execution: Run the shell with ./run_shell.