# SI 506 Lecture 02

## Useful Unix shell commands

Below is a select list of Unix shell commands that you can use from the command line to navigate your local file system as well as create, delete, copy, and move directories and files. Additional commmands are included that list the location of the current working directory or the path to an executable, view the contents of a text file, and clear the terminal screen of content.

## 1.0 Location uncertain

If your terminal prompt provides no hint and you are unsure in which directory you currently reside, use the built-in `pwd` command to print the current working directory.

### 1.1 macOS

```
pwd

/Users/arwhyte
```

### 1.2 Windows

```
pwd

/c/Users/arwhyte
```

## 2.0 List files in the current working directory

If you require basic info about subdirectories and files that reside in the current directory, use the `ls` command along with command options to print out the details.

❗ the output below represents a subset of subdirectories and files found in a macOS home directory; certain items have been excluded for display purposes only.

### 2.1 macOS

```
ls

Applications      Development      Music
Downloads         Documents        Pictures
Dropbox           Library          Postman
Desktop           Movies           Public
```

### 2.2 Windows

```
ls

'3D Objects'/
 AppData/
'Application Data'@
 Contacts/
 Cookies@
 Desktop/
 Documents/
 Downloads/
 Favorites/
 Links/
'Local Settings'@
 MicrosoftEdgeBackups/
 Music/
'My Documents'@
 OneDrive/
 Pictures/
 PrintHood@
 Recent@
 Searches/
 SendTo@
'Start Menu'@
 Templates@
 Videos/
```

### 2.3 ls command options (select list)

| Option | | Description |
|---|---|---|
| `-a`, | `--all` | List all files including hidden `.` files. |
| `-d`, | `--directory` | List only directory information (not files). |
| `-l`, | `--format=long` | Long format listing (permissions, owner, size, modification time, etc.). |
| `-R`, | `--recursive` | Recursively list subdirectories as well as current directory |

❗ the output below represents a subset of subdirectories and files found in a macOS home directory; certain items have been excluded for display purposes only.

```
ls -al

total 2848520
drwx------+  76 arwhyte  staff        2432 Sep  3 07:56 .
drwxr-xr-x   11 root     admin         352 Dec  5  2019 ..
-rw-r--r--@   1 arwhyte  staff       10244 Sep  3 07:28 .DS_Store
drwx------    2 arwhyte  staff          64 Sep  3 07:29 .Trash
```

```
-rw-------     1 arwhyte  staff       33742 Mar 16 14:58 .bash_history
-rw-r--r--@    1 arwhyte  staff        1870 Apr 19 23:15 .bash_profile
drwx------     4 arwhyte  staff         128 Jan 14  2019 .config
drwx------     3 arwhyte  staff          96 Feb 13  2019 .cups
drwx------    15 arwhyte  staff         480 Oct 16  2019 .dropbox
-rw-r--r--     1 arwhyte  staff         767 Aug 20 15:01 .gitconfig
drwxr-xr-x     3 arwhyte  staff          96 Oct 19  2019 .idlerc
drwxr-xr-x     3 arwhyte  staff          96 Mar 27 13:03 .local
drwxr-xr-x    19 arwhyte  staff         608 Aug 31 20:59 .oh-my-zsh
drwxr-xr-x   304 arwhyte  staff        9728 Sep  1 13:54 .pylint.d
-rw-------     1 arwhyte  staff        8730 Sep  1 17:02 .python_history
drwx------     8 arwhyte  staff         256 Jan 11  2019 .ssh
drwxr-xr-x     4 arwhyte  staff         128 Dec  3  2019 .vscode
drwxr-xr-x     2 arwhyte  staff          64 Aug  6  2019 .zoomus
-rw-------     1 arwhyte  staff      660463 Sep  3 07:56 .zsh_history
-rw-r--r--     1 arwhyte  staff         281 Jul  2 17:04 .zshenv
-rw-r--r--     1 arwhyte  staff        4188 Aug 14 18:15 .zshrc
drwx------@    4 arwhyte  staff         128 Jun 16 14:20 Applications
drwx------@   35 arwhyte  staff        1120 Sep  2 14:47 Desktop
drwxr-xr-x    10 arwhyte  staff         320 Aug 23 19:55 Development
drwx------@    5 arwhyte  staff         160 Sep  3 07:28 Documents
drwx------@  275 arwhyte  staff        8800 Sep  2 18:13 Downloads
drwx------@   12 arwhyte  staff         384 Aug 22 08:27 Dropbox
drwx------@   83 arwhyte  staff        2656 Jun 11 20:22 Library
drwx------+    8 arwhyte  staff         256 Jul 20 21:38 Movies
drwx------+    7 arwhyte  staff         224 Dec 20  2019 Music
drwx------+   87 arwhyte  staff        2784 Aug  5 17:58 Pictures
drwxr-xr-x     3 arwhyte  staff          96 Jul 15 12:36 Postman
drwxr-xr-x+    4 arwhyte  staff         128 Nov 28  2018 Public
```

## 3.0 Change directory

If you need to change your current location, use the cd command to change your location to a different working directory in your file system.

❗ Note that directory names and file names are case sensitive.

### 3.1 Change to a child directory

You can change to child directory using a *relative* path (i.e., relative to the current working directory).

```
pwd

/Users/arwhyte

cd Documents
pwd

/Users/arwhyte/Documents
```

💡 the current working directory is denoted by a single dot (`.`).

```
pwd

/Users/arwhyte

cd ./Documents
pwd

/Users/arwhyte/Documents
```

💡 If you need to traverse n-levels deep you can do so by extending the relative path with additional directory names separated by a slash (`/`).

```
pwd

/Users/arwhyte

cd Documents/umsi

pwd

/Users/arwhyte/Documents/umsi
```

### 3.2 Change to a parent directory

Two dots (`..`) represent the parent directory or the directory one level up.

```
pwd

/Users/arwhyte/Documents

cd ..
pwd

/Users/arwhyte
```

💡 You can concatenate the two dot parent directory notation using a slash as a separator (e.g., `../../`) *n-times* in order to traverse the directory tree *n-levels* up.

```
pwd

/Users/arwhyte/Documents

cd ../../
```

```
pwd

/Users
```

### 3.3 Change to an adjacent or sibling directory

You can switch to an adjacent or sibling directory by using the two dot notation (`..`) together with the directory name separated by a slash (`/`). In the following example the `Documents` directory contains two child directories: `umsi` and `umpy`.

```
pwd

/Users/arwhyte/Documents/umsi

cd ../umpy
pwd

/Users/arwhyte/Documents/umpy
```

### 3.4 Change directory using an absolute path

You can also change directories using an *absolute* path.

```
pwd

/Users/arwhyte

cd /Users/arwhyte/Documents
pwd

/Users/arwhyte/Documents
```

### 3.5 Change to user's home directory

You can change to your home directory by using the tilde (`~`) character.

```
pwd

/Users/arwhyte/Documents/umsi

cd ~
pwd

/Users/arwhyte
```

### 3.6 Directory names with spaces

If you need to change to a directory that includes spaces in its name you *must* either surround the name with a pair of single or double quotation marks or escape the spaces with the backslash (`\`) character.

💡 I recommend avoiding the use of spaces when naming directories or files in order to avoid having to add quotation marks or escape characters to your paths. Instead consider using underscores (`_`) if you want to separate characters in a directory or filename (e.g., `si_506` not `si 506`).

While on the subject of filenames, the Python community's naming convention for filenames or modules as they are called is as follows:

> Modules should have short, all-lowercase names. Underscores can be used in the module name if it improves readability.

```
pwd
/Users/arwhyte/Documents/umsi
➜  umsi ls

si 506

cd 'si 506'
pwd

/Users/arwhyte/Documents/umsi/si 506

cd ..
cd "si 506"
pwd

/Users/arwhyte/Documents/umsi/si 506

cd ..
cd si\ 506
pwd

/Users/arwhyte/Documents/umsi/si 506
```

## 4.0 Create a directory

To create a new director use the `mkdir` command passing the name of the new directory as an argument.

```
 pwd

/Users/arwhyte/Documents

mkdir umich
ls
```

```
umich    umpy    umsi
```

💡 You can create multiple directories at the same time by passing multiple names each separated by a space.

```
 pwd

/Users/arwhyte/Documents

mkdir msu osu
ls

msu    osu    umich    umpy    umsi
```

## 5.0 Delete a directory

### 5.1 Delete an empty directory

To delete an *empty* directory use the `rmdir` command passing the name of the directory you wish to delete as an argument.

```
 pwd

/Users/arwhyte/Documents

rmdir osu
ls

msu    umich    umpy    umsi
```

### 5.2 Delete a directory with content

To delete a directory that contains content (i.e., subdirectories and/or files) use the `rm` command together with the `-r` recursive command option and either the `-f` force option or `i` interactive command option.

#### 5.2.1 rm command options (select list)

| Option | Description |
|---|---|
| `-f`, `--force` | Remove write protected files without prompting. |
| `-i`, `--interactive` | Prompt for `y` (yes) or `n` (no) before removing a file. Overrides `-f`. |
| `-r`, `--recursive` | Remove all subdirectories and content recursively. |

```
rmdir msu

rmdir: msu: Directory not empty

cd msu
ls

spartans.txt

cd ..
rm -rf msu
ls

umich    umpy    umsi
```

## 6.0 Create a file

You can use the `touch` command to create an empty file by passing the new filename as an argument.

```
pwd

/Users/arwhyte/Documents/umich

touch wolverines.txt
ls

wolverines.txt
```

## 7.0 View the contents of a text file

To view the contents of a text file use the `cat` command.

```
cat wolverines.txt

Go Blue!
```

## 8.0 Delete a file

You can use the `rm` command to delete a file.

```
rm delete_me.txt
```

## 9.0 Move a directory or file to another location

You can use the `mv` command to move a directory or file from one location to another. Specify the *source* directory or file (i.e., the directory or file you wish to move) and the *target* location as arguments.

⚠️ if you move a file to a directory that contains a file with the same name you will overwrite the existing file.

```
pwd

/Users/arwhyte/Documents

mv umpy umich/
mv umsi umich/
cd umich
ls

umpy    umsi    wolverines.txt
```

To move directories or files up one level employ the two dot notation with a trailing slash to construct a relative path. You can also employ an absolute path (e.g., `/Users/arwhyte/Documents/`) for the *target* location.

```
pwd

/Users/arwhyte/Documents/umsi/

ls
umpy    umsi    wolverines.txt

mv umpy ../
mv umsi ../

ls

wolverines.txt

cd ../
ls

umich    umpy    umsi
```

When you move a directory or file you can also change the name by specifying a new name in the *target* path.

```
pwd

/Users/arwhyte/Documents/
```

```
cd umich
ls

wolverines.txt

mv wolverines.txt ../go_blue.txt
cd ../

ls

go_blue.txt    umich    umpy    umsi
```

💡 If you possess the requisite permissions and construct the correct *target* path you can move directories and files to any target location in your file system.

## 10.0 Copy a directory or file to another location

You can use the `cp` command to copy a directory or file to another location. Specify the *source* directory or file (i.e., the directory or file you wish to copy) and the *target* location as arguments.

💡 You can change the name of the directory or file you copy by specifying the new directory name or filename as part of the *target* path.

### 10.1 cp command options (select list)

| **Option** | **Description** |
|---|---|
| `-f`, `--force` | Remove existing files in target directory. |
| `-i`, `--interactive` | Prompt for `y` (yes) or `n` (no) before overwriting an existing file. |
| `-R`, `--recursive` | Copy directories recursively. |

### 10.2 Copy a directory to another location

When you use the `cp` command to copy a directory to another location you *must* also specify the command option `-R` in order to create a copy of the directory recursively. Otherwise, the copy operation will fail.

```
pwd

/Users/arwhyte/Documents/

mkdir program_01 program_02
cp -R program_01 umsi/msi
cp -R program_02 umsi/mhi
cd umsi
ls

mhi    msi
```

### 10.3 Copy a file to another location

Copying a file does not require use of the `-R` command option. Specify the *source* directory or file (i.e., the directory or file you wish to copy) and the *target* location as arguments.

```
pwd

/Users/arwhyte/Documents/

cp go_blue.txt umich/victors.txt

cd umich
ls

victors.txt
```

💡 you can copy multiple files to the same target path by passing the names as arguments before specifying the target path.

```
cp go_blue.txt go_green.txt cheers/
```

Alternatively, you can employ a pattern matching wildcard (`*`).

```
cp *.txt cheers/
```

## 11.0 Clear the terminal screen

There are times when clearing the terminal screen of output makes sense. Use the `clear` command to do so.

```
clear
```

## 12.0 which

The `which` command comes in handy when you need to identify the location of an executable that is associated with a given command. For example, to return the executable path for Python 3.x pass the command alias as the argument (Windows users pass `python`).

```
which python3

/usr/local/bin/python3
```

## 13.0 Start the Python interactive console

You can run the Python interactive console (a.k.a the Python shell) from the terminal. Once the console is started the prompt will change. The new prompt comprises three greater than symbols (`>>>`).

### 13.1 macOS

```
python3

Python 3.10.6 (main, Aug 11 2022, 13:36:31) [Clang 13.1.6 (clang-
1316.0.21.2.5)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

### 13.2 Windows Git Bash

❗ When using Git Bash you *must* include the `-i` interactive command option or the Python interactive console. If you fail to specify the `-i` option Git Bash will hang (terminate the application and restart it).

```
python -i

Python 3.10.6 (tags/v3.10.6:9c7b4bd, Aug  1 2022, 21:53:49) [MSC v.1932 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

You can also start the Python interactive console from Git Bash by first invoking `winpty`, a Windows software package that provides an interface for running Windows console programs.

```
winpty python

Python 3.10.6 (tags/v3.10.6:9c7b4bd, Aug  1 2022, 21:53:49) [MSC v.1932 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

### 13.3 Windows Command Prompt

You can also start the Python Interactive console using the the Command Prompt (`cmd`). The `-i` command option is not required.

```
python
```

```
Python 3.10.6 (tags/v3.10.6:9c7b4bd, Aug  1 2022, 21:53:49) [MSC v.1932 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

### 13.4 Quitting the console session

To exit the Python interactive console type `quit()` and then press enter.

```
>>> quit()
```

## Sources

A. Robbins, Unix in a Nutshell, 4th edition (O'Reilly Media, Inc., 2005).