

Unix Shell

MAS Departmental Disclaimers:

For students trying to take or audit from outside the MAS program.

Taking or auditing 400 courses is simply is not permitted because this is a self-supporting program. Sorry, unfortunately, you will NOT be able to take any of the 400 level Stats courses.

There are NO exceptions that can be made by the department. These classes were designed specifically for students who applied directly to the program.

The students of this program are also not allowed to audit or enroll in classes outside of the program as it was created for working professionals.

If you would like to apply for the program, you are welcome to do so: [https://
master.stat.ucla.edu/admissions/](https://master.stat.ucla.edu/admissions/)

Information is found here: <https://master.stat.ucla.edu/> And here: [https://master.stat.ucla.edu/
faq/](https://master.stat.ucla.edu/faq/)

The shell: why is it important??

to introduce the shell means to have a discussion about the structure of the computer, operating systems, file systems and history

the shell offers programmatic access to a computer's underlying parts, providing the ability to “do” data analysis on directories, on processes, and on their networks

as there are many flavors of the shell, it is the first time we come to a decision about choosing ‘tools’, about evaluating which shell is best for you and the shifting terrain of software development

The shell: why is it important?

as a practical matter, shell tools are an indispensable part for my own data science practice, data ‘cleaning’, data processing, exploratory analysis, automation; by design they let us deal with data on a scale that can be difficult from with R or Python

it also provides a low level link to other data platforms, data sources, and remote environments

The shell: how we'll learn

everyone needs access to the shell. everyone with a Mac and Linux already have one. those with Windows machines need something else..

there are instructions on the site for Windows users to get a version of it, but..

as we all have docker installed we can use an image that provides us all with the same environment for learning and exploration. not all functionality works in the image but overall will be useful for us (also we've seem a version in our docker Rstudio environment)

Operating Systems

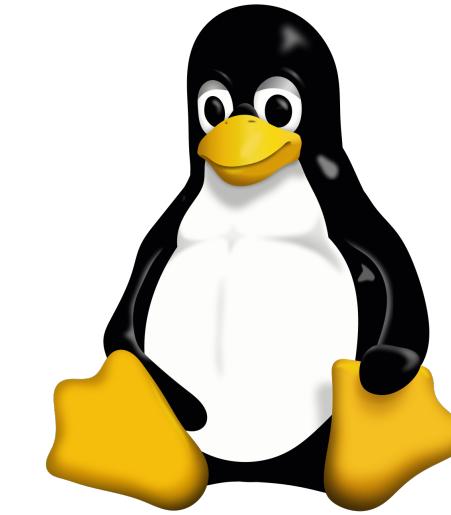
most devices that contain a computer of some kind will have an OS, they tend to emerge when the appliance will have to deal with new applications, complex user-input and possibly changing requirements on its function



Operating Systems

An operating system is a piece of software that organizes and controls hardware and other software so your computer behaves in a flexible but predictable way

maybe obviously to us all, Window, MacOS and Linux are those most commonly used for home computers. iOS and Android are most common on mobile devices



A history

In 1964, Bell Labs partnered with MIT and GE to create Multics
(Multiplexed Information and Computing Service)

“Such systems must run continuously and reliably 7 days a week, 24 hours a day in a way similar to telephone or power systems, and must be capable of meeting wide service demand from multiple man-machine interaction to the sequential processing of absentee-user jobs, from the use of the system with dedicated languages and subsystems to the programming of the stem itself”

A history

Bell Labs pulled out of the Multics project in 1969, a group of researchers at Bell Labs started work on Unica (uniplexed information and computing system) because initially it could only support one user; as the system matured it was renamed Unix, which isn't an acronym for anything

Ritchie simply says that Unix is a ‘somewhat treacherous pun on Multics’



The Unix filesystem

Multics brought about the first notion of a hierarchical file system; files were arranged in a tree structure allowing users to have control of them on areas

Unix began (more or less) as a file system and then an interactive shell emerged to let you examine its contents and perform basic operations

The kernel and the shell

the Unix kernel is the part of the operating system that carries out basic functions like accessing files, handing communication, and others

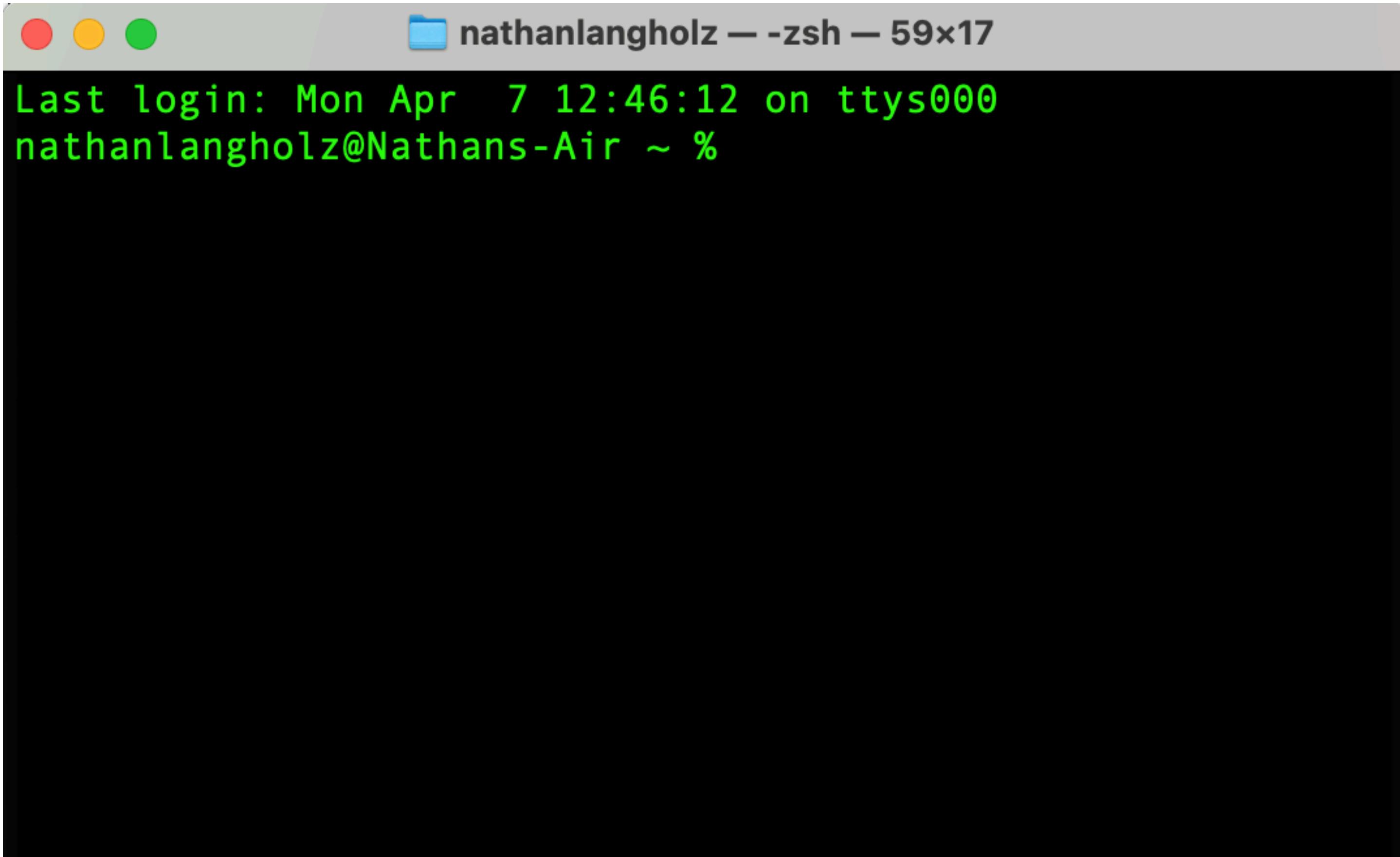
the Unix shell is a user interface to the kernel (keep in mind that Unix was designed for computer scientists and the interface is not optimized for novices)

Unix shells

A shell is a type of program called an interpreter, think of it as a text-based interface to the kernel

It operates in simple loop: it accepts a command, interprets it, executes the command and waits for another

The shell displays a prompt to tell you that it is ready to accept a command



```
Last login: Mon Apr 7 12:46:12 on ttys000
nathanlangholz@Nathans-Air ~ %
```

Not much going on but lots
of potential!

The % is the prompt

Unix shells

The shell is itself a program the the Unix operating system runs for you (a program is referred to as a process when its running)

The kernel manages many processes at once, many of which are the result of user commands (others provide services that keep the computer running)

Some commands are built into the shell, others have been added by users

Either way, the shell waits until the command is executed

Name of command

How hard the computer is thinking about it

Process ID

```
nathanlangholz — top — 94x41
Processes: 673 total, 3 running, 670 sleeping, 3054 threads
Load Avg: 3.58, 3.27, 3.37 CPU usage: 14.31% user, 7.4% sys, 78.63% idle
SharedLibs: 423M resident, 70M data, 16M linkedit.
MemRegions: 0 total, 0B resident, 1360K private, 747M shared.
PhysMem: 8079M used (2823M wired, 1238M compressor), 240M unused.
VM: 39T vsize, 5047M framework vsize, 168007066(886) swapins, 179820842(0) swapouts.
Networks: packets: 33358385/38G in, 13958509/13G out.
Disks: 35008277/1131G read, 14719577/855G written.

PID COMMAND %CPU TIME #TH #WQ #PORT MEM PURG CMPRS PGRP PPID STATE
155 WindowServer 23.2 13:57:40 14 6 4853 898M- 11M 219M- 155 1 sleeping
194 coreaudiod 10.2 51:11.67 12 4 2197 21M- 0B 11M- 194 1 sleeping
41829 top 9.2 00:09.40 1/1 0 31 9032K 0B 1036K 41829 41793 running
41677 com.apple.We 7.8 05:49.60 18/1 8 124 1487M+ 9676K 399M- 41677 1 running
0 kernel_task 4.8 10:06:26 261/4 0 0 588M- 0B 0B 0 0 running
11099 Google Drive 4.6 15:54:57 9 1 140 37M 0B 19M 1552 11095 sleeping
22553 com.apple.We 4.3 19:37.14 38 6 450- 178M- 6784K 149M 22553 1 sleeping
11103 Google Drive 3.0 09:13:04 12 1 332 27M 0B 24M 1552 11095 sleeping
22429 com.apple.Ap 2.2 17:18.66 3 2 300 776K 0B 272K 22429 1 sleeping
29078 com.apple.We 1.4 03:05.32 9 1 118- 253M- 0B 188M- 29078 1 sleeping
1474 Spotlight 1.4 03:02.20 5 2 503+ 88M 0B 62M- 1474 1 sleeping
22493 Safari 1.1 38:14.86 17 9 3001- 357M- 28K 299M- 22493 1 sleeping
40548 com.apple.We 1.0 03:46.15 11 4 108 503M 0B 435M- 40548 1 sleeping
1468 Terminal 1.0 01:02.29 11 4 540 68M+ 1792K 23M- 1468 1 sleeping
1558 Google Chrom 0.8 99:45.84 12 1 139+ 64M+ 0B 40M- 1464 1464 sleeping
113 mds 0.8 83:50.47 8 5 611 39M 0B 31M- 113 1 sleeping
22425 bluetoothd 0.8 32:33.17 11 5 617+ 9412K+ 20K 4740K- 22425 1 sleeping
22498 com.apple.We 0.7 48:21.17 10 5 191- 98M- 264K 38M- 22498 1 sleeping
5768 Google Chrom 0.6 47:45.51 14 1 401 379M+ 0B 310M- 1464 1464 sleeping
1464 Google Chrom 0.5 02:08:20 46 3 1746+ 231M 0B 166M- 1464 1 sleeping
41832 screencaptur 0.5 00:00.30 2 1 66 4160K+ 620K 0B 1470 1470 sleeping
566 sharingd 0.2 24:02.25 5 2 495+ 22M+ 0B 17M- 566 1 sleeping
86 logd 0.2 32:51.95 4 3 2363 22M 0B 27M- 86 1 sleeping
586 useractivity 0.2 01:00.43 4 3 114+ 2720K+ 0B 1376K- 586 1 sleeping
483 identityserv 0.2 11:55.14 6 2 826+ 21M+ 128K 18M- 483 1 sleeping
32877 audioanalyti 0.1 01:05.47 4 3 56 1724K+ 0B 448K- 32877 1 sleeping
166 lsd 0.1 04:20.25 2 1 140+ 6136K+ 0B 5304K- 166 1 sleeping
91 fsevents 0.1 32:59.10 14 1 199 3664K- 0B 1404K 91 1 sleeping
160 runningboard 0.1 36:11.74 7 6 944 7636K 0B 1332K 160 1 sleeping
99 powerd 0.1 12:25.51 4 3 155+ 3784K+ 0B 1872K- 99 1 sleeping
29451 com.docker.b 0.1 40:03.69 120 1 437 90M 0B 47M 29430 29430 sleeping
```

Result of the command **top**; this is a printout of all the processes running on your computer

Operating Systems

Process Management

Schedules jobs(formally referred to as processes) to be executed by the computer

Memory and storage management

Allocate space required for each running process in main memory (RAM) or in some other temporary location if space is tight and supervise the storage of data onto disk

Operating Systems

Device Management

A program called a driver translates data (files from the filesystem) into signals that

Application Programming Interface

An API (application programming interface) let programmers use functions of the computer and the operating system without having to know *how* something is done

User Interface

Finally, the operating system turns and looks at you; the UI is a program that defines how users interact with the computer; some are graphical (Windows is a GUI) and some are text-based (your Unix shell)

Unix Shell(s)

There are, in fact,
many different
kinds of Unix
shells

The table on the
right lists a few of
the most popular

KSH(1)	General Commands Manual	KSH(1)
NAME <code>ksh, ksh93</code> - KornShell, a command and programming language		
SYNOPSIS <code>ksh [abcefhikmnoprstuvwxyzBCDP] [-R file] [o option] ... [-] [arg ...] rksh [abcefhikmnoprstuvwxyzBCD] [-R file] [o option] ... [-] [arg ...]</code>		
DESCRIPTION <code>Ksh</code> is a command and programming language that executes commands read from a terminal or a file. <code>Rksh</code> is a restricted version of the command interpreter <code>ksh</code> ; it is used to set up login names and execution environments whose capabilities are more controlled than those of the standard shell. <code>Rpksh</code> is a profile shell version of the command interpreter <code>ksh</code> ; it is used to execute commands with the attributes specified by the user's profiles (see <code>pexec(1)</code>). See <code>Invocation</code> below for the meaning of arguments to the shell.		
TCSH(1)	General Commands Manual	TCSH(1)
NAME <code>tcsh</code> - C shell with file name completion and command line editing		
SYNOPSIS <code>tcsh [-bcdefFimnqstvVxX] [-Dname[=value]] [arg ...] tcsh -l</code>		
DESCRIPTION <code>tcsh</code> is an enhanced but completely compatible version of the Berkeley UNIX C shell, <code>csh(1)</code> . It is a command language interpreter usable both as an interactive login shell and a shell script command processor. It includes a command-line editor (see <code>The command-line editor</code>), programmable word completion (see <code>Completion and listing</code>), spelling correction (see <code>Spelling correction</code>), a history mechanism (see <code>History substitution</code>), job control (see <code>Jobs</code>) and a C-like syntax. The <code>NEW FEATURES</code> section describes major enhancements of <code>tcsh</code> over <code>csh(1)</code> . Throughout this manual, features of <code>tcsh</code> not found in most <code>csh(1)</code> implementations (specifically, the 4.4BSD <code>csh</code>) are labeled with '(+)', and features which are present in <code>csh(1)</code> but not usually documented are labeled with '(u)'.		
BASH(1)	General Commands Manual	BASH(1)
NAME <code>bash</code> - GNU Bourne-Again SHell		
SYNOPSIS <code>bash [options] [command_string file]</code>		
COPYRIGHT <code>Bash</code> is Copyright (C) 1989-2013 by the Free Software Foundation, Inc.		
DESCRIPTION <code>Bash</code> is an <code>sh</code> -compatible command language interpreter that executes commands read from the standard input or from a file. <code>Bash</code> also incorporates useful features from the <code>Korn</code> and <code>C</code> shells (<code>ksh</code> and <code>csh</code>). <code>Bash</code> is intended to be a conformant implementation of the Shell and Utilities portion of the IEEE POSIX specification (IEEE Standard 1003.1). <code>Bash</code> can be configured to be POSIX-conformant by default.		
ZSH(1)	General Commands Manual	ZSH(1)
NAME <code>zsh</code> - the Z shell		
OVERVIEW Because <code>zsh</code> contains many features, the <code>zsh</code> manual has been split into a number of sections:		
<code>zsh</code> Zsh overview (this section) <code>zshroadmap</code> Informal introduction to the manual <code>zshmisc</code> Anything not fitting into the other sections <code>zshepn</code> Zsh command and parameter expansion <code>zshparam</code> Zsh parameters <code>zshoptions</code> Zsh options <code>zshbuiltins</code> Zsh built-in functions <code>zshzle</code> Zsh command line editing <code>zshcomwid</code> Zsh completion widgets <code>zshcompsys</code> Zsh completion system <code>zshcomctl</code> Zsh completion control <code>zshmodules</code> Zsh loadable modules <code>zshcalsys</code> Zsh built-in calendar functions		

Why the choices?

A shell program was originally meant to take commands, interpret them and then execute some operation

Inevitably, one wants to collect a number of these operations into programs that execute compound tasks at the same time you want to make interaction on the command line as easy as possible (a history mechanism, editing capabilities and so on)

The original Bourne shell is ideal for programming; the C-shell and its variants are good for interactive use; the Korn shell is a combination of both



Steve Bourne, creator of sh, 2005 (source wikipedia)

And while we're at it

unix itself comes in different flavors; the 1980s saw an incredible proliferation of Unix versions, somewhere around 100 (System V, AIX, Berkely BSD, SunOS, Linux, ...)

vendors provided (diverging) version of Unix, optimized for their own computer architectures and supporting different features

despite the diversity it was still easier to “port” applications between versions of Unix than it was between different proprietary OS

A few common commands

First, commands to explore your file system, walk through directories and list files

`pwd`, `ls`, `cd`

`mkdir`, `rmdir`

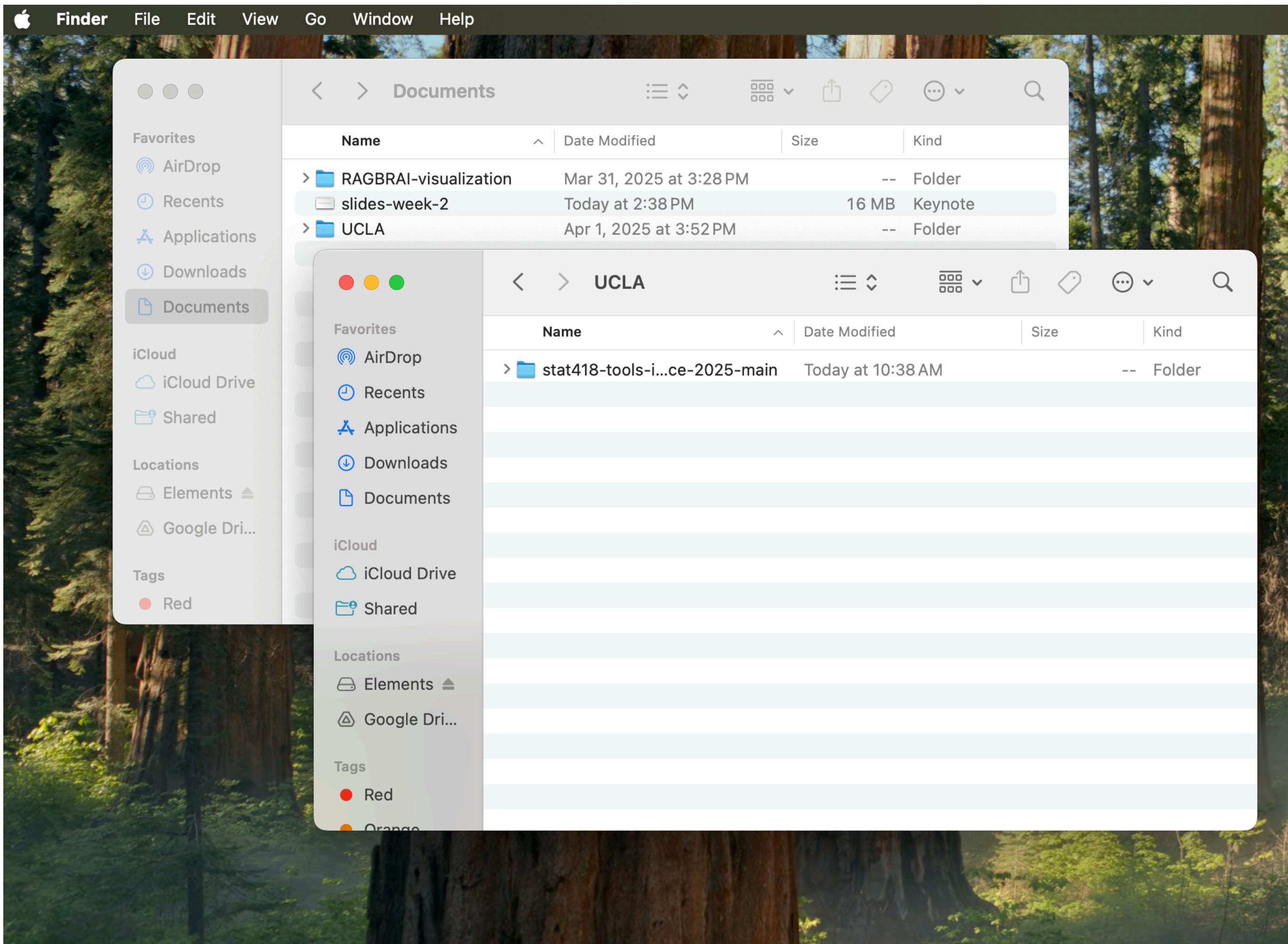
`cp`, `mv`, `rm`

The screenshot shows a terminal window titled "Documents --zsh-- 82x31". The terminal output is as follows:

```
Last login: Mon Apr  7 14:37:57 on ttys001
nathanlangholz@Nathans-Air ~ % pwd
/Users/nathanlangholz
nathanlangholz@Nathans-Air ~ % cd Documents
nathanlangholz@Nathans-Air Documents % ls
RAGBRAI-visualization  UCLA
nathanlangholz@Nathans-Air Documents % ls -l
total 33016
drwxr-xr-x@ 9 nathanlangholz  staff      288 Mar 31 15:28 RAGBRAI-visualization
drwxr-xr-x  4 nathanlangholz  staff      128 Apr  1 15:52 UCLA
-rw-r--r--@ 1 nathanlangholz  staff  15982151 Apr  7 14:38 slides-week-2.key
nathanlangholz@Nathans-Air Documents %
```

Annotations with arrows point to specific parts of the terminal output:

- A line labeled "Present working directory" points to the line "pwd" in the history.
- A line labeled "Change directory" points to the line "cd Documents".
- A line labeled "List" points to the line "ls".
- A line labeled "Long list" points to the line "ls -l".



Another view of the filesystem; here your Mac will display directories as folders and of course you navigate by clicking rather than typing commands

Read, write, execute permissions

Your username

The group you belong to that owns the file

The file's size in bytes

The file's creation date

Shorthand for your present working directory

Shorthand for the directory one level above

Filename

```
NA-nlanghol-A02:stat418-tools-in-datascience nlangholz$ NA-nlanghol-A02:stat418-tools-in-datascience nlangholz$ NA-nlanghol-A02:stat418-tools-in-datascience nlangholz$ NA-nlanghol-A02:stat418-tools-in-datascience nlangholz$ ls -al total 64 drwxr-xr-x 12 nlangholz staff 384 Apr 9 12:49 . drwx-----+ 15 nlangholz staff 480 Apr 9 17:30 .. -rw-r--r-- 1 nlangholz staff 8196 Apr 9 11:50 .DS_Store drwxr-xr-x 5 nlangholz staff 160 Apr 9 12:49 .Rproj.user drwxr-xr-x 16 nlangholz staff 512 Apr 9 12:51 .git -rw-r--r-- 1 nlangholz staff 653 Apr 9 11:50 .gitignore -rw-r--r-- 1 nlangholz staff 1057 Apr 9 11:50 LICENSE -rw-r--r-- 1 nlangholz staff 4803 Apr 9 11:50 README.md -rw-r--r-- 1 nlangholz staff 205 Apr 9 12:49 stat418-tools-in-datascience.Rproj drwxr-xr-x 4 nlangholz staff 128 Apr 9 11:50 week-1 drwxr-xr-x 4 nlangholz staff 128 Apr 9 12:13 week-2 drwxr-xr-x 3 nlangholz staff 96 Apr 9 11:50 week-3 NA-nlanghol-A02:stat418-tools-in-datascience nlangholz$
```

Kinds of Files

What you'll notice right away is that there are different types of files having different permissions

Unix filesystem conventions places (shared, commonly used) executable files in places like /usr/bin or /usr/local/bin

Different files are opened by different kinds of programs, in OSX, there is a command called open that decides which program to use

Permission bits

Unix can support many users on a single system and each user can belong to one or more groups

every file in a Unix filesystem is owned by some user and one of that user's groups; each file also has a set of permissions specifying which users can

r: read or w: write (modify) or x: execute

the file; these are specified with three 'bits' and we need three sets of bits to define what the user can do, what their group (that owns the file) can do and what others can do

the command **chmod** changes the permissions on a file but we will leave that for you to discover on your own

Permission bits

The type of file

```
drwxr-xr-x 12 nlangholz staff  
drwx-----+ 15 nlangholz staff  
-rw-r--r-- 1 nlangholz staff  
drwxr-xr-x 5 nlangholz staff  
drwxr-xr-x 16 nlangholz staff  
-rw-r--r-- 1 nlangholz staff  
-rw-r--r-- 1 nlangholz staff  
-rw-r--r-- 1 nlangholz staff  
-rw-r--r-- 1 nlangholz staff  
drwxr-xr-x 4 nlangholz staff  
drwxr-xr-x 4 nlangholz staff  
drwxr-xr-x 3 nlangholz staff
```

What you can do to
the file

What the owning
group can do

What others can do

Although we saw a terminal environment in our Rstudio docker container, now let's get another docker image that has some nice data science specific code.

we will use the docker image from the book [Data Science at the Command Line](#) by Jeroen Janssens

<https://www.datascienceatthecommandline.com/index.html>

First, we need to pull the image using the following

```
docker pull datasciencetoolbox/dsatcl2e
```

Data Science at the Command Line

To run the image

```
docker run --rm -it datasciencetoolbox/dsatcl2e
```

To leave the environment simply type **exit**

again we didn't mount a local volume so we have no connection to our computer

Now to mount our local directories, cd to desired directory, then use
on a Mac

```
docker run --rm -it -v`pwd`:~/data datasciencetoolbox/dsatcl2e
```

in Windows command line

```
C:\> docker run --rm -it -v "%cd%":~/data datasciencetoolbox/dsatcl2e
```

in Windows powershell

```
PS C:\> docker run --rm -it -v ${PWD}:~/data datasciencetoolbox/  
dsatcl2e
```

To make sure everything is running correctly, type

```
cowsay "ready to go!"
```

This environment we now have is a bash shell running on a Linux operating system.

Note: this only works in this environment not in all terminal windows

```
nlangholz — docker run --rm -it -v/Users/nlangholz:/data datascienceworkshops/data-science-at-the-command-line...
Last login: Sun Apr 14 20:50:50 on ttys001
You have new mail.
[NA-nlanghol-A02:~ nlangholz$ docker run --rm -it -v`pwd`:~/data datascienceworkshops/data-science-at-the-command-
line
[[/data]$ echo 'Hello world' | wc
      1      2     12
[/data]$
```

Redirecting output with “|”

takes output from one command and submits it as input to the next command

```
nlangholz — docker run --rm -it -v/Users/nlangholz:/data datascienceworkshops/data-science-at-the-command-line...
Last login: Sun Apr 14 22:59:39 on ttys002
You have new mail.
[NA-nlanghol-A02:~ nlangholz$ docker run --rm -it -v`pwd`:~/data datascienceworkshops/data-science-at-the-command-
line
[/data]$ echo 'Hello world' | wc
      1      2     12
[/data]$ ls
'20 datagroups LDA viz.html'  Untitled.ipynb
'20 datagroups docvec.html'   Untitled1.ipynb
Applications                  crm_customer_23JAN2018.csv
Desktop                         gensim-data
Documents                        nltk_data
Downloads                       rbtv-d2v-v1-dbbow.model
HD_log.txt                      rbtv-d2v-v2-dbbow.model
Library                          rbtv_d2v_1.model
LightGBM                         rbtv_d2v_v1.model
Movies                            spark
Music                            tensorflow
New_York_Red_Bulls.xlsx          tsne-new-values-v1.npy
Pictures                         tsne_new_values_v0.npy
Pipfile                           verse_gapminder.tar
Public                            verse_gapminder2.tar
[/data]$ cd ..
[/$] ls
bin  data  dev  drake.log  etc  home  lib  media  mnt  proc  root  run  sbin  srv  sys  tmp  usr  var
[/$] cd home/data/
[/home/data]$ head -n 3 ch02/data/movies.txt
Matrix
Star Wars
Home Alone
[/home/data]$
```

Listing pwd, then
changing to directory one
level up to access docker
container book contents

Listing the first 3 movies
from the file movies.txt

```
nlangholz — docker run --rm -it -v/Users/nlangholz:/data datascienceworkshops/data-science-at-the-command-lin...
Last login: Sun Apr 14 23:06:01 on ttys003
You have new mail.
[NA-nlanghol-A02:~ nlangholz$ docker run --rm -it -v`pwd`:~/data datascienceworkshops/data-science-at-the-command]
-line
[[/data]$ seq 5
1
2
3
4
5
[[/data]$ seq 30 | grep 3
3
13
23
30
[[/data]$ seq 200 | grep 3 | wc -l
38
[/data]$
```

You can connect pipes and have data stream from process to process

“Grep was invented for me. I was making a program to read text aloud through a voice synthesizer. As I invented phonetic rules I would check Webster's dictionary for words on which they might fail. For example, how do you cope with the digraph 'ui', which is pronounced many different ways: 'fruit', 'guile', 'guilty', 'anguish', 'intuit', 'beguine'? I would break the dictionary up into pieces that fit in ed's limited buffer and use a global command to select a list. I would whittle this list down by repeated scannings with ed to see how each proposed rule worked.”

The process was tedious, and terribly wasteful, since the dictionary had to be split (one couldn't afford to leave a split copy on line). Then ed copied each part into /tmp, scanned it twice to accomplish the g command, and finally threw it away, which takes time too.”

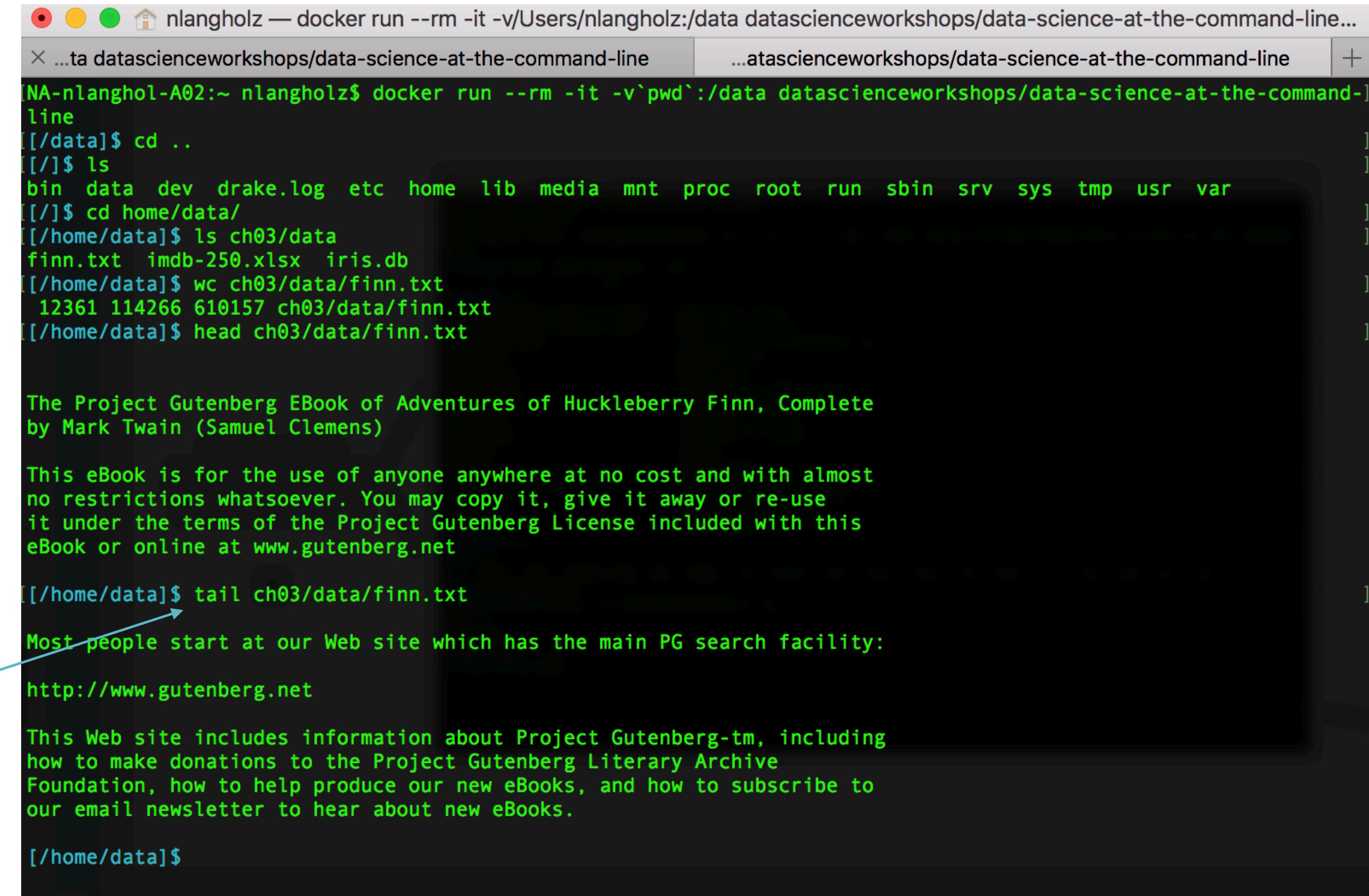
- Doug McIlroy, Adjunct Professor of Computer Science Dartmouth college

Chapter 9, On the Early History and Impact of Unix Tools to Build the Tools for a New Millennium
<http://www.columbia.edu/~rh120/ch001j.c11>

“One afternoon I asked Ken Thompson if he could lift the regular expression recognizer out of the editor and make a one-pass program to do it. He said yes. The next morning I found a note in my mail announcing a program named grep. It worked like a charm. When asked what that funny name meant, Ken said it was obvious. It stood for the editor command that it simulated, g/re/p (global regular expression print).”

- Doug McIlroy, Adjunct Professor of Computer Science Dartmouth college

Chapter 9, On the Early History and Impact of Unix Tools to Build the Tools for a New Millennium
<http://www.columbia.edu/~rh120/ch001j.c11>



```
nlangholz — docker run --rm -it -v/Users/nlangholz:/data datascienceworkshops/data-science-at-the-command-line...
...ta datascienceworkshops/data-science-at-the-command-line ...atascienceworkshops/data-science-at-the-command-line +
```

```
[NA-nlanghol-A02:~ nlangholz$ docker run --rm -it -v`pwd`:/data datascienceworkshops/data-science-at-the-command-line
line
[/data]$ cd ..
[/$] ls
bin data dev drake.log etc home lib media mnt proc root run sbin srv sys tmp usr var
[/$] cd home/data/
[/home/data]$ ls ch03/data
finn.txt imdb-250.xlsx iris.db
[/home/data]$ wc ch03/data/finn.txt
 12361 114266 610157 ch03/data/finn.txt
[/home/data]$ head ch03/data/finn.txt

The Project Gutenberg EBook of Adventures of Huckleberry Finn, Complete
by Mark Twain (Samuel Clemens)

This eBook is for the use of anyone anywhere at no cost and with almost
no restrictions whatsoever. You may copy it, give it away or re-use
it under the terms of the Project Gutenberg License included with this
eBook or online at www.gutenberg.net

[/home/data]$ tail ch03/data/finn.txt
Most people start at our Web site which has the main PG search facility:
http://www.gutenberg.net

This Web site includes information about Project Gutenberg-tm, including
how to make donations to the Project Gutenberg Literary Archive
Foundation, how to help produce our new eBooks, and how to subscribe to
our email newsletter to hear about new eBooks.

[/home/data]$
```

Similar to head we have
'tail' end of the document

```
nlangholz — docker run --rm -it -v/Users/nlangholz:/data datascienceworkshops/data-science-at-the-command-line...
...ta datascienceworkshops/data-science-at-the-command-line ...atascienceworkshops/data-science-at-the-command-line +
[~/data]$ ls ch05/data
Iris-setosa.csv    Iris-virginica.csv   iris.csv      names-commma.csv  tips.csv    wiki.html
Iris-versicolor.csv alice.txt          irismeta.csv  names.csv       users.json
[~/data]$ head ch05/data/iris.csv
sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
[~/data]$
[~/data]$
[~/data]$
[~/data]$
[~/data]$
[~/data]$
[~/data]$
[~/data]$
[~/data]$ cut -d"," -f1 ch05/data/iris.csv | head
sepal_length
5.1
4.9
4.7
4.6
5.0
5.4
4.6
5.0
4.4
[~/data]$
```

Irises again 🔥

Cut by delimiter “,” and take the first field

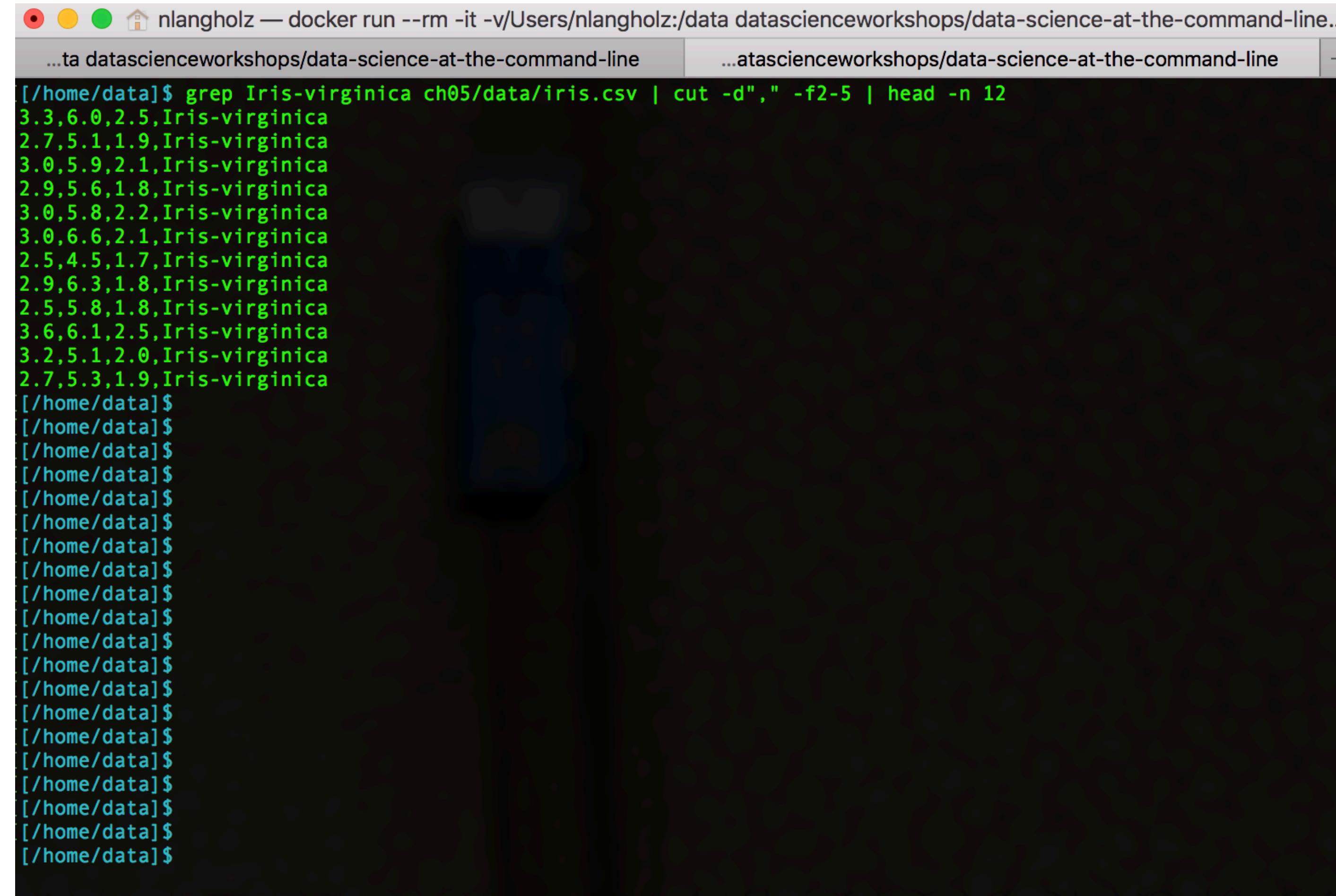
Cut by delimiter “,” and take the fifth field, sort, take unique values and finally get the word count

There are four unique lines in Iris ‘species’...

but aren't there only 3 different species?!

Well, yes, since there is no notion of header (or column names) here

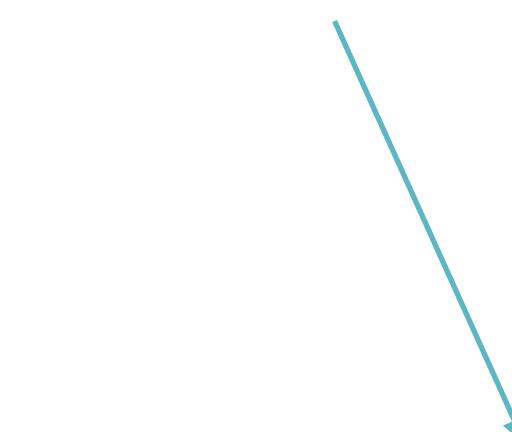
in reality there are 50 of each species and then the 1 row of ‘species’ at the top



The screenshot shows a macOS terminal window with two tabs. The active tab displays a command-line pipeline:

```
[/home/data]$ grep Iris-virginica ch05/data/iris.csv | cut -d"," -f2-5 | head -n 12
```

The output of this command is a list of 12 rows from the Iris dataset, all belonging to the 'Iris-virginica' species. The columns are separated by commas and include numerical values for Sepal Length, Sepal Width, Petal Length, and Petal Width.



You can connect pipes and have data process to process

Sending output to a file with “>”

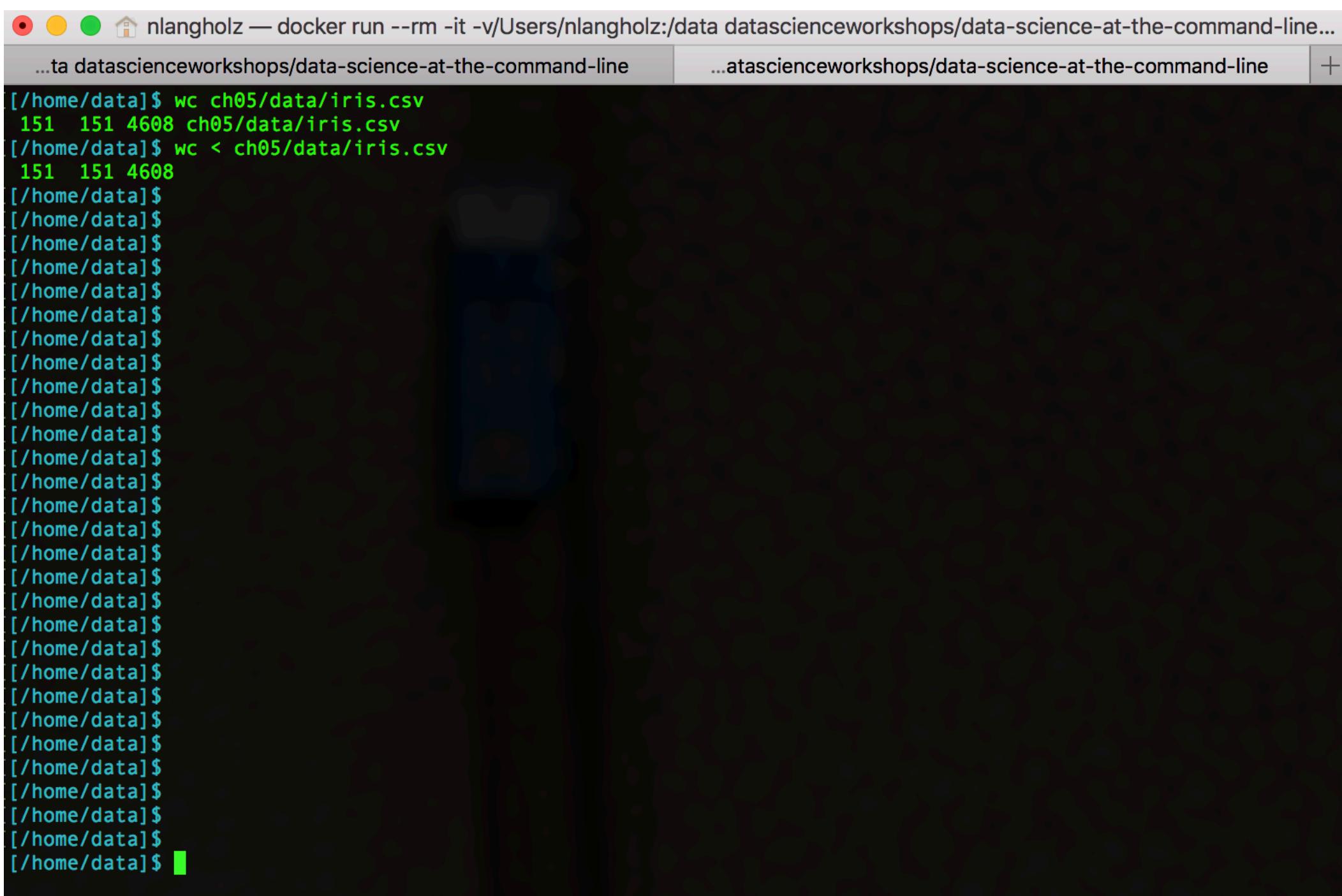
With this form of redirection, we take a stream of processed data and store it in a file

Example

```
cut -d"," -f5 ch05/data/iris.csv > species
```

Taking input from a file with “<”

With this form of redirection, we create an input stream from a file



The screenshot shows a terminal window with two tabs. The active tab displays the command `wc < ch05/data/iris.csv` being run. The output of this command is shown in green text:

```
[/home/data]$ wc < ch05/data/iris.csv
151 151 4608 ch05/data/iris.csv
```

The terminal window has a dark background and light-colored text. The tabs are labeled "...ta datascienceworkshops/data-science-at-the-command-line" and "...atascienceworkshops/data-science-at-the-command-line".

Read the man and help pages!

If the command uniq is unfamiliar you can look up its usage

`man uniq`

or

`uniq --help`

```
nlangholz — docker run --rm -it -v/Users/nlangholz:/data datascienceworkshops/data-science-at-the-command-lin...
Last login: Sun Apr 14 23:38:27 on ttys001
You have new mail.
[NA-nlanghol-A02:~ nlangholz$ docker run --rm -it -v`pwd`:~/data datascienceworkshops/data-science-at-the-command]
-line
[[/data]$ man cat | head -n 20
CAT(1)          User Commands          CAT(1)

NAME
    cat - concatenate files and print on the standard output

SYNOPSIS
    cat [OPTION]... [FILE]...

DESCRIPTION
    Concatenate FILE(s) to standard output.

    With no FILE, or when FILE is -, read standard input.

    -A, --show-all
        equivalent to -vET

    -b, --number-nonblank
        number nonempty output lines, overrides -n
[/data]$ 
```

man, short for manual, contain info for most command-line tools

here I've just shown the first 20 lines

```
nlangholz — docker run --rm -it -v/Users/nlangholz:/data datascienceworkshops/data-science-at-the-command-lin...
equivalent to -vET

-b, --number-nonblank
    number nonempty output lines, overrides -n
[[/data]$ help cd | head -n 30
cd: cd [-L|[-P [-e]] [-@]] [dir]
    Change the shell working directory.

    Change the current directory to DIR.  The default DIR is the value of the
    HOME shell variable.

The variable CDPATH defines the search path for the directory containing
DIR.  Alternative directory names in CDPATH are separated by a colon (:).
A null directory name is the same as the current directory.  If DIR begins
with a slash (/), then CDPATH is not used.

If the directory is not found, and the shell option `cachable_vars' is set,
the word is assumed to be a variable name.  If that variable has a value,
its value is used for DIR.

Options:
-L      force symbolic links to be followed: resolve symbolic
       links in DIR after processing instances of `..'
-P      use the physical directory structure without following
       symbolic links: resolve symbolic links in DIR before
       processing instances of `..'
-e      if the -P option is supplied, and the current working
       directory cannot be determined successfully, exit with
       a non-zero status
-@      on systems that support it, present a file with extended
       attributes as a directory containing the file attributes

The default is to follow symbolic links, as if '-L' were specified.
`..' is processed by removing the immediately previous pathname component
back to a slash or the beginning of DIR.
[/data]$
```

Not every command-line tool has a man page. For some shell builtins there is only a help page again I've shown only the top 30 lines

```
nlangholz — docker run --rm -it -v/Users/nlangholz:/data datascienceworkshops/data-science-at-the-command-line
```

```
[//]$  
[//]$  
[//]$  
[//]$  
[//]$  
[//]$  
[//]$  
[//]$  
[//]$  
[//]$  
[//]$  
[//]$ cd data/Documents/Egnyte/Private/nlangholz/UCLA/stat418-tools-in-datascience/week-2/hw1/homework-submissions/  
[//] B ]$  
[//]$  
[//]$  
[//]$  
[//]$ ls  
hw1_starter.sh  
[//]$ cat hw1_starter.sh  
#!/bin/bash  
  
# this fetches the file and counts the number of lines. Add to this!  
  
curl -s http://users.csc.tntech.edu/~elbrown/access_log.bz2 | bunzip2 - | wc -l  
[//]$ bash hw1_starter.sh  
234794
```

There is a lot packed into this slide

cat returns the entirety of the file in the terminal

what does bash do?

change directory all the way to the desired file location

curl goes to the specified url to download the zip file and bunzip2 will unzip

Running a shell script

There are two ways to run a shell script; you can either execute it within a new shell or make the file executable. Executing in a new shell using:

```
$ bash <filename.sh>
```

This should explain the .sh suffix we used for the homework starter filename; this naming convention will help you (and others) recognize this as a shell script

Making a file executable means making it become just like any command Unix knows. This requires changing file permissions which we saw briefly earlier but we won't cover in this lecture

What the web log
actually looks like...

```
nlangholz — docker run --rm -it -v/Users/nlangholz:/data datascienceworkshops/data-science-at-the-comm...
...atascienceworkshops/data-science-at-the-command-line ...cienceworkshops/data-science-at-the-command-line +
[[//data/Documents/Egnyte/Private/nlangholz/UCLA/stat418-tools-in-datas...ns]$ curl -s http://users.csc.tntech.edu/~elbrown/access_log.bz2 | bunzip2 - | head
74.38.188.38 - - [01/Feb/2007:00:00:32 -0600] "GET /~sjcrook/westciv/05.php HTTP/1.1" 200 9996 "http://users.csc.tntech.edu/~sjcrook/westciv/index.php" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
74.38.188.38 - - [01/Feb/2007:00:00:32 -0600] "GET /~sjcrook/westciv/default.css HTTP/1.1" 304 0 "http://users.csc.tntech.edu/~sjcrook/westciv/05.php" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
74.38.188.38 - - [01/Feb/2007:00:00:33 -0600] "GET /~sjcrook/westciv/img/bg.jpg HTTP/1.1" 304 0 "http://users.csc.tntech.edu/~sjcrook/westciv/default.css" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
74.38.188.38 - - [01/Feb/2007:00:00:33 -0600] "GET /~sjcrook/westciv/img/RenaissanceItaly.jpg HTTP/1.1" 304 0 "http://users.csc.tntech.edu/~sjcrook/westciv/05.php" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
74.38.188.38 - - [01/Feb/2007:00:00:33 -0600] "GET /~sjcrook/westciv/img/boot.gif HTTP/1.1" 304 0 "http://users.csc.tntech.edu/~sjcrook/westciv/05.php" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
74.38.188.38 - - [01/Feb/2007:00:00:33 -0600] "GET /~sjcrook/westciv/img/nijaturtles.gif HTTP/1.1" 304 0 "http://users.csc.tntech.edu/~sjcrook/westciv/05.php" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
74.38.188.38 - - [01/Feb/2007:00:00:34 -0600] "GET /~sjcrook/westciv/img/bgcontainer.jpg HTTP/1.1" 304 0 "http://users.csc.tntech.edu/~sjcrook/westciv/default.css" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
74.38.188.38 - - [01/Feb/2007:00:00:34 -0600] "GET /~sjcrook/westciv/img/gfx.jpg HTTP/1.1" 304 0 "http://users.csc.tntech.edu/~sjcrook/westciv/default.css" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
74.38.188.38 - - [01/Feb/2007:00:00:34 -0600] "GET /~sjcrook/westciv/img/bgcontent.gif HTTP/1.1" 304 0 "http://users.csc.tntech.edu/~sjcrook/westciv/default.css" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
74.38.188.38 - - [01/Feb/2007:00:00:34 -0600] "GET /~sjcrook/westciv/img/pattern.gif HTTP/1.1" 304 0 "http://users.csc.tntech.edu/~sjcrook/westciv/default.css" "Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.8.0.9) Gecko/20061206 Firefox/1.5.0.9"
[//data/Documents/Egnyte/Private/nlangholz/UCLA/stat418-tools-in-datas...ns]$
```

Combined log format (for homework)

IP address

Identity

Userid

Date

Request

Status

Bytes

Referrer

Agent

Accessing a URL

When accessing a URL (uniform resource locator) through your browser the data that is being downloaded can be interpreted; html as a website, MP3 as played music, pdf automatically downloaded

When cURL is used to access a URL the data is downloaded and is printed to standard output. As we have seen we can simply specify as an argument

```
$ curl -s http://www.gutenberg.org/files/76/76-0.txt | head -n 10
```

This downloads the book Adventures of Huckleberry Finn by Mark Twain from Project Gutenberg that we saw before. The `-s` command stands for *silent* so a progress meter is disabled so we can pipe to another command

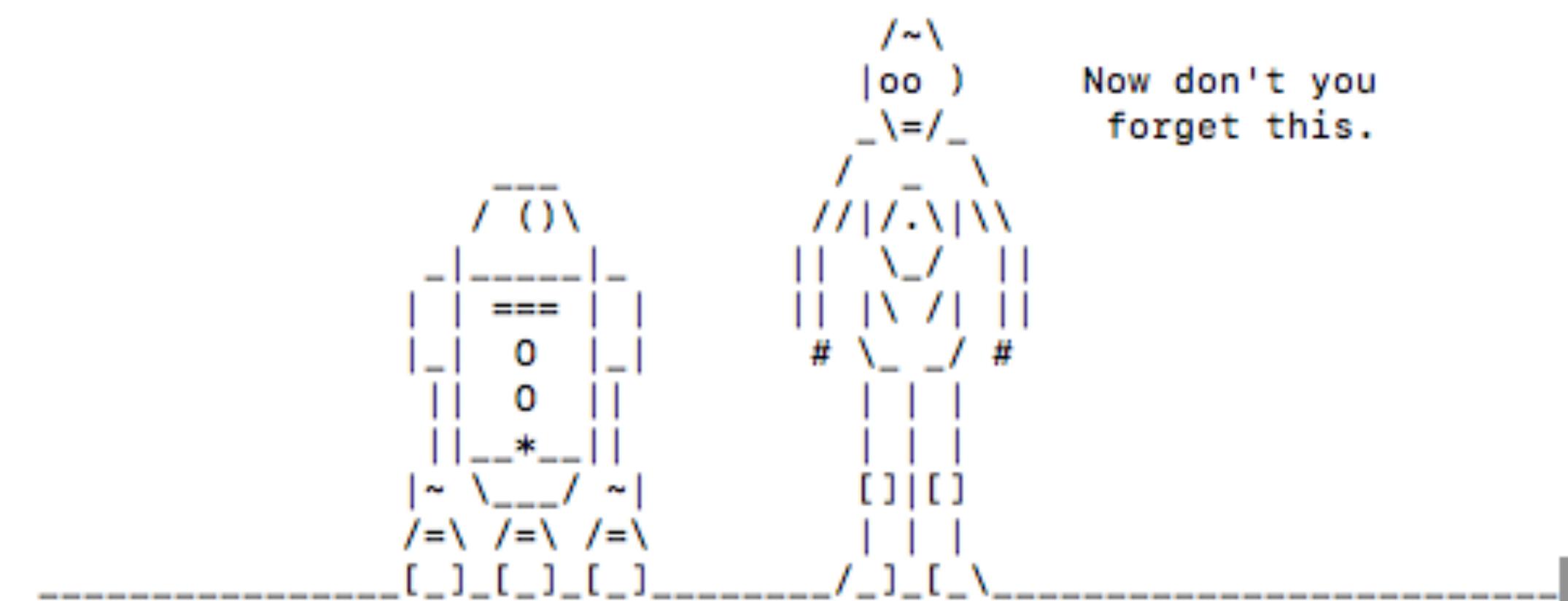
Saving a URL

Of course we can continue to combine our commands to save a file

```
$ curl -s http://www.gutenberg.org/files/76/76-0.txt > data/finn.txt
```

Don't need to specify **-s** in this case

Star Wars: Fun, but... useless?



try

```
$ nc towel.blinkenlights.nl 23
```

Web Scraping (...still in the shell)

Scraping the New York Times

The New York Times is one of the most well-known, reputable, widely circulated news outlets worldwide. They have the complete archive of all their articles available dating back to 1851. Obviously this is a huge wealth of information that we can access through their developers API.

Why does NYT (or anyone) provide an API? In their words:

3. Why are you offering APIs?

Like many organizations, we hope to encourage innovation through collaboration. When you build applications, create mashups and otherwise reveal the potential of our data, we learn more about what our readers want and gain insight into how news and information can be reimagined. We're hoping you'll show us what's next for The Times.

But we also have a simpler, more compelling reason: journalism. To inform the public or tell a story, we use articles, photos, videos, interactive graphics, slideshows and more. Data has always been the primary force behind those features, and now it can become a feature in its own right. Our APIs help us fulfill the newspaper's journalistic mission by putting more information in the hands of the public — and they also expand that mission by giving users the ability to find and tell their own stories.

Tuesday, April 16, 2019

The New York Times

ENGLISH ESPAÑOL 中文

SUBSCRIBE NOW LOG IN

World U.S. Politics N.Y. Business Opinion Tech Science Health Sports Arts Books Style Food Travel Magazine T Magazine Real Estate Video

The Daily Listen to 'The Daily' The rise and fall of Carlos Ghosn.

In the 'DealBook' Newsletter French billionaires vow to help rebuild Notre-Dame.

The Book Review Podcast Ruth Reichl dishes on Gourmet, and Emily Bazelon talks about "Charged."

S&P 500 +0.05% ↑ Dow +0.26% ↑ Nasdaq +0.30% ↑ 60°F 63° 52° Santa Monica, CA

NOTRE-DAME FIRE

Notre-Dame Is Found to Be Structurally Sound as Investigation Begins

The blaze destroyed the roof and spire, leaving three holes in the ceiling, officials said as they began "a long and complex investigation."

President Emmanuel Macron said he hoped the cathedral could be rebuilt within five years — though one expert said it could take three times that long.

16m ago 618 comments

What the Fire Reveals About the Soul of France

The shock of the potential loss has raised difficult questions about religion, secularism and European identity.

1h ago 66 comments

What We Know and Don't Know About the Fire

The fire was extinguished Tuesday morning. No one was killed, officials said, but a firefighter and two police officers were injured.

At least 600 million euros, or more than \$675 million, has already been promised to a rebuilding effort.

1h ago

2020 PRESIDENTIAL ELECTION

9 Takeaways From a Look Inside the 2020 Money Race

Bernie Sanders is the money leader. Elizabeth Warren is spending big. Pete Buttigieg emerged out of nowhere. And (almost) everyone is buying gobs of Facebook ads.

1h ago

Opinion

Sahil Chinoy 'Oh My God, That Is Unbelievable'

We used a publicly available camera stream to demonstrate how easy it is to track people without their knowledge.

2h ago

Kriston Capps Another Reason to Hate Hudson Yards

The billion-dollar luxury real estate project in Manhattan is exploiting a cash-for-visas program meant for the poor.

1h ago

Serge Schmemann Notre-Dame Survived the French Revolution. Why Not a Fire?

Agnès C. Poirier I Stood Vigil Last Night, Waiting for a Miracle

Charlie Warzel Take Back Control of Your Digital Life

Paul Krugman Republicans Are the Real Extremists

Tera W. Hunter When Slaveowners Got Reparations

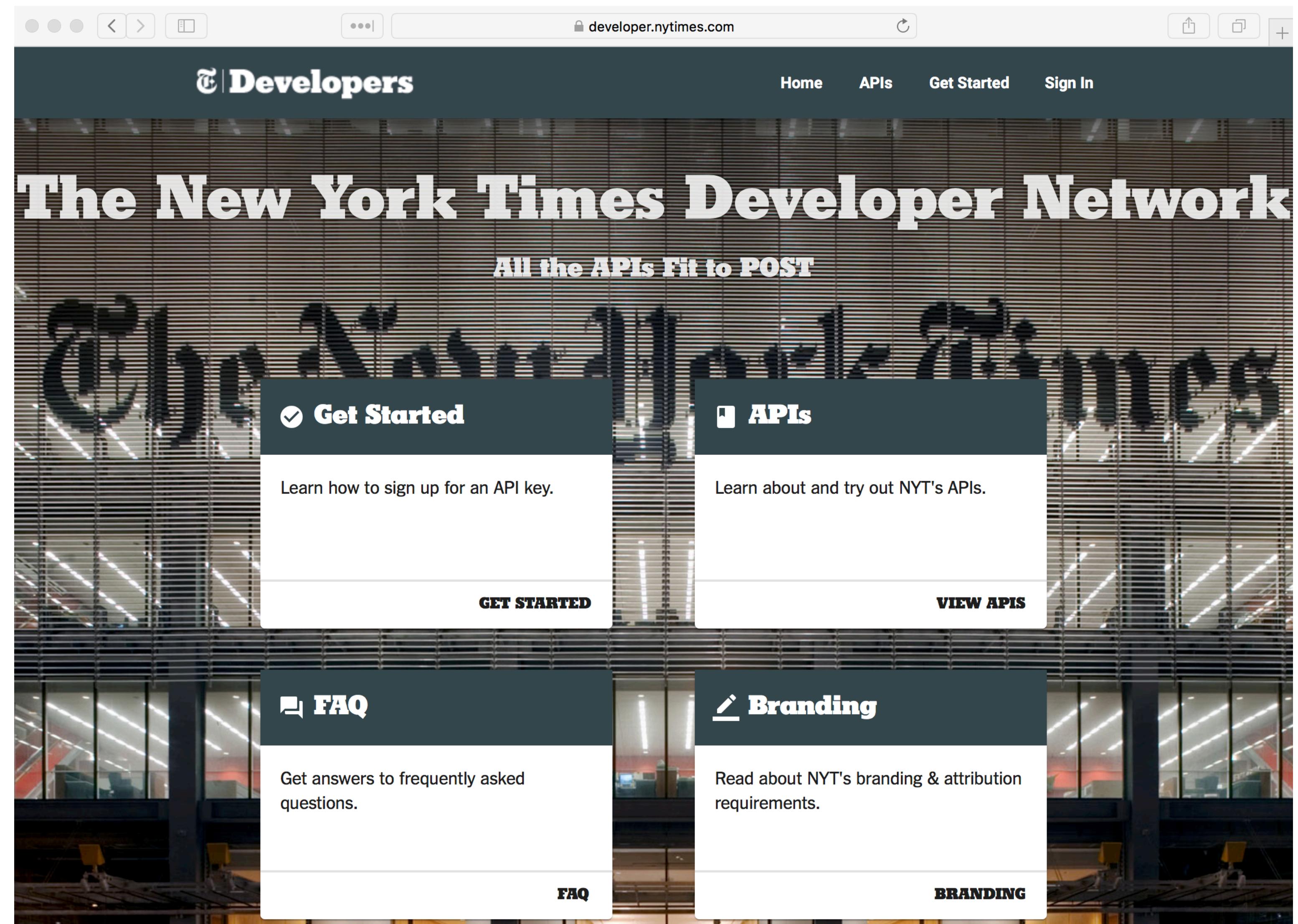
Editors' Picks

With the sad news yesterday of the Notre-Dame fire, today's NYT front page is full of articles about the cathedral.

I was curious as to how many articles are typically written about Notre-Dame given that it sees about ~12 millions visitors a year (according to the Paris Convention and Visitors Bureau) how much of an increase we saw as a result

This is the New York Times Developer Network

If you want to follow along you will need to make an account to get your own API key



You will need to create an app; name doesn't matter

We will be using the Article Search API, but it is a similar process to use any of the others

The screenshot shows a web browser window for developer.nytimes.com. The title bar says "developer.nytimes.com". The main header has "T|Developers" and a user dropdown for "nathan.langholz@redbull.com". Below the header, it says "New App". On the left, there's a sidebar with "Overview" and "APIs". Under "Overview", there's a form with "App Name *" set to "search-articles" and a "Description" field. On the right, under "APIs", there's a grid of six API cards:

API	Description
Movie Reviews API	Search for movie reviews.
Most Popular API	Get most emailed, shared, or viewed articles.
Times Wire API	Real-time feed of NYT article publishes.
Article Search API	Search for New York Times articles.
Community API	Get user comments. (DEPRECATED)
Books API	Get NYT Best Sellers Lists and lookup book reviews.

FAQ

1. What are the Times APIs?

Our APIs ([Application Programming Interfaces](#)) allow you to programmatically access New York Times data for use in your own applications. Our goal is to facilitate a wide range of uses, from custom link lists to complex visualizations. Why just read the news when you can hack it?

NYT currently has ten public APIs: Archive, Article Search, Books, Community, Geographic, Most Popular, Semantic, Times Newswire, TimesTags, and Top Stories.

2. Who is the intended audience for the Times APIs?

We've designed our APIs for the web developer community, but all non-commercial users are welcome. See our [Terms of Use](#) for more information.

3. Why are you offering APIs?

Like many organizations, we hope to encourage innovation through collaboration. When you build applications, create mashups and otherwise reveal the potential of our data, we learn more about what our readers want and gain insight into how news and information can be reimagined. We're hoping you'll show us what's next for The Times.

But we also have a simpler, more compelling reason: journalism. To inform the public or tell a story, we use articles, photos, videos, interactive graphics, slideshows and more. Data has always been the primary force behind those features, and now it can become a feature in its own right. Our APIs help us fulfill the newspaper's journalistic mission by putting more information in the hands of the public — and they also expand that mission by giving users the ability to find and tell their own stories.

4. What kinds of data can I access with Times APIs?

Please see our APIs page for the current list of available APIs.

5. How do I use the Times APIs?

Our APIs use a RESTful style and a resource-oriented architecture. Calls are made via HTTPS requests. Your request URLs should be patterned after the examples in the API documentation, and you should always include your API key in a query string. See the documentation for each API for more details on request parameters and URI structure.

6. Why can't I access all NYTimes.com content?

For each set of data we open up to the developer community, we have to consider a host of issues, ranging from load balancing to copyright law. If you have ideas for our APIs or are hoping to get access to a specific kind of data, contact us at code@nytimes.com.

7. Why do I have to register for a key to use your APIs?

Key registration allows us to monitor usage levels and ensure that developers are complying with our [Terms of Use](#). We respect your privacy and will not share your registration information with third parties.

8. Do I have to agree to your Terms of Use just to test an API?

Our [Terms of Use](#) apply to all uses of our APIs. Please agree to the terms before testing or using an API.

9. Can I use your APIs commercially?

Our APIs are for non-commercial use only. For details, see our [Terms of Use](#). If you have a commercial use case, please visit <https://nytlicensing.com/> or email code@nytimes.com.

10. What do you mean by "commercial purposes"?

Here are a few examples of what we consider commercial purposes:

- Selling New York Times content or data in any application.
- Charging a subscription fee for any New York Times content or data.
- Selling any application built with one of our APIs.
- Uses our content inside an application that is paid or has a paid tier.

If you aren't sure whether your plans constitute "commercial purposes," please contact us at code@nytimes.com. For commercial use please visit <https://nytlicensing.com/>.

11. Is there an API call limit?

Yes, there are two rate limits per API: 4,000 requests per day and 10 requests per minute. You should sleep 6 seconds between calls to avoid hitting the per minute rate limit. If you need a higher rate limit, please contact us at code@nytimes.com.

12. What response formats do you support?

Data is returned as JSON. Specific APIs may also return other formats. See the documentation for each API for more details.

This is important
to note



The screenshot shows the NYTimes API Developers portal. On the left, there's a sidebar with 'RESOURCES' and 'API REFERENCE' sections. Under 'API REFERENCE', the '/articlesearch.json' endpoint is selected. The main content area displays the 'GET /articlesearch.json' documentation, which includes an 'HTTP request' section with the URL `https://api.nytimes.com/svc/search/v2/articlesearch.json` and a 'Query Parameters' table.

Parameter	Type	Description
begin_date	string	matches <code>^\d{8}\$</code> Begin date (e.g. 20120101)
end_date	string	matches <code>^\d{8}\$</code> End date (e.g. 20121231)
facet	string	The following values are allowed: false, true Whether to show facet counts
facet_fields	string	The following values are allowed: day_of_week, document_type, ingredients, news_desk, pub_month, pub_year, section_name, source, subsection_name, type_of_material Facets
facet_filter	string	The following values are allowed: false, true Have facet counts use filters
fl	string	Field list
fq	string	Filter query
page	integer	0 ≤ value ≤ 100 Page number (0, 1, ...)
q	string	Query

To the right, there's a 'Try this API' interface with input fields for each parameter and an 'EXECUTE' button. A teal arrow points from the text 'You will need to authorize your credentials' to the 'AUTHORIZE' button in the top right corner of the interface. Another teal arrow points from the text 'You can also test query and get the resulting url to test in a browser' to the 'EXECUTE' button.

You will need to authorize your credentials

You can also test query and get the resulting url to test in a browser

"status": "OK", "copyright": "Copyright (c) 2019 The New York Times Company. All Rights Reserved.", "response": {"docs": [{"web_url": "https://www.nytimes.com/2019/04/15/opinion/notre-dame-paris-fire.html", "snippet": "A hundred years from now, people will still be talking about the fire of 2019.", "lead_paragraph": "A hundred years from now, people will still be talking about the fire of 2019.", "blog": {}, "source": "The New York Times", "multimedia": [{"rank": 0, "subType": "xlarge", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-articleLarge.jpg", "height": 450, "width": 600, "legacy": "xlarge", "crop_name": "articleLarge"}, {"rank": 0, "subType": "wide", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-articleLarge.jpg", "xlargeWidth": 600, "xlargeHeight": 450, "subType": "xlarge", "crop_name": "articleLarge"}, {"rank": 0, "subType": "wide", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-thumbWide.jpg", "height": 126, "width": 190, "legacy": "wide", "crop_name": "thumbWide"}, {"rank": 0, "subType": "wide", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-thumbWide.jpg", "widewidth": 190, "wideheight": 126}, {"rank": 0, "subType": "wide", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-thumbStandard.jpg", "height": 75, "width": 75, "legacy": "thumbnail", "crop_name": "thumbStandard"}, {"rank": 0, "subType": "articleInline", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-articleInline.jpg", "height": 142, "width": 190, "legacy": "wide", "crop_name": "articleInline"}, {"subType": "articleInline", "crop_name": "articleInline"}, {"rank": 0, "subType": "blog225", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-blog225.jpg", "height": 169, "width": 225, "legacy": "wide", "crop_name": "blog225"}, {"subType": "blog225"}, {"rank": 0, "subType": "blog427", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-blog427.jpg", "height": 320, "width": 427, "legacy": "wide", "crop_name": "blog427"}, {"subType": "blog427"}, {"rank": 0, "subType": "blog480", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-blog480.jpg", "height": 360, "width": 480, "legacy": "wide", "crop_name": "blog480"}, {"subType": "blog480"}, {"rank": 0, "subType": "blog533", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-blog533.jpg", "height": 400, "width": 533, "legacy": "wide", "crop_name": "blog533"}, {"subType": "blogSmallInline", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-blog533.jpg", "height": 113, "width": 151, "legacy": "wide", "crop_name": "blogSmallInline"}, {"subType": "blogSmallInline"}, {"rank": 0, "subType": "blogSmallThumb", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-blogSmallThumb.jpg", "height": 50, "width": 50, "legacy": "wide", "crop_name": "blogSmallThumb"}, {"subType": "blogSmallThumb"}, {"rank": 0, "subType": "facebookJumbo", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-facebookJumbo.jpg", "height": 549, "width": 1050, "legacy": "wide", "crop_name": "facebookJumbo"}, {"subType": "facebookJumbo"}, {"rank": 0, "subType": "filmstrip", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-filmstrip-v2.jpg", "height": 190, "width": 190, "legacy": "wide", "crop_name": "filmstrip"}, {"subType": "filmstrip"}, {"rank": 0, "subType": "hpLarge", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-hpLarge.jpg", "height": 287, "width": 511, "legacy": "wide", "crop_name": "hpLarge"}, {"subType": "hpLarge"}, {"rank": 0, "subType": "hpSmall", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-hpSmall.jpg", "height": 122, "width": 163, "legacy": "wide", "crop_name": "hpSmall"}, {"subType": "hpSmall"}, {"rank": 0, "subType": "jumbo", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-jumbo.jpg", "height": 768, "width": 1024, "legacy": "wide", "crop_name": "jumbo"}, {"subType": "jumbo"}, {"rank": 0, "subType": "largeHorizontal1375", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-largeHorizontal1375.jpg", "height": 250, "width": 375, "legacy": "wide", "crop_name": "largeHorizontal1375"}, {"subType": "largeHorizontal1375"}, {"rank": 0, "subType": "largeHorizontalJumbo", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-largeHorizontalJumbo.jpg", "height": 682, "width": 1024, "legacy": "wide", "crop_name": "largeHorizontalJumbo"}, {"subType": "largeHorizontalJumbo"}, {"rank": 0, "subType": "largeWidescreen573", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-largeWidescreen573.jpg", "height": 322, "width": 573, "legacy": "wide", "crop_name": "largeWidescreen573"}, {"subType": "largeWidescreen573"}, {"rank": 0, "subType": "largeWidescreen1050", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-largeWidescreen1050.jpg", "height": 1590, "width": 1050, "legacy": "wide", "crop_name": "largeWidescreen1050"}, {"subType": "largeWidescreen1050"}, {"rank": 0, "subType": "master1050", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-master1050.jpg", "height": 787, "width": 1050, "legacy": "wide", "crop_name": "master1050"}, {"subType": "master1050"}, {"rank": 0, "subType": "master180", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-master180.jpg", "height": 1135, "width": 180, "legacy": "wide", "crop_name": "master180"}, {"subType": "master180"}, {"rank": 0, "subType": "master315", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-master315.jpg", "height": 236, "width": 315, "legacy": "wide", "crop_name": "master315"}, {"subType": "master315"}, {"rank": 0, "subType": "master495", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-master495.jpg", "height": 371, "width": 495, "legacy": "wide", "crop_name": "master495"}, {"subType": "master495"}, {"rank": 0, "subType": "master768", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-master768.jpg", "height": 576, "width": 768, "legacy": "wide", "crop_name": "master768"}, {"subType": "master768"}, {"rank": 0, "subType": "master675", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-master675.jpg", "height": 506, "width": 675, "legacy": "wide", "crop_name": "master675"}, {"subType": "master675"}, {"rank": 0, "subType": "mediumFlexible177", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-mediumFlexible177.jpg", "height": 133, "width": 177, "legacy": "wide", "crop_name": "mediumFlexible177"}, {"subType": "mediumFlexible177"}, {"rank": 0, "subType": "mediumSquare149", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-mediumSquare149-v2.jpg", "height": 149, "width": 149, "legacy": "wide", "crop_name": "mediumSquare149"}, {"subType": "mediumSquare149"}, {"rank": 0, "subType": "mediumThreeByTwo210", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-mediumThreeByTwo210.jpg", "height": 1140, "width": 210, "legacy": "wide", "crop_name": "mediumThreeByTwo210"}, {"subType": "mediumThreeByTwo210"}, {"rank": 0, "subType": "mediumThreeByTwo225", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-mediumThreeByTwo225.jpg", "height": 150, "width": 225, "legacy": "wide", "crop_name": "mediumThreeByTwo225"}, {"subType": "mediumThreeByTwo225"}, {"rank": 0, "subType": "mediumThreeByTwo440", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-mediumThreeByTwo440.jpg", "height": 1293, "width": 440, "legacy": "wide", "crop_name": "mediumThreeByTwo440"}, {"subType": "mediumThreeByTwo440"}, {"rank": 0, "subType": "mediumThreeByTwo378", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-mediumThreeByTwo378.jpg", "height": 252, "width": 378, "legacy": "wide", "crop_name": "mediumThreeByTwo378"}, {"subType": "mediumThreeByTwo378"}, {"rank": 0, "subType": "mediumThreeByTwo525", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-mediumThreeByTwo525.jpg", "height": 1168, "width": 252, "legacy": "wide", "crop_name": "mediumThreeByTwo525"}, {"subType": "mediumThreeByTwo525"}, {"rank": 0, "subType": "miniMoth", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-miniMoth.jpg", "height": 70, "width": 151, "legacy": "wide", "crop_name": "miniMoth"}, {"subType": "miniMoth"}, {"rank": 0, "subType": "moth", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-moth-v2.jpg", "height": 151, "width": 151, "legacy": "wide", "crop_name": "moth"}, {"subType": "moth"}, {"rank": 0, "subType": "popup", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-popup.jpg", "height": 487, "width": 650, "legacy": "wide", "crop_name": "popup"}, {"subType": "popup"}, {"rank": 0, "subType": "sfSpan", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-sfSpan.jpg", "height": 263, "width": 395, "legacy": "wide", "crop_name": "sfSpan"}, {"subType": "sfSpan"}, {"rank": 0, "subType": "slide", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-slide.jpg", "height": 450, "width": 600, "legacy": "wide", "crop_name": "slide"}, {"subType": "slide"}, {"rank": 0, "subType": "smallSquare168", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-smallSquare168.jpg", "height": 168, "width": 168, "legacy": "wide", "crop_name": "smallSquare168"}, {"subType": "smallSquare168"}, {"rank": 0, "subType": "smallSquare252", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-smallSquare252.jpg", "height": 252, "width": 252, "legacy": "wide", "crop_name": "smallSquare252"}, {"subType": "smallSquare252"}, {"rank": 0, "subType": "square320", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-square320-v2.jpg", "height": 320, "width": 320, "legacy": "wide", "crop_name": "square320"}, {"subType": "square320"}, {"rank": 0, "subType": "square640", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-square640-v2.jpg", "height": 640, "width": 640, "legacy": "wide", "crop_name": "square640"}, {"subType": "square640"}, {"rank": 0, "subType": "superJumbo", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-superJumbo.jpg", "height": 1535, "width": 2048, "legacy": "wide", "crop_name": "superJumbo"}, {"subType": "superJumbo"}, {"rank": 0, "subType": "thumbLarge", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-thumbLarge.jpg", "height": 150, "width": 150, "legacy": "wide", "crop_name": "thumbLarge"}, {"subType": "thumbLarge"}, {"rank": 0, "subType": "tmagArticle", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/merlin_153570822_6863270c-fd2e-4786-a7c2-743068a88ecf-tmagArticle.jpg", "height": 444, "width": 592, "legacy": "wide", "crop_name": "tmagArticle"}, {"subType": "tmagArticle"}, {"rank": 0, "subType": "verticalTwoByThree735", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-verticalTwoByThree735.jpg", "height": 1102, "width": 735, "legacy": "wide", "crop_name": "verticalTwoByThree735"}, {"subType": "verticalTwoByThree735"}, {"rank": 0, "subType": "videoFifteenBySeven1305", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoFifteenBySeven1305.jpg", "height": 609, "width": 1305, "legacy": "wide", "crop_name": "videoFifteenBySeven1305"}, {"subType": "videoFifteenBySeven1305"}, {"rank": 0, "subType": "videoHpMedium", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoHpMedium.jpg", "height": 211, "width": 375, "legacy": "wide", "crop_name": "videoHpMedium"}, {"subType": "videoHpMedium"}, {"rank": 0, "subType": "videoLarge", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoLarge.jpg", "height": 507, "width": 768, "legacy": "wide", "crop_name": "videoLarge"}, {"subType": "videoLarge"}, {"rank": 0, "subType": "videoSixteenByNine1050", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoSixteenByNine1050.jpg", "height": 591, "width": 1050, "legacy": "wide", "crop_name": "videoSixteenByNine1050"}, {"subType": "videoSixteenByNine1050"}, {"rank": 0, "subType": "videoSixteenByNine150", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoSixteenByNine150.jpg", "height": 84, "width": 150, "legacy": "wide", "crop_name": "videoSixteenByNine150"}, {"subType": "videoSixteenByNine150"}, {"rank": 0, "subType": "videoSixteenByNine225", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoSixteenByNine225.jpg", "height": 126, "width": 225, "legacy": "wide", "crop_name": "videoSixteenByNine225"}, {"subType": "videoSixteenByNine225"}, {"rank": 0, "subType": "videoSixteenByNine310", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoSixteenByNine310.jpg", "height": 174, "width": 310, "legacy": "wide", "crop_name": "videoSixteenByNine310"}, {"subType": "videoSixteenByNine310"}, {"rank": 0, "subType": "videoSixteenByNine390", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoSixteenByNine390.jpg", "height": 219, "width": 390, "legacy": "wide", "crop_name": "videoSixteenByNine390"}, {"subType": "videoSixteenByNine390"}, {"rank": 0, "subType": "videoSixteenByNine480", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoSixteenByNine480.jpg", "height": 270, "width": 480, "legacy": "wide", "crop_name": "videoSixteenByNine480"}, {"subType": "videoSixteenByNine480"}, {"rank": 0, "subType": "videoSixteenByNine495", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoSixteenByNine495.jpg", "height": 278, "width": 495, "legacy": "wide", "crop_name": "videoSixteenByNine495"}, {"subType": "videoSixteenByNine495"}, {"rank": 0, "subType": "videoSixteenByNine540", "caption": null, "credit": null, "type": "image", "url": "images/2019/04/15/opinion/15Druckerman/15Druckerman-videoSixteenByNine540.jpg", "height": 304, "width": 540, "legacy": "wide", "crop_name": "videoSixteenByNine540"}, {"subType": "videoSixteenByNine540"}]}

Lets see how this works
in the shell

First, I've gone to my
class directory to make
a scraper file directory

```
week-3 — docker run --rm -it -v/Users/nlangholz/Documents/Egnyte/Private/nlangholz/UCLA/stat418-tools-in-datascien...
Last login: Tue Apr 16 10:25:52 on ttys001
You have new mail.
[NA-nlanghol-A02:~ nlangholz$ ls
20 datagroups LDA viz.html      Untitled1.ipynb
20 datagroups docvec.html       crm_customer_23JAN2018.csv
Applications                  gensim-data
Desktop                       nltk_data
Documents                      rbtv-d2v-v1-dbbow.model
Downloads                      rbtv-d2v-v2-dbbow.model
HD_log.txt                     rbtv_d2v_1.model
Library                        rbtv_d2v_v1.model
LightGBM                        results
Movies                          spark
Music                           tensorflow
New_York_Red_Bulls.xlsx        tsne-new-values-v1.npy
Pictures                        tsne_new_values_v0.npy
Pipfile                         verse_gapminder.tar
Public                          verse_gapminder2.tar
Untitled.ipynb
[NA-nlanghol-A02:~ nlangholz$ cd Documents/Egnyte/Private/nlangholz/UCLA/stat418-tools-in-datascience/week-3/
[NA-nlanghol-A02:week-3 nlangholz$ ls
[README.md          slides-week-3.key
[NA-nlanghol-A02:week-3 nlangholz$ docker run --rm -it -v`pwd`:/data datascienceworkshops/data-science-at-the-command
[-line
[/data]$ ls
[README.md  slides-week-3.key
[/data]$ mkdir sh-scraper-files
[[/data]$ ls
[README.md  sh-scraper-files  slides-week-3.key
[/data]$ cd sh-scraper-files/
[[/data/sh-scraper-files]$ ls
[[/data/sh-scraper-files]$
```

Now I have a shell command that uses curl to call the API for 'Notre Dame Cathedral'

Oops

```
week-3 — docker run --rm -it -v/Users/nlangholz/Documents/Egnyte/Private/nlangholz/UCLA/stat418-tools-in-datasience/...
Music tensorflow
New_York_Red_Bulls.xlsx tsne-new-values-v1.npy
Pictures tsne_new_values_v0.npy
Pipfile verse_gapminder.tar
Public verse_gapminder2.tar
Untitled.ipynb
[NA-nlanghol-A02:~ nlangholz$ cd Documents/Egnyte/Private/nlangholz/UCLA/stat418-tools-in-datasience/week-3/
[NA-nlanghol-A02:week-3 nlangholz$ ls
[README.md slides-week-3.key
[NA-nlanghol-A02:week-3 nlangholz$ docker run --rm -it -v`pwd`:/data datascienceworkshops/data-science-at-the-command-line
[ne
[/data]$ ls
[README.md slides-week-3.key
[/data]$ mkdir sh-scraper-files
[[/data]$ ls
[README.md sh-scraper-files slides-week-3.key
[/data]$ cd sh-scraper-files/
[[/data/sh-scraper-files]$ ls
[[/data/sh-scraper-files]$ parallel -j1 --progress --delay 7 --results results "curl -sL \"\
> \"http://api.nytimes.com/svc/search/v2/articlesearch.json?q=NotreDame\"\
> \"Cathedral&begin_date=(1)0101&end_date=(1)1231&page=(2)&api-key=\"\
> \"<your-api-key-here>\" ::: (2014..2019) ::: (0..99) > /dev/null"
Computers / CPU cores / Max jobs to run
1:local / 2 / 1

Computer:jobs running/jobs completed/%of started jobs/Average seconds to complete
local:0/600/100%/7.4s
[/data/sh-scraper-files]$ tree results | head
results
└── 1
    └── 2014
        └── 2
            ├── 0
            │   ├── seq
            │   ├── stderr
            │   └── stdout
            └── 1
                └── seq
```

How long is this going to take?

There is a lot going on in this command. I have delayed each request by 7 seconds, and am searching for articles from 2014 to 2019 with 99 pages of results

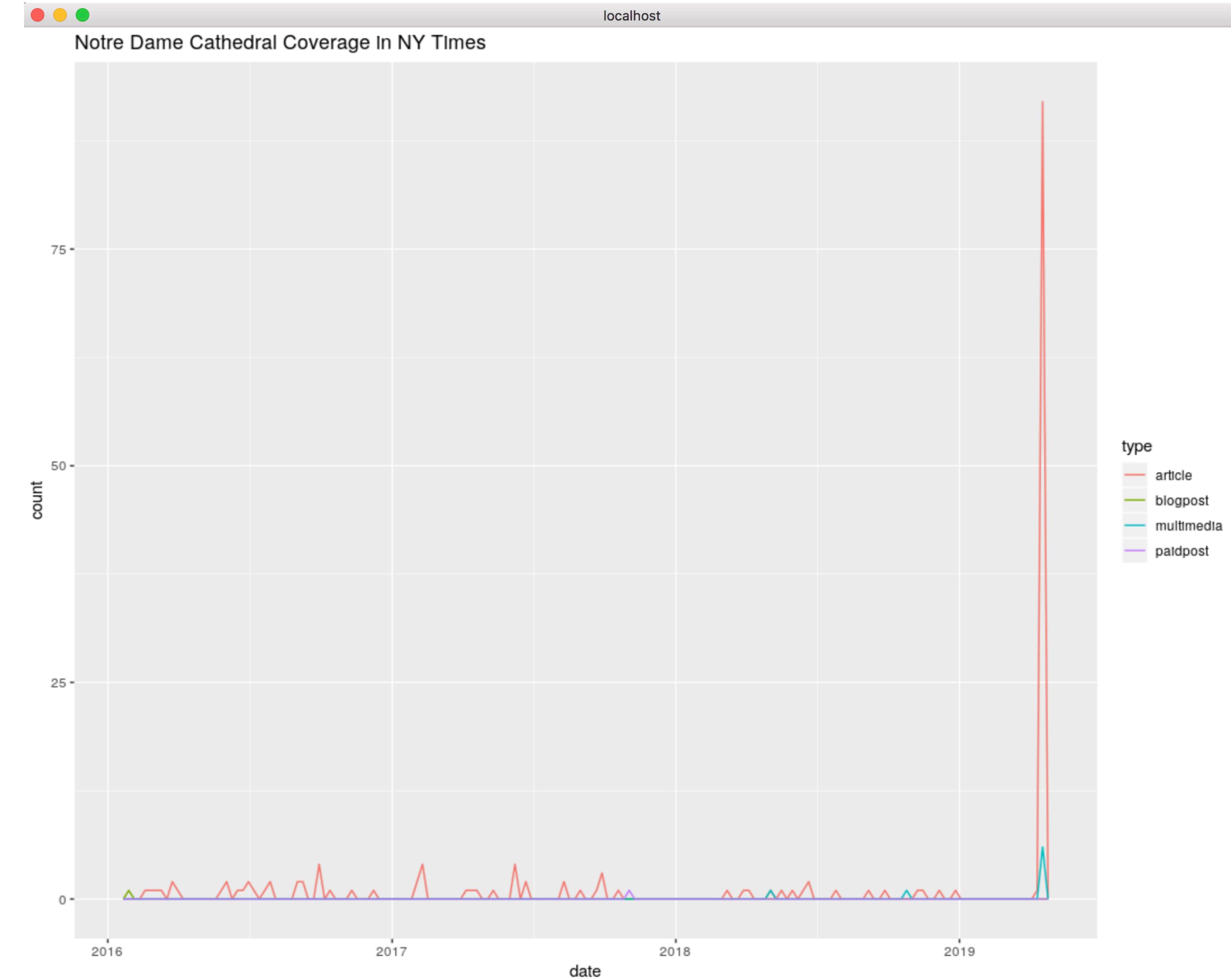
```
week-3 — docker run --rm -it -v/Users/nlangholz/Documents/Egnyte/Private/nlangholz/UCLA/stat418-tools-in-datascience/week-3:/data...
[/data/sh-scraper-files]$ wc -l notre-dame-articles.csv
1324 notre-dame-articles.csv
[/data/sh-scraper-files]$ < notre-dame-articles.csv cols -c date cut -dT -f1 | head | csvlook
+-----+-----+
| 2016-11-23 | article | Notre Dame to Appeal Ruling on Academic Cheating by Football Players
| 2016-12-08 | article | UConn Brushes Aside Notre Dame for 83rd Straight Victory
| 2016-03-27 | article | For Notre Dame's Star Guard, Family Came First
| 2016-03-21 | article | Last-Second Tip-In Shines Spotlight on a Notre Dame Guard
| 2016-06-05 | article | Bobby Williams Dies at 86; Led Undefeated Notre Dame to a Title in 1949
| 2016-02-15 | article | Johnny Lattner, Versatile Heisman-Winning Halfback With Notre Dame, Dies at 83
| 2016-03-28 | article | Top-Seeded North Carolina Fills Last Spot in Final Four
| 2016-09-09 | article | Mystery Surrounds Car With Gas Containers Found in Paris Near Notre Dame
| 2016-11-25 | article | Colombia's Peace Agreement
[[/data/sh-scraper-files]$]
```

The results

A .sh script exists in the GitHub repo to recreate this web scrape

The results

A .sh script exists in the GitHub repo to recreate this web scrape



```
Last login: Tue Apr 16 16:12:54 on ttys001
You have new mail.
NA-nlanghol-A02:~ nlangholz$ docker container ls
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS          NAMES
a0fe1a3177e0        datascienceworkshops/data-science-at-the-command-line   "/bin/sh -c bash"   3 hours ago       Up 3 hours
8e8b2056507b        datascienceworkshops/data-science-at-the-command-line   "/bin/sh -c bash"   6 hours ago       Up 6 hours
e1b252022063        datascienceworkshops/data-science-at-the-command-line   "/bin/sh -c bash"   18 hours ago      Up 18 hours
2b95a3c1c48b        datascienceworkshops/data-science-at-the-command-line   "/bin/sh -c bash"   41 hours ago      Up 41 hours
e6cda4f02042        datascienceworkshops/data-science-at-the-command-line   "/bin/sh -c bash"   41 hours ago      Up 41 hours
cb4f9295f422        datascienceworkshops/data-science-at-the-command-line   "/bin/sh -c bash"   41 hours ago      Up 41 hours
c7f95c05e64f        datascienceworkshops/data-science-at-the-command-line   "/bin/sh -c bash"   2 days ago        Up 2 days
540fd79276ed        datascienceworkshops/data-science-at-the-command-line   "/bin/sh -c bash"   2 days ago        Up 2 days
NA-nlanghol-A02:~ nlangholz$
```

...i should probably kill some of these container processes