

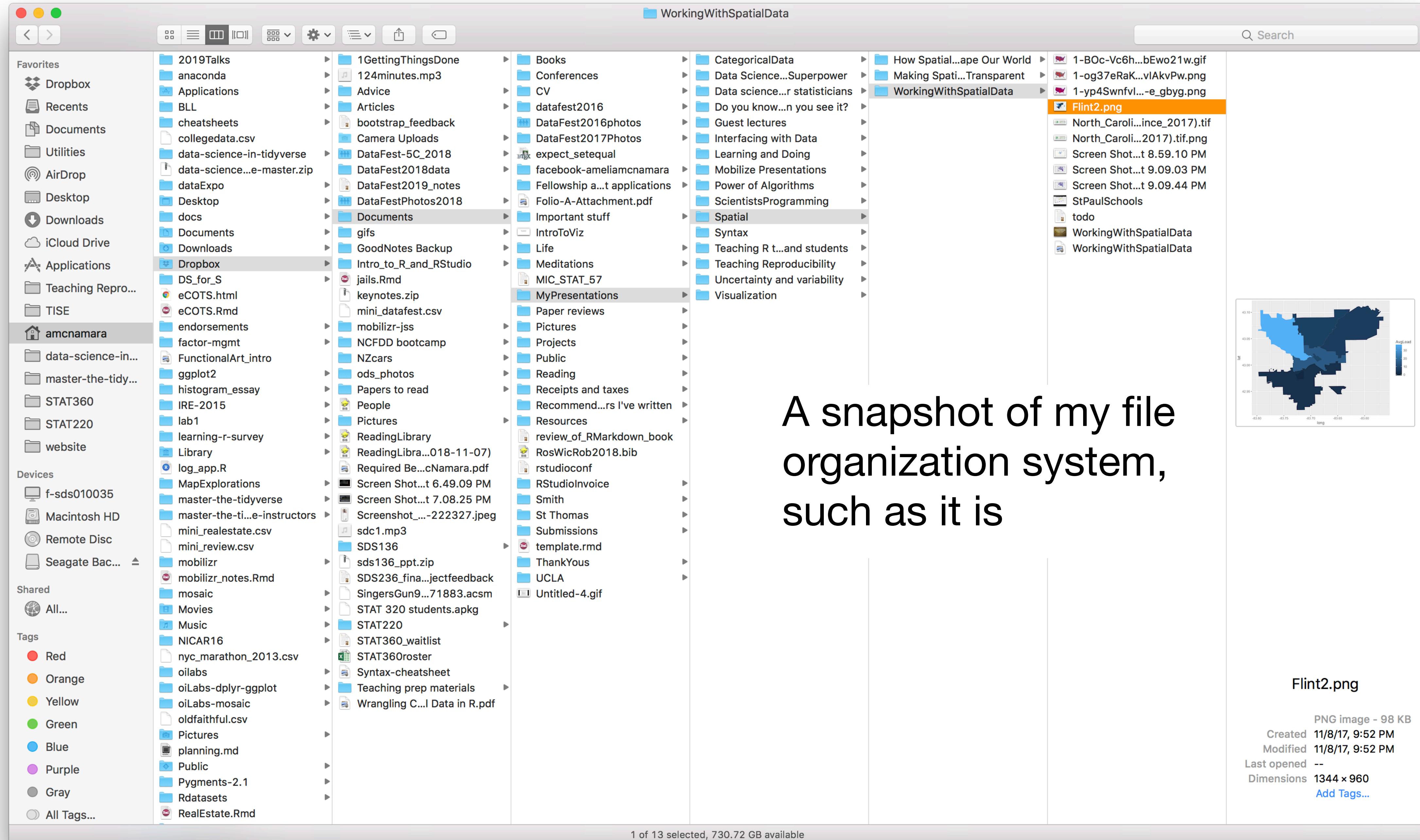
Filepaths and Projects

Filepaths are less important in today's computing landscape

If you have files stored in OneDrive, or GoogleDrive, or iCloud, they are findable with search functionality that rivals Google (or in some cases, is Google).

Computers have great search functions, like Apple's Spotlight and Windows' searchable File Explorer. iPads have actually removed the ability to walk through your filesystem.

But... it wasn't always like this. It used to be that you needed to know "where" your file was stored (scare quotes because it isn't a physical location) in order to find it. So, we used folders to keep everything organized...



Computers and programming languages still need to use
filepaths to locate things

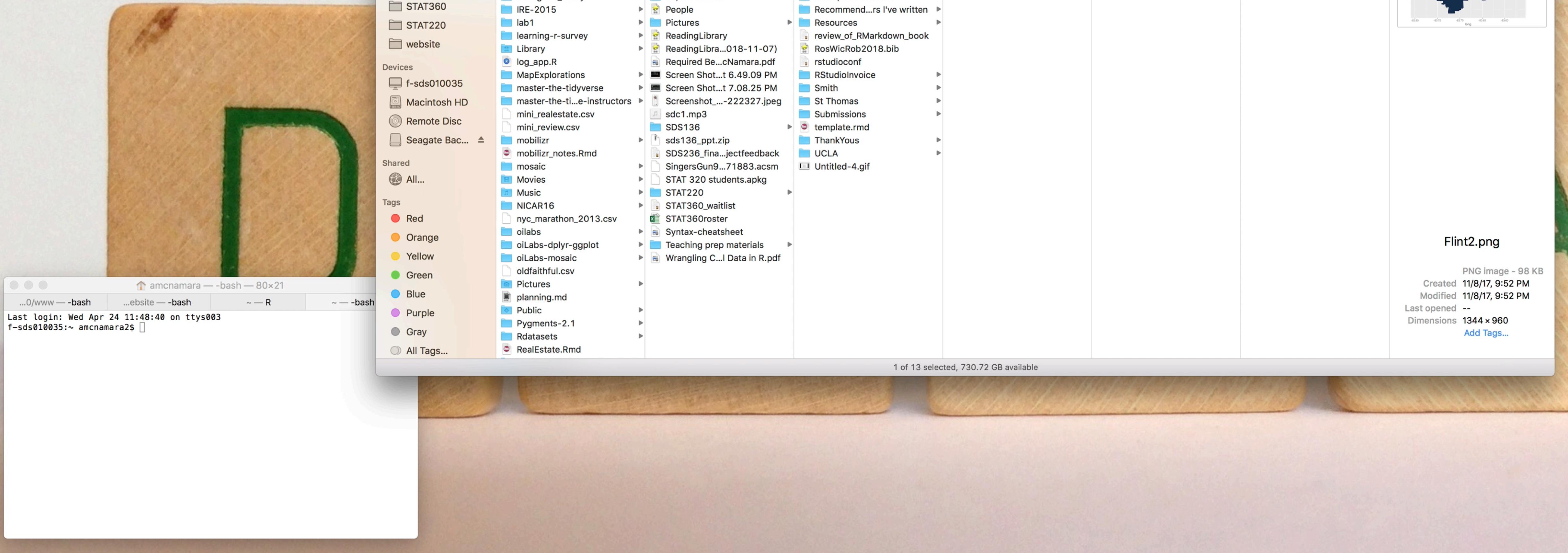
```
> read_csv("labike.csv")
```

```
Error: 'labike.csv' does not exist in current working  
directory ('/Users/amcnamara').
```

One solution is to use **absolute paths**

```
> read_csv("/Users/amcnamara/Dropbox/Documents/Projects/data/labike.csv")
```

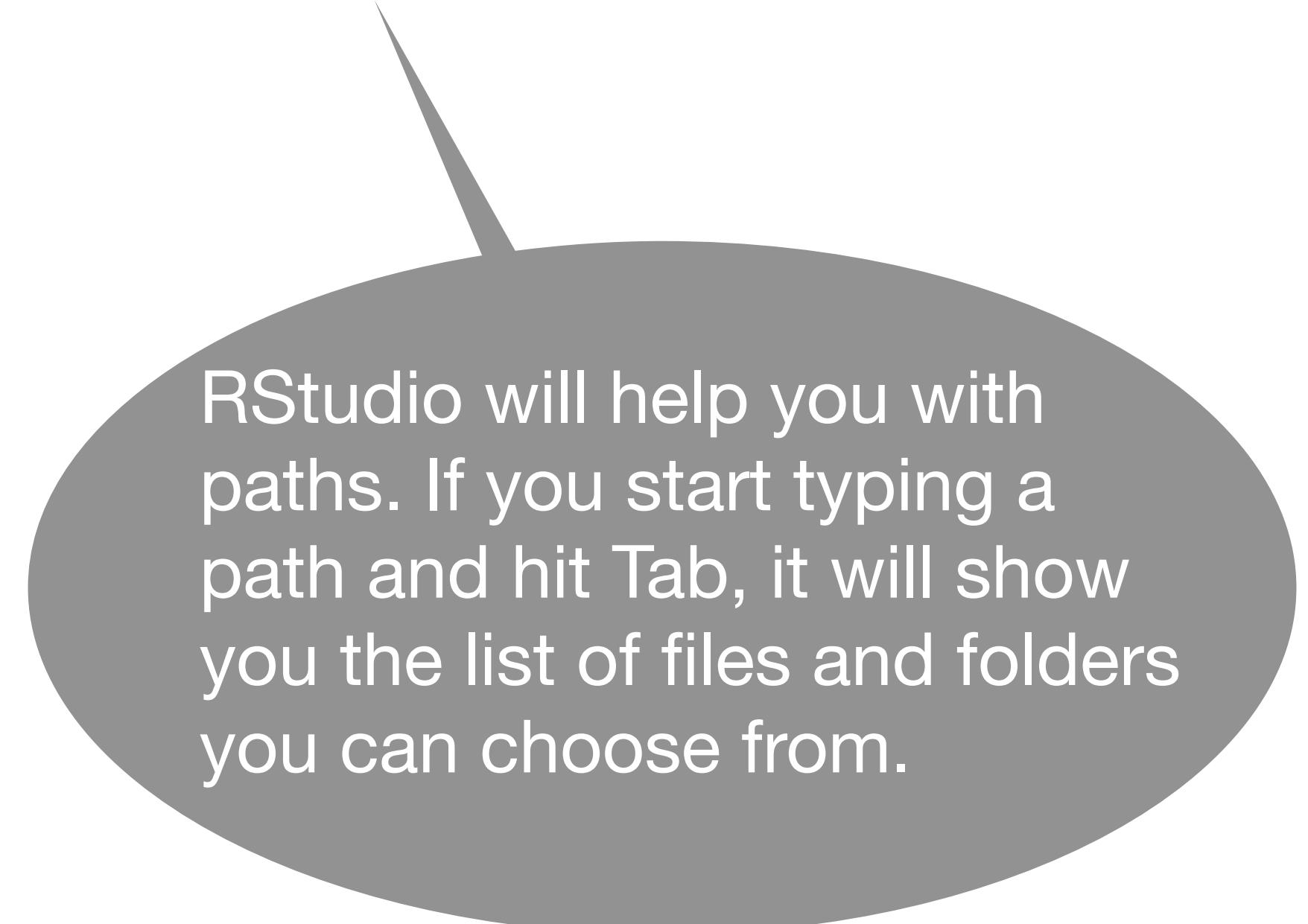
A little trick for the Mac— drag a file to the Terminal to get the filepath





It's all about the slashes!

Mac: /Users/amcnamara/Dropbox/Documents/MyPresentations/Spatial/WorkingWithSpatialData/Flint2.png
Windows (?): C:\Dropbox\Documents\MyPresentations\Spatial\WorkingWithSpatialData\Flint2.png



RStudio will help you with paths. If you start typing a path and hit Tab, it will show you the list of files and folders you can choose from.

“directory” = folder

“working directory” = the folder
your computer is looking in for stuff

```
getwd() # gets the working directory  
setwd() # sets the working directory
```

Strong words from Jenny Bryan

If the first line of your R script is

```
setwd("C:\Users\jenny\path\that\only\I\have")
```

I* will come into your office and
SET YOUR COMPUTER ON FIRE 🔥.

* or maybe Timothée Poisot will

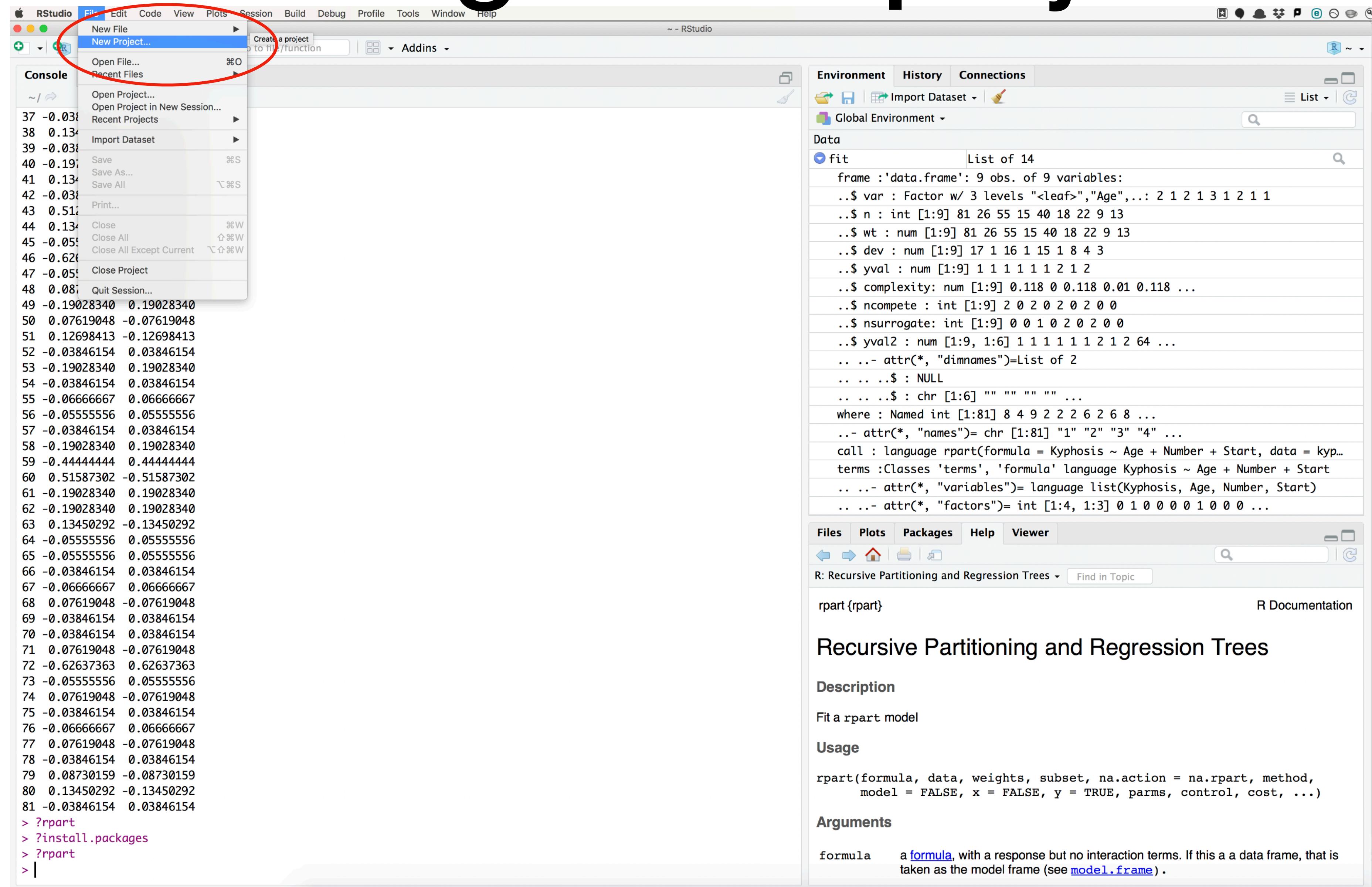
Relative paths are easier and better than absolute paths

That's why we use Projects in RStudio

When you make a Project, you're essentially making a special directory that RStudio knows things about. For example, you can set Project-specific settings, and Projects can have different tabs (e.g., the Build tab we saw working with `forcats` and the git tab we often have)

When you open a Project, RStudio sets your working directory to the folder of the project. After that, you can use relative filepaths.

Making a new project



The screenshot shows the RStudio interface. The 'File' menu is open, and the 'New Project...' option is highlighted with a red circle. The main workspace shows a console with several lines of R code and output. The Environment tab in the top right shows a list of objects, including 'fit'. The bottom right shows the R documentation for the 'rpart' function.

```
37 -0.038
38 0.134
39 -0.038
40 -0.191
41 0.134
42 -0.038
43 0.512
44 0.134
45 -0.051
46 -0.621
47 -0.051
48 0.081
49 -0.19028340 0.19028340
50 0.07619048 -0.07619048
51 0.12698413 -0.12698413
52 -0.03846154 0.03846154
53 -0.19028340 0.19028340
54 -0.03846154 0.03846154
55 -0.06666667 0.06666667
56 -0.05555556 0.05555556
57 -0.03846154 0.03846154
58 -0.19028340 0.19028340
59 -0.44444444 0.44444444
60 0.51587302 -0.51587302
61 -0.19028340 0.19028340
62 -0.19028340 0.19028340
63 0.13450292 -0.13450292
64 -0.05555556 0.05555556
65 -0.05555556 0.05555556
66 -0.03846154 0.03846154
67 -0.06666667 0.06666667
68 0.07619048 -0.07619048
69 -0.03846154 0.03846154
70 -0.03846154 0.03846154
71 0.07619048 -0.07619048
72 -0.62637363 0.62637363
73 -0.05555556 0.05555556
74 0.07619048 -0.07619048
75 -0.03846154 0.03846154
76 -0.06666667 0.06666667
77 0.07619048 -0.07619048
78 -0.03846154 0.03846154
79 0.08730159 -0.08730159
80 0.13450292 -0.13450292
81 -0.03846154 0.03846154
> ?rpart
> ?install.packages
> ?rpart
> |
```

Environment History Connections

Global Environment

Data

fit List of 14

frame : 'data.frame': 9 obs. of 9 variables:

..\$ var : Factor w/ 3 levels "<leaf>","Age",...: 2 1 2 1 3 1 2 1 1

..\$ n : int [1:9] 81 26 55 15 40 18 22 9 13

..\$ wt : num [1:9] 81 26 55 15 40 18 22 9 13

..\$ dev : num [1:9] 17 1 16 1 15 1 8 4 3

..\$ yval : num [1:9] 1 1 1 1 1 1 2 1 2

..\$ complexity: num [1:9] 0.118 0 0.118 0.01 0.118 ...

..\$ ncompete : int [1:9] 2 0 2 0 2 0 2 0 0

..\$ nsurrogate: int [1:9] 0 0 1 0 2 0 2 0 0

..\$ yval2 : num [1:9, 1:6] 1 1 1 1 1 1 2 1 2 64 ...

..-. attr(*, "dimnames")=List of 2

.. . . \$: NULL

.. . . \$: chr [1:6] "" "" "" "" ...

where : Named int [1:81] 8 4 9 2 2 2 6 2 6 8 ...

..-. attr(*, "names")= chr [1:81] "1" "2" "3" "4" ...

call : language rpart(formula = Kyphosis ~ Age + Number + Start, data = kyp...)

terms :Classes 'terms', 'formula' language Kyphosis ~ Age + Number + Start

..-. attr(*, "variables")= language list(Kyphosis, Age, Number, Start)

..-. attr(*, "factors")= int [1:4, 1:3] 0 1 0 0 0 1 0 0 0 ...

Files Plots Packages Help Viewer

R: Recursive Partitioning and Regression Trees - Find in Topic

rpart {rpart}

R Documentation

Recursive Partitioning and Regression Trees

Description

Fit a `rpart` model

Usage

```
rpart(formula, data, weights, subset, na.action = na.rpart, method,
      model = FALSE, x = FALSE, y = TRUE, parms, control, cost, ...)
```

Arguments

formula a `formula`, with a response but no interaction terms. If this is a data frame, that is taken as the model frame (see `model.frame`).

Making a new project

The screenshot shows the RStudio interface. On the left, a modal dialog titled "New Project" is open under the "Create Project" tab. It lists three options: "New Directory", "Existing Directory", and "Version Control". The "Version Control" option is highlighted with a blue border. On the right, the RStudio environment pane is visible, showing the global environment with objects like "fit" and "rpart". Below the environment pane, a help window for the "rpart" function is open, displaying its description, usage, and arguments.

All three of these are useful in different situations!

The Version Control choice can be especially useful if you have work on RStudio Cloud you've been pushing to GitHub, and you want it on your local computer.

New Project

Create Project

- New Directory**
Start a project in a brand new working directory
- Existing Directory**
Associate a project with an existing working directory
- Version Control**
Checkout a project from a version control repository

Cancel

Recursive Partitioning and Regression Trees

Description
Fit a `rpart` model

Usage
`rpart(formula, data, weights, subset, na.action = na.rpart, method, model = FALSE, x = FALSE, y = TRUE, parms, control, cost, ...)`

Arguments

formula a [formula](#), with a response but no interaction terms. If this is a data frame, that is taken as the model frame (see [model.frame](#)).

Relative paths are easier and better than absolute paths

That's why we use Projects in RStudio

When you open a Project, RStudio sets your working directory to the folder of the project. After that, you can use relative filepaths.

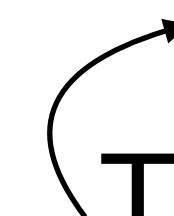
For example, if I have a Project set up in the directory “/Users/amcnamara/Dropbox/Documents/Projects”, then I can use the following relative path

```
read_csv("data/labike.csv")
```

This is better than an absolute path, and if I share that code (e.g., via GitHub), other people with Macs will be able to use my code as-is. But, it still doesn't work for Windows users.

The here package

```
> library(here)  
here() starts at /Users/amcnamara/Dropbox/Documents/Projects  
  
> here()  
[1] "/Users/amcnamara/Dropbox/Documents/Projects"  
  
> here("data", "labike.csv")  
[1] "/Users/amcnamara/Dropbox/Documents/Projects/data/labike.csv"  
  
> read_csv(here("data", "labike.csv"))
```



This will work on Macs and Windows. It will work whether you've cloned the GitHub repo into your Downloads folder, your Documents folder, onto your Desktop, etc.

here: find your
PATH!



(Wanna be oldschool?)

Use `file.path()`

```
> file.path("data", "labike.csv")
```

```
[1] "data/labike.csv"
```

Not quite as good, but no dependencies because it comes from base R