

IBiS Computational Biology Bootcamp

September 12: Cook 3118, 9 AM - 1 PM

September 13: Cook 3118, 9 AM - 1 PM

September 15: Annenberg G02, 9 AM - 1 PM



Unsolicited advice about computational work



Don't be afraid



It will be rewarding



It works



Stop using Excel

Use our wiki

The screenshot shows a GitHub repository page for `AndersenLab/IBiS-Bootcamp`. The top navigation bar includes links to various services like Apple, Google Maps, YouTube, Wikipedia, and iCloud. The repository header shows 32 commits, 1 branch, 0 releases, and 2 contributors. A red circle highlights the "Check out our Wiki" link at the bottom of the main content area.

AndersenLab/IBiS-Bootcamp

Explore Gist Blog Help

Unwatch 6 Star 0 Fork 0

<http://www.andersenlab.org> — Edit

32 commits 1 branch 0 releases 2 contributors

branch: master / +

Fixed DL238 script!

danielecook authored 18 hours ago latest commit 5e7988d6c6

File	Commit Message	Time Ago
data	Add sorter data	2 days ago
resources	atom selections example	3 days ago
scripts	Fixed DL238 script!	18 hours ago
LICENSE	Initial commit	a month ago
README.md	Update README.md	6 days ago

README.md

IBiS-Bootcamp

This repository will store files and scripts for the 2014 IBiS Computational Biology Bootcamp.

Check out our [Wiki](#)

Code Issues Pull Requests Wiki Pulse Graphs Settings

HTTPS clone URL <https://github.com/>

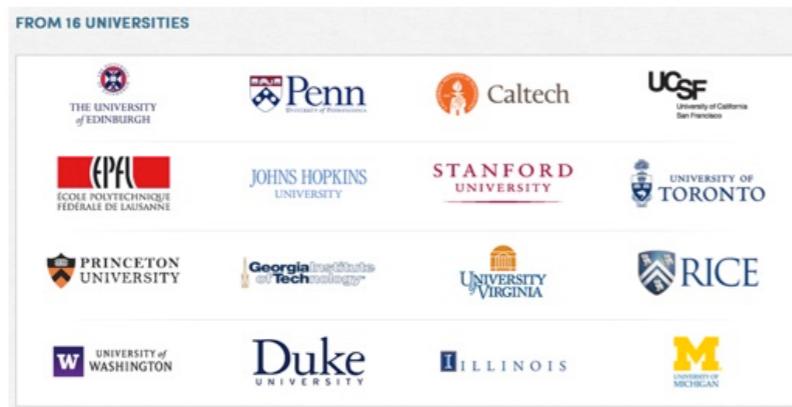
You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

Clone in Desktop Download ZIP

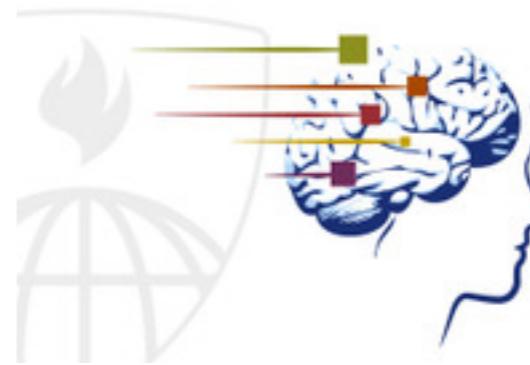
[www.github.com/AndersenLab/IBiS-Bootcamp](https://github.com/AndersenLab/IBiS-Bootcamp)

The web is filled with free books,
instructional tools, courses, etc.

Coursera



Data Science Specialization

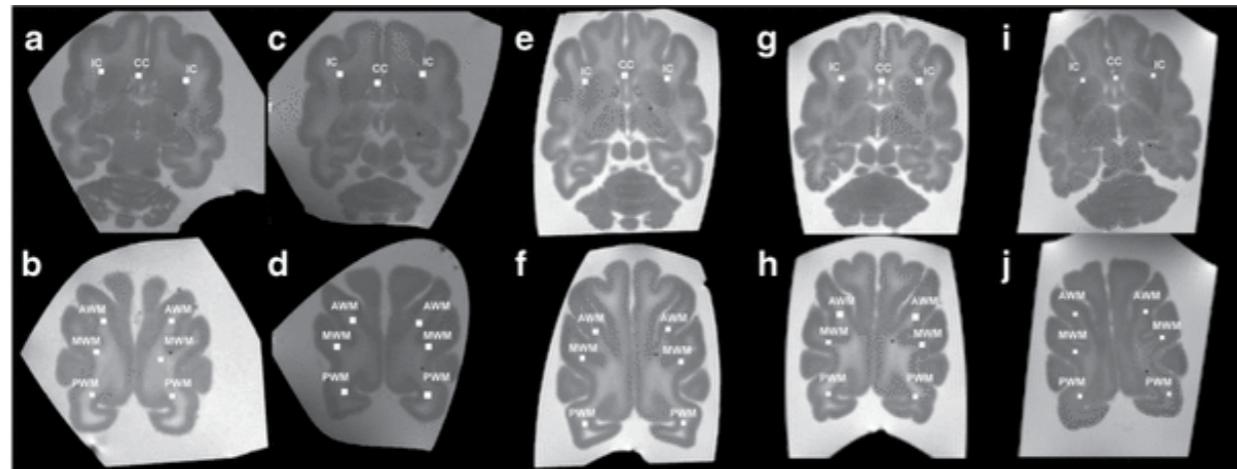


codecademy

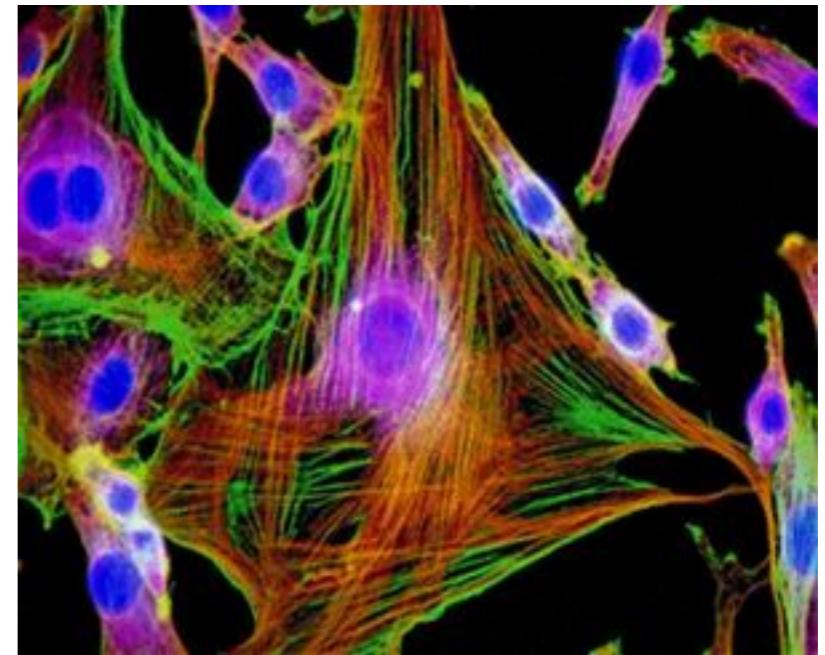
ROSALIND

software carpentry

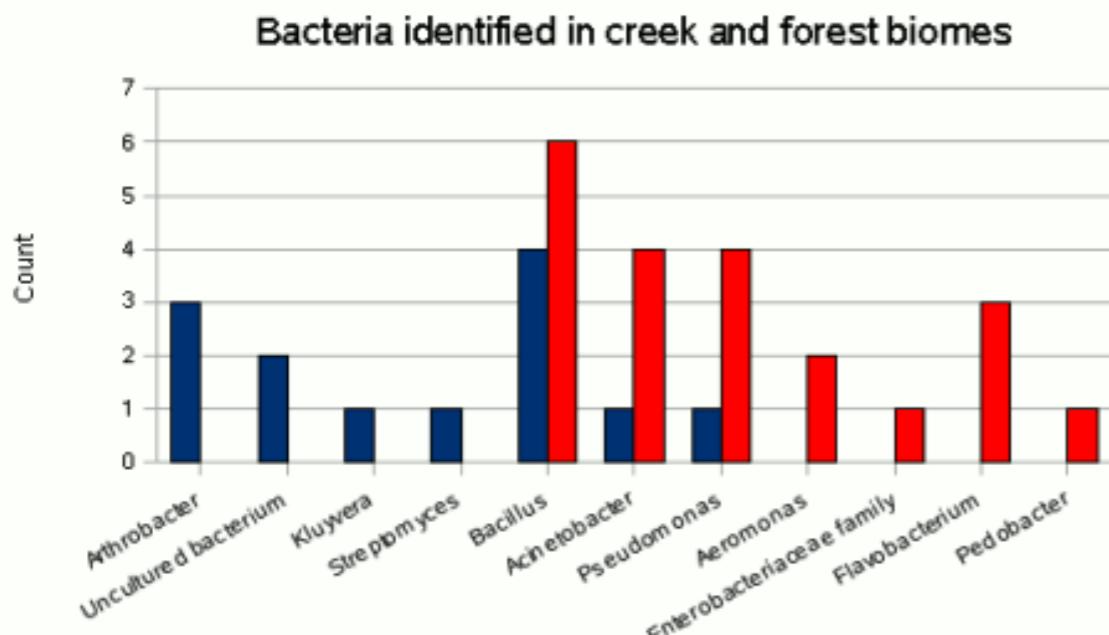
The days of descriptive biology are [almost] over



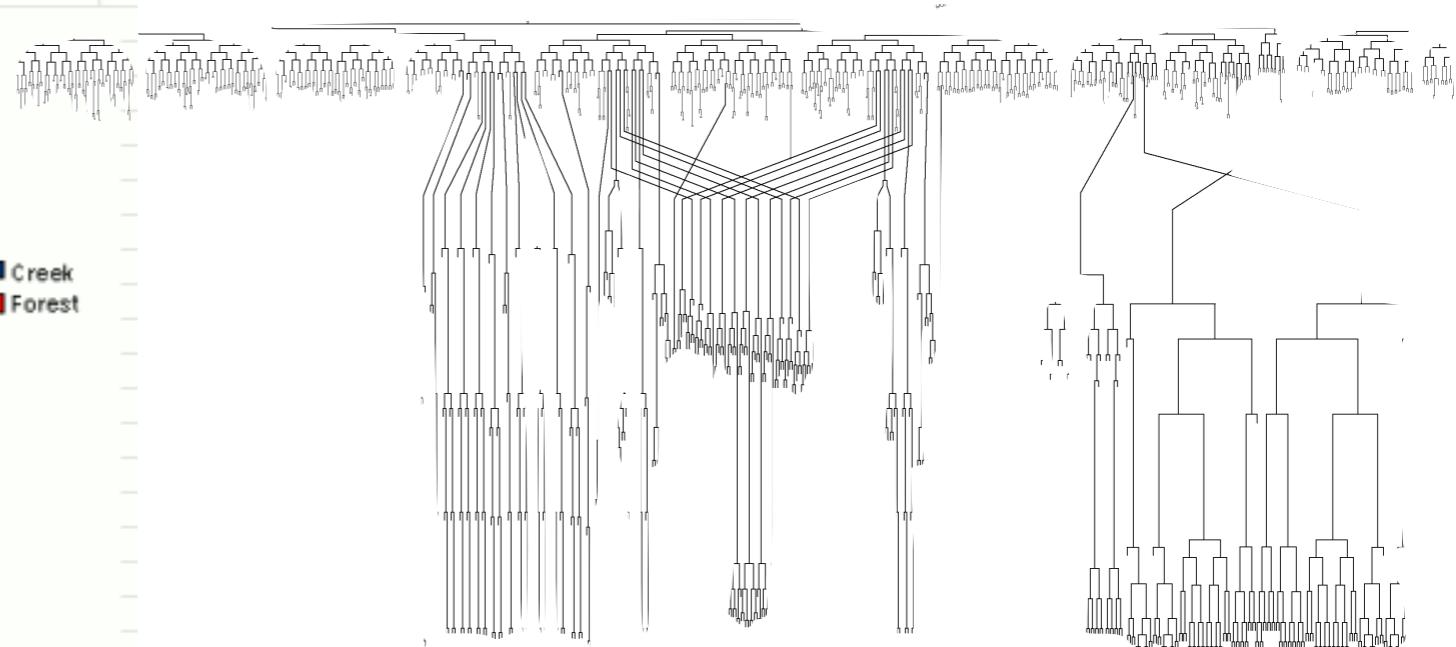
Reproducibility?



Sampling?



Error?



Automation?

NOW is the most exciting time to be a biologist



Any biologist can collect data.

Few biologists can process and analyze them.



Bootcamp outline

Day #1: Reproducible research, Command line introduction, Intro to R and RStudio, and data input

Day #2: Data cleaning, manipulation, processing

Day #3: R and data plotting, five-minute presentations

Bootcamp outline

Day #1: Reproducible research, Command line introduction, Intro to R and RStudio, and data input

- **Intro to reproducible research**
- Git and github
- Directory structures
- Basic command line
- Markdown
- More command-tool utilities
- R and Studio
- Data input

Working with biological data requires the ability to clean data.



For Big-Data Scientists, 'Janitor Work' Is Key Hurdle to Insights

By STEVE LOHR AUG. 17, 2014



Monica Rogati, Jawbone's vice president for data science, with Brian Wilt, a senior data scientist.
Peter DaSilva for The New York Times



Big data's dirty problem

by Verne Kopytoff

@vkopytoff

JUNE 30, 2014, 10:58 AM EDT



Science is in a reproducibility crisis: How do we resolve it?

Sep 20, 2013 by Fiona Fidler & Ascelin Gordon, The Conversation



About Projects Press Contact

Validating key experimental results
via independent replication

[Learn more »](#)

Reproducibility initiatives will invite increased scrutiny into data cleaning methods.

What is reproducible research?

The concept that scientific claims are published with data and code so that others can verify and build on those findings.

Please experience more at:

www.coursera.org Reproducible research course
GitHub.com Help and training pages

Microsoft Excel is destroying biology



Gene name errors are widespread in the scientific literature

Ziemann, Eren, and El-Osta. *Genome Biology* August 2016

Bootcamp outline

Day #1: Reproducible research, Command line introduction, Intro to R and RStudio, and data input

- **Intro to reproducible research**
- **Git and github**
- **Directory structures**
- **Basic command line**
- **Markdown**
- **More command-tool utilities**
- **R and Studio**
- **Data input**

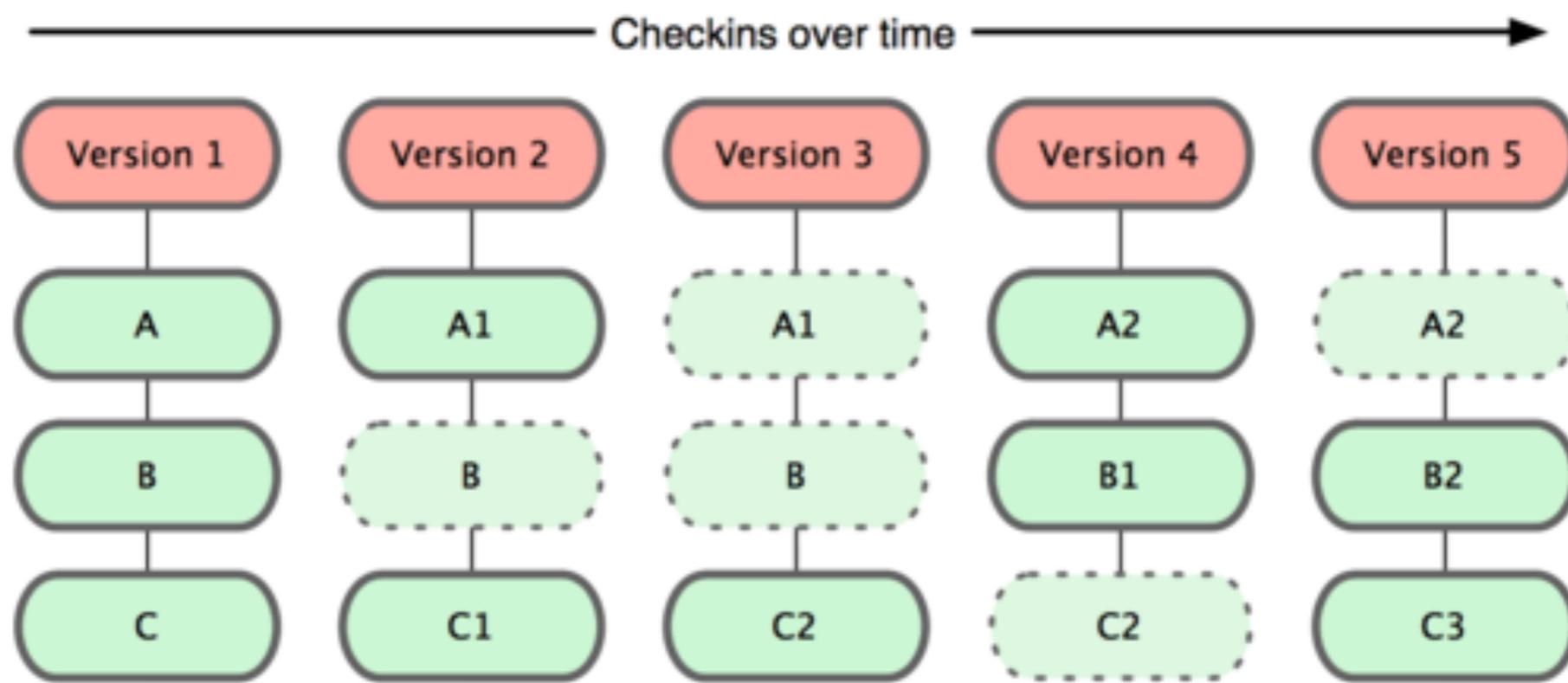
Notes vs what you type



git

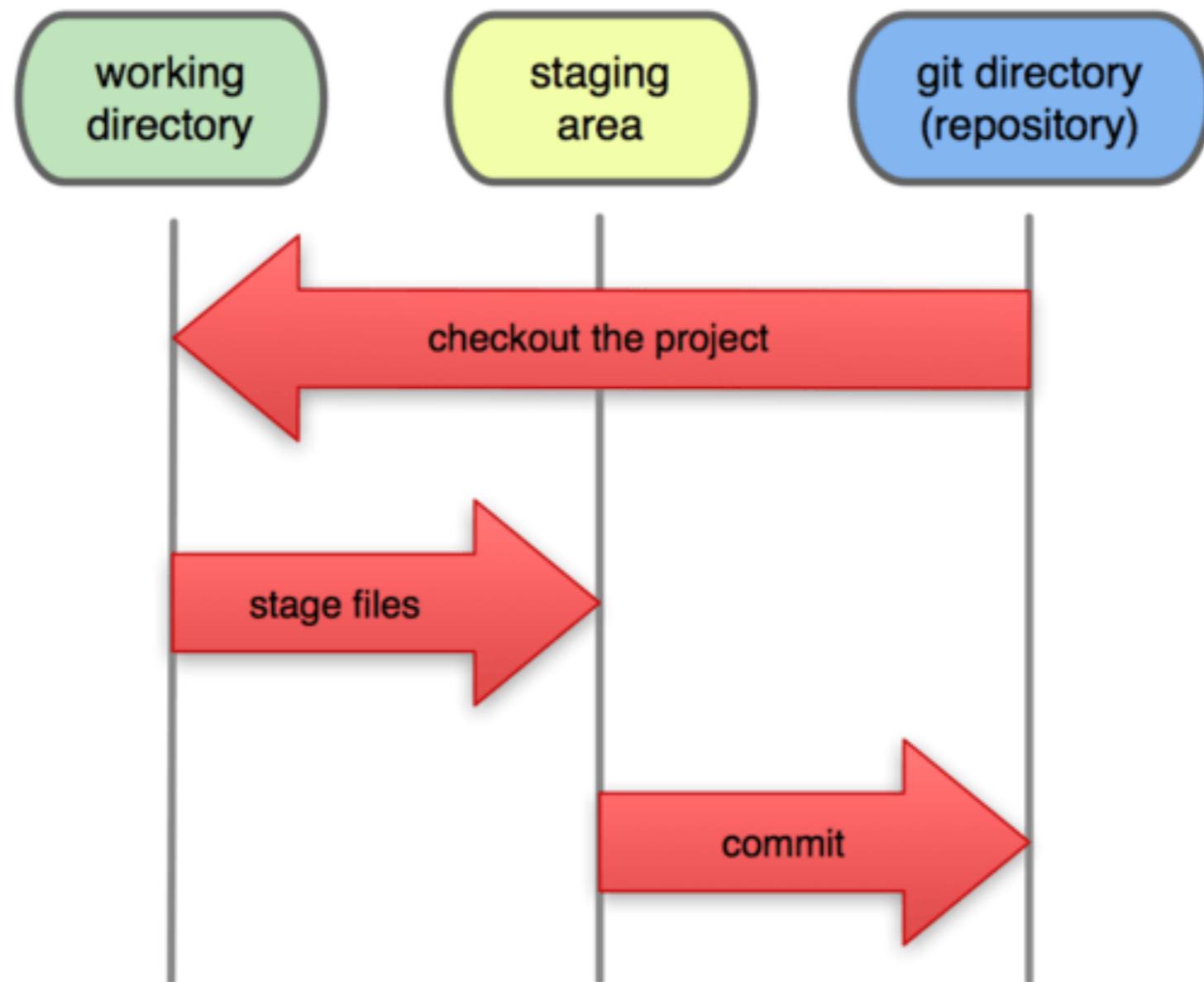
is a version control system

git tracks changes and files over time



git tracks local changes

Local Operations

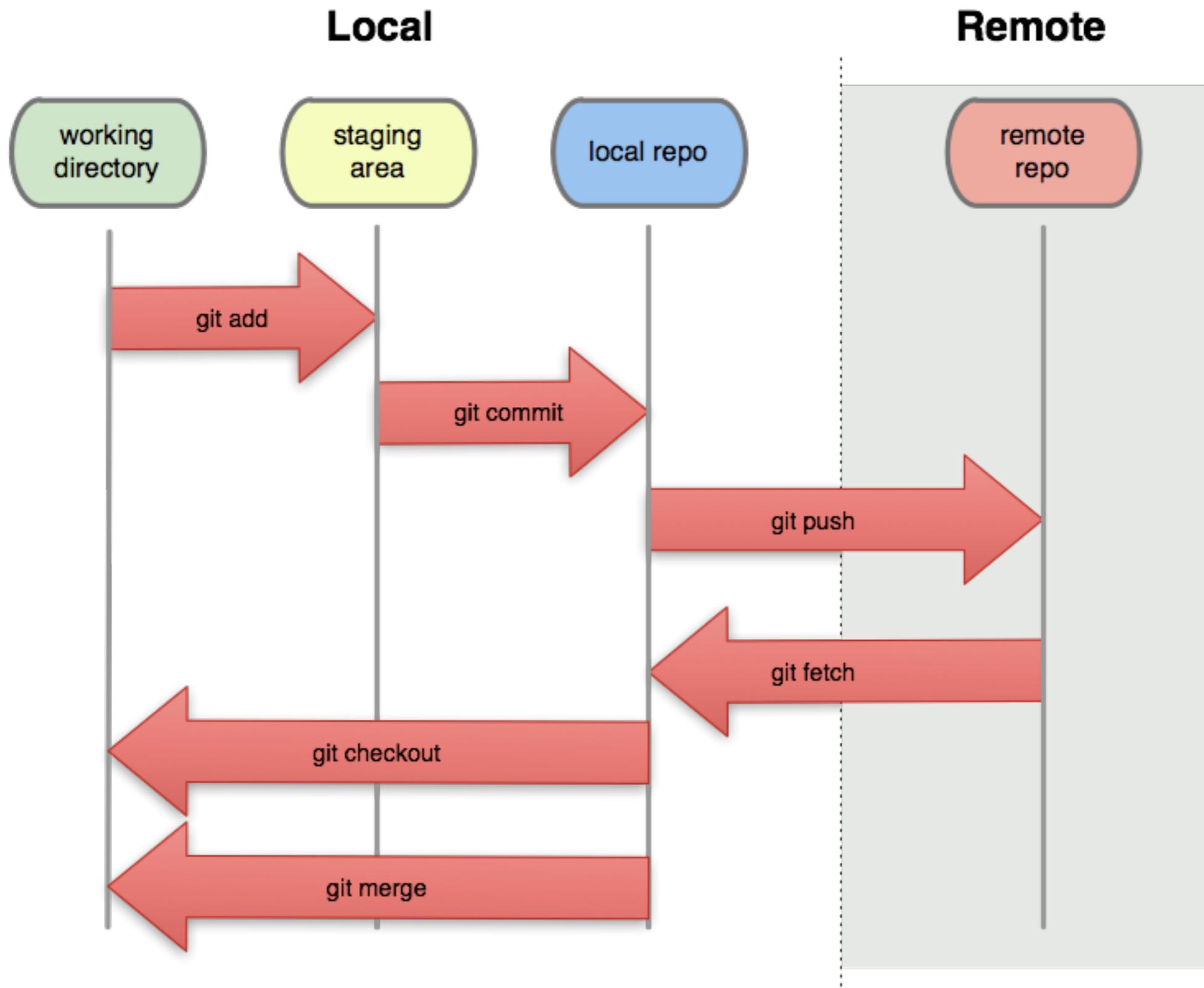




Github makes git super powerful

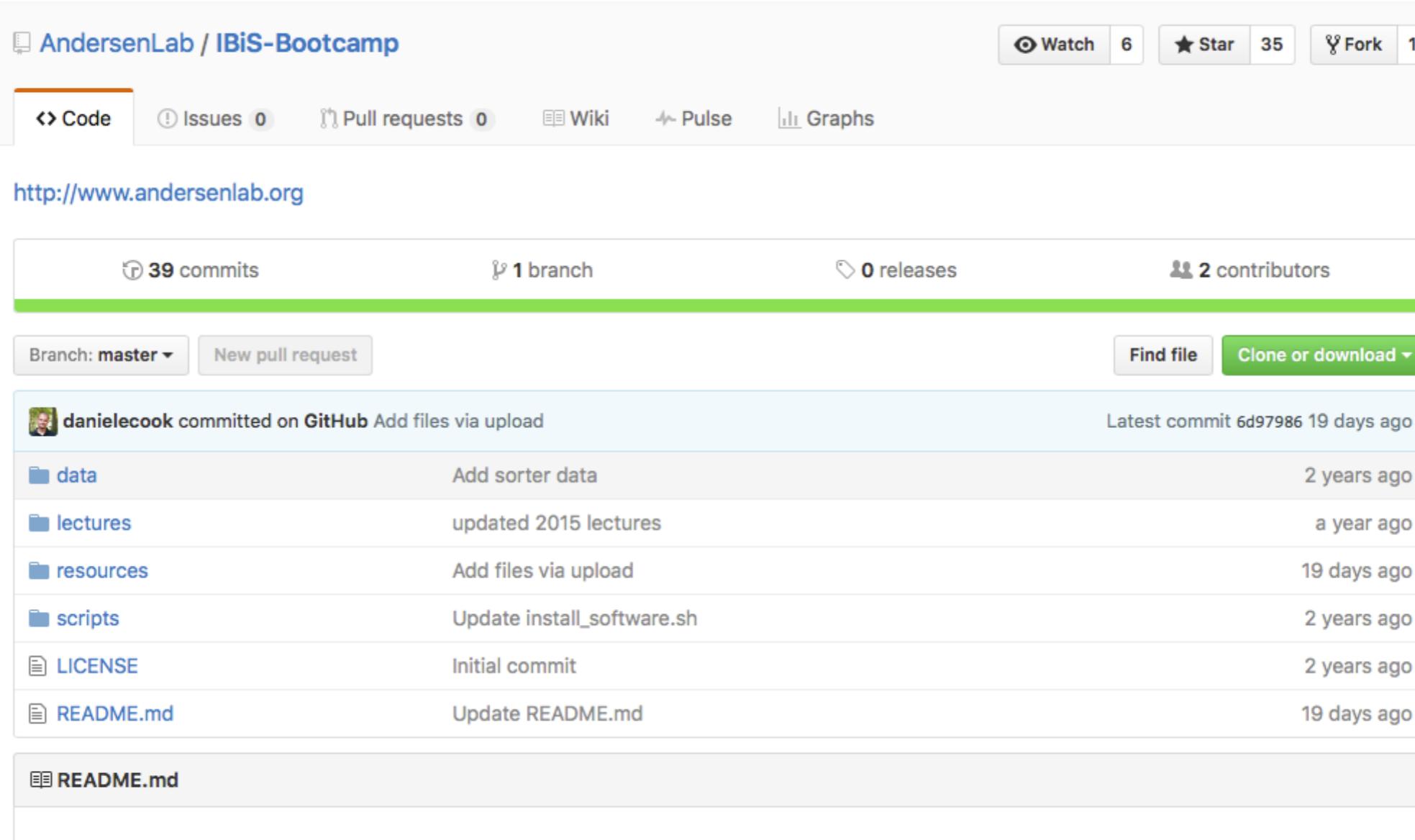
Check out: www.github.com/andersenlab

github takes git from local to the world



Let's clone the IBiS Bootcamp repository

Go to github.com/andersenlab/ibis-bootcamp

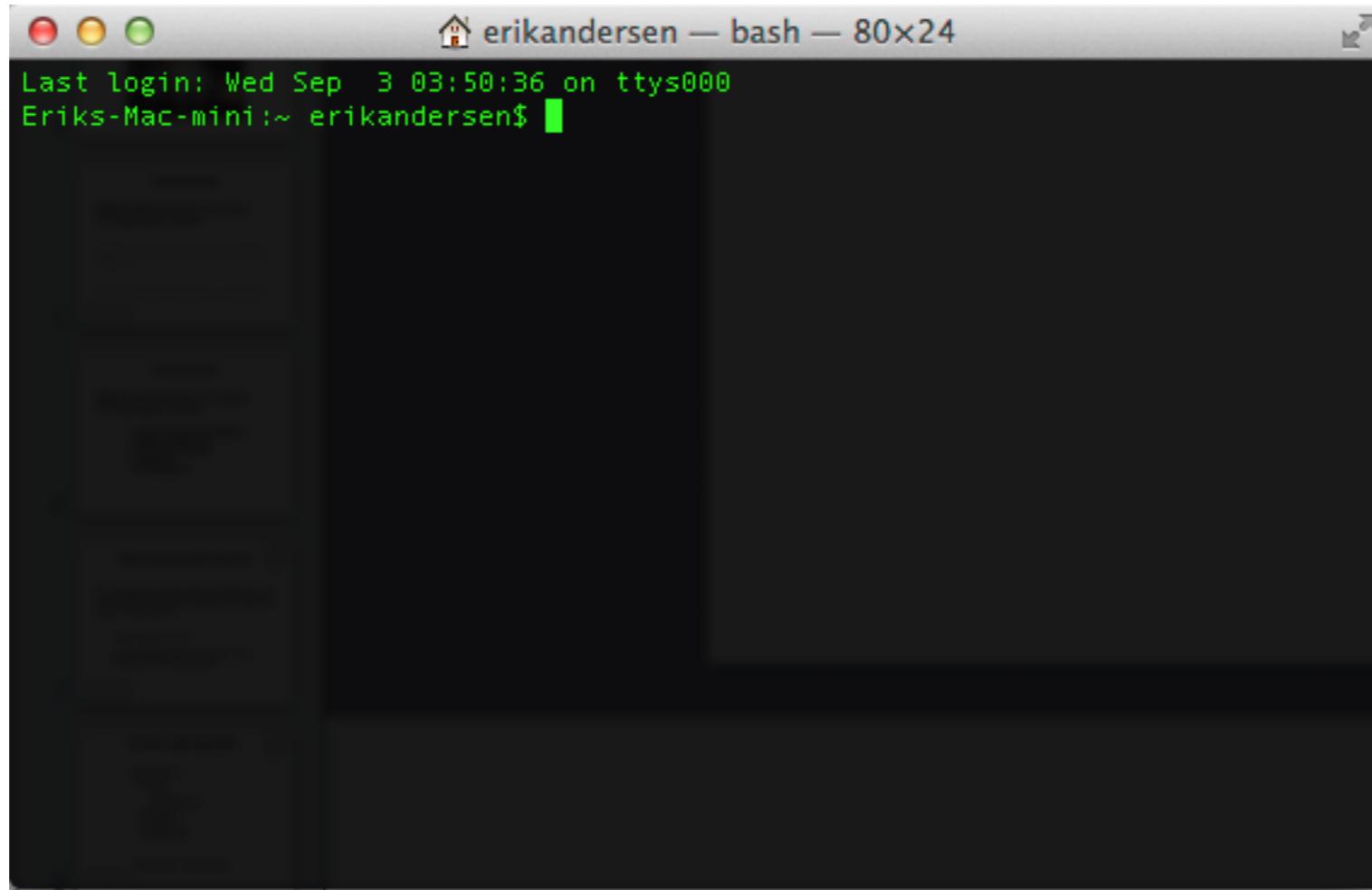


The screenshot shows the GitHub repository page for `AndersenLab / IBiS-Bootcamp`. The page includes navigation links for Code, Issues (0), Pull requests (0), Wiki, Pulse, and Graphs. It displays statistics: 39 commits, 1 branch, 0 releases, and 2 contributors. Below these, there are buttons for Branch: master ▾, New pull request, Find file, and a prominent green **Clone or download ▾** button. A red arrow points to this button. The main content area lists recent commits:

Author	Commit Message	Date
danielecook	committed on GitHub Add files via upload	Latest commit 6d97986 19 days ago
	Add sorter data	2 years ago
	updated 2015 lectures	a year ago
	Add files via upload	19 days ago
	Update install_software.sh	2 years ago
	Initial commit	2 years ago
	Update README.md	19 days ago

At the bottom, there is a single entry for `README.md`.

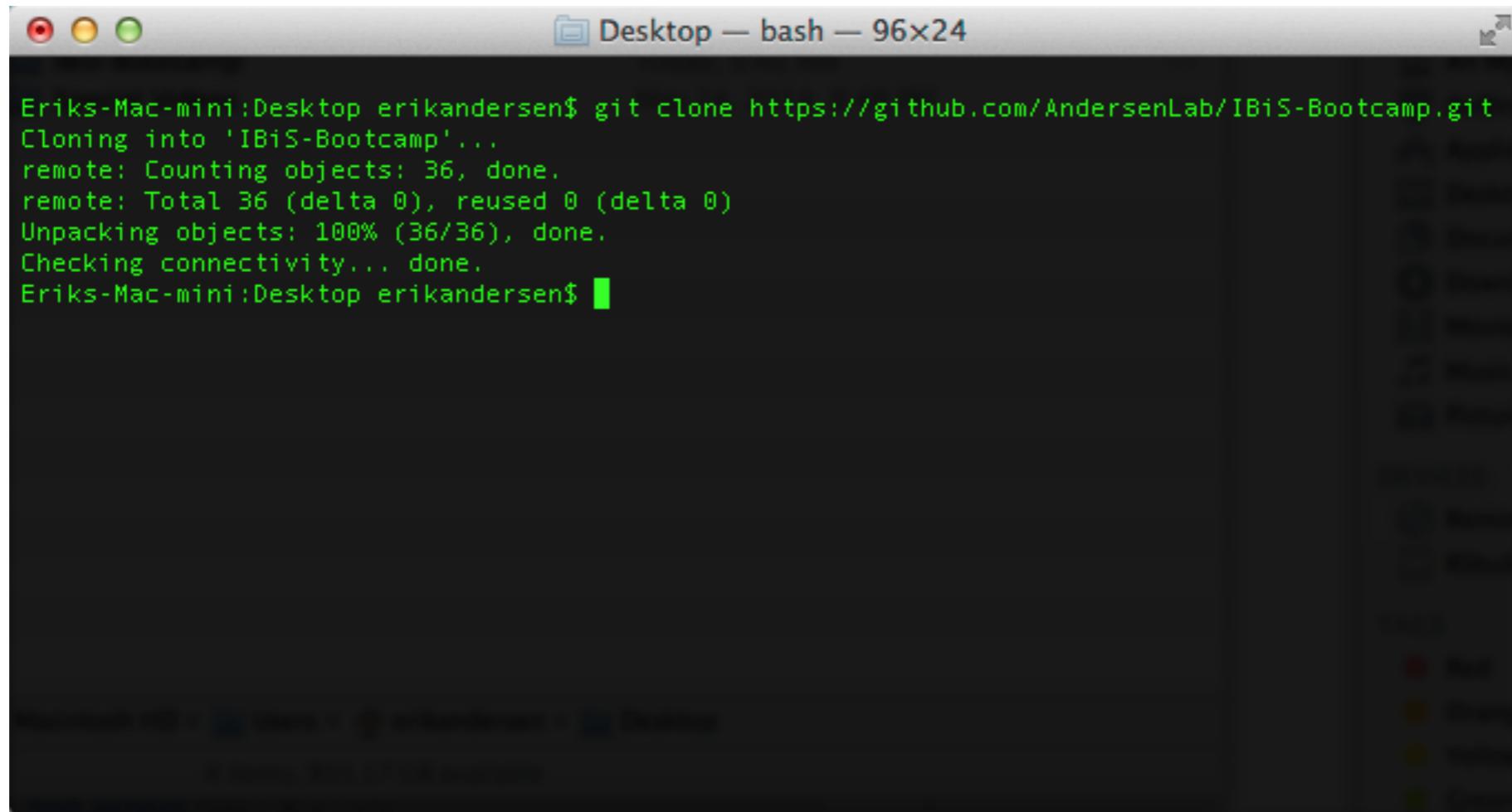
Command-line interface on a Mac



Open Terminal (in your dock)

Let's clone the IBiS Bootcamp repository

git clone invokes git to clone a repository



```
Eriks-Mac-mini:Desktop erikandersen$ git clone https://github.com/AndersenLab/IBiS-Bootcamp.git
Cloning into 'IBiS-Bootcamp'...
remote: Counting objects: 36, done.
remote: Total 36 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (36/36), done.
Checking connectivity... done.
Eriks-Mac-mini:Desktop erikandersen$
```

1. Go to home directory. Type `cd ~`
2. Type `git clone` and then paste directory after `clone`.

Git/Github Commands

Move files to staging area `git add [file/s]`

Check status of local repo `git status`

Commit files to local repo `git commit -m "short message"`

Push files to remote repo `git push origin master`

Further experiences:
<http://git.io/vGoaJ>

THIS IS GIT. IT TRACKS COLLABORATIVE WORK
ON PROJECTS THROUGH A BEAUTIFUL
DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZIZE THESE SHELL
COMMANDS AND TYPE THEM TO SYNC UP.
IF YOU GET ERRORS, SAVE YOUR WORK
ELSEWHERE, DELETE THE PROJECT,
AND DOWNLOAD A FRESH COPY.



Bootcamp outline

Day #1: Reproducible research, Command line introduction, Intro to R and RStudio, and data input

- **Intro to reproducible research**
- **Git and github**
- **Directory structures**
- **Basic command line**
- **Markdown**
- **More command-tool utilities**
- **R and Studio**
- **Data input**

Notes vs what you type

It's time to get organized...

```
MyProject/  
  data/  
    raw/  
    processed/  
  scripts/  
  results/  
  readme.md
```

```
MyProject/data/raw/
```

Basic directory commands for UNIX

man = read the manual for a command

ls = list the contents of a directory

Try typing **man ls**

Try typing **man ls -a**

Further experiences:

<http://freeengineer.org/learnUNIXin10minutes.html>
<http://www.ee.surrey.ac.uk/Teaching/Unix/>
or just google: “unix command line tools”

Basic directory commands for UNIX

man = read the manual for a command

ls = list the contents of a directory

pwd = display the present working directory

cd = change directory

mkdir = make a directory

rmdir = remove a directory

Further experiences:

<http://freeengineer.org/learnUNIXin10minutes.html>
<http://www.ee.surrey.ac.uk/Teaching/Unix/>
or just google: “unix command line tools”

Keyboard shortcuts for UNIX

Press *tab* to complete a filename

Navigation shortcuts

ctrl + a : go to beginning of line

ctrl + e : go to end of line

alt + <- (->) : skip between delimiters

alt + delete : delete previous word

command + up (down) arrow: beginning (or end) of text file

Selection shortcuts

option + shift + <- (->): highlight text to next delimiter

shift + up (down) arrow : highlight previous (next) line

command + shift <- (->) : highlight to beginning (or end of line)

command + shift + up (down) arrow : highlight to beginning (or end of text)

Further experiences:
<http://goo.gl/ukS77X>

Let's each make the directory structure using the command line.

```
MyProject/  
    data/  
        raw/  
        processed/  
        scripts/  
        results/  
        mistakes_happen/
```

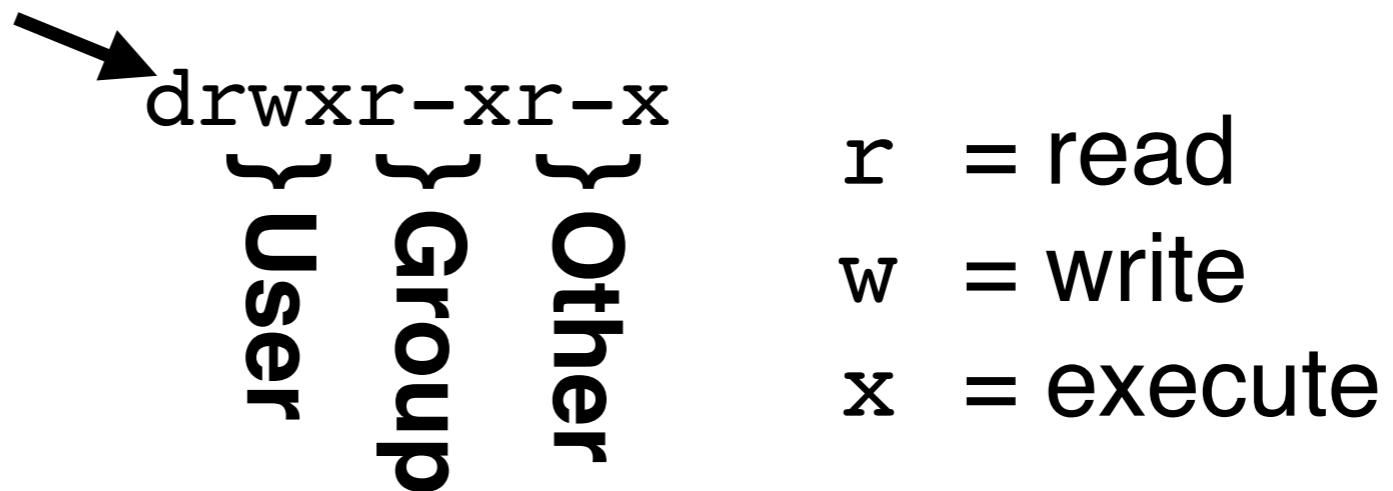
man	= read the manual for a command
ls	= list the contents of a directory
pwd	= display the present working directory
cd	= change directory
mkdir	= make a directory
rmdir	= remove a directory

cd .. allows you to go back one directory in tree

Permissions (modes) matter.

Go to your data directory, type `ls -la`

Filetype (- , d, or c)



Symbolic Notation	Octal Notation	English
-----	0000	no permissions
---x--x--x	0111	execute
--w--w--w-	0222	write
--wx-wx-wx	0333	write & execute
-r--r--r--	0444	read
-r-xr-xr-x	0555	read & execute
-rw-rw-rw-	0666	read & write
-rwxrwxrwx	0777	read, write, & execute

`chmod` changes
the mode

Bootcamp outline

Day #1: Reproducible research, Command line introduction, Intro to R and RStudio, and data input

- **Intro to reproducible research**
- **Git and github**
- **Directory structures**
- **Basic command line**
- **Markdown**
- **More command-tool utilities**
- **R and Studio**
- **Data input**

Notes vs what you type

What about that readme.md file?

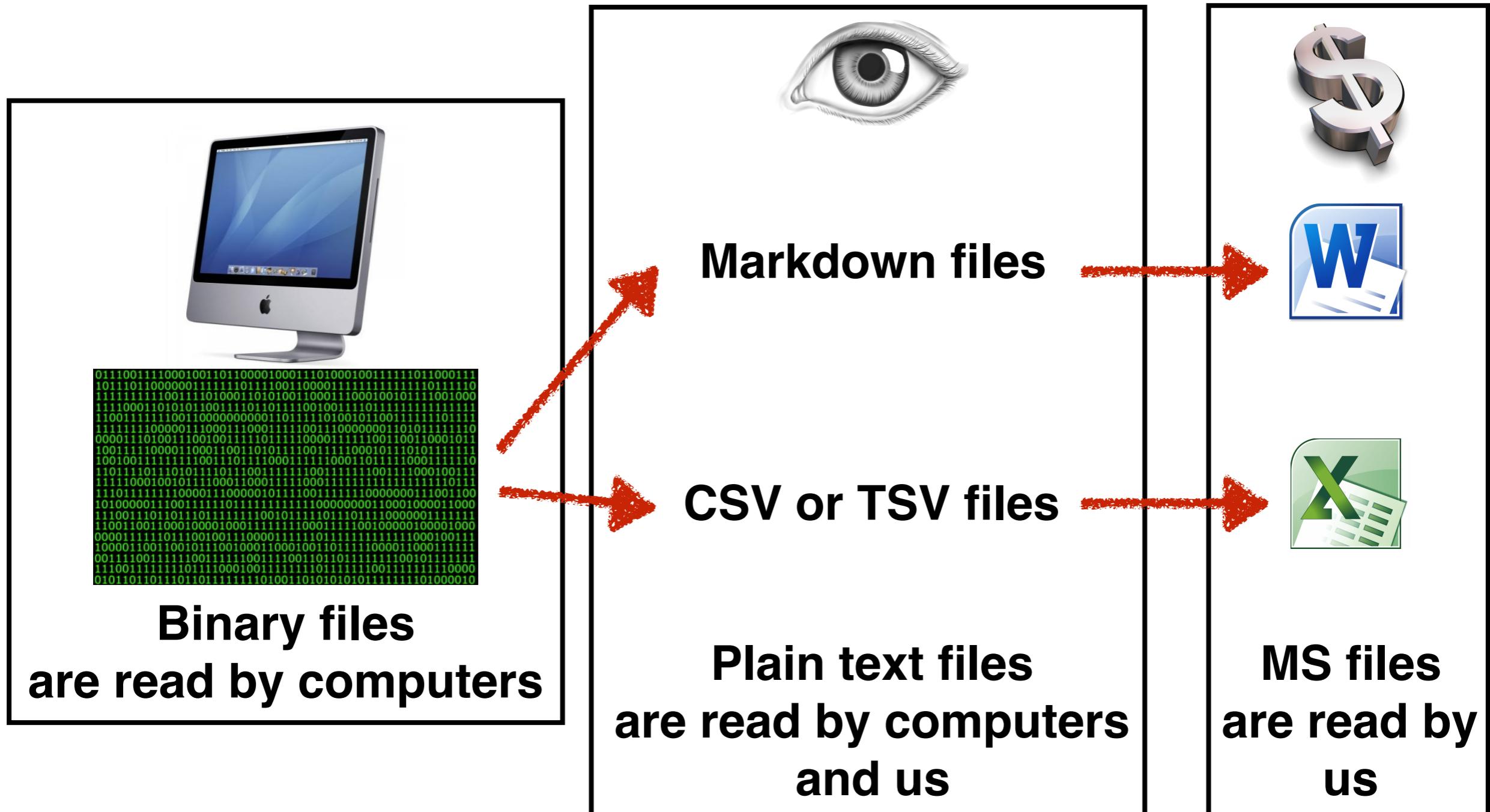
md = markdown

Markdown is the emerging standard for software documentation.

Easily written, read, and translated.

Portable to HTML, PDF, etc.

We need files that the computer and we can read.



What about that readme.md file?

md = markdown

what you type

```
Heading
=====
Sub-heading
-----
h3. Traditional html title

Paragraphs are separated
by a blank line.

Let 2 spaces at the end of a line to do a
line break

Text attributes *italic*, 
**bold**, `monospace`.

A [link](http://example.com).
<<< No space between ] and ( >>>

Shopping list:

* apples
* oranges
* pears

Numbered list:

1. apples
2. oranges
3. pears

The rain---not the reign---in
Spain.
```

what is rendered

Heading

Sub-heading

Traditional html title

Paragraphs are separated by a blank line.

Let 2 spaces at the end of a line to do a
line break

Text attributes *italic*, **bold**, `monospace`.

A [link](#).

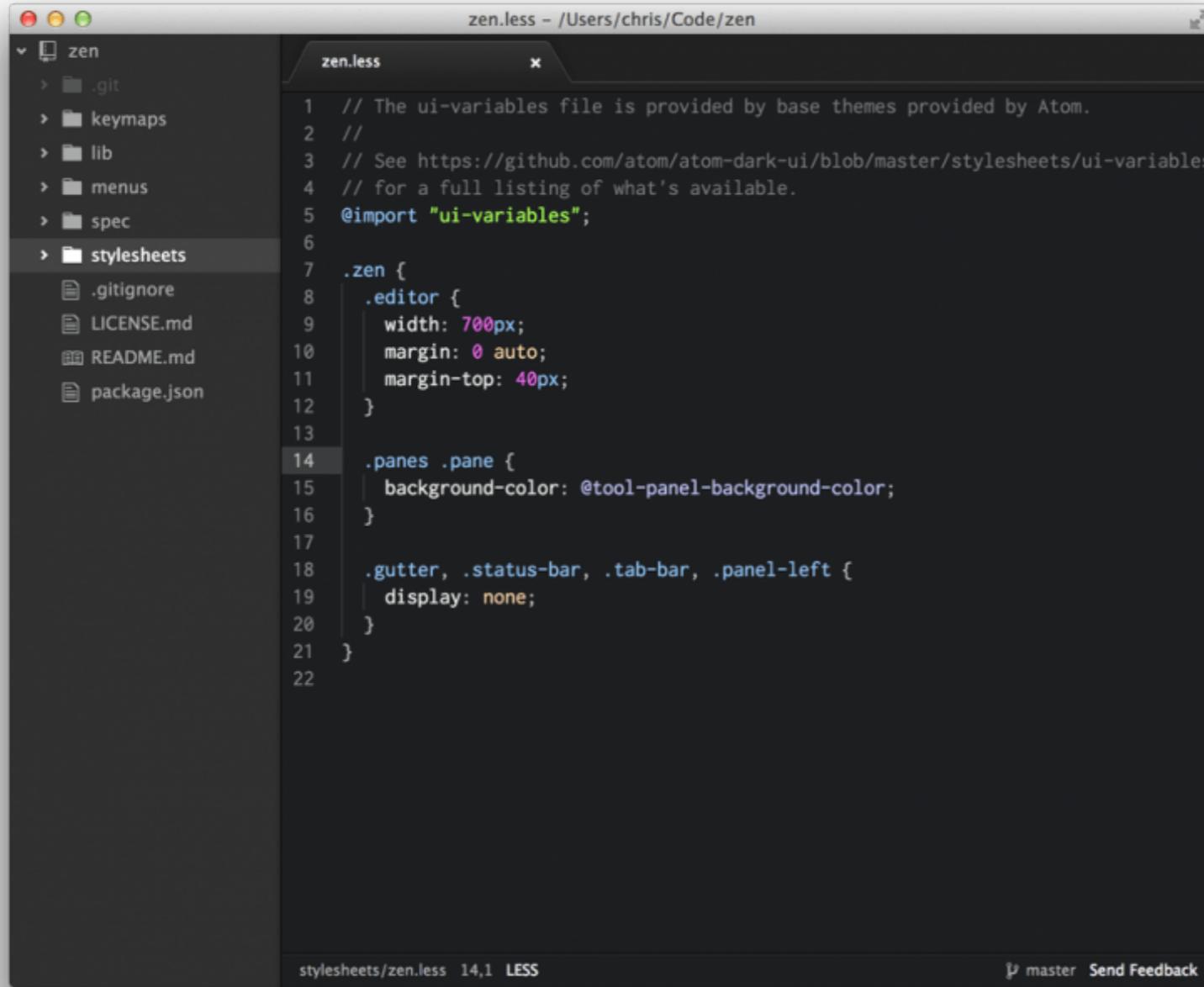
Shopping list:

- apples
- oranges
- pears

Numbered list:

1. apples
2. oranges
3. pears

The rain—not the reign—in Spain.



A screenshot of the Atom text editor interface. The title bar says "zen.less - /Users/chris/Code/zen". The left sidebar shows a project structure with "zen" as the root folder, containing ".git", "keymaps", "lib", "menus", "spec", "stylesheets" (which is selected), ".gitignore", "LICENSE.md", "README.md", and "package.json". The main editor area displays the following LESS code:

```
// The ui-variables file is provided by base themes provided by Atom.  
// See https://github.com/atom/atom-dark-ui/blob/master/stylesheets/ui-variables  
// for a full listing of what's available.  
@import "ui-variables";  
  
.zen {  
  .editor {  
    width: 700px;  
    margin: 0 auto;  
    margin-top: 40px;  
  }  
  
.panes .pane {  
  background-color: @tool-panel-background-color;  
}  
  
.gutter, .status-bar, .tab-bar, .panel-left {  
  display: none;  
}  
}
```

The status bar at the bottom shows "stylesheets/zen.less 14,1 LESS" and "master".

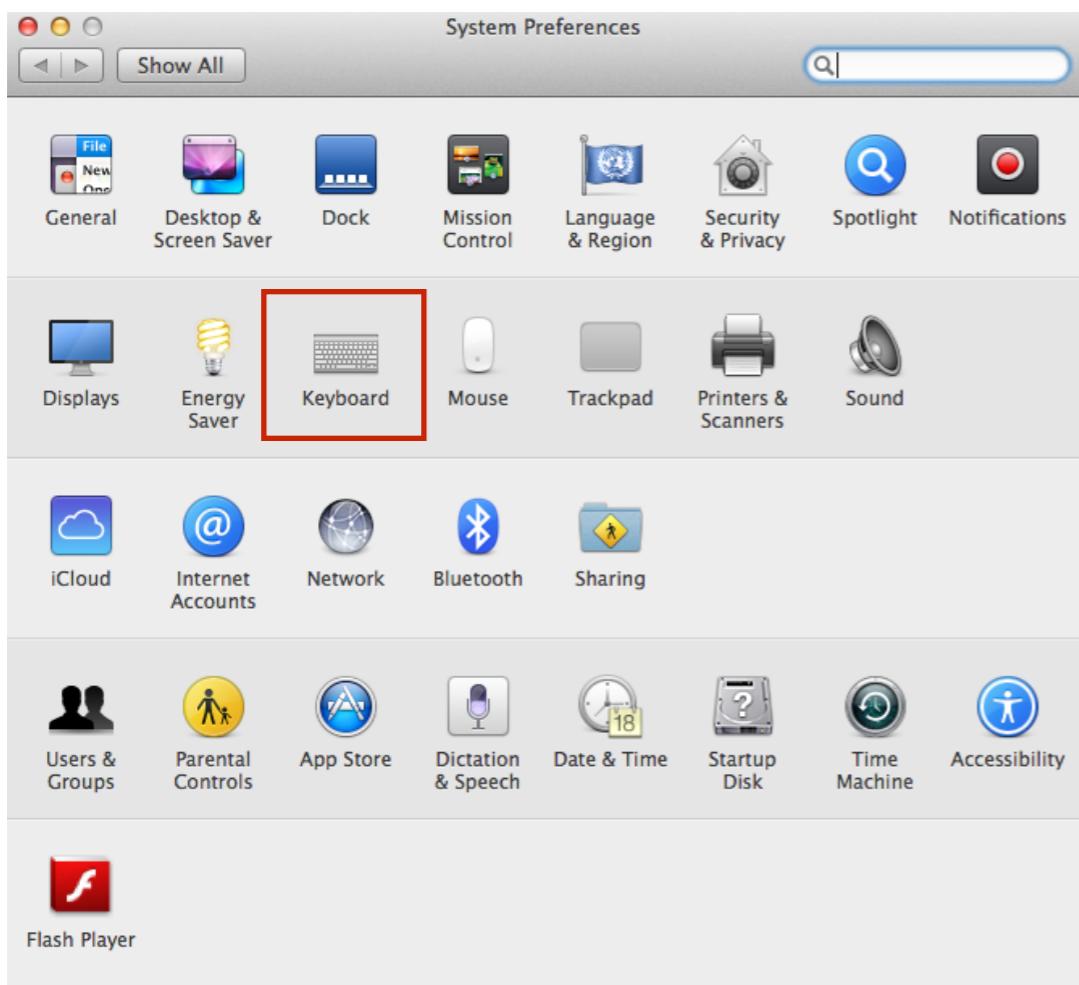
A plain text editor
different from
Microsoft Word



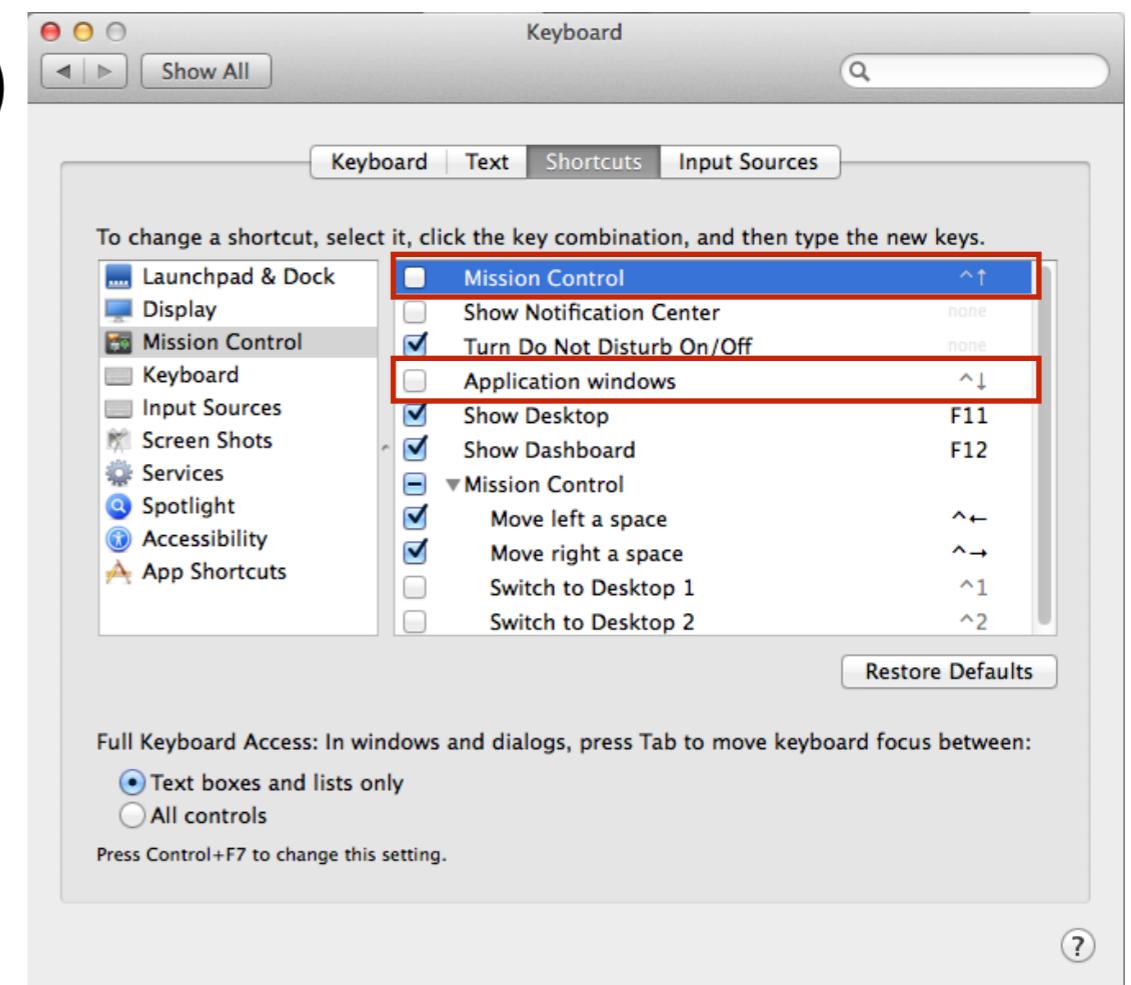
1)



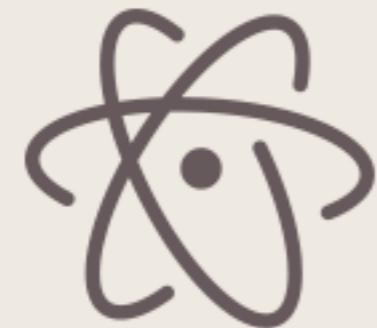
2)



3)



uncheck these two shortcuts



ATOM

Selection

Add Selection Above	⌃⇧↑
Add Selection Below	⌃⇧↓
Single Selection	⌃S
Split into Lines	⌃⌘L
Select to Top	⌃⌘↑
Select to Bottom	⌃⌘↓
Select Line	⌘L
Select Word	⌃⇧W
Select to Beginning of Word	⌃⇧B
Select to Beginning of Line	⌃⇧A
Select to First Character of Line	⌃⌘←
Select to End of Word	⌃⇧F
Select to End of Line	⌃⌘→

Open IBiS_bootcamp/resources/atom_selections_example.txt

Try the exercises, then try some of the options in the selection menu.

These functions and keyboard shortcuts are designed to *save you time*.



ATOM is customizable!

The image shows two screenshots of the Atom application. On the left, the Atom application menu is displayed, showing options like "About Atom", "View License", "Version 0.125.0", "Check for Update", "Preferences...", "Open Your Config", "Open Your Init Script", "Open Your Keymap", "Open Your Snippets", "Open Your Stylesheet", "Install Shell Commands", "Hide Atom", "Hide Others", "Show All", and "Quit". An arrow points from the "Preferences..." option in the menu to the "Settings" tab in the screenshot on the right. The right screenshot shows the "Settings" interface with sections for "Core Settings" and "Editor Settings". The "Themes" section under "Core Settings" is highlighted with a red circle, indicating it is the focus of customization. Other visible settings include "Audio Beep", "Destroy Empty Panes", "Exclude Vcs Ignored Paths", "Ignored Names (.git, .svn, .DS_Store)", "Project Home (Default: /Users/daniel/github)", "Auto Indent", "Confirm Checkout Head Revision", "Font Family", and "Font Size (14)".



ATOM is customizable!

 **Choose a Theme**

 You can also style Atom by editing [your stylesheet](#)

UI Theme Atom Light → Overall look of program
This styles the tabs, status bar, tree view, and dropdowns

Syntax Theme Proton Bat → Color of text in the editor
This styles the text inside the editor

Overall look of program

Color of text in the editor



ATOM is extensible!

A screenshot of the Atom settings interface. The sidebar on the left shows options: Settings (highlighted in blue), Keybindings, Packages (circled in red), Themes, proton, Proton Bat, and Open ~/atom. The main area shows 'Core Settings' with checkboxes for Audio Beep, Destroy Empty Panes, and Exclude Vcs Ignored Paths. It also includes sections for Ignored Names (.git, .svn, .DS_Store) and Project Home (Default: /Users/daniel/github). Below that is 'Editor Settings' with checkboxes for Auto Indent and Confirm Checkout Head Revision, and fields for Font Family and Font Size (set to 14).

Install ‘sort lines’



ATOM is extensible!

Edit

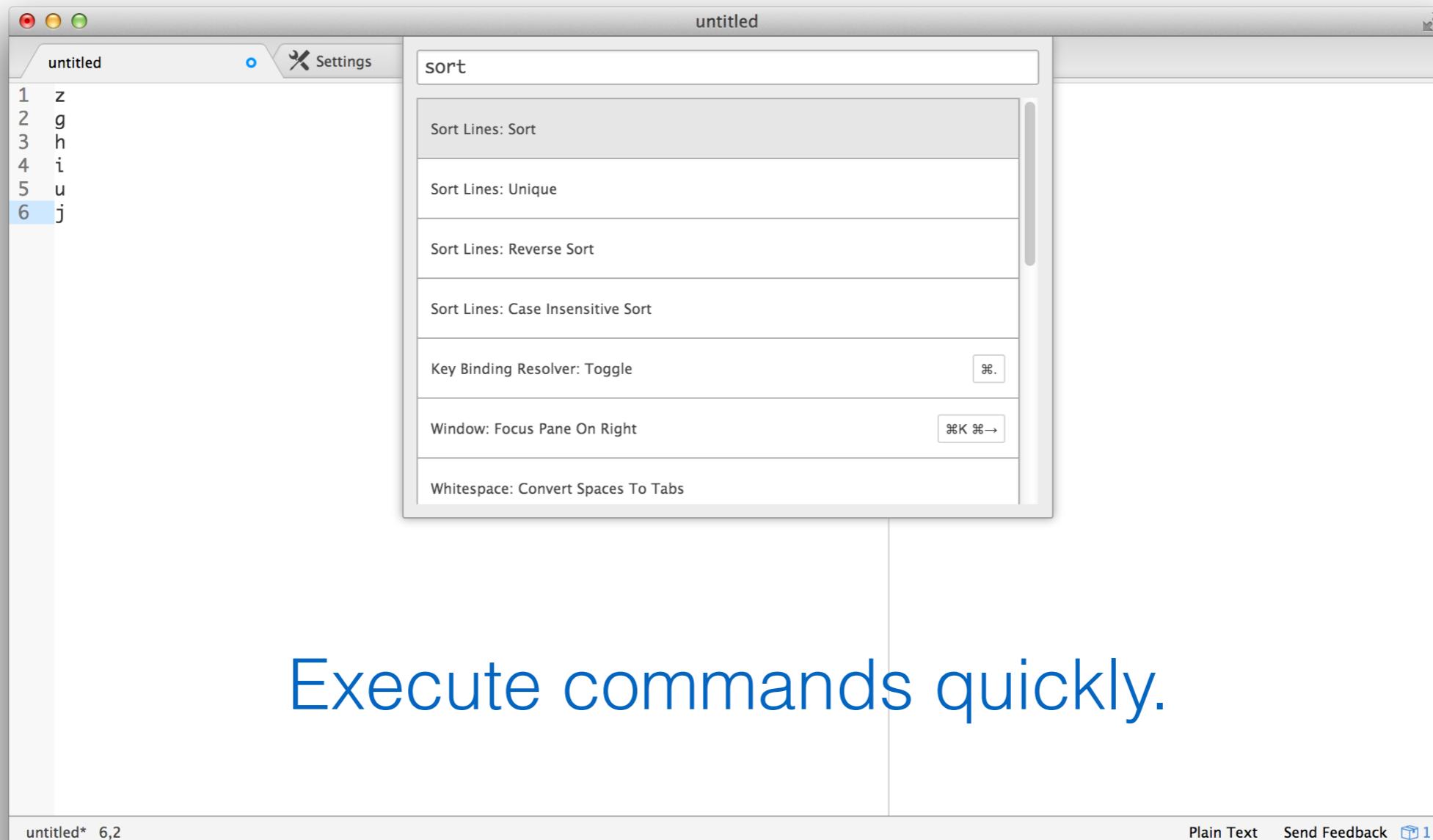
- Undo ⌘Z
- Redo ⌘Y
-
- Cut ⌘X
- Copy ⌘C
- Copy Path ⌘⇧C
- Paste ⌘V
- Select All ⌘A
-
- Toggle Comments ⌘/
- Lines ▶**
- Text ▶
- Folding ▶
- Reflow Selection ⌘⌘Q
- Bookmark ▶
- Go to Line ⌘G
- Select Grammar ⌘⇧L
-
- Start Dictation...
- Special Characters... ⌘Space

- Indent ⌘]
- Outdent ⌘[
- Auto Indent
-
- Move Line Up ⌘↑
- Move Line Down ⌘↓
- Duplicate Lines ⌘D
- Delete Line ⌘⌫
- Join Lines ⌘J**
- Sort
- Reverse Sort
- Unique
- Sort (Case Insensitive)

New functions!



ATOM is has “command palette”! Command-shift-p



Let's make readme.md from ATOM

1. Open Atom
2. Check out the welcome.md
3. Go to the untitled pane.
4. Type:

This example *project* will teach us **a lot** about basic computational biology.

5. Press command-shift-p
6. Type “markdown” and press enter
7. Check it out!
8. Close the markdown preview
9. Press control-shift-m
10. Check it out!
11. Save as “readme.md” into MyProject folder

Further experiences:

<https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet>
<http://daringfireball.net/projects/markdown/basics>

Make an ibis.md file

1. Title: IBiS is awesome
2. “awesome” in bold
3. Make it a first level header with a line underneath it
4. Add normal text below: “Our webpage is here.”
5. Add the link for the IBiS webpage to “here”.
6. Make a list of the top three reasons you love IBiS
7. Add the IBiS image to the bottom

Bootcamp outline

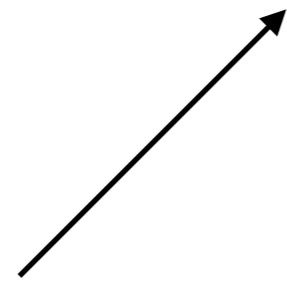
Day #1: Reproducible research, Command line introduction, Intro to R and RStudio, and data input

- **Intro to reproducible research**
- **Git and github**
- **Directory structures**
- **Basic command line**
- **Markdown**
- **More command-tool utilities**
- **R and Studio**
- **Data input**

Notes vs what you type

The command line can be personalized!

.bash_profile



A hidden file in your home directory

- Your *bash profile* is run every time you open your terminal.
- You can use your bash profile to store common settings, functions, modify commands, or personalize your terminal.

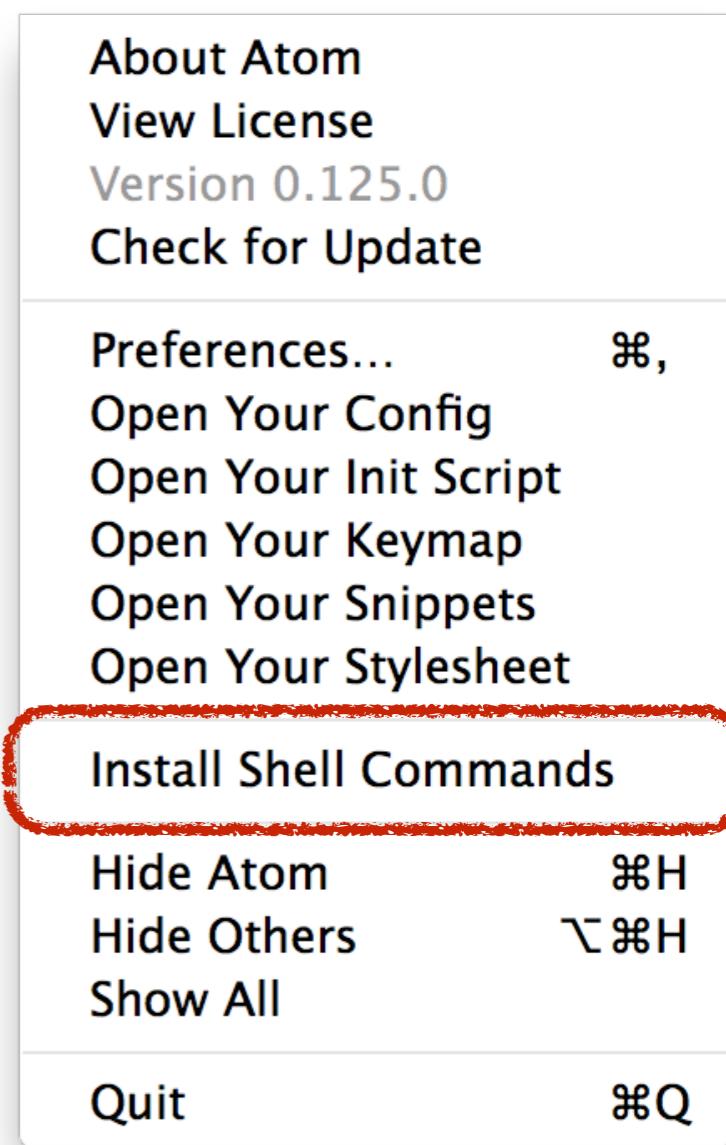
Editing your bash profile

1. Install the atom shell commands.

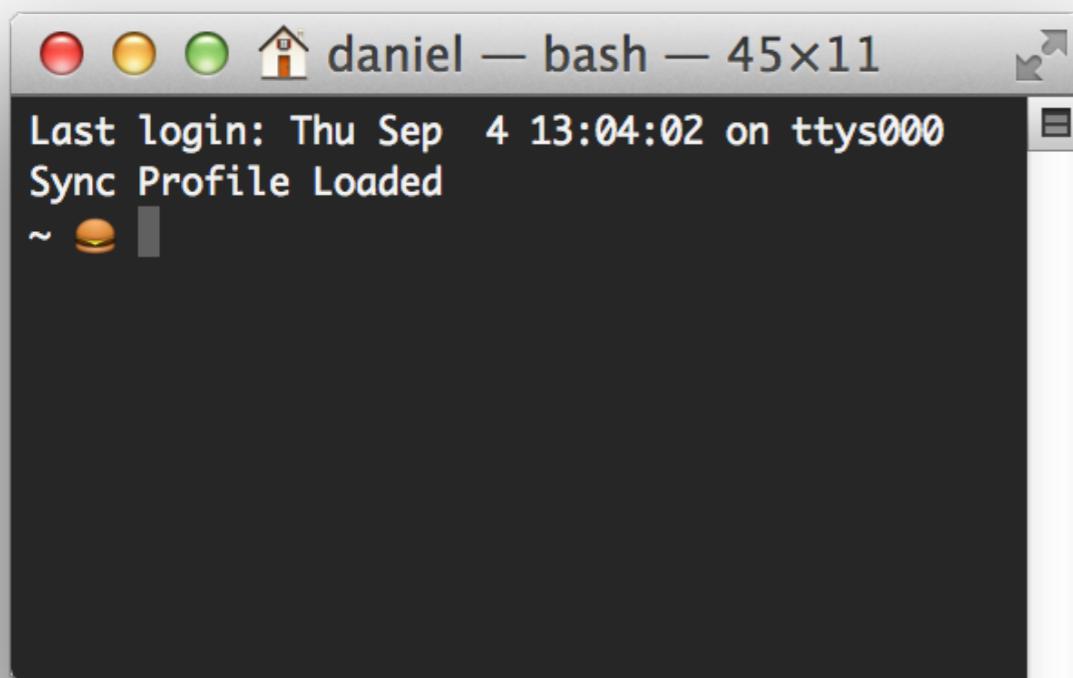
[Atom > Install Shell Commands](#)

2. Type:

`atom .bash_profile`



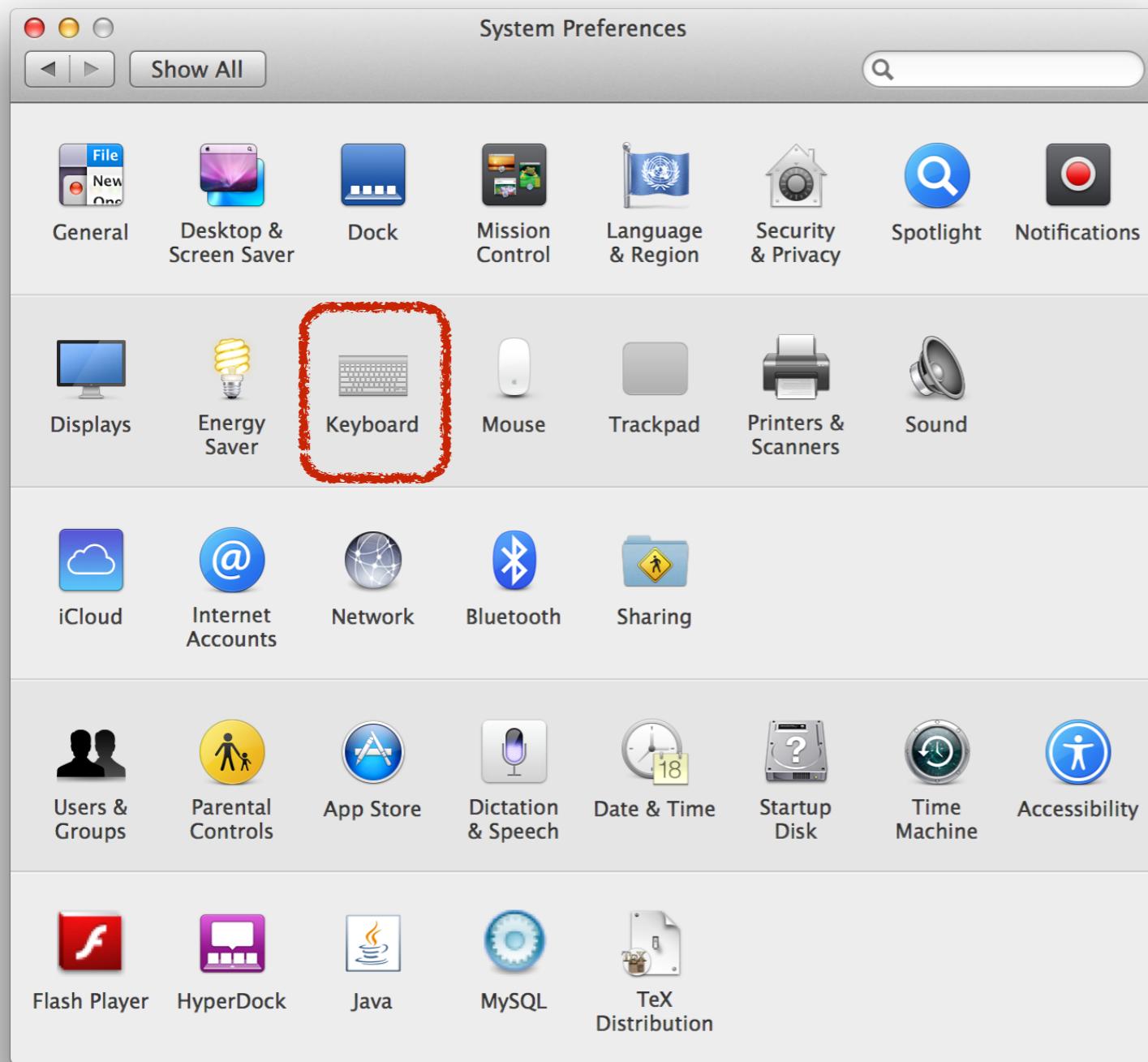
Personalize your Terminal



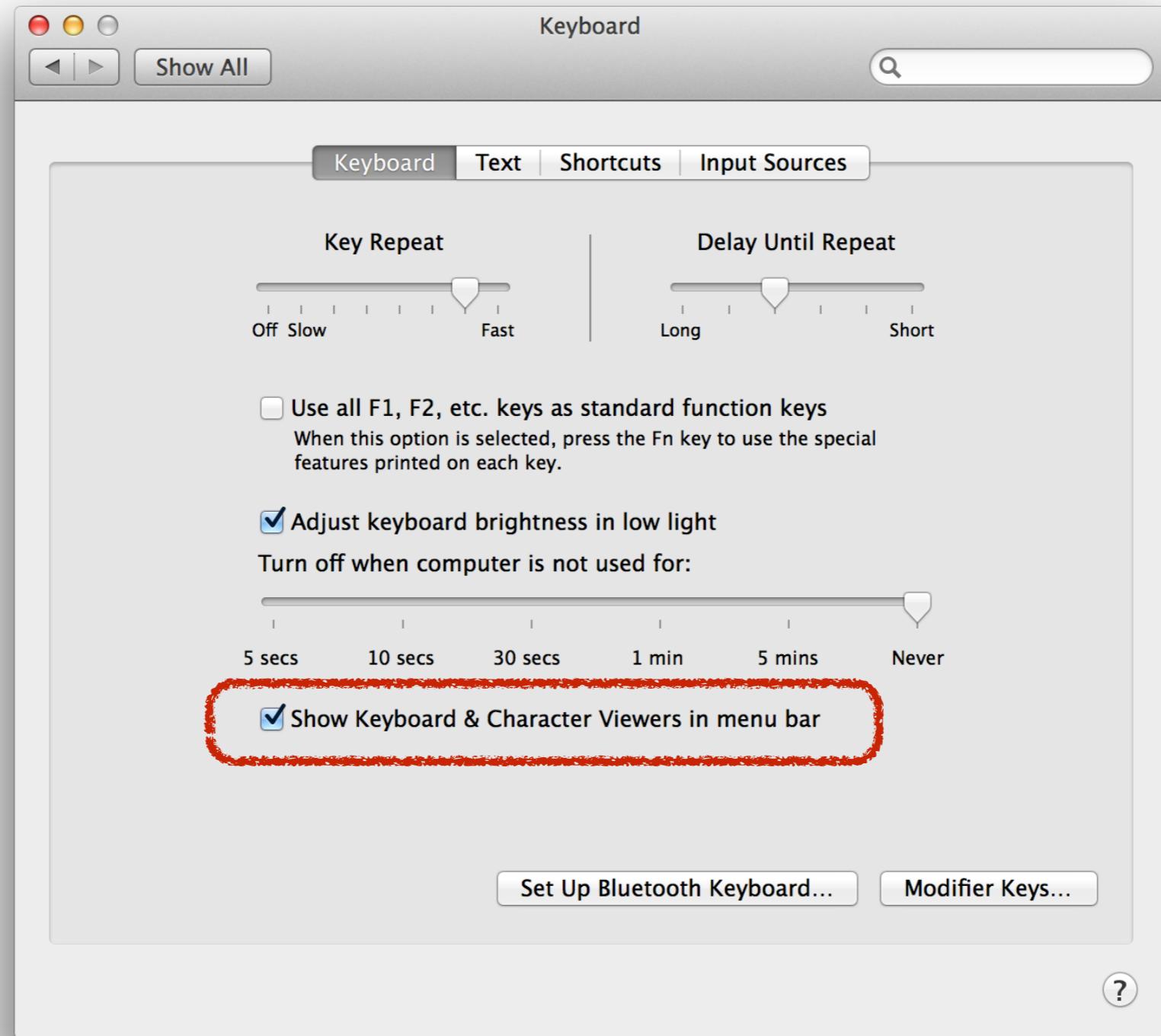
Add an icon to see command
lines



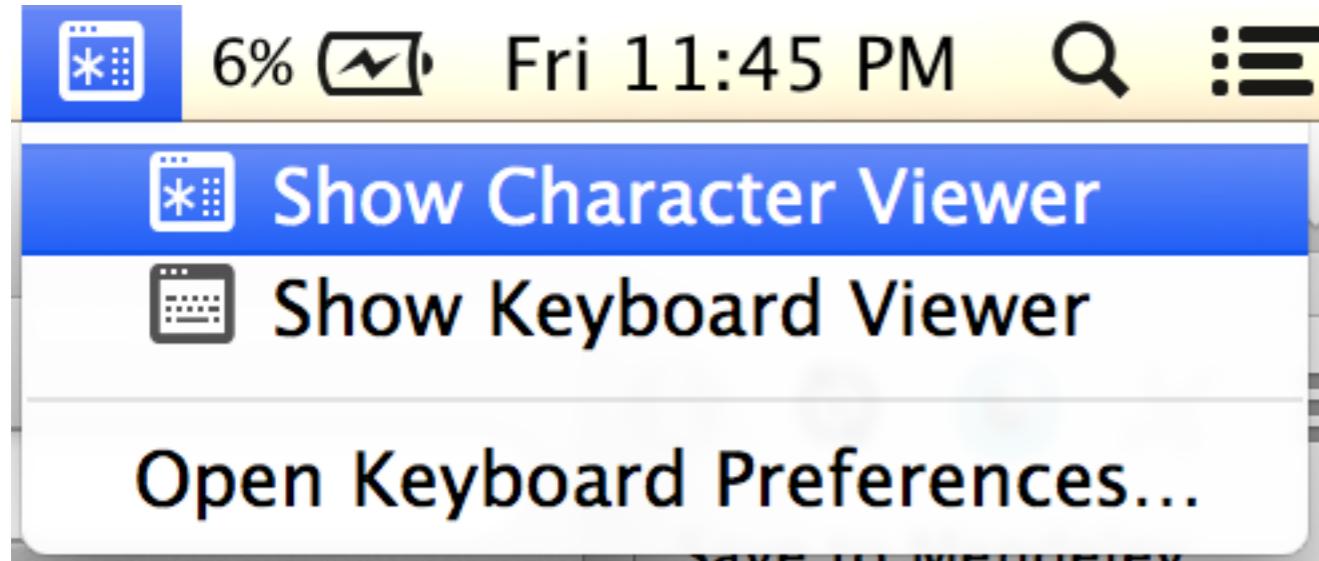
Open System Settings



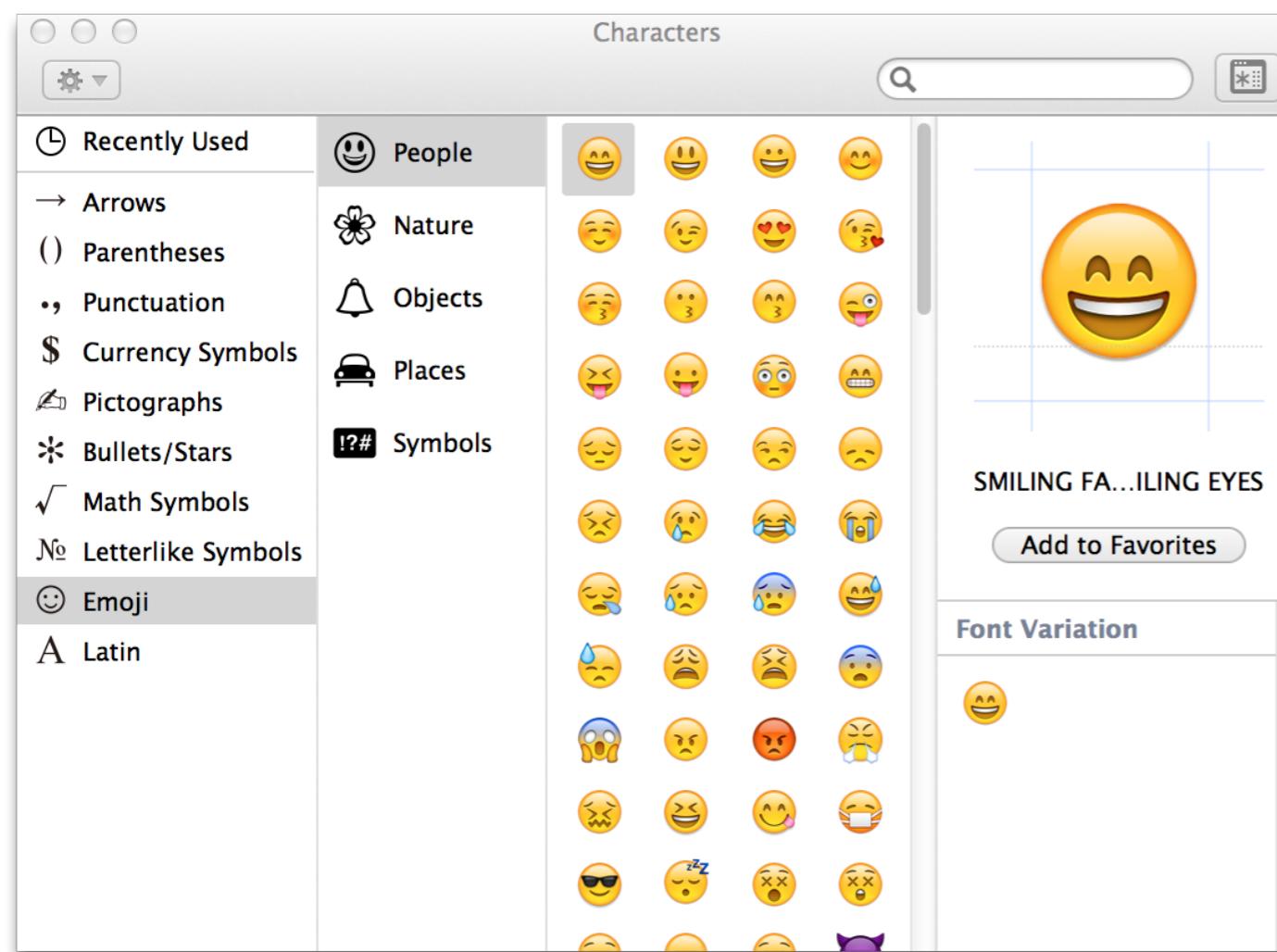
Click “Keyboard”



Check “Show Keyboard & Character Viewers in menu bar”

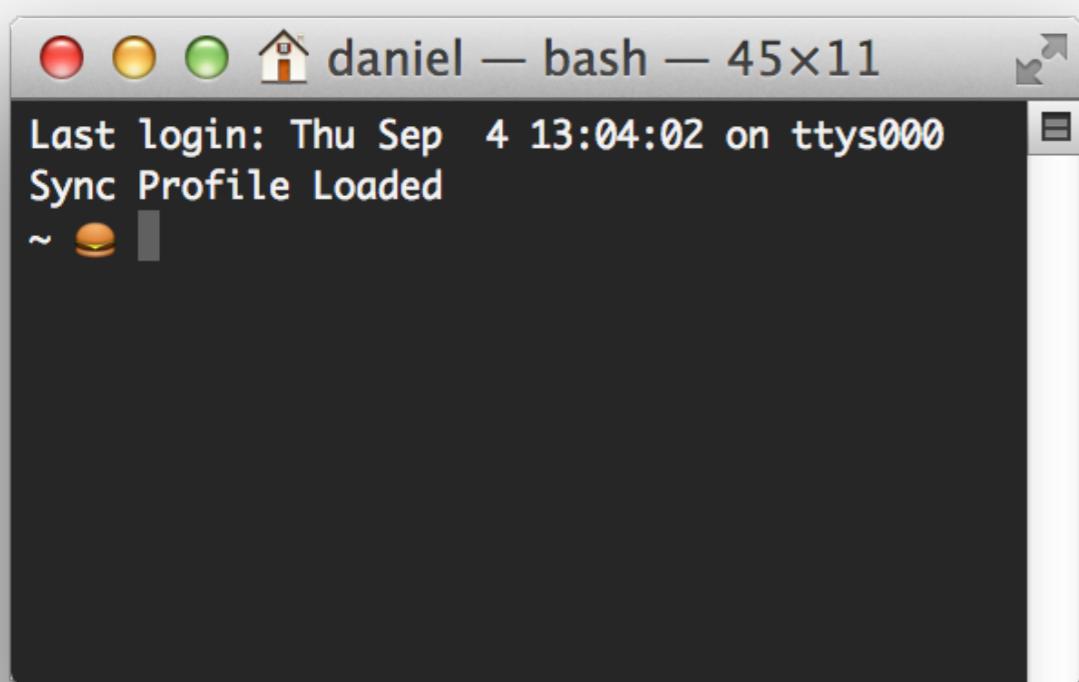


Open the character viewer.



Pick a fun icon!

Personalize your Terminal



Put your cursor here
and double click on the icon

```
export PS1="\w "
```

Shortcuts can be added with aliases

```
alias refresh="source ~/.bash_profile"  
alias dbx="cd ~/Dropbox/"
```

Typing ‘refresh’ will refresh your bash profile.

Typing ‘dbx’ will take you to the Dropbox directory.

Functions can be used to execute common tasks.

```
new_project() {  
    mkdir $1  
    touch $1/Readme.md  
    mkdir $1/Data  
    mkdir $1/Data/Raw  
    mkdir $1/Data/Processed  
    mkdir $1/Scripts  
    mkdir $1/Results  
}
```

This function will create a project directory structure.

Example:

```
new_project chip-seq-analysis
```

What to do when you are lost?



A few additional commands for UNIX

time = time how long an operation takes

screen = allows multiplexing of terminal windows

nohup = no hang ups, process errors go to a log

sed = search and edit text

awk = search, extract, or transform text

Also, check out datamash utilities

Further experiences:

<http://freeengineer.org/learnUNIXin10minutes.html>

<http://www.ee.surrey.ac.uk/Teaching/Unix/>

or just google: “unix command line tools”

Remote commands for UNIX

ssh = secure shell, remote access to UNIX systems

rsync = remote synchronization, remote copy

Let's have some fun!

1. *Cool computer voice*: Type say and whatever you want. Make sure your sound is on
2. *Get your fortune*: Type brew install fortune. Then type: fortune
3. *Download youtube videos*: Type brew install youtube-dl. Then type: youtube-dl cHK428vSIZ8
4. *Watch an asci movie*: Type: telnet towel.blinkenlights.nl 23

Bootcamp outline

Day #1: Reproducible research, Command line introduction, Intro to R and RStudio, and data input

- **Intro to reproducible research**
- **Git and github**
- **Directory structures**
- **Basic command line**
- **Markdown**
- **More command-tool utilities**
- **R and Studio**
- **Data input**

Notes vs what you type

The computational biology triumvirate



Free and open source are core principles of science

You ARE learning to code!



**is a software environment
for statistical computing**



is free!

Software



Cost

\$8,700 - \$140,000 / year

\$2150 + \$1,000s for modules

\$0



is managed by users!

Comprehensive R Archive Network

<http://cran.us.r-project.org>

Over 5000 free add-on *packages*

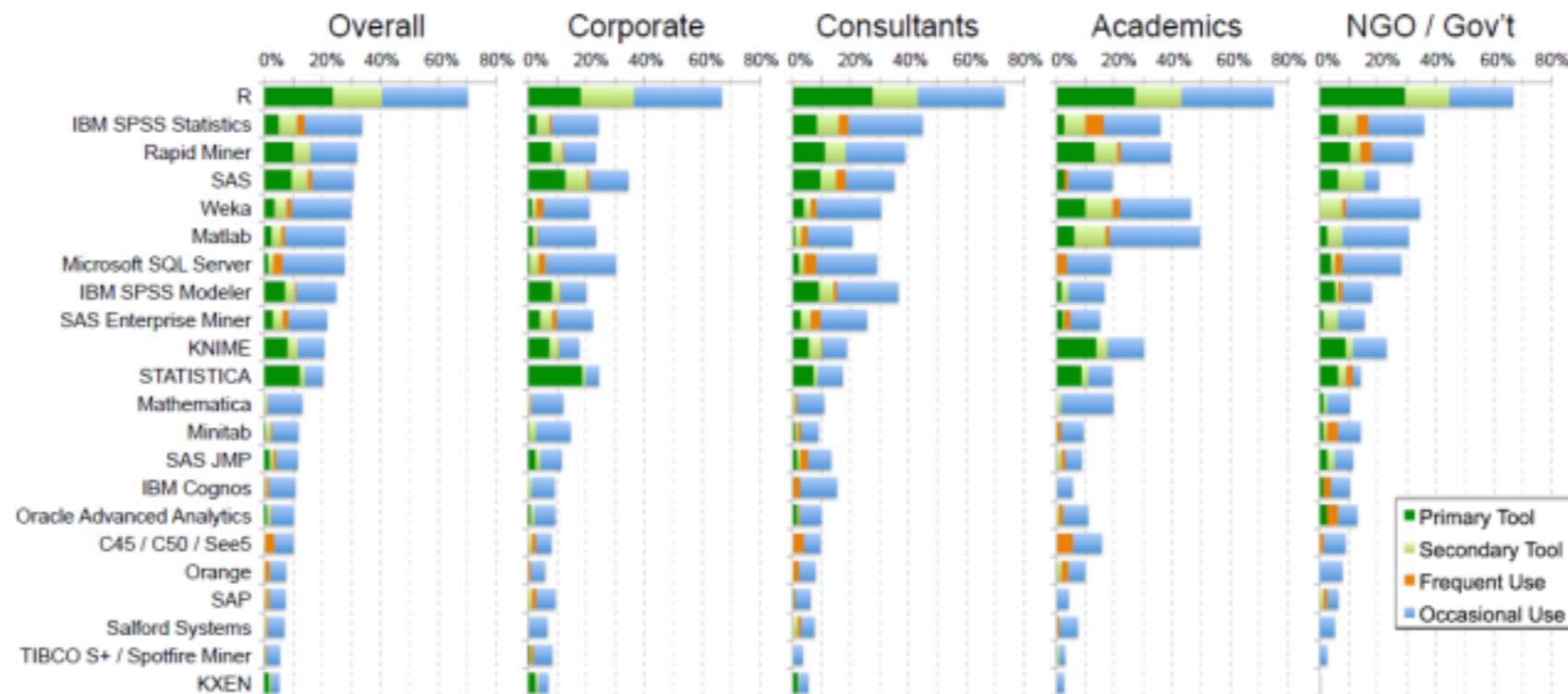
Stack Overflow

<http://stackoverflow.com>

Free help



is popular for data analysis!





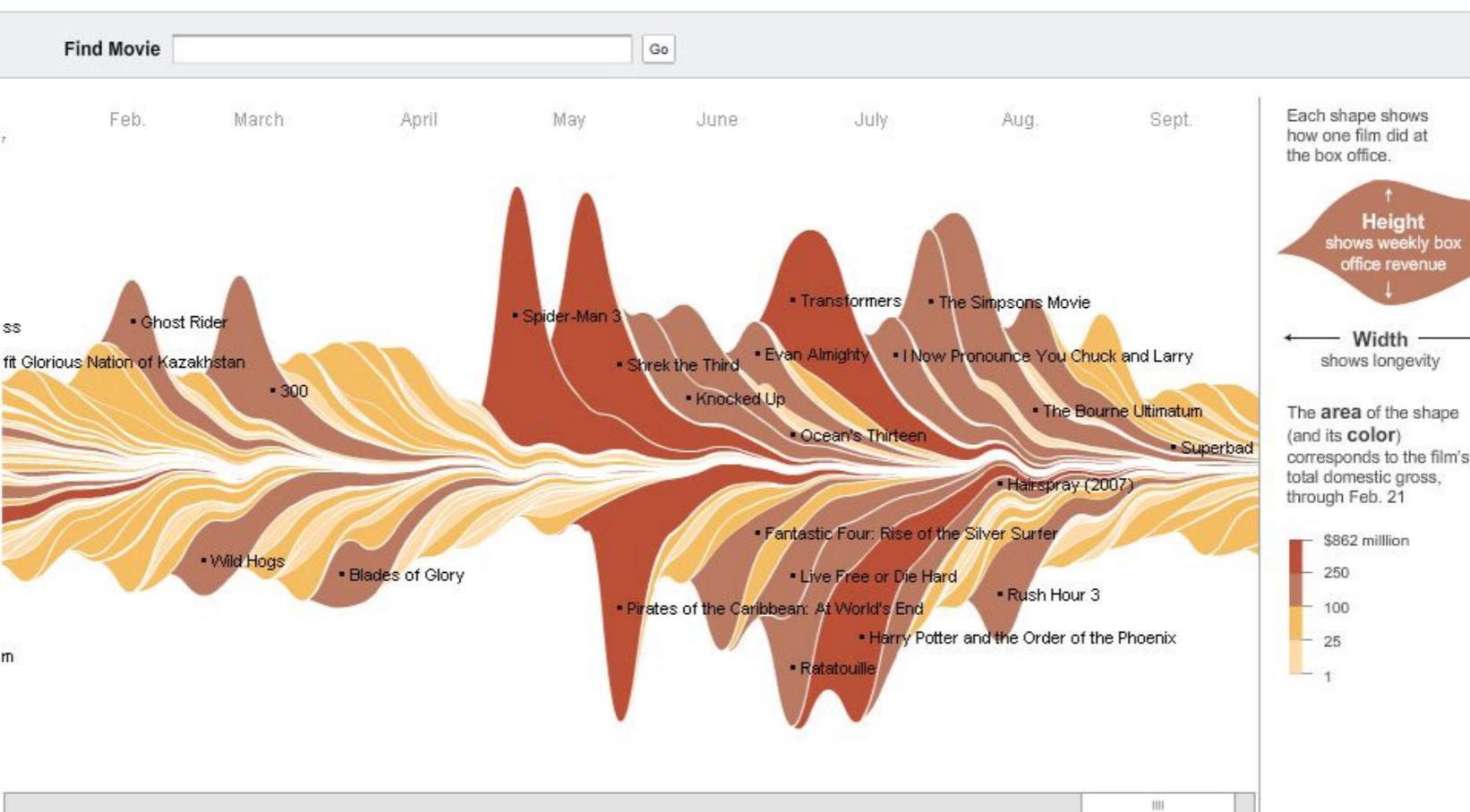
has amazing graphics!

February 23, 2008

SIGN IN TO E-MAIL OR SAVE THIS | FEEDBACK

The Ebb and Flow of Movies: Box Office Receipts 1986 — 2008

Summer blockbusters and holiday hits make up the bulk of box office revenue each year, while contenders for the Oscars tend to attract smaller audiences that build over time. Here's a look at how movies have fared at the box office, after adjusting for inflation.



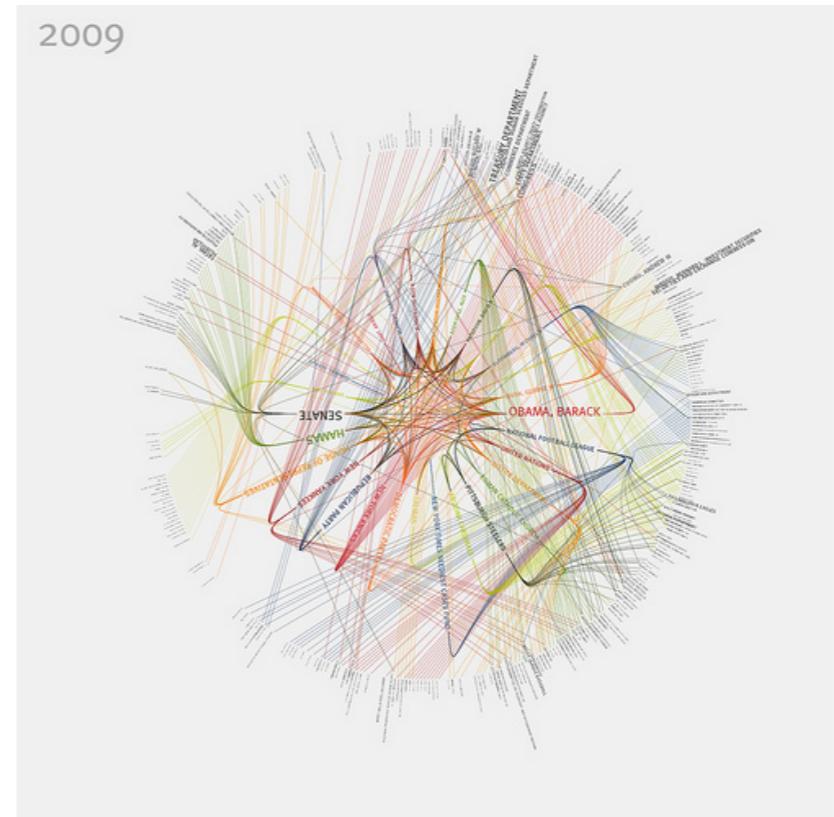
Sources: Baseline StudioSystems; Box Office Mojo

Mathew Bloch, Lee Byron, Shan Carter and Amanda Cox

New York Times favorite tool



has amazing graphics!





is a programming language!

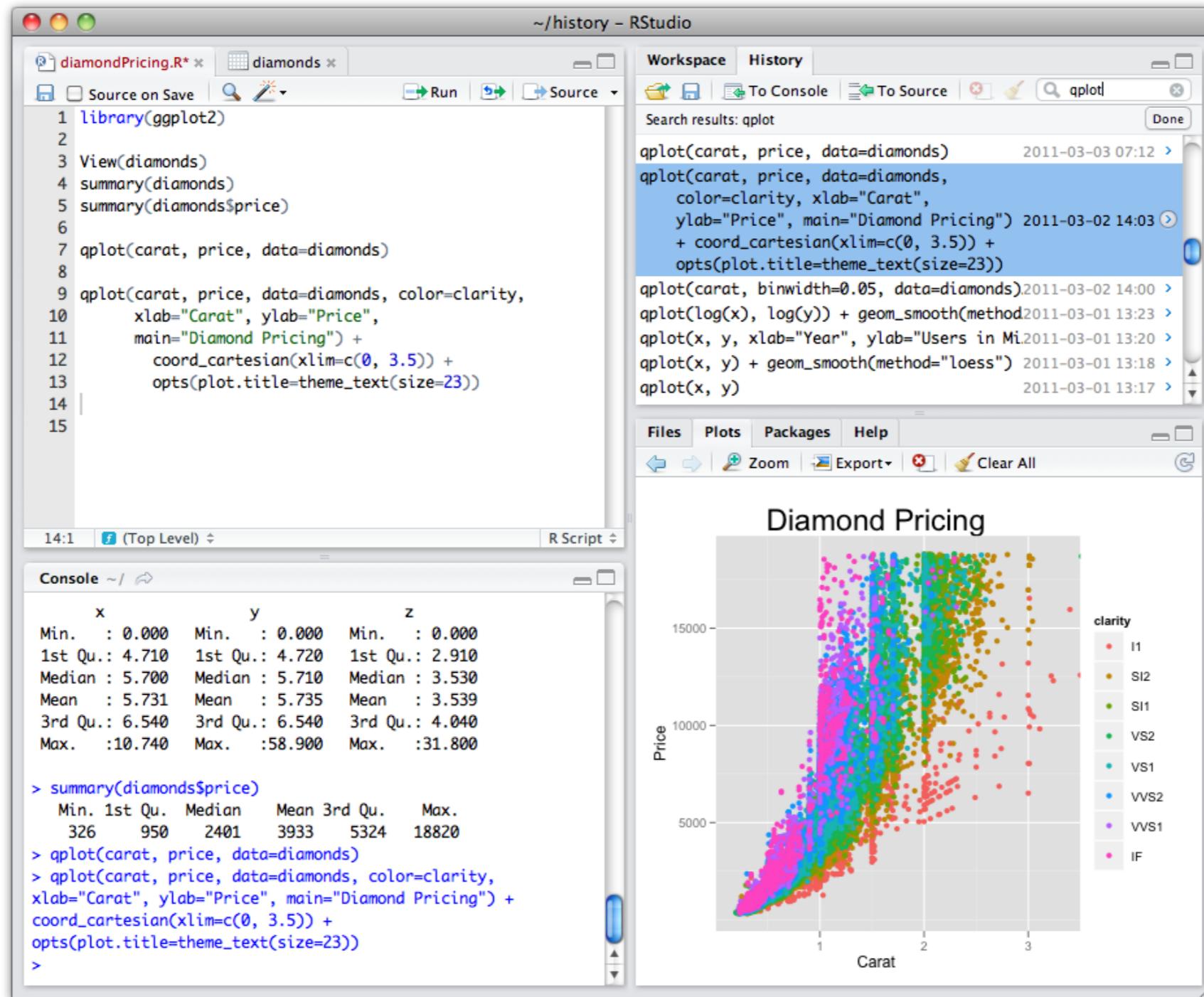
- String handling, functions, object oriented, file i/o
- Parallel programming
- Statistically orientated data structures
- Runs command line utilities, python, C, etc.
- Integrates with a variety of database systems



RStudio makes R even easier!

Editor

Console



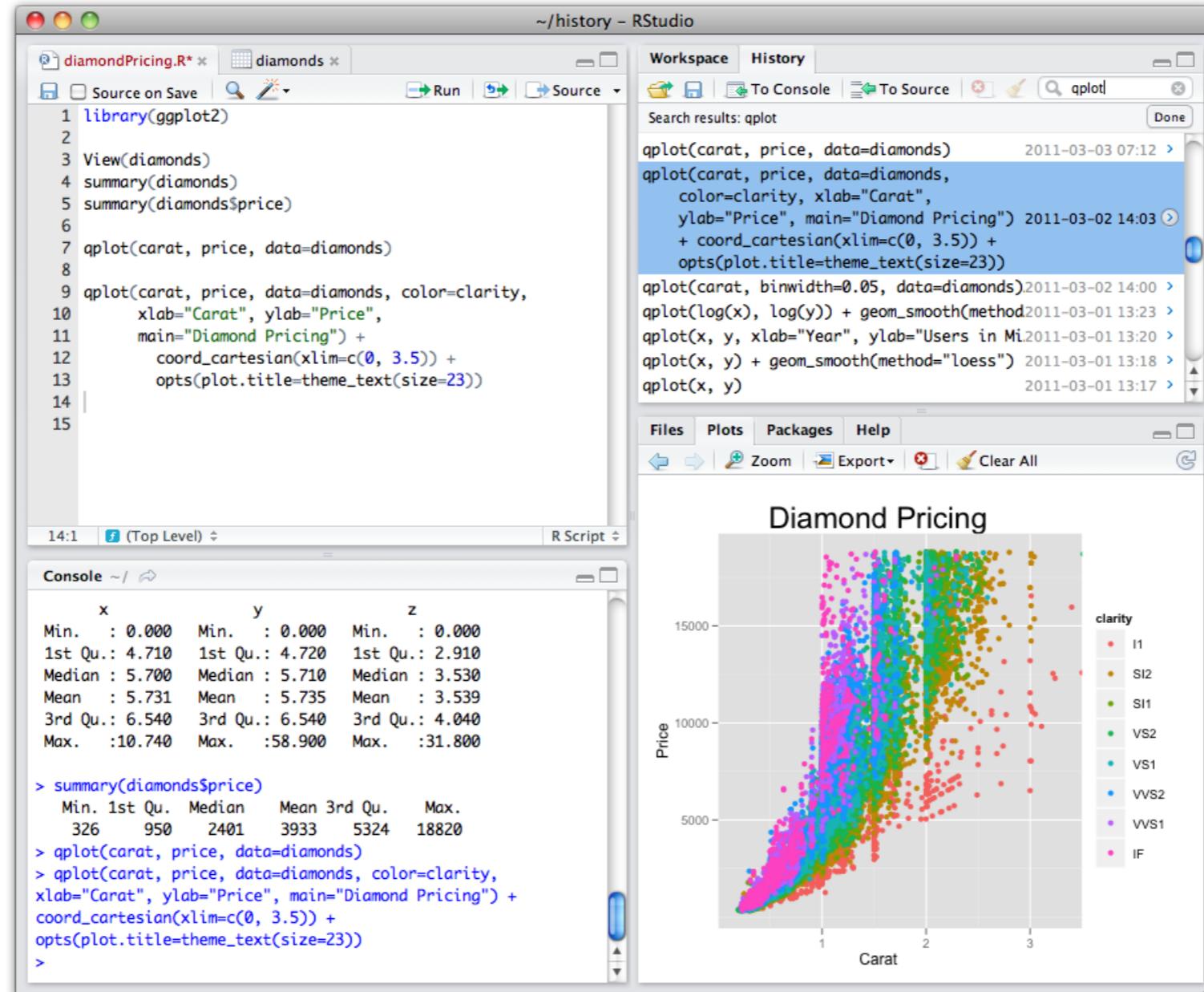
Workspace

Graphics

Two main ways to interact with R

Editor

Console

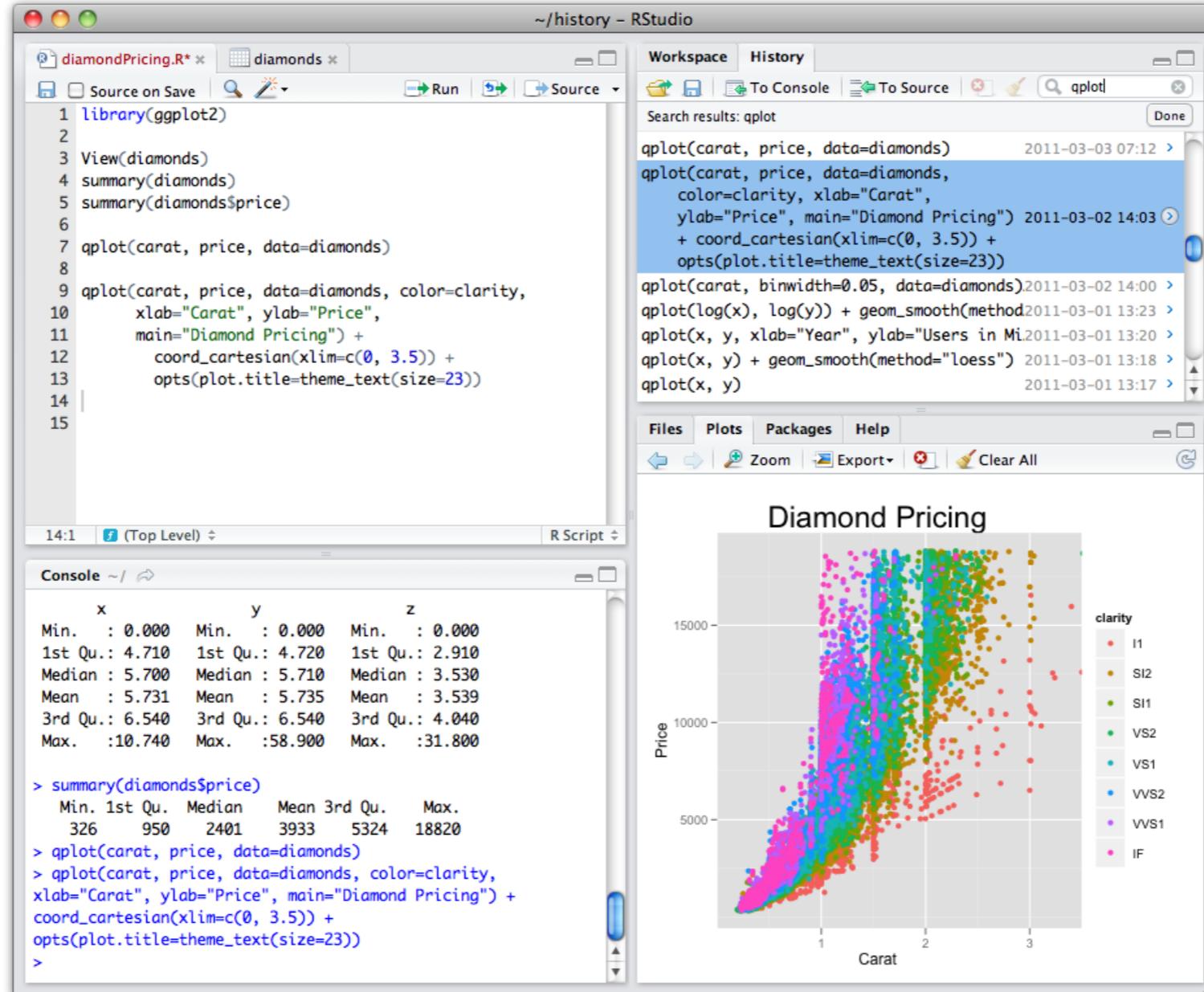


Think of the console as asking R questions about data.

Two main ways to interact with R

Editor

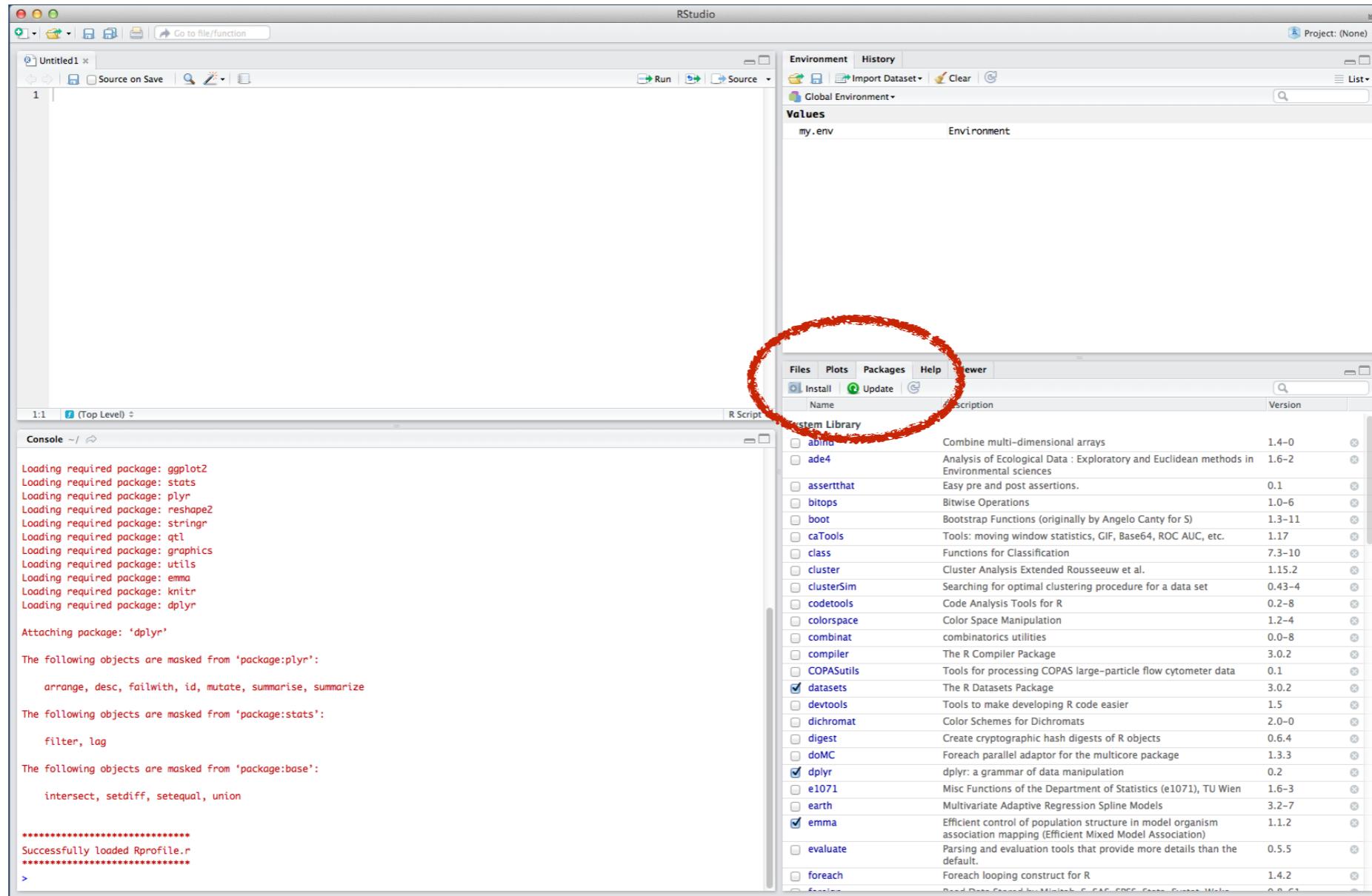
Console



The editor is a plain text file editor (like Atom)

For scripting and saving work (markdown too)

Let's install some packages



1. Click on Packages in bottom right pane
2. Click on Install
3. Make sure the Install from drop down is CRAN
4. Enter lubridate

Basic R syntax

> The prompt

+ at the prompt means that something isn't complete

Functions are words followed by (), i.e. `mean(x)` for calculating the mean of x

Equations are entered as assignments, i.e. `y <- mean(x)` for y is equal to the mean of x.

If you can't remember a function, use `apropos()`. For example, `apropos("var")` will display every item in the workspace with "var"

Same goes for `??var`

You can also use `?function`. For example, `?mean`

Try `????mean`

is used to enter comments not read by R

Basic Data Types in R

Atomic vector types

character

numeric

integer

logical

complex

raw

atomic = only holds data of one type

```
> # Assign 1 into y
```

```
> y <- 1
```

```
> y
```

```
> # Assign 1 to 10 into y
```

```
> y <- 1:10
```

```
> y
```

```
> str(y)
```

```
> view(y)
```

Basic Data Types in R

Atomic vector types

character
numeric
integer
logical
complex
raw

atomic = only holds data of one type

```
> # Assign 1 into y                      > length(y)
> y <- 1                                > y <- as.numeric(y)
> y                                         > y
                                              > str(y)
                                              > as.character(y)

> # Assign 1 to 10 into y
> y <- 1:10
> y

> str(y)
> view(y)
```

Basic Data Types in R

Atomic vector types

character
numeric
integer
logical
complex
raw

atomic = only holds data of one type

```
> length(y)          > y <- logical(5)
> y <- as.numeric(y) > y
> y                  > as.numeric(y)
> str(y)
> as.character(y)    > y <- c(1,2,3)
> str(y)
> y <- c("TRUE", "TRUE", "FALSE")
> str(y)
> y <- c("Bob", "Rick", "Ishwar")
> str(y)
```

Basic data manipulation

Adding elements:

```
> y <- c(y, "Carole")
> y
> y <- c("Greg", y)
> y
```

Missing data (NA):

```
> z <- c(0.5, NA, 0.7)
> z
> is.na(z)
```

Sequences:

```
> series <- 1:10
> series
> seq(1:10)
> seq(from =1, to =10, by = 0.1)
> LETTERS
> letters
> as.roman(2015)
```

Special values:

```
> 1/0
> 0/0
```

Data structures in R

Vectors are one-dimensional data sets of any one data type

Data frames are two-dimensional data sets of any data type

Lists are groups of vectors, data frames, or other lists.

How do we explore data objects?

`names(df)` = gives the names of columns of df

`rownames(df)` = gives the names of rows of df

`colnames(df)` = gives the names of columns of df

`dim(df)` = outputs the number of rows then columns of df

`length(df)` = outputs the length of df

`summary(df)` = outputs summary statistics of df

Let's make our own functions

```
> add20 <- function(x) {x + 20}

> add20(10)

> # This function will convert Fahrenheit to Celsius

> FtoC <- function(fahrenheit) {
  celsius = (fahrenheit - 32) * 5 / 9
  return(celsius)}
```

What do we do with “big data”?



Flow of data analysis

1. Read Today
 2. Tidy
 3. Process
 4. Plot
 5. Present
-
- ```
graph TD; A[1. Read] -- Today --> B[2. Tidy]; B -- Tomorrow --> C[3. Process]; C -- Thursday --> D[4. Plot]; D -- Thursday --> E[5. Present]
```

# We all owe Hadley Wickham a beer.



- Chief Scientist at RStudio
- Made tidyverse package to clean up data
- Made dplyr package for grammar of data manipulation
- ggplot2 package for beautiful R graphics
- Made reshape2, stringr, lubridate, devtools, plyr, rvest, etc.

# Choose your own data set for analysis

Baby names, life tables, births

```
devtools::install_github("hadley/babynames")
```

All 2015 police shootings

```
https://github.com/washingtonpost/data-police-shootings
```

Electric power consumption

Go To: <https://goo.gl/rGS5GN>

All UFO sightings

Go To: <https://goo.gl/R6KmP2>

Every Daily Show guest with Jon Stewart

Google: "analysis of guests Jon Stewart Stephen Turner", go to RPubs link

Query IMDB and pull data

```
devtools::install_github("hrbrmstr/omdbapi")
```

All LEGO sets from 1970-2014

```
devtools::install_github("seankross/lego")
```

**Others...you find them...the web is filled with data  
Check out:**

**[http://koaning.io/posts/fun\\_datasets.html](http://koaning.io/posts/fun_datasets.html)**

**<https://data.cityofchicago.org/>**

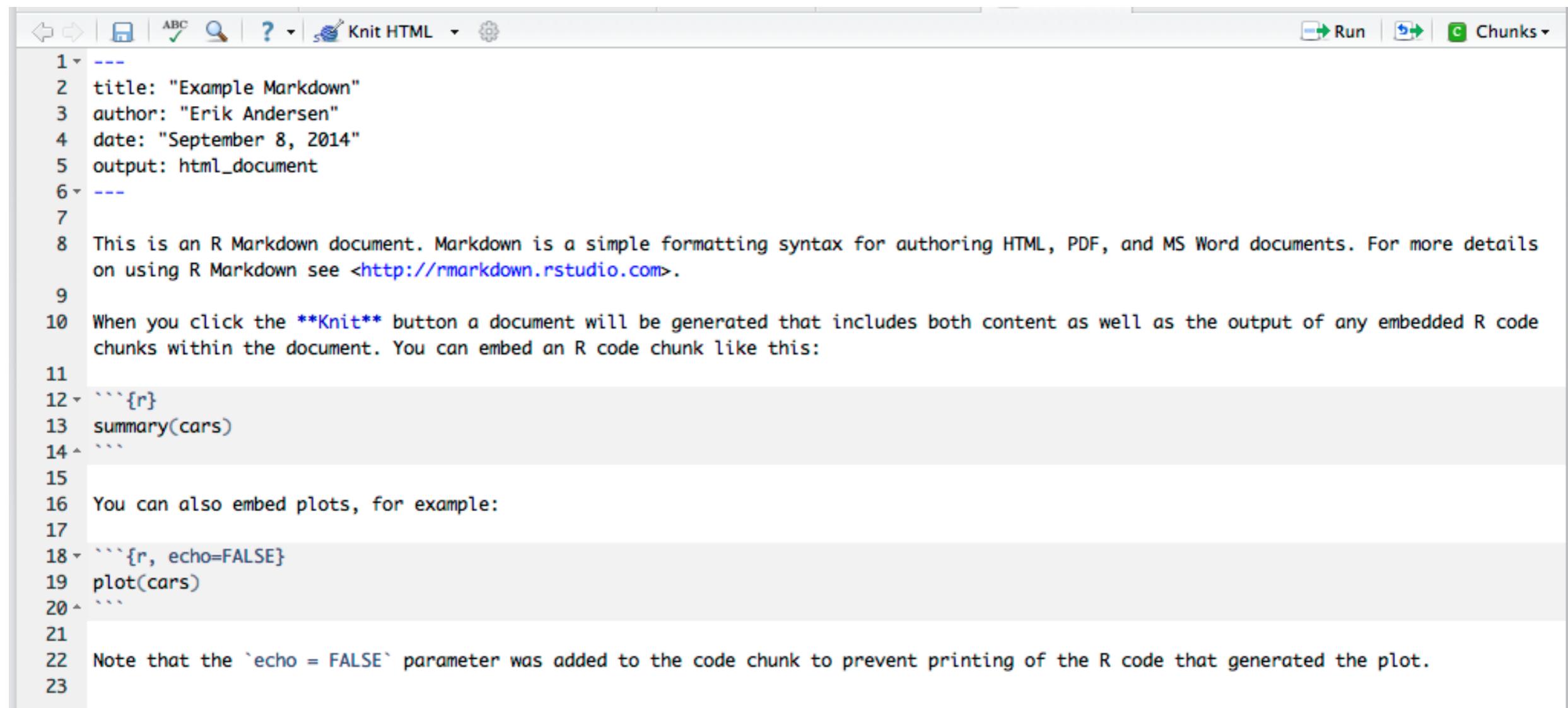
**<https://github.com/datasets>**

**<https://github.com/caesar0301/awesome-public-datasets>**

# Time to find some data



# RStudio can generate markdown reports

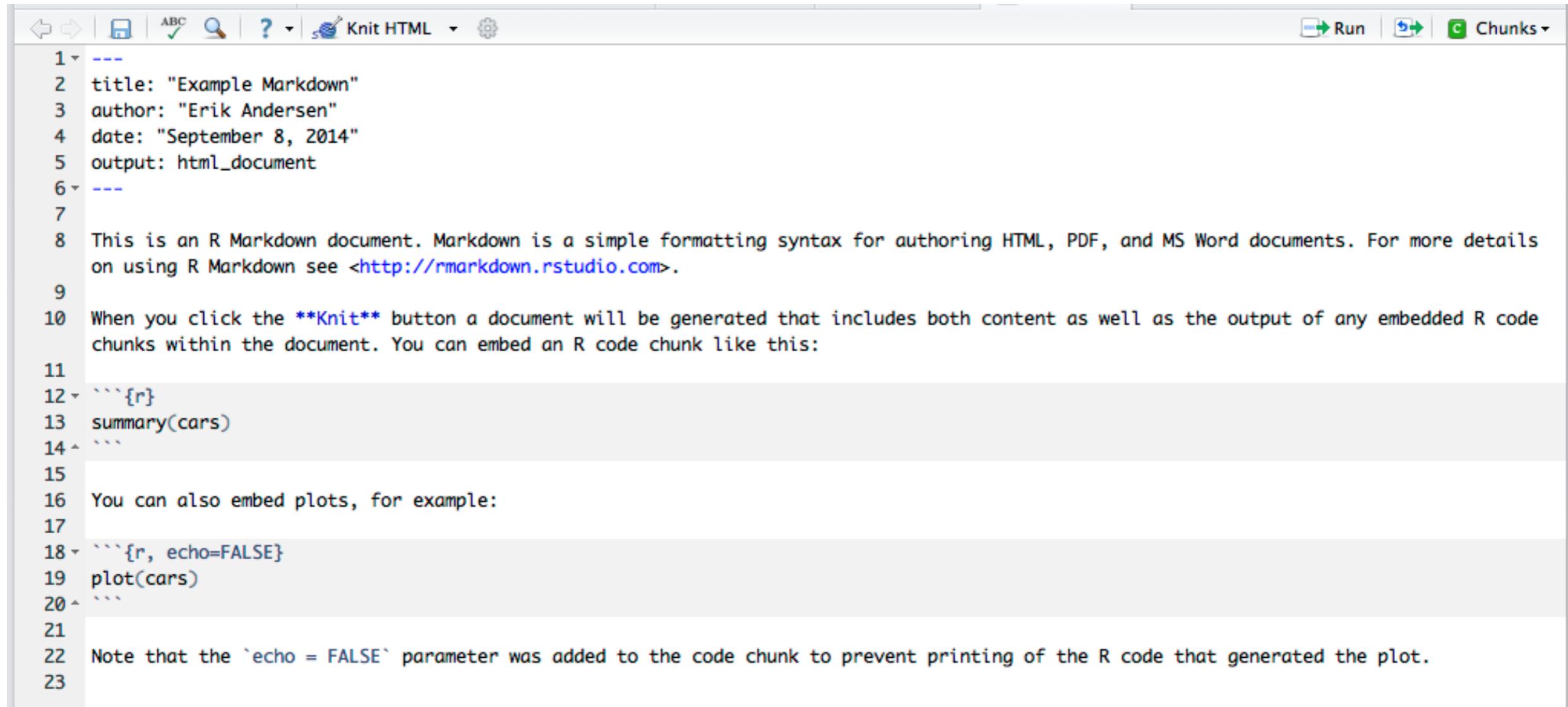


The screenshot shows the RStudio IDE interface with an R Markdown document open. The top menu bar includes icons for back, forward, file, ABC, search, help, and a Knit HTML button. The main code editor area contains the following R Markdown code:

```
1 ---
2 title: "Example Markdown"
3 author: "Erik Andersen"
4 date: "September 8, 2014"
5 output: html_document
6 ---
7
8 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details
on using R Markdown see <http://rmarkdown.rstudio.com>.
9
10 When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code
chunks within the document. You can embed an R code chunk like this:
11
12 ```{r}
13 summary(cars)
14```
15
16 You can also embed plots, for example:
17
18 ```{r, echo=FALSE}
19 plot(cars)
20```
21
22 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
23
```

- 1. Go to File, New File, R Markdown**
- 2. Install the necessary packages**
- 3. Write your R code and Markdown text**
- 4. Knit an HTML report**

# Make a markdown for your data analysis project



The screenshot shows the RStudio interface with an R Markdown file open. The toolbar at the top includes icons for back, forward, ABC, search, help, Knit HTML (which is currently selected), and run. The code editor displays the following R Markdown content:

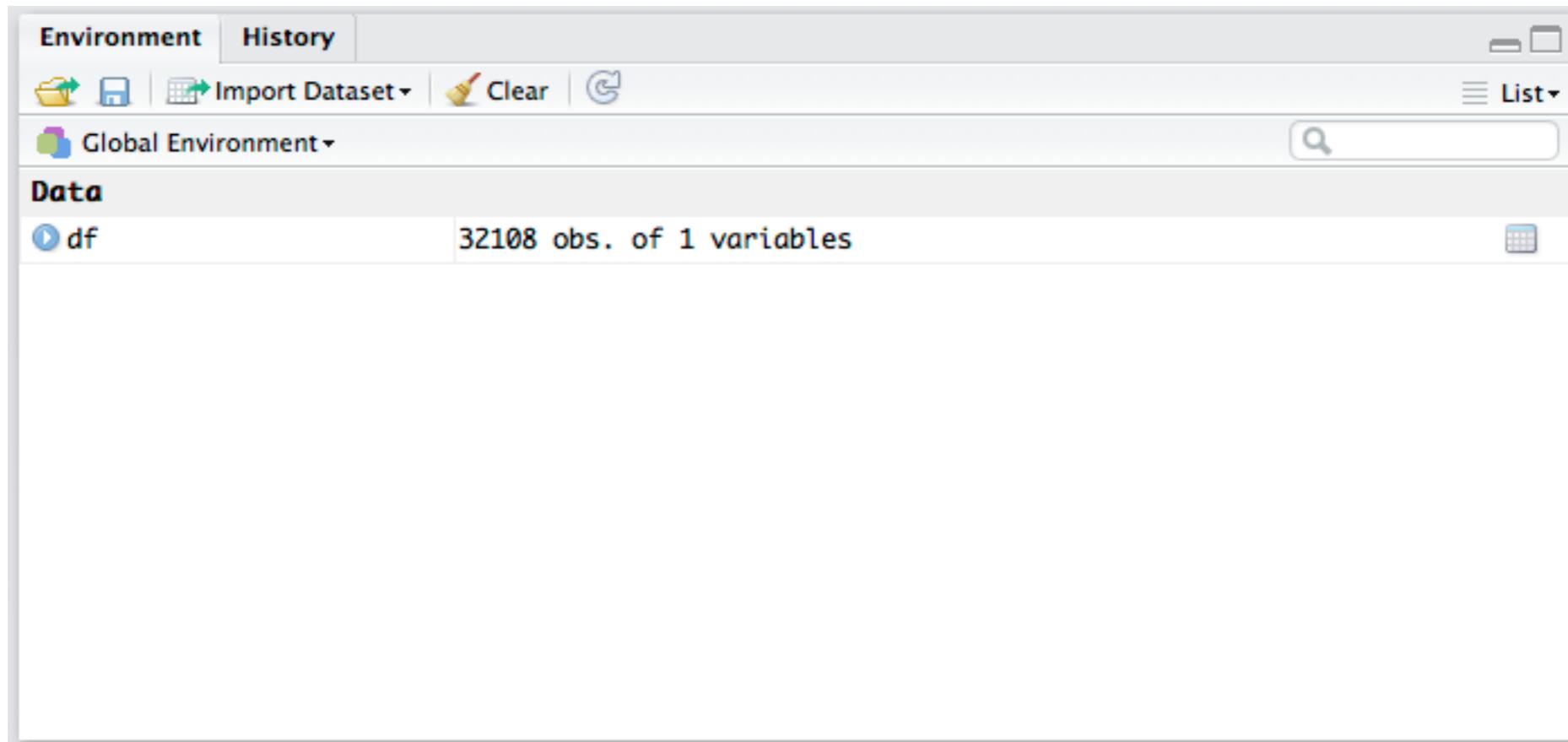
```
1 ---
2 title: "Example Markdown"
3 author: "Erik Andersen"
4 date: "September 8, 2014"
5 output: html_document
6 ---
7
8 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details
on using R Markdown see <http://rmarkdown.rstudio.com>.
9
10 When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code
chunks within the document. You can embed an R code chunk like this:
11
12 ```{r}
13 summary(cars)
14 ```
15
16 You can also embed plots, for example:
17
18 ```{r, echo=FALSE}
19 plot(cars)
20 ```
21
22 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
23
```

- 1. Write a short description of the data and ideas you might have**
- 2. Take a chunk to read in the data**
- 3. Think of analyses you might want to do  
(time series, averages, etc.)**
- 4. Prepare to clean the data!**

# Reading in data

**read.delim()** is a generic function to read in simple text files, like csv or tsv files.

```
df <- read.delim(file="~/Desktop/IBiS-Bootcamp/data/sorter_data.txt", header=TRUE, sep="\t")
```



A new data object is in your workspace. Click on it. Type `df` to display all of it.

Try `head(df)` and `tail(df)`.

Try `summary(df)`.

Try `View(df)`.

# Reading in data

Other packages have much more powerful data reading capabilities.

Try `readr`, `read_csv` or `read_tsv()`

Try `rio`, `import()`

A new data object is in your workspace. Click on it. Type `df` to display all of it.

Try `head(df)` and `tail(df)`.

Try `summary(df)`.

Try `view(df)`.

# Reading in data

```
1 v ---
2 title: "20160911_nutrition"
3 author: "Erik Andersen"
4 date: "September 11, 2016"
5 output: html_document
6 v ---
7 For the 2016 bootcamp, I will analyze my nutrition data for the last year. The data were entered manually using the MyFitnessPal app.
8 |
9 v ````{r, echo=F}
10 library(rio)
11 library(lubridate)
12 library(dplyr)
13 library(ggplot2)
14 ^```
15
16 v ##### Read in data
17 v ````{r}
18 nut <- import("~/Dropbox/Personal/QuantifiedSelf/Nutrition-Summary2-2015-08-26-to-2016-08-28.csv")
19 ^````
```

# Observing data

It looks like the tail has some data that do not match the pattern of the rest of the file.

```
df <- read.delim(file "~/Desktop/IBiS-Bootcamp/data/sorter_data.txt", header=TRUE, sep="\t", nrows=32093)
```

Type `str(df)` to look at the structure of the file

```
Console ~/
' ends field 1 on line 1 when detecting types: Green PMT voltage 600
> str(df)
'data.frame': 32093 obs. of 27 variables:
 $ Id : int 0 1 2 3 4 5 6 7 8 9 ...
 $ Plate : int 1 1 1 1 1 1 1 1 1 1 ...
 $ Row : chr "A" "A" "A" "A" ...
 $ Column : int 1 1 1 1 1 1 1 1 1 1 ...
 $ Clog : chr "N" "N" "N" "N" ...
 $ Scan.rate: int 2500 2500 2500 2500 2500 2500 2500 2500 2500 ...
 $ Status.sort: int 0 0 0 0 0 0 0 0 0 ...
 $ Status.sel : int 40 40 40 40 40 40 40 40 40 ...
 $ TOF : int 52 556 88 639 531 34 247 152 96 79 ...
 $ EXT : int 120 2375 130 2697 2520 36 534 474 210 178 ...
 $ Green : int 2 148 3 205 151 1 16 66 20 4 ...
 $ Yellow : int 3 193 5 241 170 3 21 26 7 5 ...
 $ Red : int 0 90 2 102 71 2 14 11 4 2 ...
 $ PH.Ext : int 0 0 0 0 0 0 0 0 0 ...
 $ PW.Ext : int 0 0 0 0 0 0 0 0 0 ...
 $ PC.Ext : int 0 0 0 0 0 0 0 0 0 ...
 $ PH.Green : int 0 0 0 0 0 0 0 0 0 ...
 $ PW.Green : int 0 0 0 0 0 0 0 0 0 ...
 $ PCGreen : int 0 0 0 0 0 0 0 0 0 ...
 $ PH.Yellow: int 0 0 0 0 0 0 0 0 0 ...
 $ PW.Yellow: int 0 0 0 0 0 0 0 0 0 ...
 $ PCYellow : int 0 0 0 0 0 0 0 0 0 ...
 $ PH.Red : int 0 0 0 0 0 0 0 0 0 ...
 $ PW.Red : int 0 0 0 0 0 0 0 0 0 ...
 $ PCRed : int 0 0 0 0 0 0 0 0 0 ...
 $ Time.Stamp: int 2933417 2933417 2933432 2933432 2933463 2933463 2933463 2933495 2933495 2933495 ...
 $ X : logi NA NA NA NA NA NA ...>
```

# Data structures in R

Data.frames are data organized into rows and columns.

Data frames are organized by [row,column]. You must use the comma!

You can look at a vector of the data frame using \$, i.e. `df$TOF`

Or you can specify the row or column to output, i.e. `df[, 9]` outputs the ninth column and `df[, "TOF"]` outputs the column named “TOF”

```
Console ~ /
>
> df[,"TOF"]
 [1] 52 556 88 639 531 34 247 152 96 79 54 133 64 471 417 21 491 400 504 163 25 436 57 36
[25] 190 29 574 476 394 51 584 74 172 373 26 49 92 31 25 521 177 108 266 59 57 435 26 135
[49] 62 197 52 38 236 97 545 438 82 28 129 76 67 48 105 140 147 22 88 74 24 265 162 21
[73] 31 122 50 20 70 53 329 27 71 80 158 517 523 448 40 30 86 49 519 134 26 59 49 78
[97] 65 46 91 219 259 21 85 128 244 216 59 188 182 56 43 75 51 60 136 114 80 77 52 58
[121] 219 26 203 81 25 127 190 53 63 54 37 40 468 197 172 59 85 196 45 23 24 245 58 209
[145] 314 400 229 488 99 35 260 46 103 199 421 116 481 133 112 40 55 164 51 161 212 161 128 212
[169] 64 23 491 47 20 22 69 471 23 120 28 51 524 124 265 20 69 22 62 481 103 32 53 143
[193] 48 37 203 116 89 31 45 255 73 281 296 105 472 49 20 43 20 46 113 92 143 194 46 87
[217] 194 23 70 525 54 52 51 46 36 59 71 27 506 22 331 73 847 438 58 160 50 55 402 31
[241] 208 524 58 50 78 479 54 102 55 73 295 516 109 38 121 142 127 132 486 44 101 335 66 393
[265] 94 205 77 172 205 565 29 133 54 443 511 34 97 139 182 25 39 27 182 50 60 101 56 59
[289] 146 44 206 48 118 324 36 310 127 537 194 59 46 111 411 61 439 74 32 51 91 115 46 250
[313] 467 229 47 58 181 25 103 35 184 42 295 185 406 119 32 61 549 635 84 186 111 179 49 142
```

# **Take some time to move through your data**

1. Are there missing values? `is.na()`
2. What are the column names? `colnames()`
3. What are the dimensions? `dim()`
4. What are the row values? `rownames()`
5. Are the data organized long or wide?

# Some thoughts about organizing raw data

- Be consistent.
- Write dates as YYYY-MM-DD or YYYYMMDD.
- Fill in all of the cells.
- Put just one thing in a cell.
- Make it a rectangle.
- Create a data dictionary.
- No calculations in the raw data files.
- Don't use font color or highlighting as data.
- Choose good names for things.
- Make backups.
- Use data validation to avoid data entry mistakes. Unit tests!
- Save the data in plain text files.

from Karl Broman (UW Madison), check out his fantastic blog  
Other great rules here: <https://github.com/jtleek/datasharing>

**The most important part of data analysis  
is to *think* about your data**

What do you want to test?

How will you show that conclusion?

Put the goal at the top of your markdown report

**Time to read in data, explore your data,  
and plan analyses**



# What to do when you are lost?



<http://www.r-bloggers.com>

<http://gettinggeneticsdone.blogspot.com>

<http://rseek.org>

# Day #1 Homework

1. Sign up for github
2. Star our class repository
3. Download Github desktop and login with your Github login
4. Go and explore github! Find some cool code.
5. Install the swirl package
6. Type `swirl()`
7. Take the introduction to R programming class
8. Play with your new data set