

[301] The Terminal

Tyler Caraza-Harter

```
settern — -bash — 80x24  
Last login: Wed Feb 24 10:56:19 on ttys003  
new-host-6:~ settern$
```

```
E:\Windows\system32\cmd.exe  
Microsoft Windows [Version 6.1.7600]  
Copyright (c) 2009 Microsoft Corporation. All rights reserved.  
E:\Users\ACK>shutdown /?  
Usage: shutdown /l /i /s /r /g /a /p /h /e  
/m \\computer [/t xxx] [/d [p]:xx:yy [/c "comment"]]  
  
No args Display help. This is the same as typing /?.  
/? Display help. This is the same as not typing /?.  
/i Display the graphical user interface (GUI). This must be the first option.  
/l Log off. This cannot be used with /m or /d or /s.  
/s Shutdown the computer.  
/r Shutdown and restart the computer.  
/g Shutdown and restart the computer. After the rebooted, restart any registered application.  
/a Abort a system shutdown. This can only be used during the time-out period.  
/p Turn off the local computer with no time-out. Can be used with /d and /f options.  
/h Hibernate the local computer. Can be used with the /f option.  
/e Document the reason for an unexpected shutdown.  
/m \\computer Specify the target computer.  
/t xxx Set the time-out period before shutdown to xxx. The valid range is 0-315360000 (10 years). If the timeout period is greater than 0, the implied.  
/c "comment" Comment on the reason for the restart or shutdown. Maximum of 512 characters allowed.  
/f Force running applications to close without warning. The /f parameter is implied when a value greater than 0 is specified for the /t parameter.  
/d [p]:xx:yy Provide the reason for the restart or shutdown. p indicates that the restart or shutdown is planned. u indicates that the reason is user defined. If neither p nor u is specified the restart or shutdown is unplanned.  
xx is the major reason number (positive integer less than 256). yy is the minor reason number (positive integer less than 65536).
```

```
Windows PowerShell  
Copyright (c) 2009 Microsoft Corporation. All rights reserved.  
PS C:\Users\NullByte> get-help  
TOPIC  
Get-Help  
SHORT DESCRIPTION  
Displays help about Windows PowerShell cmdlets and concepts.  
LONG DESCRIPTION  
SYNTAX  
get-help <<CmdletName> ! <TopicName>  
help <<CmdletName> ! <TopicName>  
<<CmdletName> -?  
"Get-help" and "-?" display help on one page.  
"Help" displays help on multiple pages.  
Examples:  
get-help get-process : Displays help about the get-process cmdlet.  
help where-object : Displays help about the where-object cmdlet.  
help about-foreach : Displays help about the about-foreach topic.  
set-service -? : Displays help about the set-service cmdlet.  
You can use wildcard characters in the help topic.  
If multiple help topics match, PowerShell displays the first match.  
Examples:  
get-help * : Displays all help topics.  
get-help get-* : Displays topics that begin with "get-".  
help where-object* : Displays topics with "where-object" in the name.  
get-help about* : Displays all conceptual help topics.  
For information about wildcards, type:  
get-help about_wildcard
```

```
hello world  
stuart@stuart-desktop:~$
```

Today's Topics

Terminal Emulators and Shells

- Terminal history
- Shells
- Running programs from a shell

Navigation

Running Programs and Commands

Demos

History: the Original Terminals



Mainframe
(powerful computer)

History: the Original Terminals



**Mainframe
(powerful computer)**

How to share it?

History: the Original Terminals

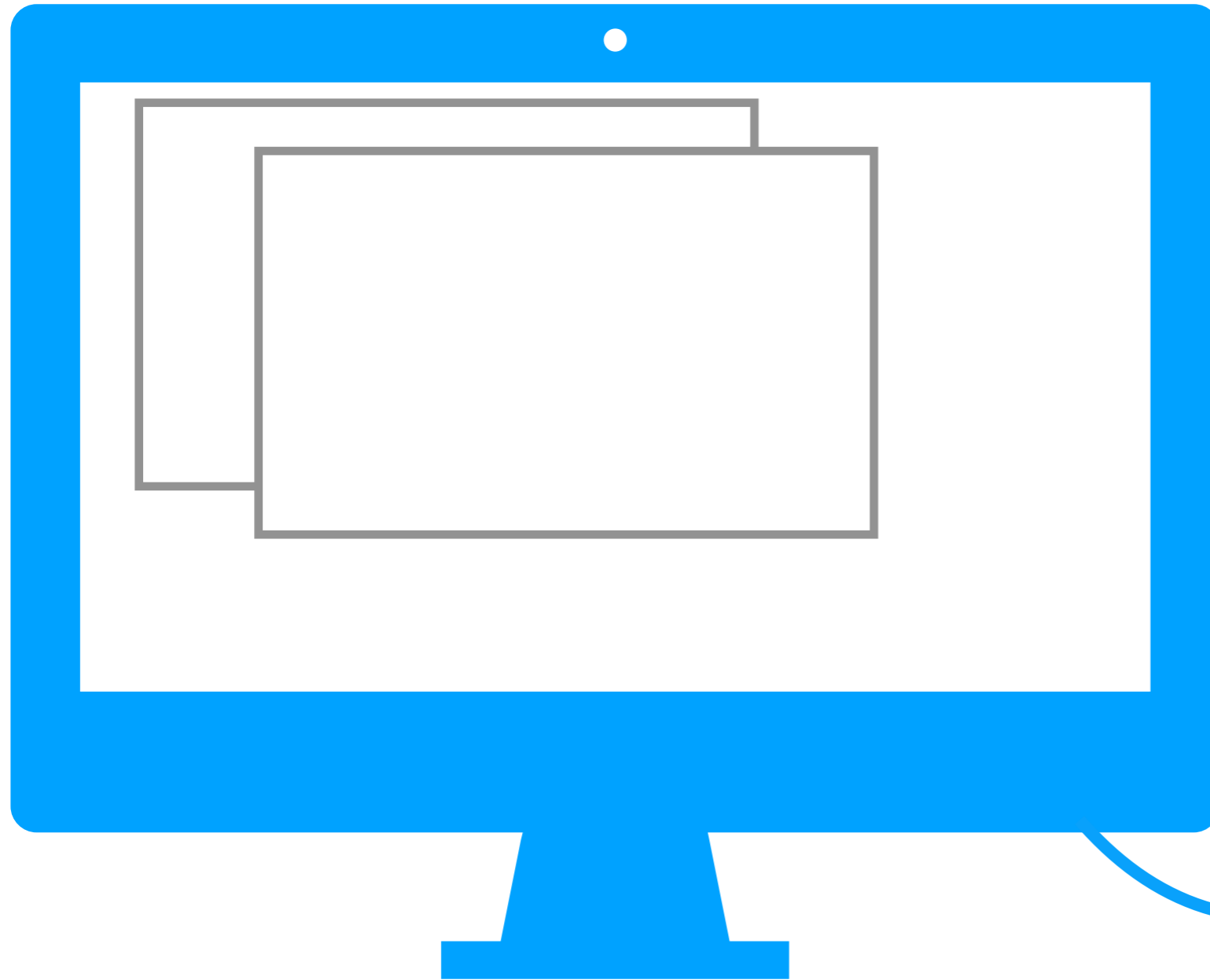


**Mainframe
(powerful computer)**

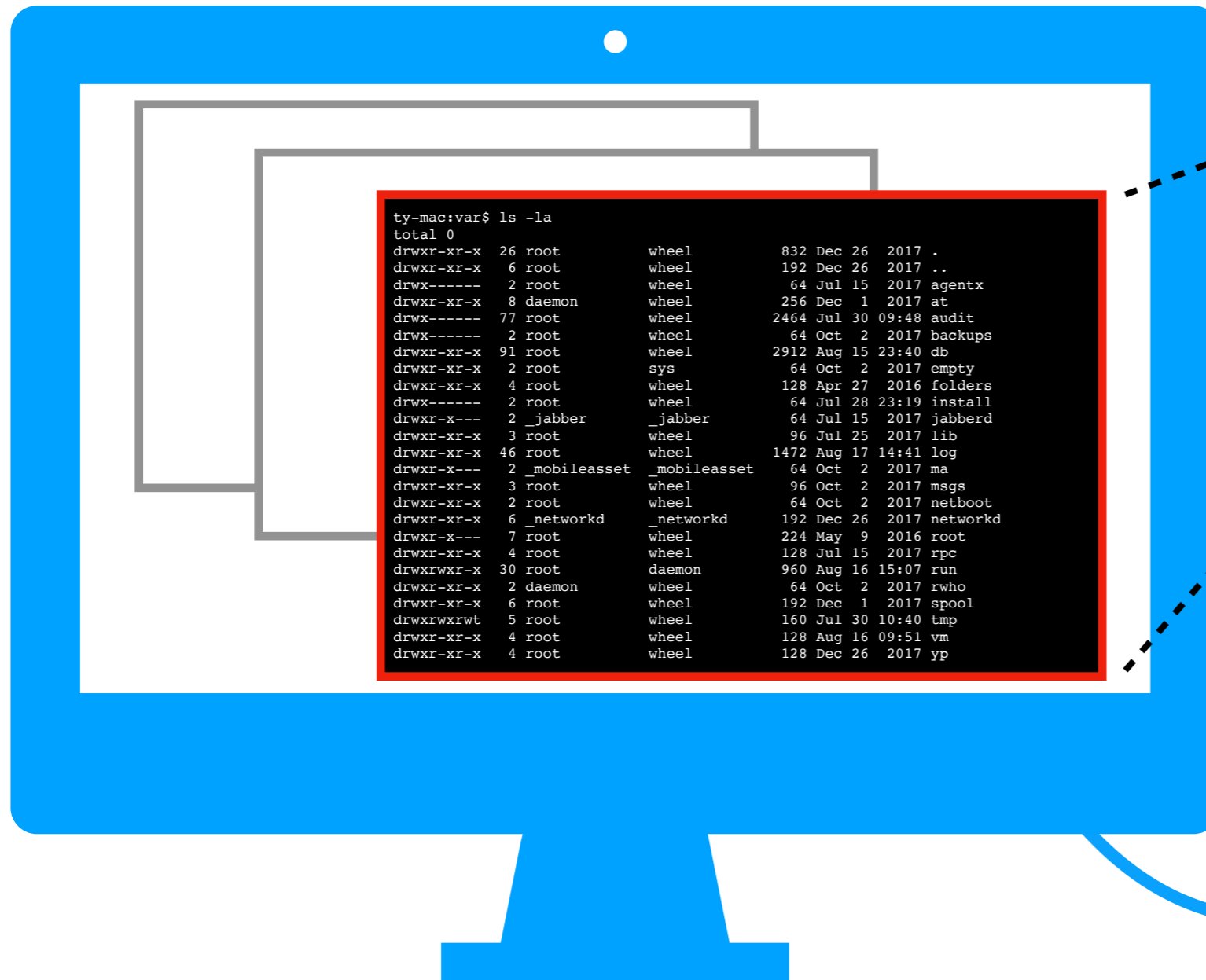


**dumb terminals
(text based)**

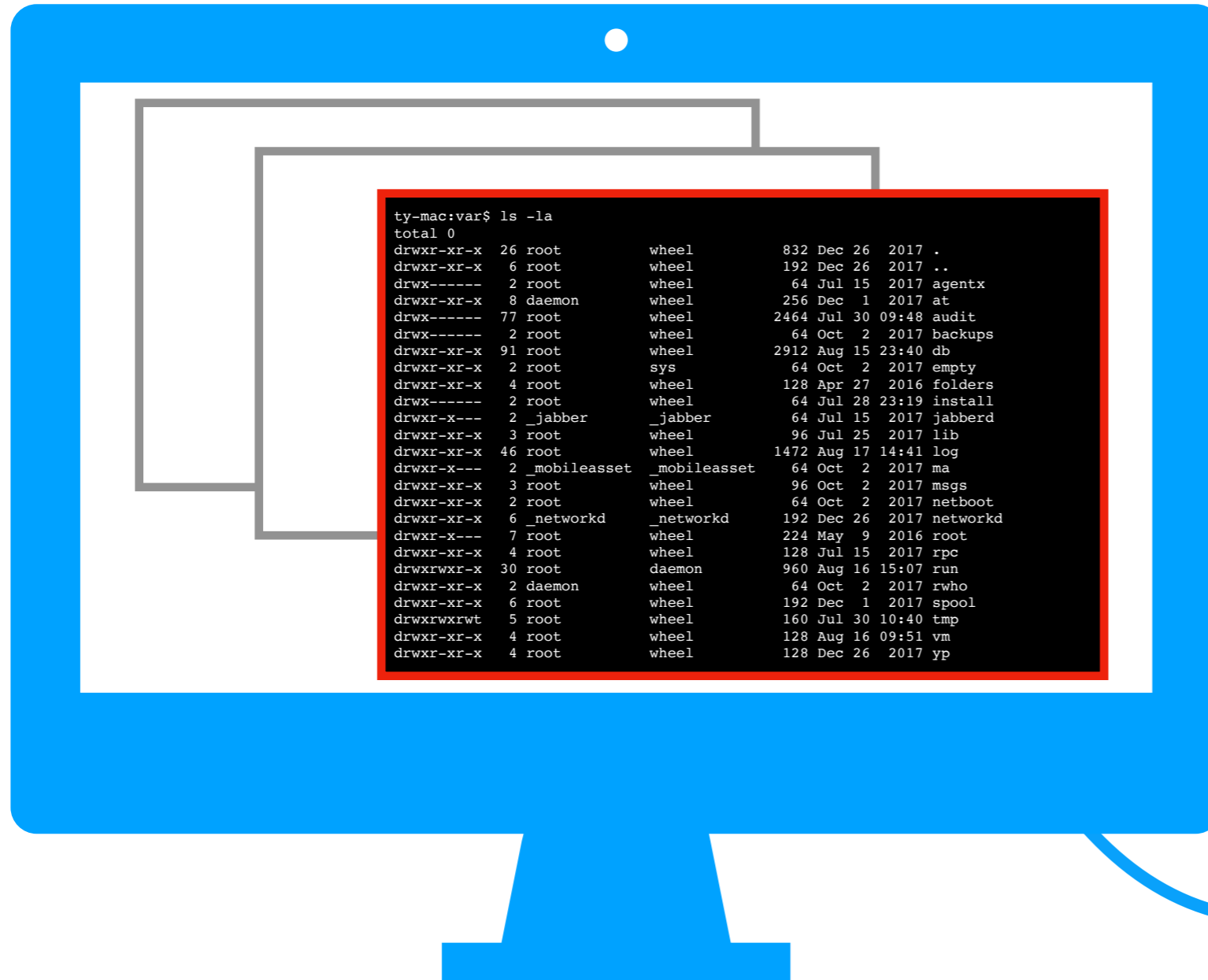
Terminal Emulators



Terminal Emulators

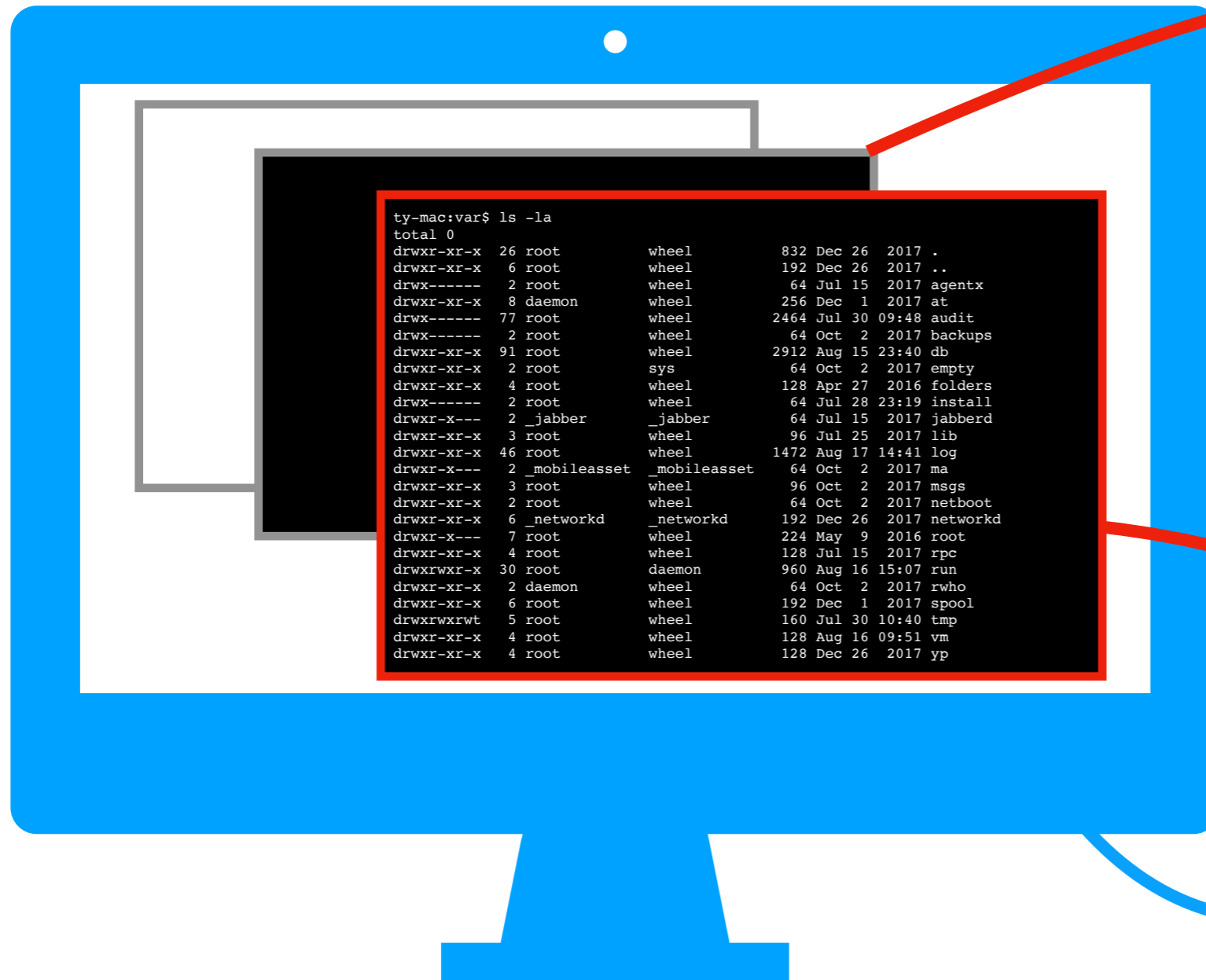


Terminal Emulators



local computer
(e.g., personal)

Terminal Emulators



remote computer
(e.g., CS lab)

OR



local computer
(e.g., personal)

Today's Topics

Terminal Emulators and Shells

- Terminal history
- **Shells**
- Running programs from a shell

Navigation

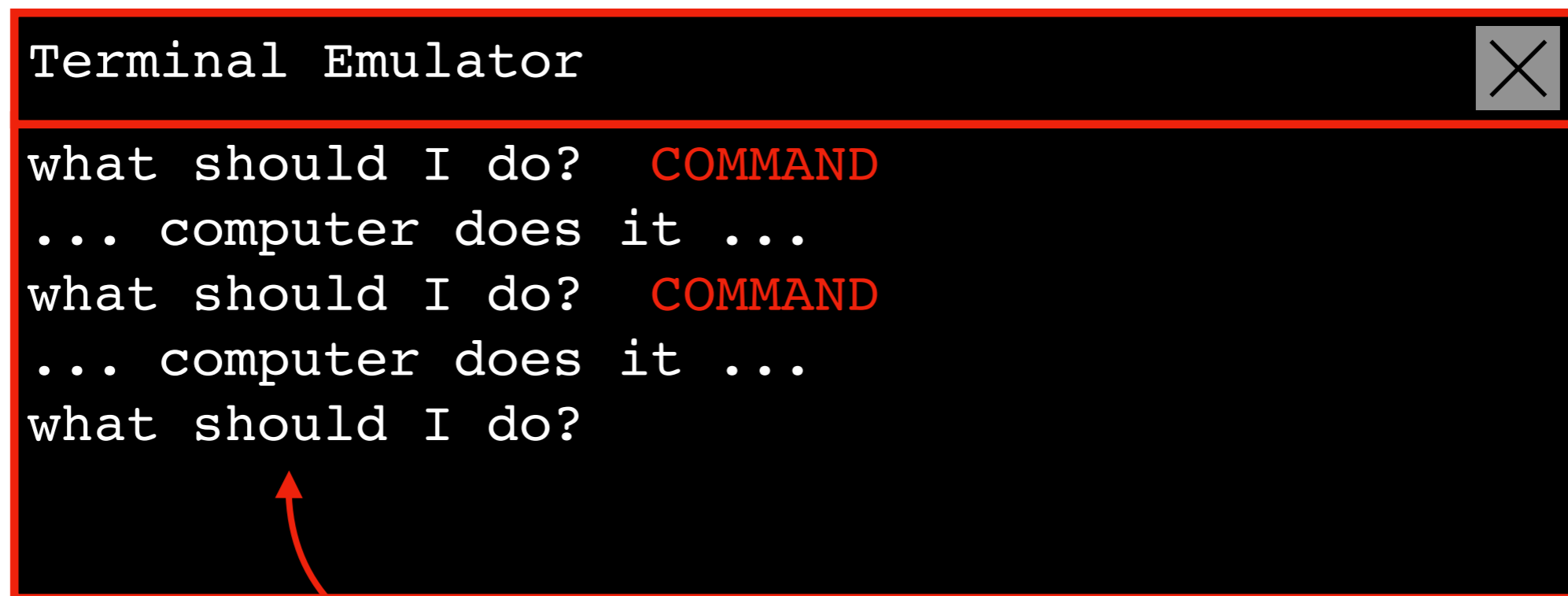
Running Programs and Commands

Demos

Shells

Inside a terminal, a program called a “shell” runs

- The shell lets users type commands, then carries out the appropriate actions
- Exploring files and running programs are common activities
- **You will be running Python programs from a shell in a terminal!**



```
Terminal Emulator
what should I do?  COMMAND
... computer does it ...
what should I do?  COMMAND
... computer does it ...
what should I do?
```

the shell is the program running inside the terminal emulator

Shells

Inside a terminal, a program called a “shell” runs

- The shell lets users type commands, then carries out the appropriate actions
- Exploring files and running programs are common activities
- **You will be running Python programs from a shell in a terminal!**
- Different shells have minor (or major) variations

Shells

Inside a terminal, a program called a “shell” runs

- The shell lets users type commands, then carries out the appropriate actions
- Exploring files and running programs are common activities
- **You will be running Python programs from a shell in a terminal!**
- Different shells have minor (or major) variations

Windows Shells

- cmd

type “dir” to view files

- PowerShell

type “ls” (for list) to view files

Shells

Inside a terminal, a program called a “shell” runs

- The shell lets users type commands, then carries out the appropriate actions
- Exploring files and running programs are common activities
- **You will be running Python programs from a shell in a terminal!**
- Different shells have minor (or major) variations

Windows Shells

- cmd
- PowerShell

UNIX Shells (Mac and Linux)

- bash
- csh
- zsh
- many more

Shells


Inside a terminal, a program called a “shell” runs

- The shell lets users type commands, then carries out the appropriate actions
- Exploring files and running programs are common activities
- **You will be running Python programs from a shell in a terminal!**
- Different shells have minor (or major) variations

Windows Shells

- cmd
- PowerShell 

UNIX Shells (Mac and Linux)

- bash 
- csh
- zsh
- many more

Today's Topics

Terminal Emulators and Shells

- Terminal history
- Shells
- Running programs from a shell

Navigation

Running Programs and Commands

Demos

Running Programs

Running programs is easy, just type name of the program and hit enter:

```
ty-mac:var$
```

Running Programs

Running programs is easy, just type name of the program and hit enter:

```
ty-mac:var$ ls
```

Running Programs

Running programs is easy, just type name of the program and hit enter:

```
ty-mac:var$ ls
agentx      jabberd    root
at          lib        rpc
audit       log        run
backups     ma         rwho

ty-mac:var$
```

Running Programs

Running programs is easy, just type name of the program and hit enter:

program name

```
prompt ty-mac:var$ ls
agentx    jabberd    root
at        lib        rpc
audit     log        run
backups   ma        rwho

prompt ty-mac:var$
```

a "prompt" is the question, *what should I do?*

Today's Topics

Terminal Emulators and Shells

Navigation

- Storage Drives (Windows)
- Files
- Directories (aka Folders)
- Windows vs. Mac

Running Programs and Commands

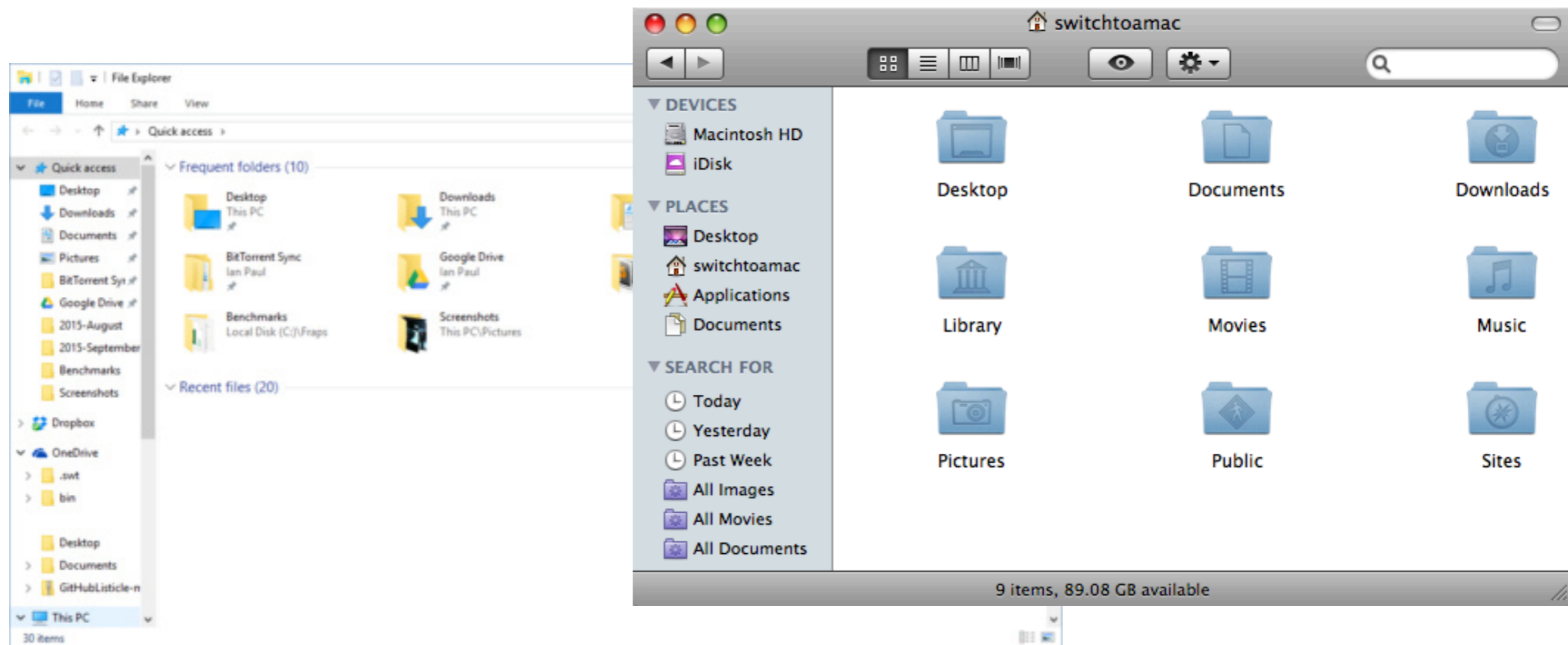
Demos

What is navigation?

Navigation is looking around for files/folders you want

Navigation programs

- File Explorer (Windows)
- Finder (Mac)



What is navigation?

Navigation is looking around for files/folders you want

Navigation programs

- File Explorer (Windows)
- Finder (Mac)

With shell, navigate w/ various commands...

`ls`

`pwd`

`cat`

`...`

`cd`

`mkdir`

Today's Topics

Terminal Emulators and Shells

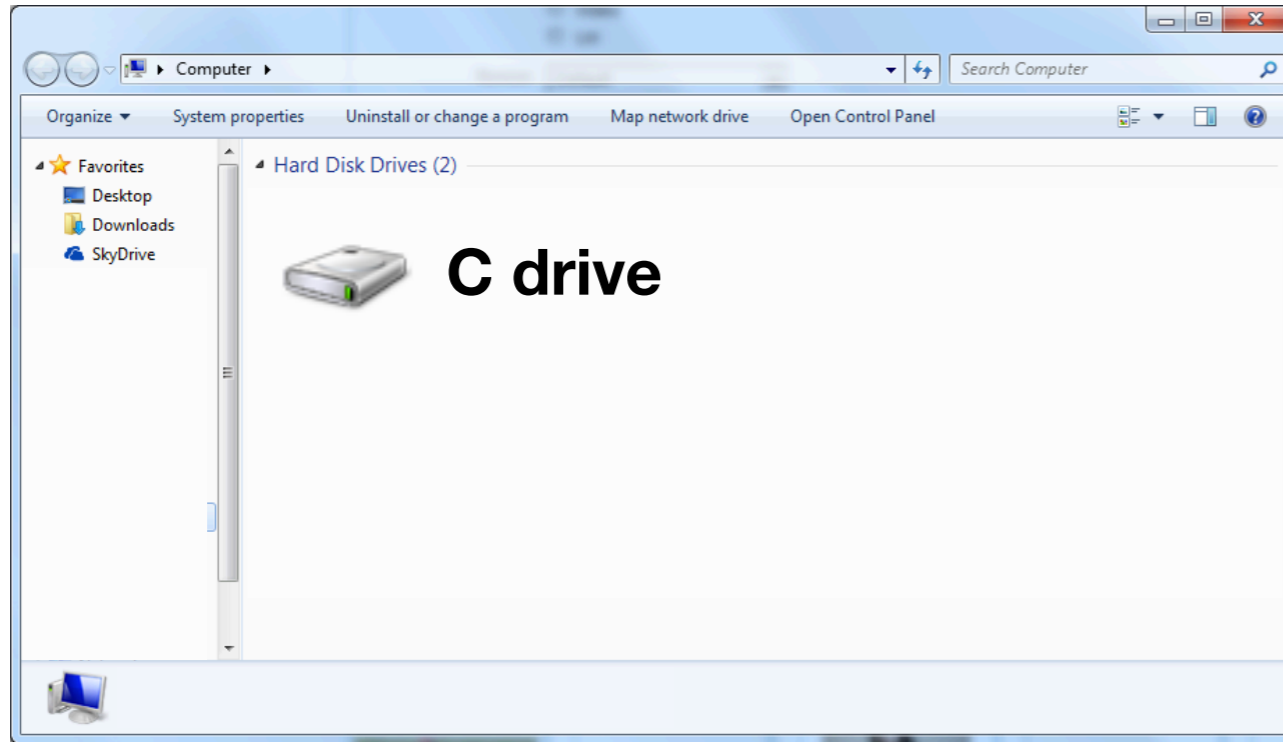
Navigation

- Storage Drives (Windows)
- Files
- Directories (aka Folders)
- Windows vs. Mac

Running Programs and Commands

Demos

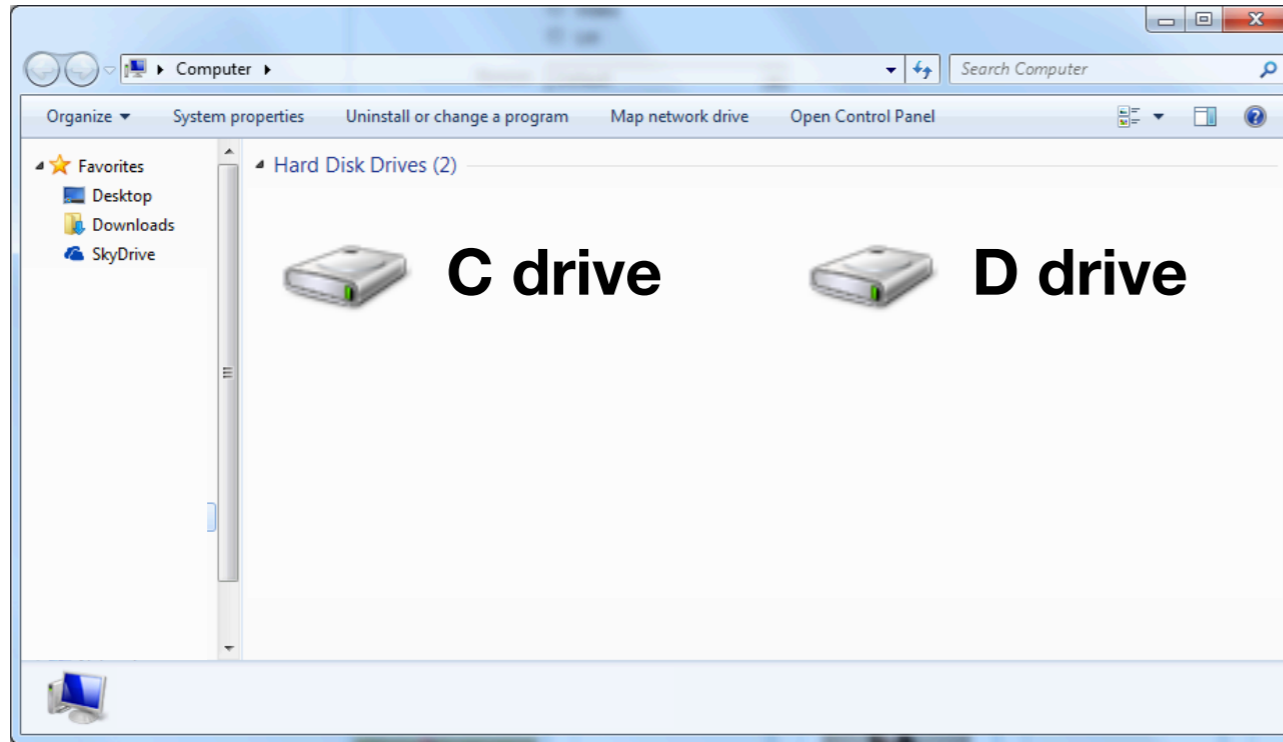
Windows Storage Drives



Each added drive is given its own drive letter



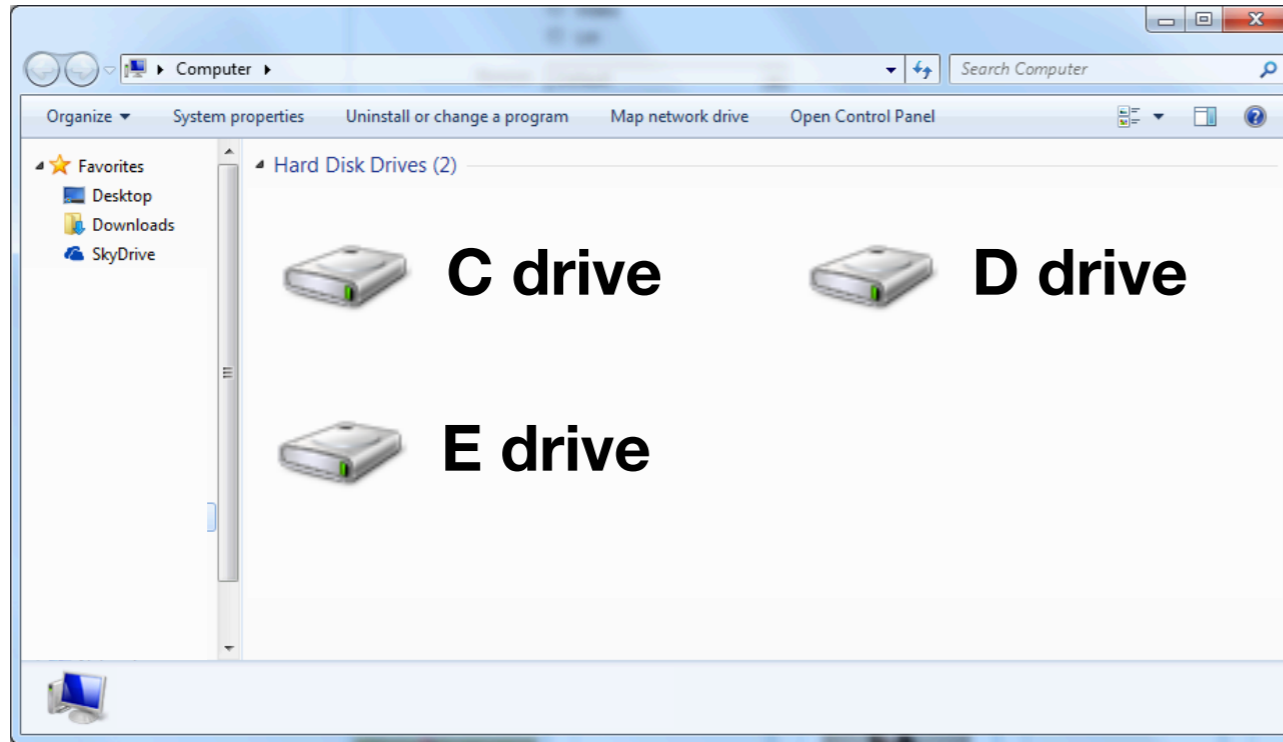
Windows Storage Drives



Each added drive is given its own drive letter



Windows Storage Drives



Each added drive is given its own drive letter



Today's Topics

Terminal Emulators and Shells

Navigation

- Storage Drives (Windows)
- **Files**
- Directories (aka Folders)
- Windows vs. Mac

Running Programs and Commands

Demos

Files

Each file has a name, called a “path name”

c:\README.txt

c:\hw.docx

d:\page.html

e:\main.py

Files

Each file has a name, called a “path name”

filename

c:\README.txt

c:\hw.docx

d:\page.html

e:\main.py

Files

Each file has a name, called a “path name”

filename
c:\README.txt
pathname

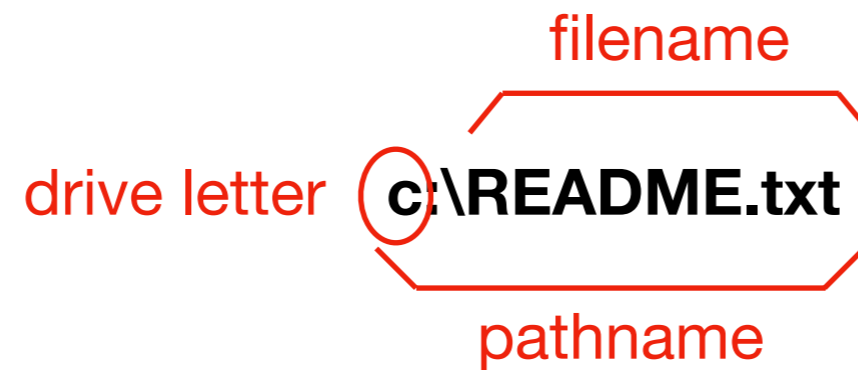
c:\hw.docx

d:\page.html

e:\main.py

Files

Each file has a name, called a “path name”



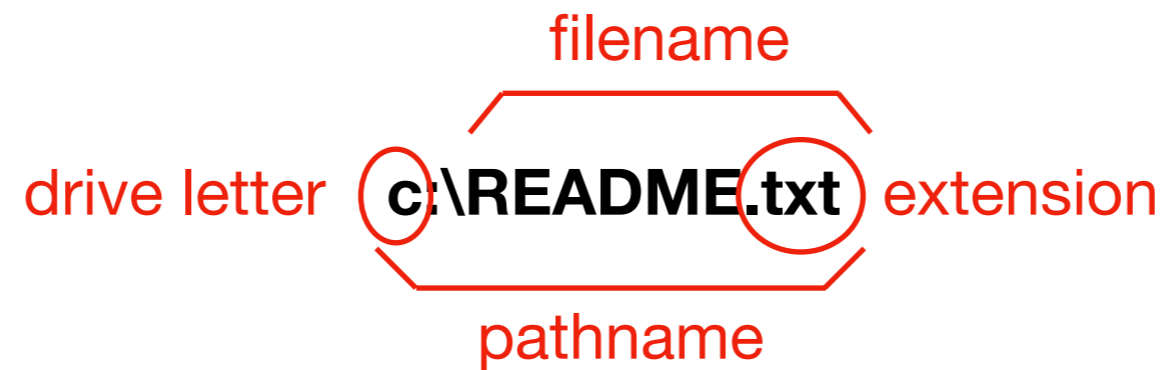
`c:\hw.docx`

`d:\page.html`

`e:\main.py`

Files

Each file has a name, called a “path name”



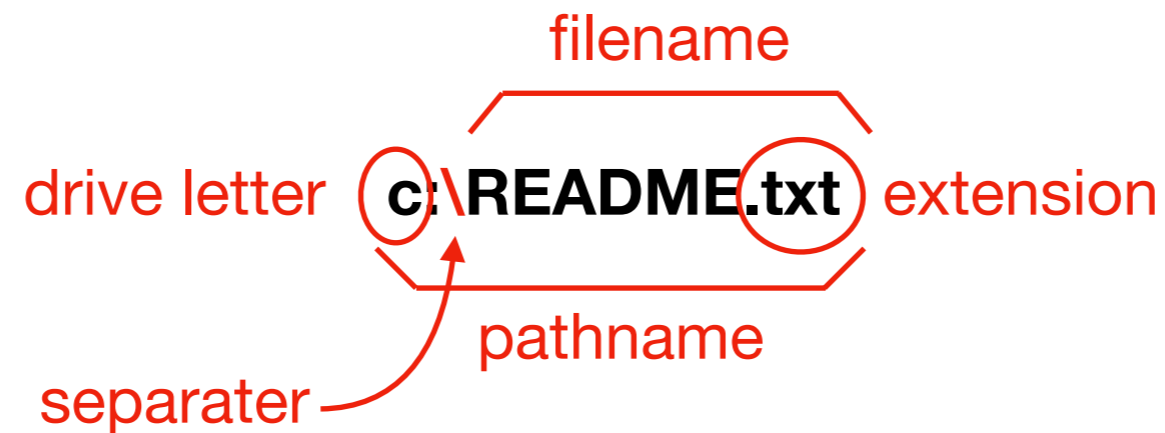
`c:\hw.docx`

`d:\page.html`

`e:\main.py`

Files

Each file has a name, called a “path name”



`c:\hw.docx`

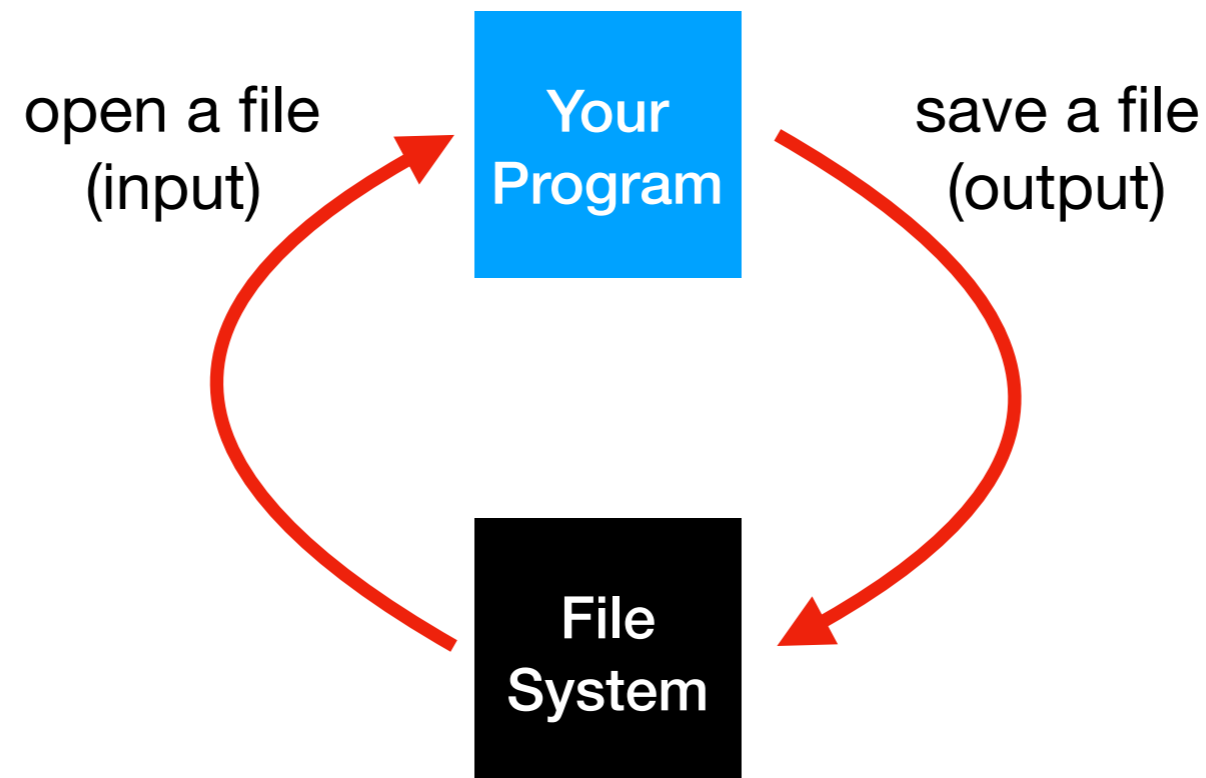
`d:\page.html`

`e:\main.py`

Files

Files might be either **input** or **output** for your programs

Files are managed by a part of the operating system called the **“file system”**



Today's Topics

Terminal Emulators and Shells

Navigation

- Storage Drives (Windows)
- Files
- Directories (aka Folders)
- Windows vs. Mac

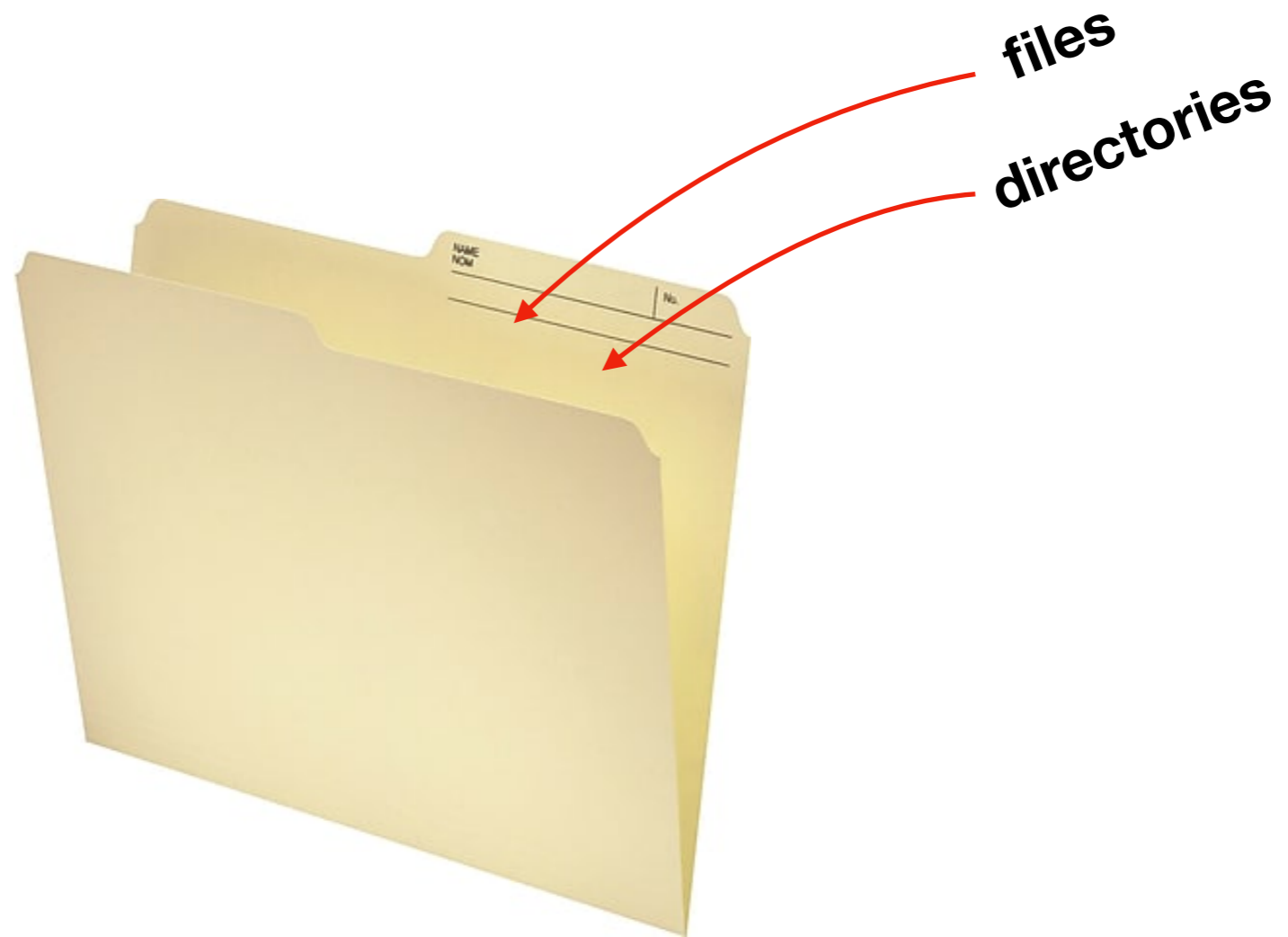
Running Programs and Commands

Demos

Directories

Directories are used to organize files and sub directories

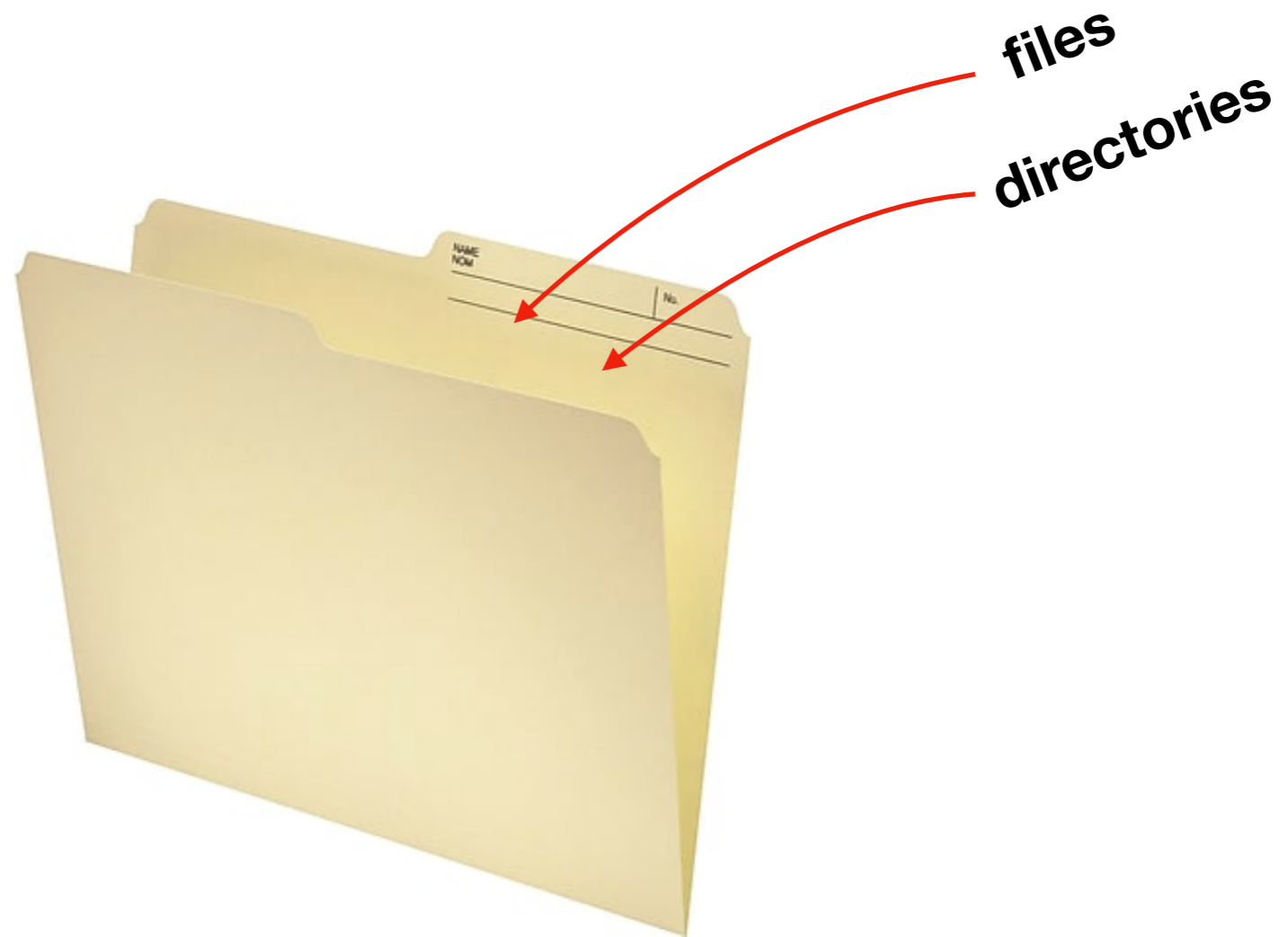
- Also called “folders”



Directories

Directories are used to organize files and sub directories

- Also called “folders”
- A directory also has pathname



Directories

Directories are used to organize files and sub directories

- Also called “folders”
- A directory also has pathname

Example paths:

- c:\my-directory\file1.docx
- c:\my-directory\file2.docx
- c:\my-directory\file3.docx



in my-directory

Directories

Directories are used to organize files and sub directories

- Also called “folders”
- A directory also has pathname

Example paths:

- c:\my-directory\file1.docx
- c:\my-directory\file2.docx
- c:\my-directory\file3.docx
- c:\directory1\directory2\file1.docx
- c:\same-dir\same-dir\readme.txt

Directories

Directories are used to organize files and sub directories

- Also called “folders”
- A directory also has pathname

Example paths:

- c:\my-directory\file1.docx
- c:\my-directory\file2.docx
- c:\my-directory\file3.docx
- c:\directory1\directory2\file1.docx
- c:\same-dir\same-dir\readme.txt

two types of paths: **relative** or **absolute**

Relative Paths

Where is the Computer Science building?

- **Answer 1:** 1210 W Dayton St, Madison, WI 53706
- **Answer 2:** on the other side of Johnson street



When is Answer 2 appropriate?

Relative Paths

Where is the Computer Science building?

- **Answer 1:** 1210 W Dayton St, Madison, WI 53706
- **Answer 2:** on the other side of Johnson street



When is Answer 2 appropriate?

- When you're in the psychology building
- It may be more convenient

Relative Paths

Where is the Computer Science building?

- **Answer 1:** 1210 W Dayton St, Madison, WI 53706
- **Answer 2:** on the other side of Johnson street



When is Answer 2 appropriate?

- When you're in the psychology building
- It may be more convenient

Pathnames are absolute (answer 1) or relative (answer 2)

- Absolute paths: always possible
- Relative paths: **if current location is known**

Relative Paths

Where is the Computer Science building?

- **Answer 1:** 1210 W Dayton St, Madison, WI 53706
- **Answer 2:** on the other side of Johnson street



When is Answer 2 appropriate?

- When you're in the psychology building
- It may be more convenient

Pathnames are absolute (answer 1) or relative (answer 2)

- Absolute paths: always possible
- Relative paths: **if current location is known**
- **Current Working Directory**

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	
c:\x\y\z\my.docx	c:\x\y	
c:\x\y\z	c:\x	

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	
c:\x\y\z	c:\x	

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z

Two special directory names

- “..” means up a directory
- “.” means current directory

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	
c:\x\y\z	c:\x	
c:\x	c:\x\y\z	

Two special directory names

- “..” means up a directory
- “.” means current directory

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	
c:\x	c:\x\y\z	

Two special directory names

- “..” means up a directory
- “.” means current directory

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	.\y\z
c:\x	c:\x\y\z	

Two special directory names

- “..” means up a directory
- “.” means current directory

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	.\y\z
c:\x	c:\x\y\z	..\..

Two special directory names

- “..” means up a directory
- “.” means current directory

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	.\y\z
c:\x	c:\x\y\z	..\..
c:\B\file.txt	c:\A	

Two special directory names

- “..” means up a directory
- “.” means current directory

Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	.\y\z
c:\x	c:\x\y\z	..\..
c:\B\file.txt	c:\A	..\B\file.txt

Two special directory names

- “..” means up a directory
- “.” means current directory

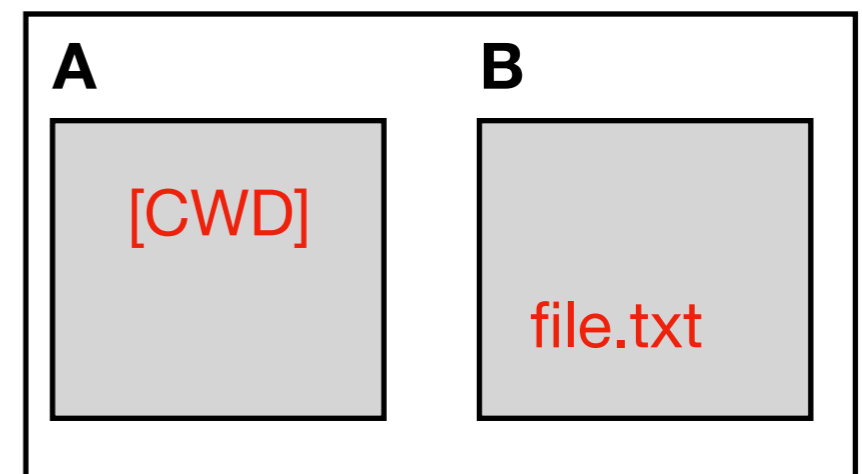
Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	.\y\z
c:\x	c:\x\y\z	..\..
c:\B\file.txt	c:\A	..\B\file.txt

Two special directory names

- “..” means up a directory
- “.” means current directory

c:\

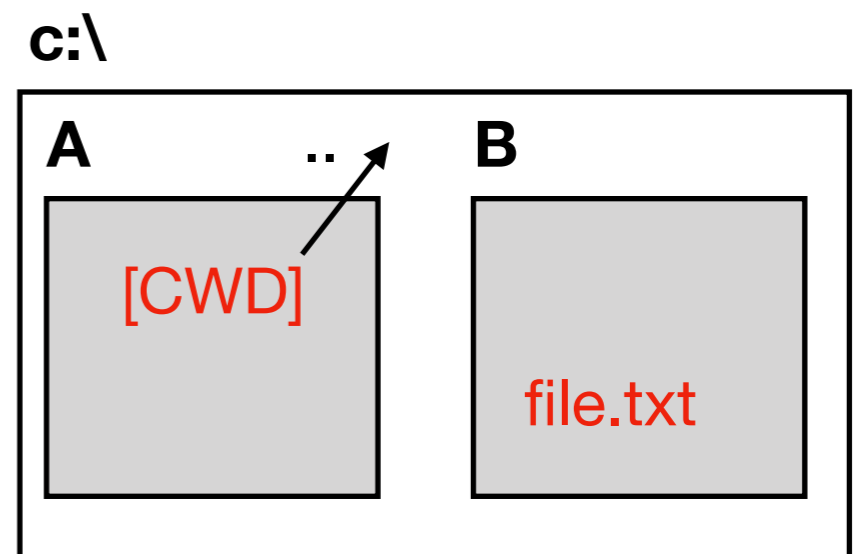


Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	.\y\z
c:\x	c:\x\y\z	..\..
c:\B\file.txt	c:\A	..\B\file.txt

Two special directory names

- “..” means up a directory
- “.” means current directory

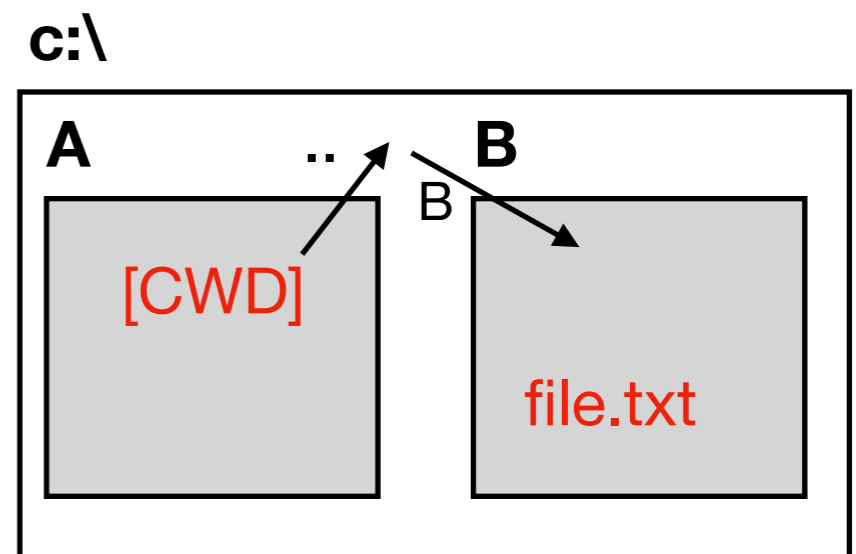


Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	.\y\z
c:\x	c:\x\y\z	..\..
c:\B\file.txt	c:\A	..\B\file.txt

Two special directory names

- “..” means up a directory
- “.” means current directory

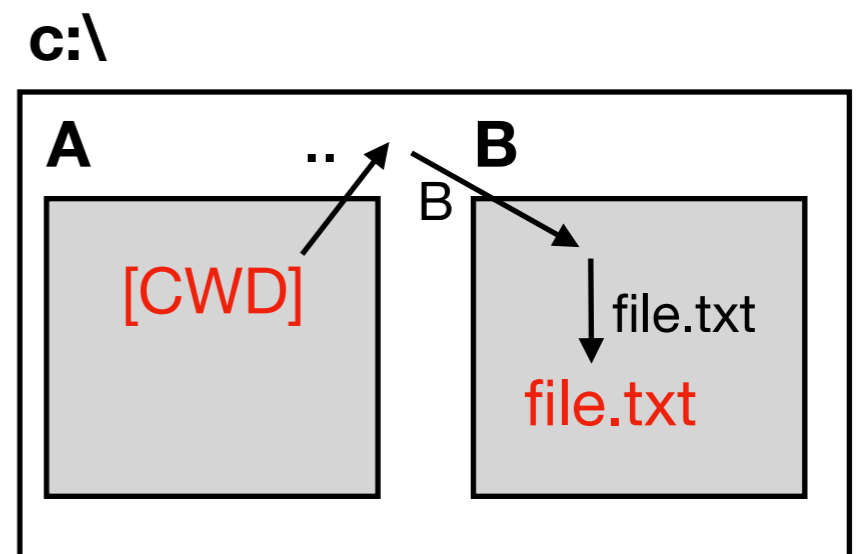


Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	.\y\z
c:\x	c:\x\y\z	..\..
c:\B\file.txt	c:\A	..\B\file.txt

Two special directory names

- “..” means up a directory
- “.” means current directory



Absolute vs. Relative

Absolute Path	Working Directory	Relative Path
c:\test.txt	c:\	test.txt
c:\x\y\z\my.docx	c:\x\y\z	my.docx
c:\x\y\z\my.docx	c:\x\y	z\my.docx
c:\x\y\z	c:\x	y\z
c:\test.txt	c:\	.\test.txt
c:\test.txt	c:\	..\test.txt
c:\x\y\z	c:\x	.\y\z
c:\x	c:\x\y\z	..\..
c:\B\file.txt	c:\A	..\B\file.txt

Two special directory names

- “..” means up a directory
- “.” means current directory

more examples in demo later...

Today's Topics

Terminal Emulators and Shells

Navigation

- Storage Drives (Windows)
- Files
- Directories (aka Folders)
- **Windows vs. Mac**

Running Programs and Commands

Demos

Multiple Drives in Mac

Windows

- Absolute paths start with **c:** or **d:**
- Indicates which drive

Mac

- Absolute paths start with **/**
- Example: **/Users/tyler/my-file.docx**
- Don't know which drive

How can we use multiple drives if every file paths starts the same???

/.....

Multiple Drives in Mac

Windows

- Absolute paths start with **c:** or **d:**
- Indicates which drive

Mac

- Absolute paths start with **/**
- Example: **/Users/tyler/my-file.docx**
- Don't know which drive

How can we use multiple drives if every file paths starts the same???
/.....

Answer: different drives feel like different directories

Comparison

Windows

Mac

Drives

c:\Users\tyler\file.txt

/Users/tyler

c:\Program Files

/usr/local/bin

c:\Windows\...\Logs

/var/log



d:\

/Volumes/backup

d:\A

/Volumes/backup/A



e:\movies

/Volumes/movies

e:\movies\demo1.mov

/Volumes/movies/demo1.mov



Comparison

on a Mac, a path doesn't tell you what drive you're on

Windows

Mac

Drives

c:\Users\tyler\file.txt
c:\Program Files
c:\Windows\...\Logs

/Users/tyler
/usr/local/bin
/var/log



d:\
d:\A

/Volumes/backup
/Volumes/backup/A



e:\movies
e:\movies\demo1.mov

/Volumes/movies
/Volumes/movies/demo1.mov



Today's Topics

Terminal Emulators and Shells

Navigation

Running Programs and Commands

- Navigational commands
- Arguments
- Saving output

Demos

We'll cover a few simple examples for reference in the slides, then go into more detail in the demo...

Most of these examples work in both PowerShell (Windows) and bash (Mac)

Today's Topics

Terminal Emulators and Shells

Navigation

Running Programs and Commands

- Navigational commands
- Arguments
- Saving output

Demos

Where am I? (What directory am I in?)

Command: `pwd`

```
PS /Users/trh/scratch>
```

Where am I? (What directory am I in?)

Command: `pwd`

“print working directory”

```
PS /Users/trh/scratch> pwd
```

Where am I? (What directory am I in?)

Command: `pwd`

```
PS /Users/trh/scratch> pwd
```

```
Path
```

```
----
```

```
/Users/trh/scratch
```



this is the current directory

```
PS /Users/trh/scratch>
```


Go up a directory

Command: `cd ..`

```
PS /Users/trh/scratch> pwd
```

```
Path
```

```
----
```

```
/Users/trh/scratch
```

```
PS /Users/trh/scratch>
```

Go up a directory

Command: `cd ..`

```
PS /Users/trh/scratch> pwd
```

```
Path
```

```
----
```

```
/Users/trh/scratch
```

```
PS /Users/trh/scratch> cd ..
```

Go up a directory

Command: `cd ..`

```
PS /Users/trh/scratch> pwd
```

```
Path
```

```
----
```

```
/Users/trh/scratch
```

```
PS /Users/trh/scratch> cd ..
```

```
PS /Users/trh>
```

Clear the screen

Command: `clear`

```
PS /Users/trh/scratch> pwd
```

```
Path
```

```
----
```

```
/Users/trh/scratch
```

```
PS /Users/trh/scratch> cd ..
```

```
PS /Users/trh> clear
```

Clear the screen

Command: `clear`

```
PS /Users/trh>
```

Go inside a directory

Command: `cd directory-name`

```
PS /Users/trh>
```

Go inside a directory

Command: `cd directory-name`

name of directory we started in

```
PS /Users/trh> cd scratch
```

Go inside a directory

Command: `cd directory-name`

```
PS /Users/trh> cd scratch  
PS /Users/trh/scratch>
```


Go to top directory

Command: `cd /`

is this Windows or Mac?

```
PS /Users/trh> cd scratch  
PS /Users/trh/scratch> cd /
```

Go to top directory

Command: `cd /`

```
PS /Users/trh> cd scratch  
PS /Users/trh/scratch> cd /  
PS />
```

View contents of current directory

Command: **ls**

```
PS /Users/trh> cd scratch  
PS /Users/trh/scratch> cd /  
PS />
```

View contents of current directory

Command: **ls**

```
PS /Users/trh> cd scratch  
PS /Users/trh/scratch> cd /  
PS /> ls
```

View contents of current directory

Command: **ls**

```
PS /Users/trh> cd scratch
PS /Users/trh/scratch> cd /
PS /> ls
Applications          etc
Library               home
Network              installer.failurerequests
System               net
Users                README.txt
PS />
```

View contents of a file

Command: `cat file-name`

```
PS /Users/trh> cd scratch
PS /Users/trh/scratch> cd /
PS /> ls
Applications          etc
Library               home
Network               installer.failurerequests
System                net
Users                 README.txt
PS />
```

View contents of a file

Command: `cat file-name`

```
PS /Users/trh> cd scratch
PS /Users/trh/scratch> cd /
PS /> ls
Applications          etc
Library               home
Network              installer.failurerequests
System               net
Users                README.txt
PS /> cat README.txt
```

View contents of a file

Command: `cat file-name`

```
PS /Users/trh> cd scratch
PS /Users/trh/scratch> cd /
PS /> ls
Applications          etc
Library               home
Network              installer.failurerequests
System               net
Users                README.txt
PS /> cat README.txt
The file says Hello!

PS />
```


View contents of a file

Command: `cat file-name`

```
PS /Users/trh> cd scratch
PS /Users/trh/scratch> cd /
PS /> ls
Applications          etc
Library               home
Network               installer.failurerequests
System               net
Users                 README.txt
PS /> cat README.txt
The file says Hello!
```

data saved in README.txt

Today's Topics

Terminal Emulators and Shells

Navigation

Running Programs and Commands

- Navigational commands
- **Arguments**
- Saving output

Demos

Arguments (program input)

```
PS /Users/trh> cd scratch
PS /Users/trh/scratch> cd /
PS /> ls
Applications          etc
Library               home
Network              installer.failurerequests
System               net
Users                README.txt
PS /> cat README.txt
The file says Hello!

PS />
```

Arguments (program input)

```
PS /Users/trh> cd scratch
PS /Users/trh/scratch> cd /
PS /> ls
Applications          etc
Library               home
Network               initramfs
                       initramfs-6.11.0-rc1-pererequests
Users                 README.txt
PS /> cat README.txt
The file says Hello!

PS />
```

program name (cat)

an argument (README.txt)

echo Example

```
PS /Users/trh>
```

echo Example

```
PS /Users/trh> echo hello
```

echo Example

program is "echo"

argument is "hello"

```
PS /Users/trh> echo hello
```

echo Example

```
PS /Users/trh> echo hello  
hello  
PS /Users/trh>
```


echo Example

```
PS /Users/trh> echo hello  
hello
```

```
PS /User
```

the echo program prints
whatever it's argument is

Today's Topics

Terminal Emulators and Shells

Navigation

Running Programs and Commands

- Navigational commands
- Arguments
- **Saving output**

Demos

Saving output

Format: `program > file-name`

```
PS /Users/trh>
```

Saving output

Format: **program > file-name**

```
PS /Users/trh> echo hello
```

Saving output

Format: **program** > **file-name**

```
PS /Users/trh> echo hello  
hello  
PS /Users/trh>
```

Saving output

Format: **program** > **file-name**

```
PS /Users/trh> echo hello
```

```
hello
```

```
PS /Users/trh> echo hello > output.txt
```

“redirect” operator, sends output to a file

Saving output

Format: **program > file-name**

```
PS /Users/trh> echo hello  
hello  
PS /Users/trh> echo hello > output.txt  
PS /Users/trh>
```

Saving output

Format: **program > file-name**

```
PS /Users/trh> echo hello
```

```
hello
```

```
PS /Users/trh> echo hello > output.txt
```

```
PS /Users/trh>
```

without redirect, output
was printed to the screen

with redirect, output was
saved in the output.txt file

Saving output

Format: **program > file-name**

```
PS /Users/trh> echo hello  
hello  
PS /Users/trh> echo hello > output.txt  
PS /Users/trh>
```

Saving output

Format: **program** > **file-name**

```
PS /Users/trh> echo hello
hello
PS /Users/trh> echo hello > output.txt
PS /Users/trh> cat output.txt
```

Saving output

Format: **program > file-name**

```
PS /Users/trh> echo hello
hello
PS /Users/trh> echo hello > output.txt
PS /Users/trh> cat output.txt
hello
PS /Users/trh>
```

Today's Topics

Terminal Emulators and Shells

Navigation

Running Programs and Commands

Demos

Conclusion

Today we covered

- What a terminal and shell is
- What it looks like to have multiple storage drives attached to your computer
- How to navigate between directories/folders
- How to run programs in the terminal