

# How to Read Manual Pages in Linux Using the man Command

Learn how to effectively use the man command in Linux to read manual pages, understand command syntax, options, and examples for better system navigation.

# Contents

01	Introduction	3
02	manCommand Syntax	3
03	Practical Examples of themanCommand	4
04	Advanced Usage Scenarios	9
05	Conclusion	12

# Introduction

The `man` (manual) command is an important tool that provides detailed information about Linux commands and programs. It provides comprehensive documentation, including descriptions, options, usage examples, and related information about commands or a specific program.

This article explains how to use the `man` command in Linux to access and read manual pages.

## `man` Command Syntax

Below is the basic `man` command syntax:

### CONSOLE

```
$ man [option] command_name
```

Within the above command, `command_name` sets the name of the command or program to view its manual page.

## `man` Command Options

### Option Description

- `-a` Displays all manual pages for a specific command.
- `-k` Search the short descriptions and manual page names for a specified keyword (equal to `apropos`).
- `-f` Display the manual page titles for the specified command (equal to `whatis`).
- `-M` Specifies a custom path to the manual pages.
- `-P` Specifies a pager command other than the default.
- `-u` Updates the database of manual pages.
- `-w` Displays the location of the manual page files.

# Practical Examples of the `man` Command

1. Read the manual page of a specific command.

CONSOLE

```
$ man ls
```

The above command displays the manual page for the [ls command](#), which includes detailed information about its usage.

Output:

```
LS(1) User Commands LS(1)
NAME
  ls - list directory contents
SYNOPSIS
  ls [OPTION]... [FILE]...
DESCRIPTION
  List information about the FILES (the current directory by default). Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.
  Mandatory arguments to long options are mandatory for short options too.
  -a, --all
    do not ignore entries starting with .
  -A, --almost-all
    do not list implied . and ..
  --author
    with -l, print the author of each file
  -b, --escape
    print C-style escapes for nongraphic characters
  --block-size=SIZE
    with -l, scale sizes by SIZE when printing them; e.g., '--block-size-M'; see SIZE format below
  -B, --ignore-backups
    do not list implied entries ending with ~
  -c
    with -lt: sort by, and show, ctime (time of last change of file status information); with -l: show ctime and sort by name; otherwise: sort by ctime, newest first
  -C
    list entries by columns
Manual page ls(1) line 1/221 15% (press h for help or q to quit)
```

2. Search for a command using a specific keyword in its description.

CONSOLE

```
$ man -k copy
```

The above command looks up a command's short description or manual page names with the keyword `copy` and lists all related commands.

Output:

```
vultr_user@vultr:~$ man -k copy
bcopy (3)          - copy byte sequence
btrfs-select-super (8) - overwrite primary superblock with a backup copy
copy_file_range (2) - Copy a range of data from one file to another
copysign (3)       - copy sign of a number
copysignf (3)      - copy sign of a number
copysignl (3)      - copy sign of a number
cp (1)            - copy files and directories
cpgr (8)          - copy with locking the given file to the password or group file
cpio (1)          - copy files to and from archives
cppw (8)          - copy with locking the given file to the password or group file
dd (1)           - convert and copy a file
debconf-copydb (1) - copy a debconf database
getunwind (2)     - copy the unwind data to caller's buffer
getutmp (3)       - copy utmp structure to utmpx, and vice versa
getutmpx (3)      - copy utmp structure to utmpx, and vice versa
git-checkout-index (1) - Copy files from the index to the working tree
install (1)       - copy files and set attributes
memccpy (3)       - copy memory area
memcpy (3)        - copy memory area
memmove (3)       - copy memory area
```

3. Display manual page titles.

CONSOLE

```
$ man -f ls
```

The above command displays the titles of manual pages for the `ls` command with a brief description for each.

Output:

```
vultr_user@vultr:~$ man -f ls
ls (1)          - list directory contents
vultr_user@vultr:~$ █
```

4. View all the manual pages for a specific command.

CONSOLE

```
$ man -a intro
```

The above command displays all manual pages for the `intro` command.

Output:

```
intro(1)                                General Commands Manual                                intro(1)
NAME
  intro - introduction to user commands

DESCRIPTION
  Section 1 of the manual describes user commands and tools, for example, file manipulation tools, shells, compilers, web browsers, file and image viewers and editors, and so on.

NOTES
  Linux is a flavor of UNIX, and as a first approximation all user commands under UNIX work precisely the same under Linux (and FreeBSD and lots of other UNIX-like systems).

  Under Linux, there are GUIs (graphical user interfaces), where you can point and click and drag, and hopefully get work done without first reading lots of documentation. The traditional UNIX environment is a CLI (command line interface), where you type commands to tell the computer what to do. That is faster and more powerful, but requires finding out what the commands are. Below a bare minimum, to get started.

Login
  In order to start working, you probably first have to open a session by giving your username and password. The program login(1) now starts a shell (command interpreter) for you. In case of a graphical login, you get a screen with menus or icons and a mouse click will start a shell in a window. See also xterm(1).

The shell
  One types commands to the shell, the command interpreter. It is not built-in, but is just a program and you can change your shell. Everybody has their own favorite one. The standard one is called sh. See also ash(1), bash(1), chsh(1), csh(1), dash(1), ksh(1), zsh(1).

A session might go like:

knuth login: aeb
Password: *****
$ date
Tue Aug 6 23:50:44 CEST 2002
$ cal
    August 2002
Su Mo Tu We Th Fr Sa
Manual page intro(1) line 1/144 24% (press h for help or q to quit)
```

5. Display the location of manual page files.

CONSOLE

```
$ man -w ls
```

The above command outputs the path to the `ls` manual page.

Output:

```
vultr_user@vultr:~$ man -w ls
/usr/share/man/man1/ls.1.gz
vultr_user@vultr:~$
```

6. Use `man` with a custom pager command.

## CONSOLE

```
$ man -P cat ls
```

The above command uses `cat` instead of the default pager to display the `ls` manual page.

Output:

```
vultr_user@vultr:~$ man -P cat ls
LS(1)                                     User Commands                               LS(1)
NAME
  ls - list directory contents
SYNOPSIS
  ls [OPTION]... [FILE]...
DESCRIPTION
  List information about the FILES (the current directory by default). Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

  Mandatory arguments to long options are mandatory for short options too.

  -a, --all
      do not ignore entries starting with .
  -A, --almost-all
      do not list implied . and ..
  --author
      with -l, print the author of each file
  -b, --escape
      print C-style escapes for nongraphic characters
  --block-size=SIZE
      with -l, scale sizes by SIZE when printing them; e.g., '--block-size-M'; see SIZE format below
  -B, --ignore-backups
      do not list implied entries ending with ~
  -c
      with -lt: sort by, and show, ctime (time of last change of file status information); with -l: show ctime and sort by name; otherwise: sort by ctime, newest first
  -C
      list entries by columns
```

7. Specify a manual page section to look up a specific command.

## CONSOLE

```
$ man 5 passwd
```

The above command displays section 5 of the `passwd` manual page that includes the file format information.

Output:

```
PASSWD(5) File Formats and Configuration PASSWD(5)
NAME
passwd - the password file
DESCRIPTION
/etc/passwd contains one line for each user account, with seven fields delimited by colons (":"). These fields are:
  • login name
  • optional encrypted password
  • numerical user ID
  • numerical group ID
  • user name or comment field
  • user home directory
  • optional user command interpreter

If the password field is a lower-case "x", then the encrypted password is actually stored in the shadow(5) file instead; there must be a corresponding line in the /etc/shadow file, or else the user account is invalid.

The encrypted password field may be empty, in which case no password is required to authenticate as the specified login name. However, some applications which read the /etc/passwd file may decide not to permit any access at all if the password field is blank.

A password field which starts with an exclamation mark means that the password is locked. The remaining characters on the line represent the password field before the password was locked.

Refer to crypt(3) for details on how this string is interpreted.

If the password field contains some string that is not a valid result of crypt(3), for instance ! or *, the user will not be able to use a unix password to log in (but the user may log in the system by other means).

The comment field, also known as the gecos field, is used by various system utilities, such as finger(1). The use of an ampersand here
Manual page passwd(5) line 1 (press h for help or q to quit)
```

## Navigating the Manual Pages

Use the following keyboard shortcuts to navigate and search through the manual page contents.

- **Spacebar:** Scroll down one screen at a time.
- **b:** Scroll up one screen at a time.
- **Enter:** Scroll down one line at a time.
- **k:** Scroll up one line at a time.
- **/pattern:** Search forward for a pattern.
- **?pattern:** Search backward for a pattern.
- **n:** Repeat the last search in the same direction.
- **N:** Repeat the last search in the opposite direction.
- **q:** Quit the manual page viewer.

## Understanding Manual Page Sections

Manual pages are divided into sections that include different types of information. Common sections include:

- **1:** User commands.

- **2:** System calls.
- **3:** Library functions.
- **4:** Special files (usually found in /dev).
- **5:** File formats and conventions.
- **6:** Games and screensavers.
- **7:** Miscellaneous.
- **8:** System administration commands.

## Advanced Usage Scenarios

1. View the manual pages for library functions.

CONSOLE

```
$ man 3 printf
```

The above command displays the manual page for the `printf` function, which is part of the C standard library.

Output:

```
printf(3)                                Library Functions Manual                                printf(3)
NAME
printf, fprintf, dprintf, sprintf, snprintf, vprintf, vfprintf, vdprintf, vsprintf, vsnprintf - formatted output conversion
LIBRARY
Standard C library (libc, -lc)
SYNOPSIS
#include <stdio.h>

int printf(const char *restrict format, ...);
int fprintf(FILE *restrict stream,
            const char *restrict format, ...);
int dprintf(int fd,
            const char *restrict format, ...);
int sprintf(char *restrict str,
            const char *restrict format, ...);
int snprintf(char str[restrict .size], size_t size,
            const char *restrict format, ...);

int vprintf(const char *restrict format, va_list ap);
int vfprintf(FILE *restrict stream,
            const char *restrict format, va_list ap);
int vdprintf(int fd,
            const char *restrict format, va_list ap);
int vsprintf(char *restrict str,
            const char *restrict format, va_list ap);
int vsnprintf(char str[restrict .size], size_t size,
            const char *restrict format, va_list ap);

Feature Test Macro Requirements for glibc (see feature_test_macros(7)):

snprintf(), vsnprintf():
    _XOPEN_SOURCE >= 500 || _ISOC99_SOURCE
    || /* glibc <= 2.19: */ _BSD_SOURCE

Manual page printf(3) line 1 (press h for help or q to quit)
```

## 2. Read the manual pages for system calls.

CONSOLE

```
$ man 2 open
```

The above command displays the manual page for the `open` system call and provides information on how files are opened in the kernel.

Output:

```
open(2)                               System Calls Manual                               open(2)
NAME
    open, openat, creat - open and possibly create a file
LIBRARY
    Standard C library (libc, -lc)
SYNOPSIS
    #include <fcntl.h>

    int open(const char *pathname, int flags, ...
              /* mode_t mode */);

    int creat(const char *pathname, mode_t mode);

    int openat(int dirfd, const char *pathname, int flags, ...
              /* mode_t mode */);

    /* Documented separately, in openat2(2): */
    int openat2(int dirfd, const char *pathname,
               const struct open_how *how, size_t size);

Feature Test Macro Requirements for glibc (see feature_test_macros(7)):

    openat():
        Since glibc 2.10:
            _POSIX_C_SOURCE >= 200809L
        Before glibc 2.10:
            _ATFILE_SOURCE
DESCRIPTION
    The open() system call opens the file specified by pathname. If the specified file does not exist, it may optionally (if O_CREAT is specified in flags) be created by open().

    The return value of open() is a file descriptor, a small, nonnegative integer that is an index to an entry in the process's table of open file descriptors. The file descriptor is used in subsequent system calls (read(2), write(2), lseek(2), fcntl(2), etc.) to refer
Manual page open(2) line 1 (press h for help or q to quit)
```

## 3. Use `man` in scripts to display specific documentation information about commands.

- Create a new script file using a text editor such as Nano

CONSOLE

```
$ nano my_script1.sh
```

- Add the following contents to the file.

```
BASH
```

```
#!/bin/bash  
echo "Displaying the manual page for the 'ls' command:"  
man ls
```

- Enable execute permissions on the file.

```
CONSOLE
```

```
$ chmod +x my_script1.sh
```

- Run the script

```
CONSOLE
```

```
$ ./my_script1.sh
```

The above script displays the manual page for the `ls` command when executed.

Output:

```
vultr_user@vultr:~$ nano my_script1.sh  
vultr_user@vultr:~$ chmod +x my_script1.sh  
vultr_user@vultr:~$ ./my_script1.sh  
Displaying the manual page for the 'ls' command:  
vultr_user@vultr:~$ █
```

4. Combine the `man` command with `grep` to search pages.

```
CONSOLE
```

```
$ man ls | grep "list"
```

The above command pipes the manual page output for the ls command to [grep](#) and searches for lines that contain the string list.

Output:

```
vultr_user@vultr:~$ man ls | grep "list"
ls - list directory contents
    do not list implied . and ..
    do not list implied entries ending with ~
-C    list entries by columns
    list directories themselves, not their contents
-f    list all entries in directory order
-g    like -l, but do not list owner
    in a long listing, don't print group names
    follow symbolic links listed on the command line
    do not list implied entries matching shell PATTERN (overridden by -a or -A)
    do not list implied entries matching shell PATTERN
-l    use a long listing format
-m    fill width with a comma separated list of entries
    like -l, but list numeric user and group IDs
-o    like -l, but do not list group information
    list subdirectories recursively
-U    do not sort; list entries in directory order
-x    list entries by lines instead of by columns
-1    list one file per line
vultr_user@vultr:~$
```

5. Update the manual pages database.

CONSOLE

```
$ sudo mandb
```

The above command updates the database of all manual pages to ensure new or removed pages are available in the search results.

## Conclusion

You have used the `man` command to search for documentation and command usage pages. For more information about the `man` command, run `man man` to view the command's manual page.



VULTR

