

## Linux Commands Cheat Sheet

### File and Directory Operations

- List Files and Directories
- Change Directory
- Copy Files and Directories
- Move/Rename Files and Directories
- Remove Files and Directories
- Create Files and Directories

### File Content Operations

- View File Content
- Edit Files

### File Permissions and Ownership

- Change Permissions
- Change Owner

### File Search and Filtering

- Find Files
- Search Content in Files

### User and Group Management

- User Operations
- Group Operations

### Process Management

- View Running Processes
- Kill Processes
- Background and Foreground Jobs

### Disk and Storage Management

- Check Disk Usage
- Mount/Unmount Filesystems

### Network Configuration and Troubleshooting

- Network Information
- Test Connectivity
- Network Configuration

### System Information

- Display System Information

### Package Management

- Install, Update, Remove Packages

### Text Processing and Manipulation

- String and Text Operations
- Archive Operations

### System Administration

- Service Management
- System Logs

### Environment Variables and Shell

- Environment Variables
- Aliases and Functions

### Advanced Operations

- File Transfer
- SSH and Remote Access
- Pipes and Redirection
- Regular Expressions

Cron Jobs and Scheduling

    Cron Job Management

Tips and Tricks

    Command History

    Command Substitution

## **Linux Commands Cheat Sheet**

# File and Directory Operations

---

## List Files and Directories

These commands help you view and explore the contents of directories, which is essential for navigation and understanding file structure in Linux.

**Use Cases:** - Quick viewing of directory contents - Detailed inspection of file permissions and sizes - Hidden file discovery

```
ls                # List files in current directory
ls -l            # List with details (permissions, owner, size, date)
ls -a            # List all files including hidden files
ls -R            # List recursively (all subdirectories)
ls -lh           # List with human-readable file sizes
```

---

## Change Directory

Navigation through the filesystem is fundamental to working efficiently in Linux environments.

**Use Cases:** - Move between project directories - Access different parts of the filesystem - Return to previous locations quickly

```
cd /path/to/dir  # Change to specific directory
cd ~              # Change to home directory
cd -              # Change to previous directory
cd ..             # Change to parent directory
pwd              # Print current working directory
```

---

## Copy Files and Directories

Copying is critical for backing up data, duplicating configurations, and managing file deployments across the system.

**Use Cases:** - Backup important files and configurations - Duplicate files for modification - Deploy applications and scripts - Archive data for safekeeping

```
cp file1 file2    # Copy file
cp -r dir1 dir2    # Copy directory recursively
cp -v file1 file2  # Copy with verbose output
cp -p file1 file2  # Copy and preserve permissions/ownership
cp -u file1 file2  # Copy only if source is newer
```

---

## Move/Rename Files and Directories

Moving and renaming files are essential operations for organizing files and managing directory structures.

**Use Cases:** - Organize files into appropriate directories - Rename files with meaningful names - Reorganize project structure - Clean up temporary files

```
mv oldname newname # Rename file or directory
mv file /path/to/destination # Move file to another directory
mv -v file destination # Move with verbose output
mv -i file destination # Move with confirmation prompt
```

---

## Remove Files and Directories

Deletion operations help maintain disk space and remove unwanted files from your system.

**Use Cases:** - Clean up temporary or test files - Remove outdated backups - Manage disk space - Delete sensitive files securely

```
rm file                # Delete file
rm -r directory        # Delete directory and contents
rm -f file             # Force delete without confirmation
rm -i file             # Delete with confirmation prompt
rm -v file             # Delete with verbose output
```

---

## Create Files and Directories

Creating new files and directories forms the foundation of organizing and storing data.

**Use Cases:** - Initialize new project directories - Create configuration files - Create scripts for automation - Organize project structure

```
touch filename          # Create empty file or update timestamp
mkdir dirname           # Create new directory
mkdir -p /path/to/nested/dir # Create nested directories
cat > filename << EOF   # Create file with content
content here
EOF
```

---

## File Content Operations

### View File Content

Viewing file content is essential for reading configurations, logs, and code without modifying them.

**Use Cases:** - Read configuration files - View application logs - Inspect code and scripts - Verify file contents

```
cat filename            # Display entire file
cat file1 file2         # Display multiple files
head -n 20 filename     # Display first 20 lines
tail -n 20 filename     # Display last 20 lines
tail -f filename        # Follow file (useful for logs)
more filename           # Display file with pagination
less filename           # Display with better navigation
wc -l filename          # Count lines in file
```

---

### Edit Files

Text editing is crucial for modifying configurations, scripts, and code files in a Linux environment.

**Use Cases:** - Modify configuration files - Write and edit scripts - Update system files - Edit application settings

```
nano filename          # Open nano editor (beginner-friendly)
vim filename            # Open vim editor (powerful but steep learning curve)
vi filename             # Open vi editor (basic text editor)
sed 's/old/new/' filename # Replace text (stream editor)
```

---

## File Permissions and Ownership

### Change Permissions

File permissions control who can read, write, and execute files—critical for system security.

**Use Cases:** - Make scripts executable - Restrict access to sensitive files - Set appropriate permissions for applications - Enforce security policies

```
chmod 755 filename           # Set permissions (rwx for owner, rx for others)
chmod +x filename            # Add execute permission
chmod u+w filename           # Add write permission for owner
chmod -R 755 directory       # Change permissions recursively
```

---

## Change Owner

Changing ownership is important for managing access rights and application permissions.

**Use Cases:** - Transfer file ownership - Set application-specific permissions - Fix permission issues - Manage multi-user environments

```
chown user filename          # Change file owner
chown user:group filename    # Change owner and group
chown -R user:group directory # Change recursively
```

---

## File Search and Filtering

### Find Files

Finding files efficiently is essential for locating specific files in large directory structures.

**Use Cases:** - Locate configuration files - Find all files of a specific type - Search by modification date - Find large files consuming disk space

```
find /path -name filename    # Find file by name
find /path -type f -name "*.txt" # Find all text files
find /path -type d -name dirname # Find directories
find /path -mtime -7         # Find files modified in last 7 days
find /path -size +100M       # Find files larger than 100MB
find /path -exec command {} \; # Execute command on found files
```

---

### Search Content in Files

Text searching helps you find specific information within file contents.

**Use Cases:** - Find error messages in logs - Locate specific configurations - Debug code by searching patterns - Extract relevant information

```
grep "pattern" filename      # Search for pattern in file
grep -r "pattern" /path      # Search recursively
grep -i "pattern" filename    # Case-insensitive search
grep -n "pattern" filename    # Show line numbers
grep -c "pattern" filename    # Count matching lines
grep -E "regex_pattern" filename # Use regular expressions
```

---

## User and Group Management

### User Operations

Managing users is fundamental for multi-user systems and access control.

**Use Cases:** - Create accounts for team members - Manage user permissions - Control system access - Audit user activities

```
useradd username             # Create new user
userdel username             # Delete user
passwd username              # Set/change user password
id username                  # Display user ID and groups
whoami                      # Display current user
```

---

## Group Operations

Groups allow you to manage permissions for multiple users efficiently.

**Use Cases:** - Grant group-based access - Organize users by role - Simplify permission management - Control resource access

```
groupadd groupname      # Create new group
groupdel groupname      # Delete group
usermod -g groupname username # Add user to group
usermod -aG groupname username # Add user to group (keep other groups)
id -Gn username         # List user's groups
```

---

## Process Management

### View Running Processes

Monitoring processes helps you understand system activity and troubleshoot performance issues.

**Use Cases:** - Monitor running applications - Identify resource-intensive processes - Troubleshoot system issues - Manage background tasks

```
ps                # List current processes
ps aux            # List all processes with details
ps -ef            # List all processes (alternative format)
top               # Interactive process monitor
htop              # Enhanced interactive process monitor
pgrep processname # Find process by name
```

---

### Kill Processes

Terminating processes allows you to stop stuck or unwanted applications.

**Use Cases:** - Stop runaway processes - Restart services - Free system resources - Troubleshoot hung applications

```
kill PID          # Terminate process by ID
kill -9 PID       # Force kill process
kill -STOP PID    # Pause process
kill -CONT PID    # Resume process
killall processname # Kill all instances of process
pkill processname  # Kill process by name
```

---

### Background and Foreground Jobs

Job control allows you to manage multiple tasks simultaneously in a terminal session.

**Use Cases:** - Run long-running tasks in background - Manage multiple terminal jobs - Resume suspended tasks - Optimize terminal workflow

```
command &        # Run command in background
jobs              # List background jobs
fg %jobnumber     # Bring job to foreground
bg %jobnumber     # Resume job in background
Ctrl+Z           # Suspend current job
Ctrl+C           # Terminate current job
nohup command &  # Run command immune to hangups
```

---

## Disk and Storage Management

### Check Disk Usage

Monitoring disk usage helps prevent storage problems and optimize system performance.

**Use Cases:** - Monitor disk space availability - Identify space-consuming directories - Plan disk capacity - Troubleshoot storage issues

```
df -h           # Display disk space usage (human-readable)
df -i           # Display inode usage
du -sh directory # Show directory size
du -h directory # Show size of all subdirectories
lsblk           # List block devices
fdisk -l        # List partitions
```

---

## Mount/Unmount Filesystems

Mounting allows you to access additional storage devices and filesystems.

**Use Cases:** - Mount external drives - Access network shares - Connect USB devices - Manage multiple filesystems

```
mount /dev/device /mount/point # Mount filesystem
umount /mount/point            # Unmount filesystem
mount -t type device /mount/point # Mount with specific type
mount | grep device             # Check mounted filesystems
fstab                           # View permanent mount configuration
```

---

## Network Configuration and Troubleshooting

### Network Information

Understanding network configuration is essential for connectivity troubleshooting.

**Use Cases:** - Verify network connectivity - Check IP addresses - Diagnose network issues - Monitor network traffic

```
ifconfig           # Display network interfaces (older systems)
ip addr show        # Display IP addresses (modern systems)
ip route show       # Display routing table
netstat -tuln       # Display listening ports
ss -tuln            # Socket statistics (modern alternative)
hostname            # Display system hostname
cat /etc/hostname   # View hostname configuration
```

---

### Test Connectivity

Network testing helps you diagnose connectivity problems quickly.

**Use Cases:** - Verify host reachability - Test DNS resolution - Troubleshoot network connectivity - Validate network configuration

```
ping hostname      # Send ICMP packets to test connectivity
traceroute hostname # Trace route to destination
nslookup hostname  # Query DNS records
dig hostname        # Advanced DNS query tool
curl URL            # Download file or test URL
wget URL            # Download file with resume capability
telnet hostname port # Test port connectivity
```

---

## Network Configuration

Configuring network settings enables proper communication and connectivity.

**Use Cases:** - Set IP addresses - Configure DNS servers - Manage network interfaces - Establish VPN connections

```
ip addr add IP/SUBNET dev interface      # Add IP address
ip addr del IP/SUBNET dev interface      # Remove IP address
ip link set interface up                 # Bring interface up
ip link set interface down               # Bring interface down
ifup interface                           # Enable interface
ifdown interface                         # Disable interface
systemctl restart networking             # Restart network service
```

---

## System Information

### Display System Information

System information helps you understand hardware and software configuration.

**Use Cases:** - Verify system specifications - Check OS information - Monitor system uptime - Document hardware configuration

```
uname -a                                # Display system information
cat /etc/os-release                      # Display OS information
lsb_release -a                           # Display Linux distribution info
hostnamectl                              # Display hostname and OS info
uptime                                   # Display system uptime
free -h                                  # Display memory usage (human-readable)
lscpu                                     # Display CPU information
cat /proc/cpuinfo                        # Display detailed CPU information
```

---

## Package Management

### Install, Update, Remove Packages

Package management allows you to install software and maintain system updates efficiently.

**Use Cases:** - Install new software - Update system packages - Remove unwanted software - Manage dependencies

```
# For Debian/Ubuntu systems:
apt update                               # Update package lists
apt upgrade                               # Upgrade installed packages
apt install package_name                  # Install package
apt remove package_name                   # Remove package
apt autoremove                            # Remove unused dependencies

# For Red Hat/CentOS systems:
yum update                                # Update system
yum install package_name                  # Install package
yum remove package_name                   # Remove package
yum groupinstall "group_name"             # Install package group
```

---

## Text Processing and Manipulation

### String and Text Operations

Text processing is fundamental for working with configuration files and log analysis.

**Use Cases:** - Extract specific columns from data - Process log files - Transform data formats - Manipulate text files

```
cut -d: -f1 filename                     # Extract columns from file
sort filename                             # Sort lines in file
uniq filename                             # Remove duplicate lines
sed 's/pattern/replacement/' file        # Replace text (stream editor)
awk '{print $1}' filename                 # Extract fields (awk)
tr 'a-z' 'A-Z' < filename                 # Translate characters
```



---

## Archive Operations

Compression and archiving help you save space and transport files efficiently.

**Use Cases:** - Create backups - Compress files for storage - Transport multiple files - Archive old data

```
tar -cvf archive.tar files      # Create tar archive
tar -xvf archive.tar            # Extract tar archive
tar -czvf archive.tar.gz files # Create gzip compressed archive
tar -xzvf archive.tar.gz       # Extract gzip archive
tar -cjvf archive.tar.bz2 files # Create bzip2 compressed archive
zip -r archive.zip files        # Create zip archive
unzip archive.zip               # Extract zip archive
gzip filename                   # Compress file
gunzip filename.gz              # Decompress gzip file
```

---

## System Administration

### Service Management

Managing services is essential for running applications and system daemons.

**Use Cases:** - Start/stop services - Enable services at boot - Check service status - Manage system daemons

```
systemctl start service        # Start service
systemctl stop service         # Stop service
systemctl restart service      # Restart service
systemctl enable service       # Enable at boot
systemctl disable service      # Disable at boot
systemctl status service       # Check service status
systemctl list-units --type=service # List all services
```

---

### System Logs

Logs provide crucial information for troubleshooting and monitoring system events.

**Use Cases:** - Troubleshoot system issues - Monitor service failures - Audit user activities - Debug application errors

```
journalctl                    # View system logs
journalctl -u service_name    # View logs for specific service
journalctl -n 50              # View last 50 log entries
journalctl -f                 # Follow logs in real-time
journalctl --since "2 hours ago" # View logs from last 2 hours
tail -f /var/log/syslog       # View system log (older systems)
tail -f /var/log/messages     # View messages log (Red Hat systems)
```

---

## Environment Variables and Shell

### Environment Variables

Environment variables control application behavior and system configuration.

**Use Cases:** - Set application settings - Configure PATH for executables - Pass credentials securely - Customize shell behavior

```
echo $VARIABLE          # Display variable value
export VARIABLE=value    # Set environment variable
printenv                 # Display all environment variables
env command              # Run command with custom environment
unset VARIABLE           # Remove variable
source ~/.bashrc         # Load shell configuration
```

---

## Aliases and Functions

Aliases and functions allow you to create shortcuts and automate command sequences.

**Use Cases:** - Create command shortcuts - Automate routine tasks - Simplify complex commands - Improve command-line efficiency

```
alias shortcut='long command' # Create command alias
unalias shortcut              # Remove alias
function_name() { commands; } # Define shell function
alias                         # List all aliases
```

---

## Advanced Operations

### File Transfer

Secure file transfer is essential for remote system administration and data synchronization.

**Use Cases:** - Copy files to/from remote servers - Synchronize directories - Backup data remotely - Deploy applications

```
scp local_file user@host:/path # Copy to remote server
scp user@host:/path/file .     # Copy from remote server
scp -r directory user@host:/path # Copy directory recursively
rsync -avz source/ destination/ # Synchronize directories
rsync -avz user@host:source/ dest/ # Remote synchronization
sftp user@host                  # Interactive SFTP session
```

---

### SSH and Remote Access

SSH enables secure remote administration and execution of commands on distant systems.

**Use Cases:** - Remote system administration - Execute commands on remote servers - Secure remote access - Automated deployments

```
ssh user@hostname          # Connect to remote server
ssh -p port user@hostname  # Connect using custom port
ssh -i key_file user@hostname # Connect using specific key
ssh user@host 'command'    # Execute remote command
ssh-keygen -t rsa           # Generate SSH key pair
ssh-copy-id user@host       # Copy SSH key to remote server
ssh-agent                  # SSH authentication agent
```

---

### Pipes and Redirection

Pipes and redirection are powerful tools for chaining commands and managing input/output.

**Use Cases:** - Chain multiple commands together - Redirect output to files - Process command output - Filter and transform data

```
command > file              # Redirect output to file (overwrite)
command >> file              # Redirect output to file (append)
command < file              # Use file as input
command1 | command2         # Pipe output of command1 to command2
command 2> errors.txt        # Redirect errors to file
command 2>&1                 # Redirect errors to output
command &> file              # Redirect both output and errors
```

---

## Regular Expressions

Regular expressions enable pattern matching and complex text manipulation.

**Use Cases:** - Validate input patterns - Extract specific data - Find and replace patterns - Parse log files

```
grep '^pattern' file           # Match at start of line
grep 'pattern$' file          # Match at end of line
grep 'pat.*rn' file           # Match with wildcards
grep '[0-9]' file              # Match any digit
grep '[a-z]' file              # Match any lowercase letter
grep -E '[a-z]{3}[0-9]+$' file # Extended regex pattern
```

---

## Cron Jobs and Scheduling

### Cron Job Management

Cron allows you to schedule tasks to run automatically at specified times.

**Use Cases:** - Automated backups - Scheduled maintenance - Report generation - System monitoring

```
crontab -e                     # Edit cron jobs
crontab -l                     # List cron jobs
crontab -r                     # Remove cron jobs
# Cron syntax: minute hour day month day_of_week command
0 2 * * * /path/to/backup.sh   # Run daily at 2 AM
*/5 * * * * /path/to/monitor.sh # Run every 5 minutes
```

---

## Tips and Tricks

### Command History

Using command history efficiently speeds up your workflow.

**Use Cases:** - Reuse previous commands - Search command history - Avoid retyping long commands

```
history                       # Display command history
history 20                    # Display last 20 commands
!command                      # Execute last command starting with 'command'
!!                            # Execute previous command
Ctrl+R                        # Search command history interactively
```

---

### Command Substitution

Command substitution allows you to use command output as input for other commands.

**Use Cases:** - Dynamic command execution - Process command output - Create complex command chains

```
$(command)                   # Modern syntax
`command`                    # Legacy syntax
command $(subcommand)        # Nested substitution
```