What They Forgot to Teach You About R

rstudio::conf 2018 San Diego

Training Days

https://www.rstudio.com/conference/

Yes, all materials are available.

All will be revealed and downloaded.

You have my word.

Wed Jan 31 & Thurs Feb 1, 2018

- 8am registration & light breakfast
- 9am We start!
- 10:30-11am snacks and coffee
- 12-1pm lunch
- 3-3:30pm snacks and coffee
- 5pm We stop!

Wed Jan 31

- 9-10:30 blah blah blah
- 10:30-11am Extinguish Git fires
- 11am-12pm blah blah blah
- 12-1pm **Really** extinguish Git fires
- 1-5pm Use Git and GitHub like a boss

rstd.io/forgot

Everyone is encouraged to open issues here:

rstd.io/forgot
https://github.com/jennybc/what-they-forgot

Record glitches, gotchas,
good sidebar discussions, etc.
to address now or later.

Go here now:

rstd.io/forgot
https://github.com/jennybc/what-they-forgot

Go to the Issues.
What's your OS? Put an emoji on your OS.

What if I'm in over my head?

What if I could teach this workshop?

Meet your helpers

Meet your helpers*

* your neighbours

Go here now:

rstd.io/forgot
https://github.com/jennybc/what-they-forgot

Go to the Issues.
What's your OS? Put an emoji on your OS.

Day 1, morning

Jennifer Bryan
RStudio, University of British Columbia

@jennybc
@JennyBryan

What *Did* They Forget to Teach You?

Everything else

Statistical analysis


Deep
Thoughts

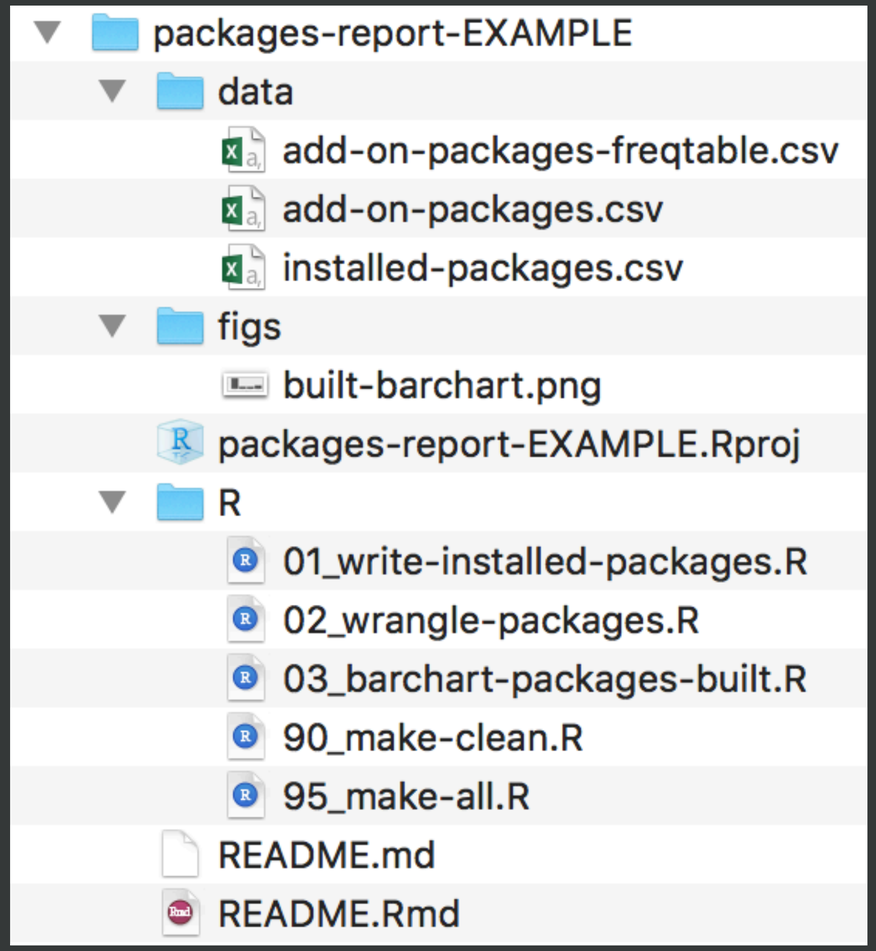
Be organized

do this as you go, not "tomorrow"

but also don't fret over past mistakes
raise the bar for *new* work

Be organized

self-explaining >>> wordy, needy explainers



>>>

file salad

+ an out-of-date README

Good enough practices in scientific computing

Wilson, Bryan, Cranston, Kitzes, Nederbragt, Teal

https://doi.org/10.1371/journal.pcbi.1005510

http://bit.ly/good-enuff

Day 1 Practical Example: Explore your R installation

R package = the natural unit for distributing R code

base R ≈ 14 base + 15 recommended packages
ship with all binary distributions of R

can use right out of the box:

```
library(lattice)
```

CRAN has > 12K additional packages

install, then attach:

```
install.packages("devtools")
library(devtools)
```

And then there's GitHub ...

install via devtools, then attach:

```
devtools::install_github("tidyverse/dplyr")
library(dplyr)
```

Where do packages live locally?

By default, in the default library

```
.Library
```

All libraries for current session:

```
.libPaths()
```

All installed packages:

```
installed.packages()
```

```
install.packages("usethis")
library(usethis)
use_course("rstd.io/forgot_1")
```

```
install.packages("usethis")
library(usethis)
use_course("rstd.io/forgot_1")
```

Pick one to open and flesh out:

```
01_explore-libraries_spartan.R
01_explore-libraries_comfy.R*
```

* worst case, there's always `jenny`

```
install.packages("usethis")
library(usethis)
use_course("rstd.io/forgot_1")
```

Stay relaxed.

We will refine this code and where it lives soon enough.

You **want** to leave rough edges and gaps to address later.

work on challenge


Deep
Thoughts

Adopt a project-oriented workflow

Why?

- work on more than 1 thing at a time
- collaborate, communicate, distribute
- start and stop

Adopt a project-oriented workflow

How?

- dedicated directory
- RStudio **P**roject
- Git repo, probably syncing to a remote

If you do this at the top of your scripts, I might set your computer on 🔥:

```
setwd("C:\Users\jenny\path\that\only\I\have")
rm(list = ls())
```

Project-oriented workflow designs this away:

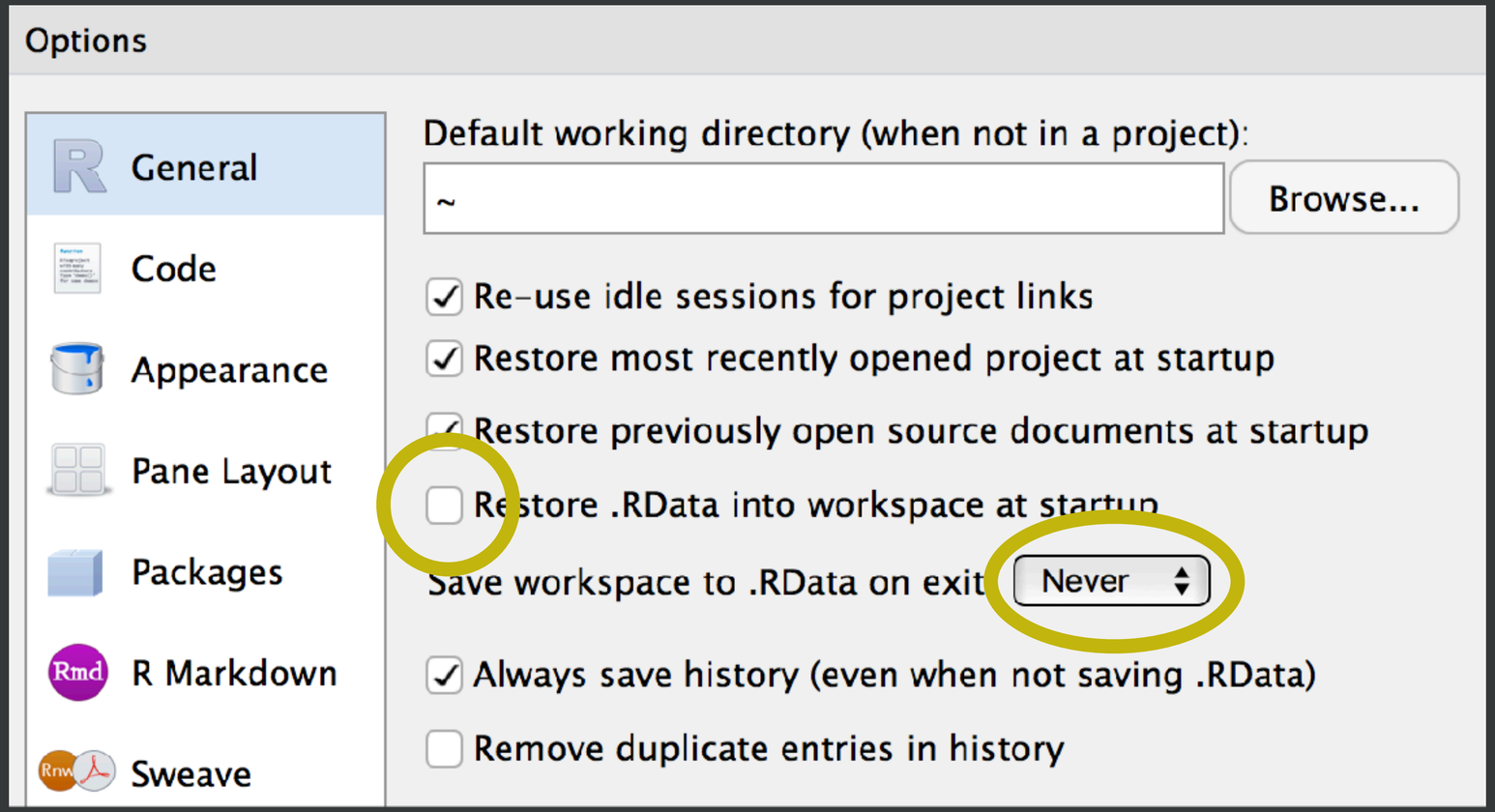
https://www.tidyverse.org/articles/2017/12/workflow-vs-script/

What does it mean to be an RStudio **P**roject?

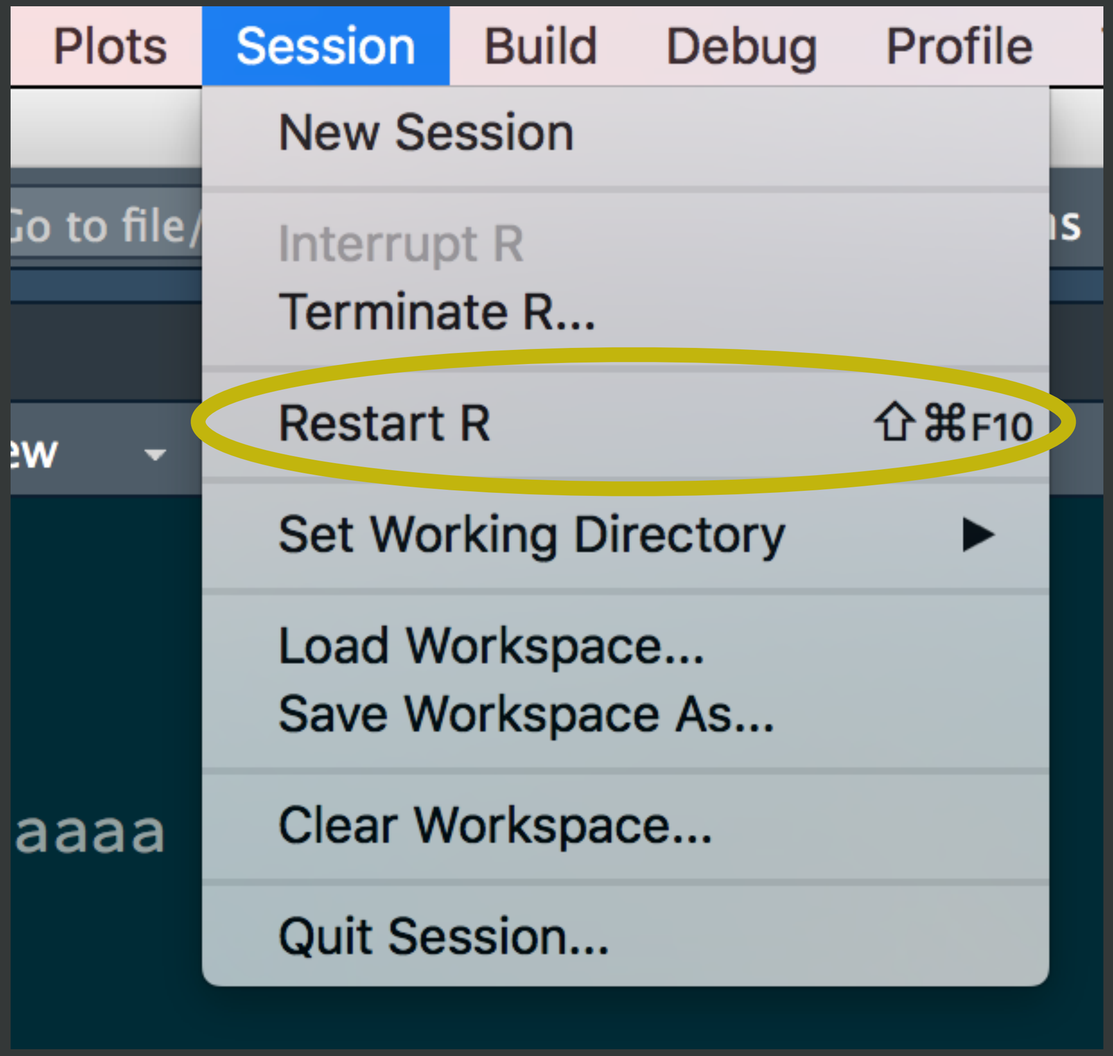
- RStudio leaves notes to itself in `foo.Rproj`
- Open Project = dedicated instance of RStudio
  - dedicated R process
  - file browser pointed at Project directory
  - working directory set to Project directory


rexcel
blarg
jenny
eigencoder
foofy
STAT545-
happy-git-with-

Use "fresh starts"



Restart R often



Project initiation: the local case

New folder + make it an RStudio Project

- `usethis::create_project("~/i_am_new")`
- RStudio > New Project... > New Directory > New Project

Make existing folder into an RStudio Project

- `usethis::create_project("~/i_exist")`
- RStudio > New Project... > Existing Directory

Project initiation strategies, the local case

New folder + make it an RStudio Project

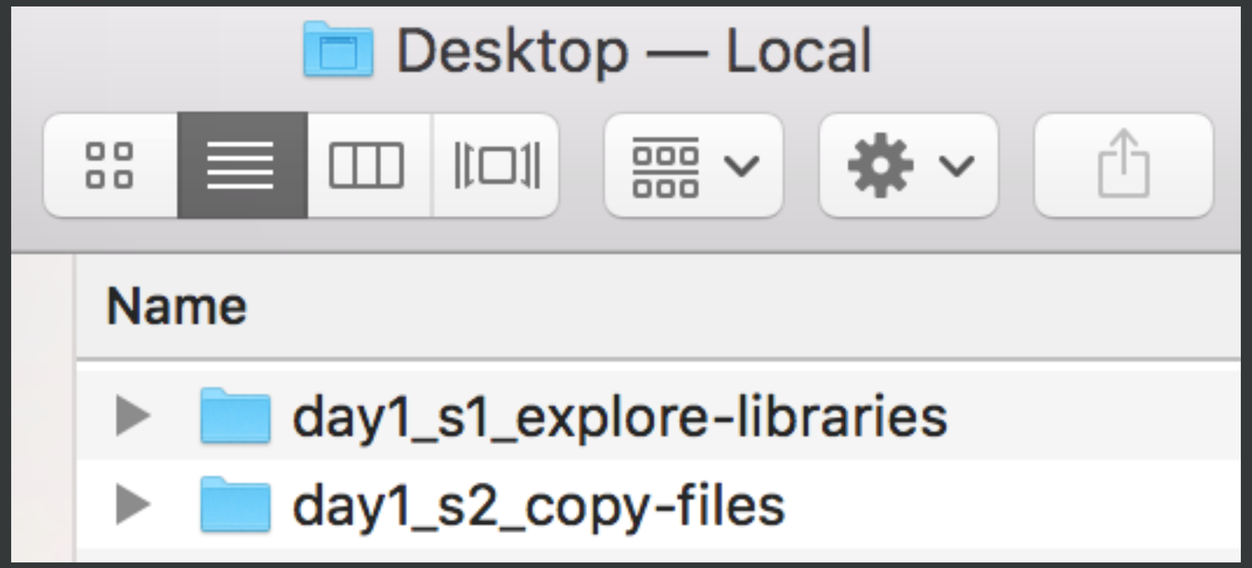
- `usethis::create_project("~/i_am_new")`
- RStudio > New Project... > New Directory > New Project

Make existing folder into an RStudio Project

- `usethis::create_project("~/i_exist")`
- RStudio > New Project... > Existing Directory

```
library(usethis)
use_course("rstd.io/forgot_2")
```

- you know the drill
- download to same location as previous



```
use_course("rstd.io/forgot_2")
```

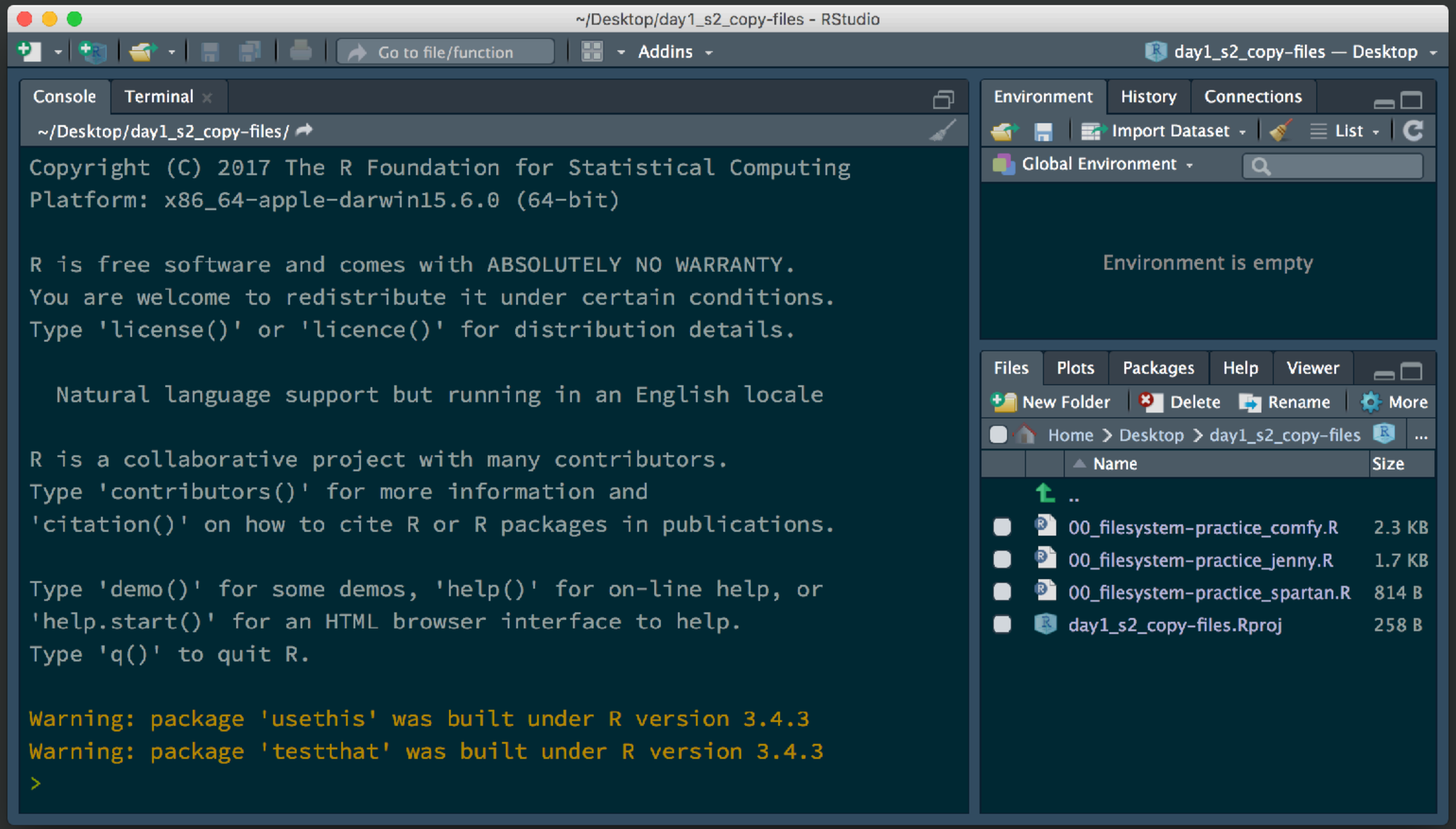
- make this new folder an RStudio Project
  - `usethis::create_project("~/i_exist")`
  - RStudio > New Project... > Existing Directory

You should now be in RStudio,
in the new RStudio Project named
day1_s2_copy-files.

How to launch an RStudio Project?

- double-click on .Rproj file
- use File > Open Project and friends
- use Project drop down in upper right corner
- Alfred trick 😉

We are here!

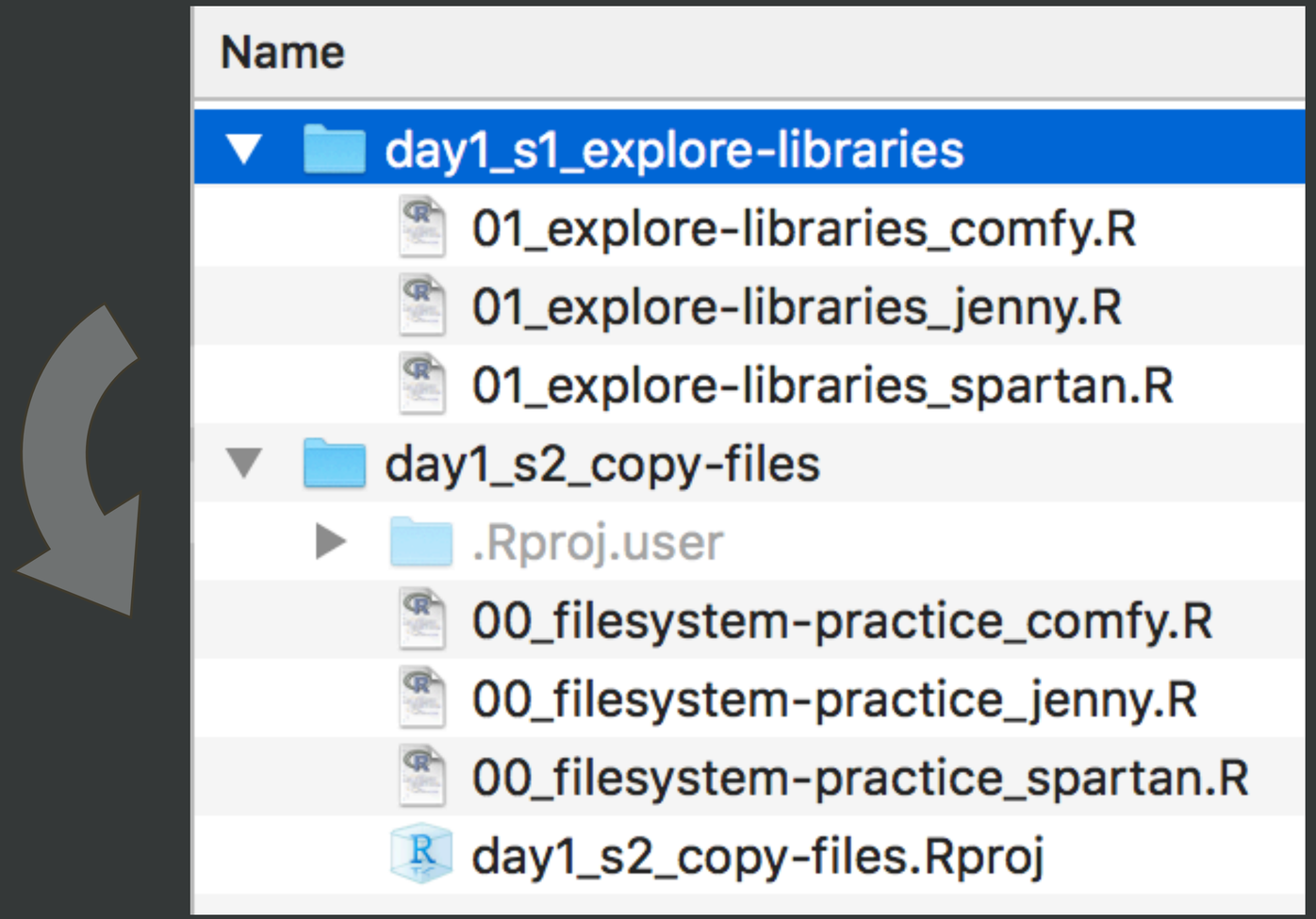

~/Desktop/day1_s2_copy-files - RStudio
Go to file/function
Addins
day1_s2_copy-files — Desktop
Console
Terminal
~/Desktop/day1_s2_copy-files/
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
Warning: package 'usethis' was built under R version 3.4.3
Warning: package 'testthat' was built under R version 3.4.3
>
Environment
History
Connections
Import Dataset
List
Global Environment
Environment is empty
Files
Plots
Packages
Help
Viewer
New Folder
Delete
Rename
More
Home > Desktop > day1_s2_copy-files
Name
Size
..
00_filesystem-practice_comfy.R
2.3 KB
00_filesystem-practice_jenny.R
1.7 KB
00_filesystem-practice_spartan.R
814 B
day1_s2_copy-files.Rproj
258 B

```
use_course("rstd.io/forgot_2")
```

Contains `00_filesystem-practice_*.R` files in the usual variants (spartan, comfy, jenny).

Open the file you want and develop your code there.

Copy the `.R` files from earlier into the current project.



Programmatically, with R. Don't use the mouse!


Deep
Thoughts

Practice "safe paths"

Do you know where your files are?

What is working directory?

https://devs.mailchimp.com/blog/the_freddie_mercury_project/

Our second launch was supposed to be The One, but we ran into a critical issue with some of our code.... The code worked fine as we were testing on command line since the location of PHP was known by our shell, but once the code was added to cron for automation, the **location of PHP wasn't known, and the scripts continuously failed to send.**

Practice "safe paths"

relative to a stable base

use file system functions,

not `paste()`, `strsplit()`, etc.

Examples of a stable base

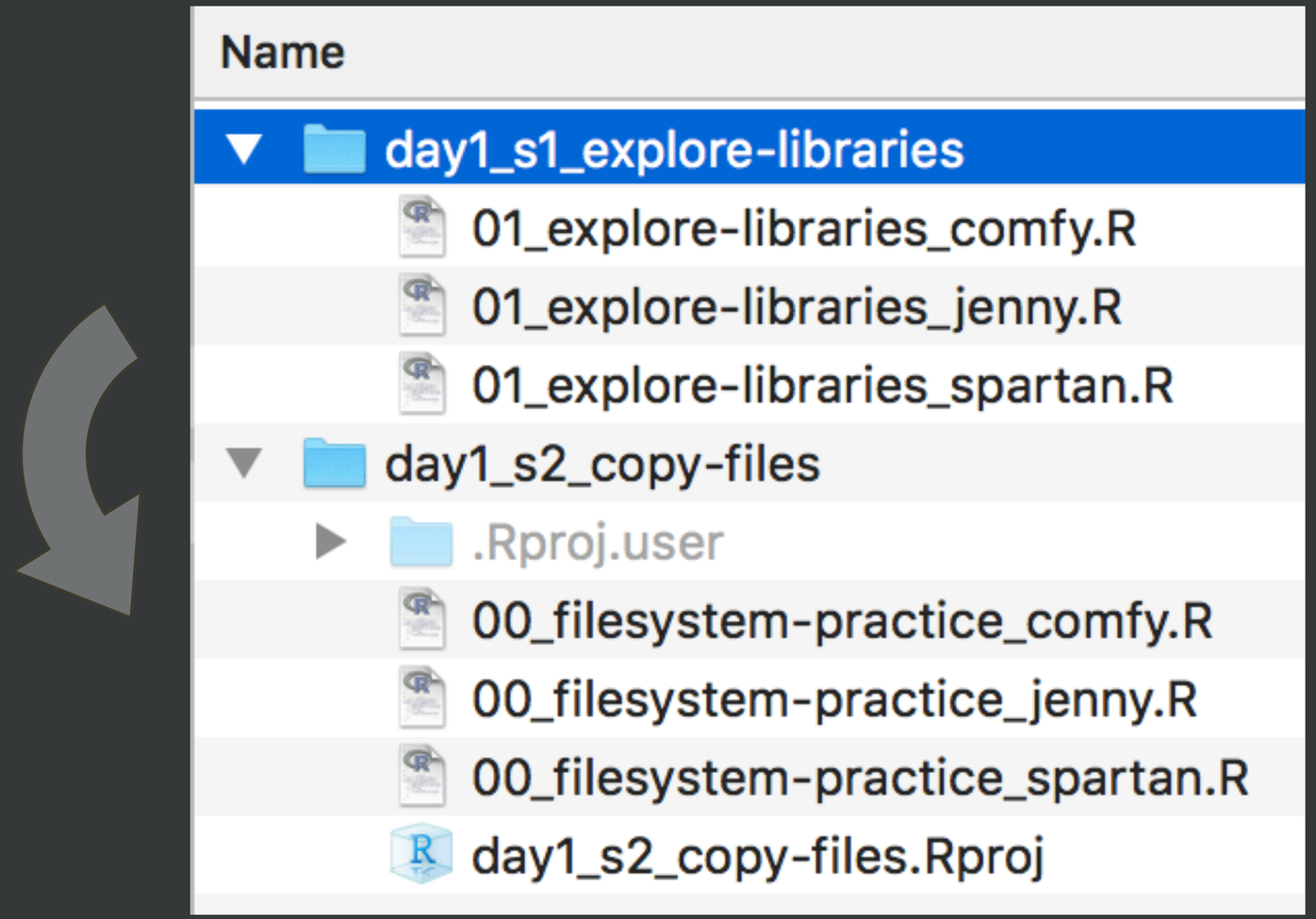
- Project directory
  - `here::here("data", "raw-data.csv")`
- User's home directory
  - `file.path("~", ...)`
  - `fs::path_home(...)`
- Official location for installed s/w
  - `library(thingy)`
  - `system.file(..., package = "thingy")`

I have nothing against absolute paths.
Some of my best friends are absolute paths!
But don't hard-wire them into your scripts.
Instead, form at runtime relative to a stable base.

```
> (GOOD <- normalizePath("~/tmp/test.csv"))
[1] "/Users/jenny/tmp/test.csv"

> (BAD <- "/Users/jenny/tmp/test.csv")
[1] "/Users/jenny/tmp/test.csv"
```

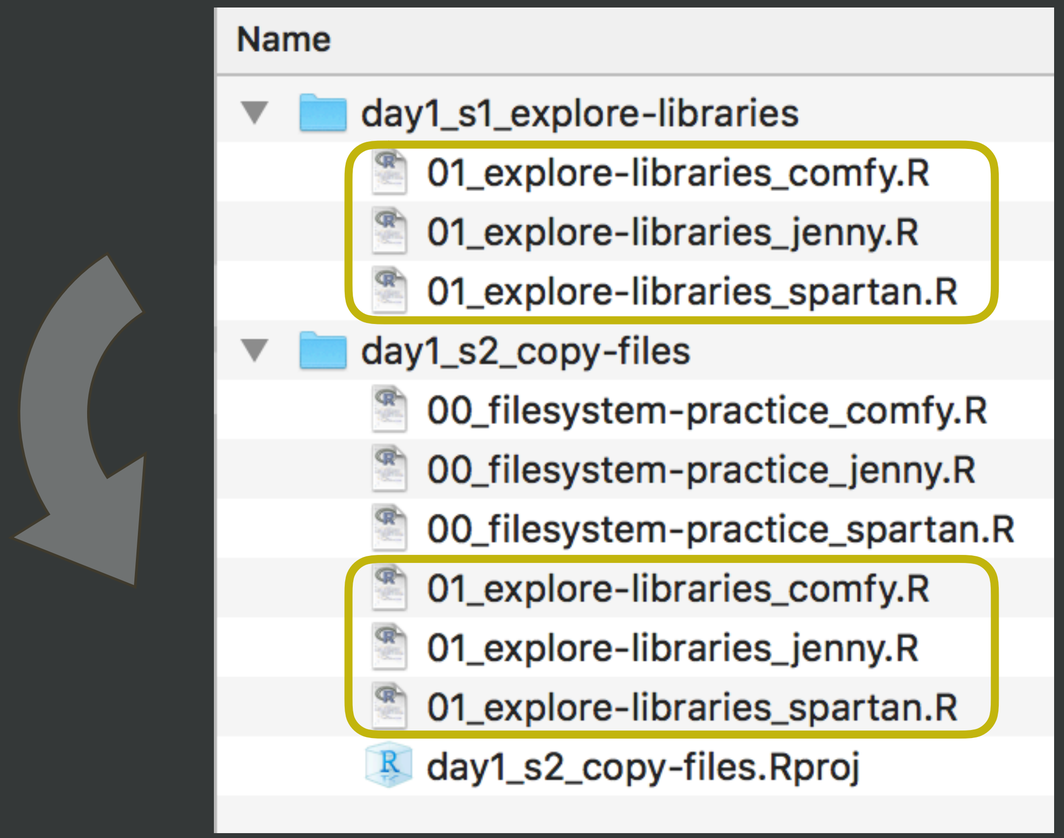
Copy the `.R` files from earlier into the current project.



Programmatically, with R. Don't use the mouse!

```
use_course("rstd.io/forgot_2")
```

Success looks like this (and the code is saved in `00_filesystem-practice_*.R`):



work on challenge


Deep
Thoughts

Be open to new things

I predict fs is my new BFF for file system work

I choose to use it today,
even though I have no idea what I'm doing

I will provide useful feedback

Names matter

machine readable
human readable
sort nicely

myabstract.docx
Joe’s Filenames Use Spaces and Punctuation.xlsx
figure 1.png
homework1.R
JW7d^(2sl@deletethisandyourcareerisoverWx2*.txt

2018-01_bryan-abstract-rstudio-conf.docx
joes-filenames-are-getting-better.xlsx
fig01_scatterplot-talk-length-vs-interest.png
bryan_hw01.R
1986-01-28_raw-data-from-challenger-o-rings.txt

"machine readable"

regular expression and globbing friendly

- avoid spaces, punctuation, accented characters, case sensitivity

easy to compute on

- deliberate use of delimiters

"human readable"

name contains info on content
name anticipates context
concept of a slug from user-friendly URLs

1986-01-28_raw-data-from-challenger-o-rings.txt

"sort nicely"

put something numeric in there
left pad with zeros for constant width
use the ISO 8601 standard for dates
order = chronological or ... consider common sense

```
> ft <- tibble(files = dir_ls(glob = "*.R"))
> ft
# A tibble: 6 x 1
  files
  <fs::path>
1 00_filesystem-practice_comfy.R
2 00_filesystem-practice_jenny.R
3 00_filesystem-practice_spartan.R
4 01_explore-libraries_comfy.R
5 01_explore-libraries_jenny.R
6 01_explore-libraries_spartan.R
```

👍 file names

```
> ft <- tibble(files = dir_ls(glob = "*.R"))
> ft
# A tibble: 6 x 1
  files
  <fs::path>
1 00_filesystem-practice_comfy.R
2 00_filesystem-practice_jenny.R
3 00_filesystem-practice_spartan.R
4 01_explore-libraries_comfy.R
5 01_explore-libraries_jenny.R
6 01_explore-libraries_spartan.R
```

Anyone can guess at file's purpose

```
> ft %>%
+   filter(str_detect(files, "explore"))
# A tibble: 3 x 1
  files
  <fs::path>
1 01_explore-libraries_comfy.R
2 01_explore-libraries_jenny.R
3 01_explore-libraries_spartan.R
```

Easy to filter in R (or the shell or whatever)

```
> ft %>%
+   mutate(files = path_ext_remove(files)) %>%
+   separate(files, into = c("i", "topic", "flavor"), sep = "_")
# A tibble: 6 x 3
  i     topic                flavor
* <chr> <chr>                <chr>
1 00    filesystem-practice comfy
2 00    filesystem-practice jenny
3 00    filesystem-practice spartan
4 01    explore-libraries   comfy
5 01    explore-libraries   jenny
6 01    explore-libraries   spartan
```

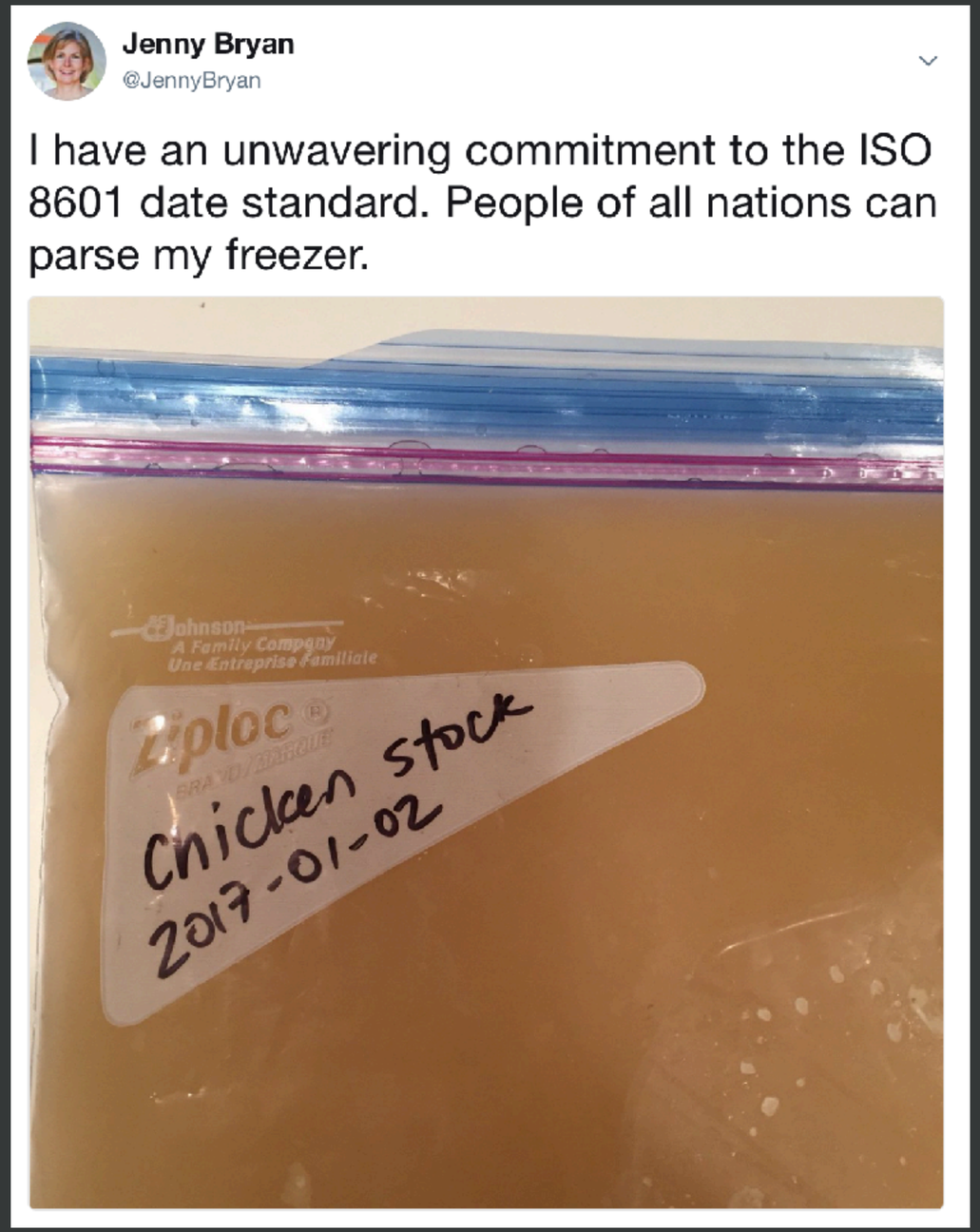
Intentional use of delimiters = meta-data easy to recover

"_" delimits fields

"-"delimits words so my eyes don’t bleed

```
> dirs <- dir_ls(path_home("Desktop"), type = "directory")
> (dt <- tibble(dirs = path_file(dirs)))
# A tibble: 2 x 1
  dirs
  <fs::path>
1 day1_s1_explore-libraries
2 day1_s2_copy-files
```

Sorts in the same order as you experience in real life



YYYY-MM-DD

ISO 8601

Names matter

machine readable
human readable
sort nicely

Names matter

easy to implement NOW

payoffs accumulate as your skills evolve and projects get more complex