

UNIX Beginner's Handbook

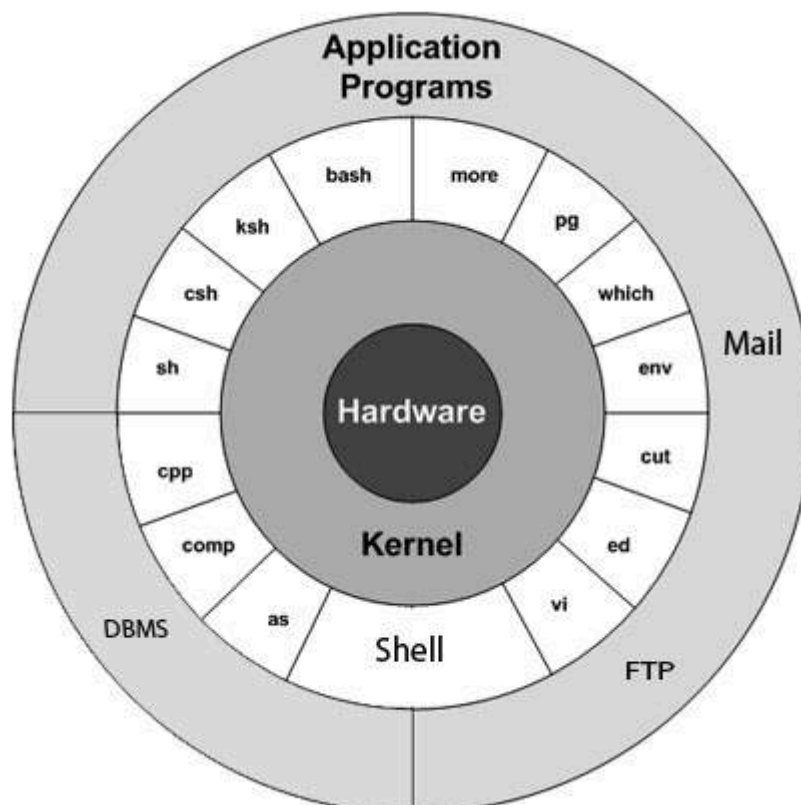
Welcome to the UNIX Beginner's Handbook, a comprehensive guide designed to help newcomers navigate the world of UNIX and its command-line interface. This handbook will provide detailed notes and explanations in simple English to ensure that even those with no prior experience can understand and utilise UNIX effectively.

Getting Started

Unix is a multiuser, multitasking operating system that was first developed in the 1960s and has been under constant development ever since. It is known for its stability, reliability, and flexibility, making it suitable for servers, desktops, and laptops.

UNIX Architecture

The Unix operating system comprises three parts: the **kernel**, the **standard utility programs**, and the **system configuration files**.



The computer programs that allocate the system resources and coordinate all the details of the computer's internals is called the **operating system(OS)** or **Kernel**.

Users communicate with the kernel through a program known as the **shell**. The shell is the command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel.

0x00-shell_basics

General Concepts

RTFM: [Read The Manual](#)

RTFM stands for "Read The Manual" and is an acronym used to suggest that the answer to a question can be found in the available documentation or manual. It's a reminder to utilise the resources at hand before seeking help from others.



Shebang: The Script's Starting Point

A **shebang** is a character sequence (**#!**) at the beginning of a script file that specifies which interpreter should be used to execute the script.

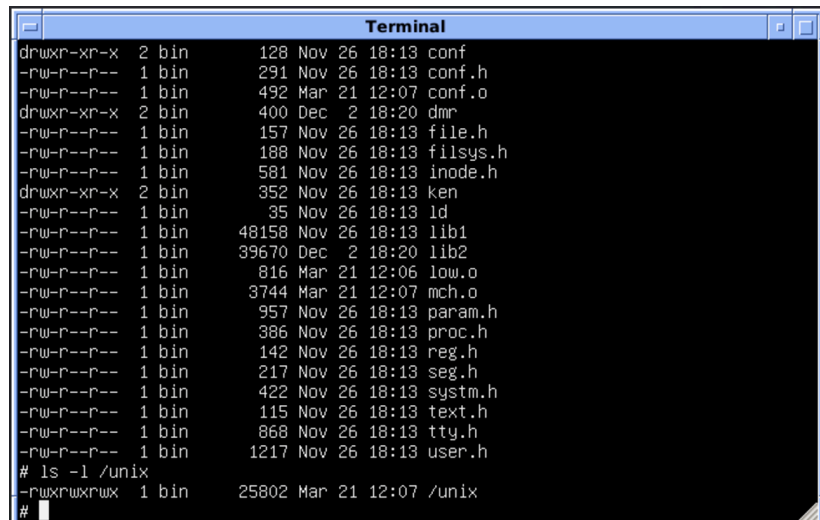
```
#!/bin/bash
```

The Shell: Your Command-Line Interface

The **shell** is a command-line interface that allows users to interact with the operating system. It interprets commands and acts as an intermediary between the user and the system's services.

Terminal vs. Shell: Understanding the Difference

A **terminal**, or terminal emulator, is a program that allows you to interact with the shell. The shell is the interface that processes commands and returns output. While they work together, they are distinct; the terminal is the graphical window, and the shell is the command-line interface within that window .



```
Terminal
drwxr-xr-x  2 bin      128 Nov 26 18:13 conf
-rw-r--r--  1 bin      291 Nov 26 18:13 conf.h
-rw-r--r--  1 bin      492 Mar 21 12:07 conf.o
drwxr-xr-x  2 bin      400 Dec  2 18:20 dmr
-rw-r--r--  1 bin      157 Nov 26 18:13 file.h
-rw-r--r--  1 bin      188 Nov 26 18:13 filsys.h
-rw-r--r--  1 bin      581 Nov 26 18:13 inode.h
drwxr-xr-x  2 bin      352 Nov 26 18:13 ken
-rw-r--r--  1 bin        35 Nov 26 18:13 ld
-rw-r--r--  1 bin    48158 Nov 26 18:13 lib1
-rw-r--r--  1 bin   39670 Dec  2 18:20 lib2
-rw-r--r--  1 bin      816 Mar 21 12:06 low.o
-rw-r--r--  1 bin    3744 Mar 21 12:07 mch.o
-rw-r--r--  1 bin      957 Nov 26 18:13 param.h
-rw-r--r--  1 bin     386 Nov 26 18:13 proc.h
-rw-r--r--  1 bin     142 Nov 26 18:13 reg.h
-rw-r--r--  1 bin     217 Nov 26 18:13 seg.h
-rw-r--r--  1 bin     422 Nov 26 18:13 systm.h
-rw-r--r--  1 bin     115 Nov 26 18:13 text.h
-rw-r--r--  1 bin     868 Nov 26 18:13 tty.h
-rw-r--r--  1 bin    1217 Nov 26 18:13 user.h
# ls -l /unix
-rwxrwxrwx  1 bin    25802 Mar 21 12:07 /unix
#
```

Shell Prompt: Ready for Commands

The **shell prompt** is the text displayed in a terminal to indicate that the shell is ready to receive commands. It often contains information like the current user and the working directory .

Using History: Command Recall

The **history** feature in the shell allows you to view and recall previously entered commands, which can save time and reduce typing errors. You can navigate through your command history using the 'Up' and 'Down' arrow keys .

Navigation

Basic Commands: cd, pwd, ls

cd (change directory): Changes the current working directory .

pwd (print working directory): Displays the current working directory .

ls (list): Lists files and directories in the current directory .

Filesystem Navigation

To **navigate** the filesystem, you use commands like **cd** to move between directories. The **.** directory refers to the current directory, and **..** refers to the parent directory .

Working Directory: Your Current Location

The **working directory** is the directory you are currently operating in. You can print it with `pwd` and change it with `cd` .

Root Directory: The Top of the Filesystem

The **root directory** is the topmost directory in the filesystem, denoted by `/`. It contains all other directories and files .

Home Directory: Your Personal Space

Each user has a **home directory**, typically denoted by `~`, which is their personal space on the system .

Hidden Files: The Invisible Data

Hidden files start with a dot (`.`) and are not shown by default when using the `ls` command. To list them, you can use `ls -a` .

Navigating Back: The `cd -` Command

The `cd -` command takes you back to the previous directory you were in, making it easy to toggle between two locations .

Looking Around

Exploring Files: `ls`, `less`, `file`

`ls`: Lists files and directories .

`less`: Allows you to view the contents of a file page by page .

`file`: Determines the type of a file .

Options and Arguments: Command Customization

Commands can be **customised** with options (usually preceded by a `-`) and arguments to perform specific tasks .

Understanding `ls` Long Format

The `ls -l` command displays files and directories in long format, which includes permissions, ownership, size, and modification date .

A Guided Tour

Creating Links: `ln` Command

The **ln** command is used to create links between files. There are two types of links: **symbolic links** and **hard links** .

Common Directories: What's Inside

Common directories like **/bin**, **/etc**, **/var**, and **/home** contain essential system files, configurations, and user data .

Symbolic vs. Hard Links

Symbolic links (soft links) point to another file or directory and are a reference to the original file .

Hard links are direct pointers to the data on the disk and behave as if they are the actual file. The difference between them is that deleting the original file does not affect hard links, but it does invalidate symbolic links .

Manipulating Files

File Operations: cp, mv, rm, mkdir

cp: Copies files or directories .

mv: Moves or renames files or directories .

rm: Removes files or directories .

mkdir: Creates new directories .

Wildcards: Pattern Matching

Wildcards are characters like ***** and **?** that can match multiple characters in filenames, allowing you to perform operations on groups of files .

Working with Commands

Discovering Commands: type, which, help, man

type: Displays information about command type .

which: Shows the location of a command's executable file .

help: Provides help for shell built-ins .

man: Displays the manual pages for commands .

Command Types: Built-in, Alias, and More

Commands can be built-in to the shell, **aliases** created by the user, or executable programs located in the system's path .

Using Help and Man: Getting Command Information

Use **help** for built-in shell commands .

Use **man** for a comprehensive manual of commands, which includes usage, options, and examples .

Reading Man Pages: The Manual

Man pages are structured documents that provide detailed information about commands. They are divided into sections for user commands, system calls, and library functions, among others .

Keyboard Shortcuts for Bash

Common shortcuts like **Ctrl+C** (interrupt process), **Ctrl+Z** (suspend process), and **Ctrl+L** (clear screen) enhance the command-line experience .

LTS: Long-Term Support

LTS stands for **Long-Term Support**, which refers to versions of software that are maintained with updates and bug fixes for an extended period, providing stability and reliability .

This handbook aims to provide a solid foundation for new UNIX users. Remember, practice is key to becoming proficient in navigating and using the UNIX command-line interface.

Happy learning! 😊