



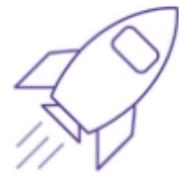
Welcome to Working with the File System.

## What you will learn

### At the core of the lesson

You will learn how to:

- Navigate files and directories in Linux
- Explain basic commands for managing files and directories
- Compare absolute and relative paths



In this lesson, you will learn how to:

- Navigate files and directories in Linux
- Explain basic commands for managing files and directories
- Compare absolute and relative paths

## Navigating files and directories

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing the Linux file system.

## Files

- Everything in Linux is a file.
- Commands, hardware, and directories are represented as files.
- Most system configurations are in files.



4

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws re/start

Files allow for transparency: drives, processes, and other elements are all represented as files. They can be browsed and accessed for information (for example, `ls /proc` gives you access to processes).

Files allow for interoperability: the same tools can be used for different types of files and can be combined (for example, `ls -l | grep .txt`).

## Files

```
[ec2-user@myLinux ~]$ ls -l  
total 12  
drwxr-xr-x 2 ec2-user ec2-user 6 May 28 08:39 Desktop  
-rw-rw-r-- 1 ec2-user ec2-user 21 Jun 1 08:12 myfile
```

```
[ec2-user@myLinux bin]$ ls -l ls  
-rwxr-xr-x 1 root root 109288 Jan 23 2020 ls  
[ec2-user@myLinux bin]$ 
```

```
[ec2-user@myLinux ~]$ ls | grep .txt  
myfile.txt  
[ec2-user@myLinux ~]$ ls >> myFilesList.txt  
[ec2-user@myLinux ~]$ more myFilesList.txt  
Desktop  
myfile  
myFile  
myFilesList.txt  
myfile.txt  
[ec2-user@myLinux ~]$ 
```

5

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.



- In the first screenshot, both the directory and the text file are considered as files. The directory is a special kind of file, hence the d and the blue color.
- In the second screenshot, you see that you can list the ls command because it is also a file.
- The third screenshot shows how you can operate between commands.

## Linux file names and extensions

- Case sensitive
- Must be unique within the directory
- Should not contain / or spaces
- Extensions are optional and not necessarily mapped to applications

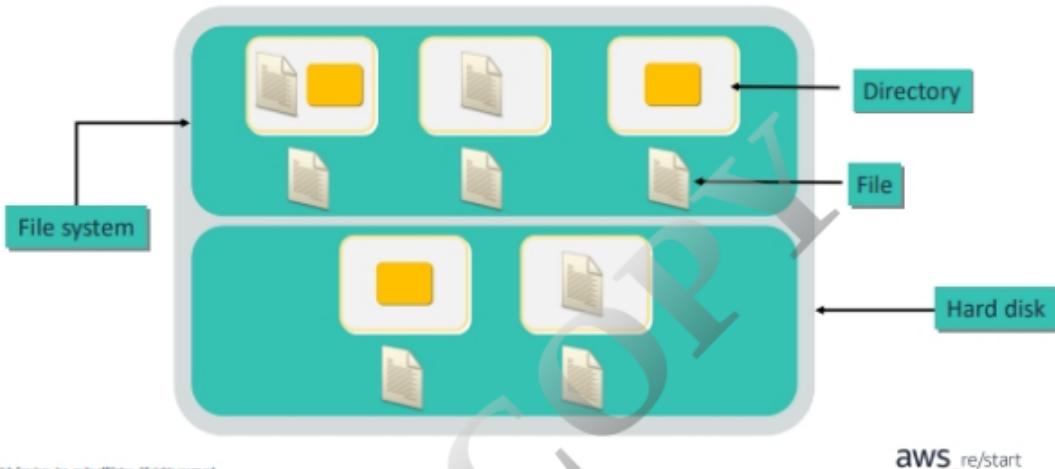
```
[ec2-user@myLinux ~]$ ls -l m*
-rw-rw-r-- 1 ec2-user ec2-user 21 Jun  1 08:12 myfile
-rw-rw-r-- 1 ec2-user ec2-user 15 Jun  1 08:12 myFile
-rw-rw-r-- 1 ec2-user ec2-user 49 Jun  1 11:56 myFilesList.txt
-rw-rw-r-- 1 ec2-user ec2-user 25 Jun  1 08:12 myfile.txt
[ec2-user@myLinux ~]$
```

The screen capture shows three text files: the name is case sensitive and the extension is optional.

You are strongly advised to have consistent extensions. For instance, a .jpg image could be named image.txt even though this extension would not make sense. A user might think that this file is a description file and try to open it with a text editor instead of an image viewer. A better option is to name the image.jpeg or image.jpg.

## File systems

File systems: a way of naming, retrieving, and organizing data on the storage disk



The file system organizes how files are stored on the hard drive. A file is located inside a directory.

## File system hierarchy standard (FHS)

- Standard location and function of directories across Linux distributions
  - Allows software to be compatible with various distributions
  - Allows administrators to predict where certain types of files will be found
- Examples:
  - /etc typically contains configuration files
  - /var/log typically contains log files

```
[ec2-user@myLinux ~]$ ls /
bin  dev  home  lib64  media  opt  root  sbin  sys  usr
boot  etc  lib  local  mnt  proc  run  srv  tmp  var
[ec2-user@myLinux ~]$
```

Most Linux distributions use this standard, but some distributions might differ slightly or intentionally use a different file system.

## Some key FHS directories

Directory	Function
/	Root of the file system
/boot	Boot files and kernel
/dev	Devices
/etc	Configuration files
/home	Standard users' home directories
/media	Removable media
/mnt	Network drives
/root	Root user home directory
/var	Log files, print spool, network services

The FHS has many other directories. This list shows a select few.

## Commands for managing files and directories

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The following commands are important. They are used to manage files and directories.

## Understanding command syntax with the `ls` command

- The `ls` command displays a list of files in a directory.
- Different colors represent different types of files.
- `ls` lists the content of the current directory.
- `ls dir` lists the content of the `dir` directory.

```
[userA@server00 ~]$ ls
backups  Documents  Music  Public  Videos
Desktop  Downloads  Pictures  Templates
[userA@server00 ~]$ ls /var
account  crash  empty  kerberos  lock  nis
adm      db      games  lib      log   opt
cache    develweb gopher local  mail  preserve
[userA@server00 ~]$ █
```

Colors are not defined according to a standard. They depend on the configuration of the shell that you are using.

You can list several multiple directories: `ls directory1 directory2`

## Useful options for the `ls` command

Option	Description
<code>-l</code>	Long format (shows permissions)
<code>-h</code>	File sizes reported in a human-friendly format
<code>-a</code>	Shows all files, including hidden files
<code>-R</code>	Lists subdirectories
<code>--sort=extension or -X</code>	Sorts alphabetically by file extension
<code>--sort=size or -S</code>	Sorts by file size
<code>--sort=time or -t</code>	Sorts by modification time
<code>--sort=version or -v</code>	Sorts by version number

By default, the `ls` command uses the natural sort order. To get results in the reverse order, add the `-r` option.

You can combine options: `ls -al` displays hidden files and file details.

## Examples of the `ls` command

```
[ec2-user@myLinux ~]$ ls -a
.
..           .bashrc   .ICEauthority    myfile.txt  .Xauthority
..           .cache    .local          .ssh
.bash_history .config   myfile         .swp
.bash_logout  .dbus     myFile        .viminfo
.bash_profile .Desktop  myFilesList.txt .vnc
[ec2-user@myLinux ~]$ ls -al
total 60
drwx----- 9 ec2-user ec2-user   315 Jun  1 11:56 .
drwxr-xr-x  5 root    root      48 May 20 19:30 ..
-rw-----  1 ec2-user ec2-user 3185 Jun  1 10:54 .bash_history
-rw-r--r--  1 ec2-user ec2-user  18 Jul 15 2020 .bash_logout
-rw-r--r--  1 ec2-user ec2-user 193 Jul 15 2020 .bash_profile
-rw-r--r--  1 ec2-user ec2-user 231 Jul 15 2020 .bashrc
drwx----- 3 ec2-user root       25 May 28 08:36 .cache
drwxrwxr-x  7 ec2-user ec2-user  74 May 28 09:15 .config
drwx----- 3 ec2-user ec2-user  25 May 28 08:39 .dbus
drwxr-xr-x  2 ec2-user ec2-user  6 May 28 08:39 Desktop
-rw-----  1 ec2-user ec2-user 624 Jun  1 10:55 .ICEauthority
drwx----- 3 ec2-user ec2-user 19 May 28 08:39 .local
-rw-rw-r--  1 ec2-user ec2-user 21 Jun  1 08:12 myfile
-rw-rw-r--  1 ec2-user ec2-user 15 Jun  1 08:12 myFile
-rw-rw-r--  1 ec2-user ec2-user 49 Jun  1 11:56 myFilesList.txt
```

13 © 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.



- **`ls -l`**: Lists the contents of the current directory with details. It does not display the hidden files.
- **`ls -a`**: Displays the hidden files.
- **`ls -al`**: Displays the hidden files and the file's details (not all of the list is displayed on the screenshot because it would take too much space).

## Demonstration: Exploring files and directories



In this demonstration, the instructor will show how to use the `ls` command with various options to view information about files on a Linux system.

```
[labuser@centos ~]$ ls
asmlib companyA logins work
[labuser@centos ~]$ ls -l
total 12
drwxr-x--- 3 584731 labuser 4096 Jan 5 22:37 asmlib
drwxr-x--- 1 labuser 1001 4096 Aug 26 2019 companyA
lrwxrwxrwx 1 labuser labuser 33 Mar 30 10:02 logins -> companyA/SharedFolders/logins.csv
drwxrwx--- 4 labuser 48 4096 Mar 5 08:19 work
[labuser@centos ~]$ ls -a
.
.. .asmlib .bash_logout .bashrc logins .vlog
.. .bash_history .bash_profile companyA .ssh work
[labuser@centos ~]$ ls -al
total 86
drwxrwx--- 1 labuser 48 4096 Mar 30 10:02 .
drwxr-x--- 1 root 4096 Aug 26 2019 ..
drwxr-x--- 3 584731 labuser 4096 Jan 5 22:37 asmlib
-rw-r--r-- 1 labuser 1001 1248 Aug 26 2019 .bash_history
-rw-r--r-- 1 labuser 48 18 Apr 11 2018 .bash_logout
-rw-r--r-- 1 labuser 1001 199 Aug 26 2019 .bash_profile
-rw-r--r-- 1 labuser 48 246 Aug 26 2019 .bashrc
drwxr-x--- 1 labuser 1001 4096 Aug 26 2019 companyA
lrwxrwxrwx 1 labuser labuser 33 Mar 30 10:02 logins -> companyA/SharedFolders/logins.csv
drwxrwx--- 1 labuser 48 4096 Mar 30 09:50 .ssh
-rw-r--r-- 1 labuser 1001 171 Aug 26 2019 .vlog
drwxrwx--- 4 labuser 48 4096 Mar 5 08:19 work
[labuser@centos ~]$
```

Follow along as the instructor demonstrates the use of the `ls` command.

## The more command

- View file contents that don't fit on one screen
- Loads entire contents of files before displaying results
- Can only scroll down
- `more` can be used in conjunction with other commands:  
`cat file.txt | more`

```
# $OpenBSD: sshd_config,v 1.100 2016/08/15 12:32:04 naddy Exp $
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
--More-- (19%)
```

Usage:

`more [-options] [-num] [+pattern] [+linenum] [file_name]`

- Options:
  - `-d`: Displays information about how to navigate at the bottom of the screen
  - `-f`: Prevents line wrap
  - `-p`: Clears the screen before displaying the content
  - `-S`: Squeezes multiple blank lines into one line
- `num`: Number of lines to display
- `/pattern`: String to find in the file
- `linenum`: Line number where to start displaying the content
- `file_name`: Name of the file to display the content of

## The **less** command

- View file contents that don't fit on one screen
- Can scroll up and down through content
- Loads faster than more because **less** doesn't load every page before it displays results
- Mostly used for large files

```
# $OpenBSD: sshd_config,v 1.100 2016/08/15 12:32:04 naddy Exp $
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
sshd_config
```

Usage:

**less [OPTIONS] filename**

- Use Q to quit
- Options:
  - -N: Shows line numbers
  - -X: Displays the content after the last command and does not clear screen when exiting
  - +F: Watches for file content changes

## The head command

- Displays the first 10 lines of a file by default
- With the **-n** option, you can specify a number of lines to display
- Can display multiple files

```
[ec2-user@myLinux ~]$ head myfile myfile.txt  
==> myfile <==  
this is another file  
  
==> myfile.txt <==  
this is yet another file  
[ec2-user@myLinux ~]$
```

```
[ec2-user@myLinux ~]$ sudo head -n 5 /etc/passwd  
root:x:0:0:root:/root:/bin/bash  
bin:x:1:1:bin:/bin:/sbin/nologin  
daemon:x:2:2:daemon:/sbin:/sbin/nologin  
adm:x:3:4:adm:/var/adm:/sbin/nologin  
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin  
[ec2-user@myLinux ~]$
```

Usage:

**head [OPTIONS] filename(s)**

- Options:
  - **-n <number>**: First n lines to display
  - **-c <number>**: First C bytes to display

## The `tail` command

- Displays the last 10 lines of a file by default
- With the `-n` option, you can specify a number of lines to display

```
[ec2-user@myLinux ~]$ sudo tail -5 /var/log/boot.log
Starting Wait for Plymouth Boot Screen to Quit...
[ OK ] Started Job spooling tools.
Starting Job spooling tools...
[ OK ] Started System Logging Service.
[ OK ] Started Finds and configures elastic network interfaces.
[ec2-user@myLinux ~]$ █
```

Usage:

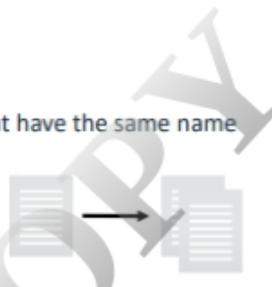
`tail [OPTIONS] filename(s)`

- Options:
  - `-n <number>`: Last `n` lines to display
  - `-c <number>`: Last `C` bytes to display
  - `-f`: Monitor for file changes

The `tail -f` command is useful for log files that are regularly updated and where the most recent entries are at the bottom of the file.

## The cp command

- The **cp** command copies files and directories
- By default, the **cp** command overwrites existing files that have the same name
- Example: **cp <file-name> <destination>**
- For more options, refer to the next slide



```
[ec2-user@myLinux ~]$ ls folderA/  
srcfile  
[ec2-user@myLinux ~]$ ls folderB/  
[ec2-user@myLinux ~]$ cp folderA/srcfile folderB/destfile  
[ec2-user@myLinux ~]$ cp folderA/srcfile folderB/  
[ec2-user@myLinux ~]$ ls folderB/  
destfile srcfile  
[ec2-user@myLinux ~]$ █
```

19 © 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws re/start

### Usage:

**cp folderA/srcfile folderB/destfile**

- Copies the srcfile that is located in folderA to folderB and names it destfile.

**cp folderA/srcfile folderB/**

- Copies the srcfile that is located in folderA to folderB (both files have the same name).

**cp folderA/srcfile folderB/ folderC/destfile**

- Copies the srcfile that is located in folderA to folderB and to folderC with the name destfile.

## The cp command: Additional options

Option	Description
cp -a	Archive files
cp -f	Force copy by overwriting the destination file if needed
cp -i	Interactive – Ask before overwrite
cp -l	Link files instead of copy
cp -L	Follow symbolic links
cp -n	No file overwrite
cp -R	Recursive copy (including hidden files)
cp -u	Update – Copy when source is newer than destination
cp -v	Verbose – Print informative messages

The slide shows additional options for the cp command that you might use.

## The rm command

- Deletes a file
- There is no undelete or Recycle Bin

```
[userA@server00 Documents]$ ls  
file1.txt  
[userA@server00 Documents]$ rm file1.txt  
[userA@server00 Documents]$ ls  
[userA@server00 Documents]$ █
```



Usage:

`rm [OPTIONS] filename(s)`

- Options:
  - `-d`: Removes a directory; the directory must be empty: `rm -d dir`
  - `-r`: Allows you to remove a non-empty directory: `rm -r dir`
  - `-f`: Never prompt user (useful when deleting a directory with many files)
  - `-i`: Prompts the user for confirmation for each file
  - `-v`: Display the names of deleted files
- If a file is write protected, a prompt will ask the user for confirmation.
- Several files can be removed at once.
- If you want to remove a complete directory, use the `-r` and `-f` option: `rm -rf dir`
- You can use a regular expression: `rm *.png` removes all files that end with .png.

## The `mkdir` command

Creates new directories



```
[userA@server00 ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos  
[userA@server00 ~]$ mkdir NewProject  
[userA@server00 ~]$ ls  
Desktop Downloads NewProject Public Videos  
Documents Music Pictures Templates  
[userA@server00 ~]$
```

Usage:

`mkdir [OPTIONS] filename(s)`

- Options:
  - `-m <mask>`: Sets a permission to the directory
  - `-p`: Creates parent directory
- You can create several directories with one command: `mkdir dir1 dir2 dir3`
  - `mkdir -m 700 dir1`: Creates the `dir1` directory with the mask 700 for permissions
  - `mkdir -p /home/user/dir1/dir2`: If `dir1` does not exist, the creation will fail without the `-p` option

## The mv command

- Moves a file from one directory to another
- Renames a file if the source and destination are the same

```
[userA@server00 Documents]$ ls  
file1.txt  file1.txt.backup  
[userA@server00 Documents]$ mv file1.txt.backup ~/backups/  
[userA@server00 Documents]$ ls ~/backups/  
file1.txt.backup  
[userA@server00 Documents]$ █
```

Note: By default, the mv command overwrites existing files that have the same name.

Usage:

**mv [OPTIONS] source destination**

- Options:
  - **-i**: Prompts before overwriting a file
  - **-f**: Avoid being prompted
  - **-n**: Do not overwrite existing files
  - **-v**: Verbose option, prints the name of files that are moved or renamed
  - **mv file1 dir1**: Moves file1 to dir1
  - **mv dir1 dir2**: Moves dir1 to dir2
  - **mv file1 file2 dir1 dir2**: Moves file1, file2, and dir1 to dir2; there can only be one target directory here, dir2
  - **mv file1 dir1/file2**: Moves file1 to dir1 and renames it file2
  - **mv file1 file2**: Renames file1 as file2
- You can use a regular expression to move files of the same type:
  - **mv \*.png dir1**: Moves all files with extension .png into dir1

## The rmdir command

- Deletes existing empty directories: `rmdir <DirectoryName>`
- If a directory isn't empty, use `rm -r <DirectoryName>`
  - This command removes a directory and all of its contents.



```
[userA@server00 ~]$ ls  
Desktop  Downloads  NewProject  Public  Videos  
Documents  Music  Pictures  Templates  
[userA@server00 ~]$ rmdir NewProject  
[userA@server00 ~]$ ls  
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos  
[userA@server00 ~]$ █
```

`rmdir` is equivalent to `rm -d`

## The **pwd** command

- Output of the **pwd** command: Absolute path to your current location in the file system.
- Essential for navigation: You must know where you are in the file system to move to other directories.

```
[userA@server00 projects]$ pwd  
/home/userA/Documents/projects  
[userA@server00 projects]$ █
```

Use the **pwd** command to know where you are in the **file** directory structure.

## Demonstration: Managing files and directories



26

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

In this demonstration, the instructor will show you how to:

- Create, move, copy, and delete files
- Create and delete directories

```
[labsuser@centos companyA] $ ls
FolderListing.csv  IA      Roster.csv  securecopy.txt  Shipping
HR                  Management  Sales      SharedFolders
[labuser@centos companyA] $ touch employeeList.csv
[labuser@centos companyA] $ mkdir Employees
[labuser@centos companyA] $ mv employeeList.csv  Employees/
[labuser@centos companyA] $ rm -rf IA
[labuser@centos companyA] $ ls
Employees  HR      Roster.csv  securecopy.txt  Shipping
FolderListing.csv  Management  Sales      SharedFolders
[labuser@centos companyA] $ ls Employees/
employeeList.csv
[labuser@centos companyA] $
```

aws re/start

Follow along as the instructor creates, moves, copies, and deletes files and creates and deletes directories.

## Absolute versus relative paths

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

You must know the difference between absolute and relative paths.

## Paths

- Paths define directories to be traversed to get to a particular resource.
- In a graphical user interface (GUI), you navigate by opening directories.
- In a command line interface (CLI), you also navigate through directories, but you specify them by name.

```
[userA@server00 projects]$ pwd  
/home/userA/Documents/projects  
[userA@server00 projects]$
```

You must know how to navigate directories by both a GUI and a CLI.

## Types of paths

- Absolute path (complete path to the resource from the root of the file system):
  - The absolute path to access the **projects** directory from the root of the file system
  - Example: `/home/userA/Documents/projects`
- Relative path (path to the resource from the current directory):
  - The relative path to access the **projects** directory from the **Documents** directory
  - Example: `Documents/projects`

Suppose the command `pwd` tells you that you are in the folder `/home/ec2-user`.

`cd /home/userA/Documents/projects` will navigate to the `/home/userA/Documents/projects` folder.

`cd Documents/projects` will navigate to the `/home/ec2-user/Documents/projects` folder (currentfolder/Document/projects, where current folder is `/home/ec2-user`).

## The **cd** command

- The change directory or **cd** command is used to move from one directory to another
- Using the **cd** command with the absolute path:

```
[userA@server00 etc]$ cd /home/userA/Documents/projects/  
[userA@server00 projects]$
```

- Using the **cd** command with the relative path:

```
[userA@server00 ~]$ pwd  
/home/userA  
[userA@server00 ~]$ cd Documents/projects/
```

Tip: Use **.. /** to go up a single directory at a time.

For example, if you are in the **/home/userA** folder, **cd .. /** will navigate to **/home**.

## Demonstration: Absolute and relative paths



31

© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

In this demonstration, the instructor will show you the difference between absolute and relative paths.

```
[labsuser@centos companyA]$ pwd  
/home/labsuser/companyA  
[labsuser@centos companyA]$ cd ..  
[labsuser@centos ~]$ pwd  
/home/labsuser  
[labsuser@centos ~]$ cd companyA/SharedFolders/  
[labsuser@centos SharedFolders]$ pwd  
/home/labsuser/companyA/SharedFolders  
[labsuser@centos SharedFolders]$
```

aws re/start

Follow along as the instructor demonstrates the difference between absolute and relative paths.

## Checkpoint questions



What is the difference between an absolute path and a relative path?

When would you use the **less** command instead of the **more** command? Why?

### Answers:

1. The absolute path shows the entire folder structure to the resource that is being used. The absolute path to `my_file.txt` in the `Documents` directory would be something like `/Users/user_name/Documents/Labwork/my_file.txt`. The relative path shows only from the current directory to the file that is being used. From the previous example, within the `user_name` directory, the relative path to the file is `/Documents/Labwork/my_file.txt`.
2. You use the **less** command if you want to scroll backward through a file. With the **more** command, you can only scroll forward through a file.

## Key takeaways



33 © 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

- Everything in Linux is a file.
- The Linux file system:
  - Is case sensitive
  - Has the key-like directories:
    - » /
    - » /home
    - » /mnt
- Linux contains many commands to help work with files.  
Some are:
  - `ls` – Lists the contents of a directory
  - `cat` – Shows the contents of a file
  - `cp` – Copies a file
  - `rm` – Removes a file
  - `mkdir` – Creates a directory
- Linux has both absolute and relative directory paths.

aws/re/start

Some key takeaways from this lesson include the following:

- Everything in Linux is a file.
- The Linux file system is case sensitive and has key-like directories.
- Linux contains many commands to help work with files.
- Linux has both absolute and relative directory paths.

Thank you

© 2021 Amazon Web Services, Inc. or its affiliates. All rights reserved. This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, reverse engineering, or selling is prohibited. Corrections, feedback, or other questions? Contact us at . All trademarks are the property of their owners.

