

Using the Linux Bash Terminal

Learn how to navigate, manage, and automate with Bash

by Your Name

What is Bash?

- Bash stands for **Bourne Again Shell**
- It's a command-line interface (CLI) used to interact with the Linux system
- Provides tools for file manipulation, process control, scripting, and automation

Why Use the Terminal?

- More powerful and flexible than GUIs
- Allows automation and scripting
- Remote administration via SSH
- Essential for servers, development, and system management

Bash Prompt Structure

```
$ user@hostname:~/directory$
```

- `user` — your username
- `hostname` — the machine name
- `~/directory` — current working directory



Basic Navigation

```
pwd      # Print current directory
ls       # List files
cd /path # Change directory
cd ..    # Go up one directory
cd ~     # Go to home directory
```



File and Directory Management

```
touch file.txt      # Create file
mkdir folder        # Create directory
cp file1 file2      # Copy file
mv old new          # Move or rename
rm file.txt         # Remove file
rm -r folder        # Remove directory recursively
```



Viewing Files

```
cat file.txt          # Print file contents
less file.txt         # Scroll through file
head file.txt         # Show first 10 lines
tail file.txt         # Show last 10 lines
grep "text" file.txt  # Search for text
```

Permissions and Ownership

```
ls -l          # Show permissions
chmod 755 file.sh # Change permissions
chown user:group file # Change owner and group
```

Example permission format: - rwxr-xr - -

Process Management

```
ps                # Show processes
top               # Interactive process viewer
kill PID          # Terminate process
kill -9 PID       # Force kill process
jobs              # List background jobs
fg / bg           # Bring job to foreground/background
```



Redirection and Pipes

```
command > file      # Redirect output to file  
command >> file     # Append output to file  
command < file      # Read input from file  
command1 | command2 # Pipe output of one into another
```

Example: `cat file.txt | grep "error"`



Environment Variables

```
echo $HOME  
echo $PATH  
export MYVAR="Hello"  
echo $MYVAR  
unset MYVAR
```

Writing Shell Scripts

```
#!/bin/bash
# My first script
echo "Hello, $USER!"
pwd
ls
```

Save as `script.sh`, make executable: `chmod +x script.sh`

Run: `./script.sh`

Control Structures

```
if [ -f file.txt ]; then
    echo "File exists"
else
    echo "File not found"
fi

for file in *.txt; do
    echo $file
done
```

History and Aliases

```
history          # View command history
!45              # Run command #45 from history
alias ll='ls -lh' # Create alias
unalias ll       # Remove alias
```

Keyboard Shortcuts

- `Ctrl + C` – Cancel command
- `Ctrl + L` – Clear screen
- `Ctrl + R` – Reverse search history
- `Tab` – Autocomplete
- `!!` – Repeat last command



Tips and Tricks

- Use `man` command for help
- Chain commands with `&&` or `;`
- Use `sudo` for admin tasks
- Learn `tmux` or `screen` for multitasking

Summary

- Bash is powerful for interacting with Linux systems
- Mastering it improves efficiency and automation
- Explore scripting and customization next!

“The command line is where power users live.”