# File commands

Here are some commonly used commands for Linux file and directory operations, classified according to their functionalities:

**Navigating and viewing directories**:

- `ls`: List directory contents
- `cd`: Change the current working directory
- `pwd`: Show the full path of the current working directory
- `tree`: Display directory content in a tree-like structure
- `cat`: View file content
- `more` or `less`: View file content one page at a time

**Managing files and directories**:

- `touch`: Create new files or update file access and modification times
- `mkdir`: Create new directories

**Viewing file and directory attributes**:

- `ls` (with different options): View detailed information about files and directories
- `file`: Determine the file type
- `stat`: Display detailed properties of files and directories
- `du`: Display disk usage of files and directories
- `df`: Display disk usage of file systems

**Compressing and decompressing files**:

- `tar`: Create archive files or extract files from archives
- `gzip`: Compress files

- `cp`: Copy files and directories
- `mv`: Move or rename files and directories
- `rm`: Delete files and directories
- `ln`: Create links or shortcuts
- `chmod`: Modify file and directory permissions
- `chown`: Change file and directory owner
- `chgrp`: Change file and directory group

- `gunzip`: Decompress files
- `zip`: Create ZIP archive files or extract files from ZIP archives
- `unzip`: Decompress ZIP archive files

**Searching for files and directories**:

- `find`: Search for files and directories in a specified directory
- `grep`: Search for specified patterns or text in files
- `locate`: Quickly find files using a database (requires updating the database)
- `whereis`: Show the location of binary files, source files, and help files
- `which`: Display the full path of a specified command

These are just some of the commonly used Linux commands for file and directory operations, categorized based on their functionalities. There are many more commands and options available for different scenarios and requirements. You can refer to relevant documentation to learn more about the usage and options of each command.

---

# cheat sheet

| Command | Description |
| --- | --- |
| `ls` | List files in the directory |
| `ls -a` | List all files (shows hidden files) |
| `ls -l \| sort -k 5 -n` | Sort files by size linux |
| `locate [name]` | Find all files and directories related to a particular name: |
| `pwd` | Show directory you are currently working in |
| `mkdir [directory]` | Create a new directory |
| `rm [file_name]` | Remove a file |
| `rm -r [directory_name]` | Remove a directory recursively |
| `rm -rf [directory_name]` | Recursively remove a directory without requiring confirmation |
| `cp [file_name1] [file_name2]` | Copy the contents of one file to another file |
| `cp -r [directory_name1] [directory_name2]` | Recursively copy the contents of one directory to a second directory |
| `mv [file_name1] [file_name2]` | Rename [file_name1] to [file_name2] with the command |
| `ln -s /path/to/[file_name] [link_name]` | Create a symbolic link to a file |
| `touch [file_name]` | Create a new file using touch |
| `more [file_name]` | Show the contents of a file |
| `cat [file_name]` | or use the cat command |
| `cat [file_name1] >> [file_name2]` | Append file contents to another file |
| `head [file_name]` | Display the first 10 lines of a file with head command |
| `tail [file_name]` | Show the last 10 lines of a file |
| `wc` | Show the number of words, lines, and bytes in a file using wc |
| `ls \| xargs wc` | List number of lines/words/characters in each file in a directory with the xargs command |
| `cut -d[delimiter] [filename]` | Cut a section of a file and print the result to standard output |
| `[data] \| cut -d[delimiter]` | Cut a section of piped data and print the result |

| | |
|---|---|
| | to standard output |
| `awk '[pattern] {print $0}' [filename]` | Print all lines matching a pattern in a file |
| `diff [file1] [file2]` | Compare two files and display differences |
| `source [filename]` | Read and execute the file content in the current shell |
| `[command] \| tee [filename] >/dev/null` | Store the command output in a file and skip the terminal output |
| `cd ~` | Change the directory to the user's home directory. |
| `cd ..` | Move one directory level up (parent directory). |
| `cd -` | Switch to the previous working directory. |
| `find . -type f -exec du -h {} + \| sort -rh \| head -n 1` | find the largest files in linux |

# Absolute Path vs Relative Path

### Absolute Path

- An absolute path is a complete and unambiguous path that starts from the root directory (e.g., ).
- It specifies the full location of a file or directory in the file system, starting from the top-level.
- Absolute paths always remain the same regardless of the current working directory.

**Example**:

| |
|---|
| `/home/user/documents/file.txt` |
| `/var/www/html/index.html` |
| `/usr/bin/python3` |

### Relative Path

- A relative path is a path that is specified relative to the current working directory.
- It does not start with the root directory and depends on the current location in the file system.
- Relative paths are typically shorter and are used within the context of a specific working directory.

**Example:**

| |
|---|
| `documents/file.txt` |
| `../parent_directory/file.txt` |
| `subdirectory/subfile.txt` |

# Linux file system

| Directory | Description | Example |
|---|---|---|
| `/bin` | Essential user binaries (commands) | `/bin/ls`, `/bin/cp`, `/bin/mkdir` |
| `/boot` | Boot loader files and kernel images | `/boot/vmlinuz`, `/boot/initrd.img` |
| `/dev` | Device files for hardware devices | `/dev/sda`, `/dev/tty`, `/dev/null` |
| `/etc` | System configuration files and directories | `/etc/passwd`, `/etc/hosts`, `/etc/ssh` |
| `/home` | Home directories for regular users | `/home/user1`, `/home/user2` |
| `/lib` | Shared libraries needed by system binaries | `/lib/libc.so.6`, `/lib64/ld-linux-x86-64.so.2` |
| `/media` | Mount point for removable media devices | `/media/usb`, `/media/cdrom` |
| `/mnt` | Temporary mount point for other file systems | `/mnt/external`, `/mnt/iso` |
| `/opt` | Optional application software packages | `/opt/java`, `/opt/apache` |
| `/proc` | Virtual file system containing kernel and process information | `/proc/cpuinfo`, `/proc/meminfo` |
| `/root` | Home directory for the root user | `/root/.bashrc`, `/root/.profile` |
| `/run` | Run-time data for processes (e.g., PID files, sockets) | `/run/lock`, `/run/user/1000` |
| `/sbin` | Essential system binaries (usually for system administration tasks) | `/sbin/ifconfig`, `/sbin/reboot` |
| `/srv` | Data for services provided by the system | `/srv/www`, `/srv/ftp` |
| `/tmp` | Temporary files | `/tmp/tempfile`, `/tmp/socket` |
| `/usr` | User-related programs and data, including user/binaries, libraries, and documentation | `/usr/bin/gcc`, `/usr/share/man` |
| `/var` | Variable data, such as log files, spool files, and temporary files | `/var/log/syslog`, `/var/spool/mail` |

# File permission in Linux

file permissions are represented using a 10-character string that consists of three sets of permissions for the owner, group, and others, respectively. The format is as follows:

- `- rwxrwxrwx`

Here's what each part of the string means:

- The first character indicates the type of the file: `-` for a regular file, `d` for a directory, `l` for a symbolic link, and so on.
- The next three characters (`rwx`) represent the read, write, and execute permissions for the file's owner.
- The next three characters (`rwx`) represent the read, write, and execute permissions for the group that the file belongs to.
- The last three characters (`rwx`) represent the read, write, and execute permissions for others, i.e., users who are neither the owner nor part of the group.

| Permission | Symbol | Description |
|---|---|---|
| Read | r | The ability to read the contents of a file or view the contents of a directory |
| Write | w | The ability to modify the contents of a file or create, rename, or delete files within a directory |
| Execute | x | The ability to execute a file or enter a directory |

File permission examples:

| Permission | Owner | Group | Others |
|---|---|---|---|
| 777 | rwx | rwx | rwx |
| 755 | rwx | r-x | r-x |
| 644 | rw- | r– | r– |

Directory permission examples:

| Permission | Owner | Group | Others |
|---|---|---|---|
| 777 | rwx | rwx | rwx |

| | | | |
|---|---|---|---|
| 755 | rwx | r-x | r-x |