

Rstudio::conf 2018

“What They Forgot to Teach You About R” Training Days – Jenny Bryan (+ Jim Hester)

Recap by Jessica Minnier, PDX R Meetup 2/26/18

Jenny's repo: <https://github.com/jennybc/what-they-forgot>

My notes: https://github.com/jminnier/rstudioconf18_jennybryan_forgot/blob/master/notes.md

My rstudio::conf notes: https://github.com/jminnier/rstudioconf18_jennybryan_forgot/blob/master/notes_rstudioconf18.md

These slides are a subset of Jenny + Jim's

Not included in slides: live purr coding, API example, dotfiles/start up files (see notes/repo)

What They Forgot to Teach You About R

rstudio::conf
from R Studio

rstudio::conf 2018 San Diego

Training Days

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

This work is licensed under a **Creative Commons Attribution-ShareAlike 4.0 International License**.

To view a copy of this license, visit
<http://creativecommons.org/licenses/by-sa/4.0/>

Day 1, morning

Jennifer Bryan

RStudio, University of British Columbia

 @jennybc

 @JennyBryan

Everything else



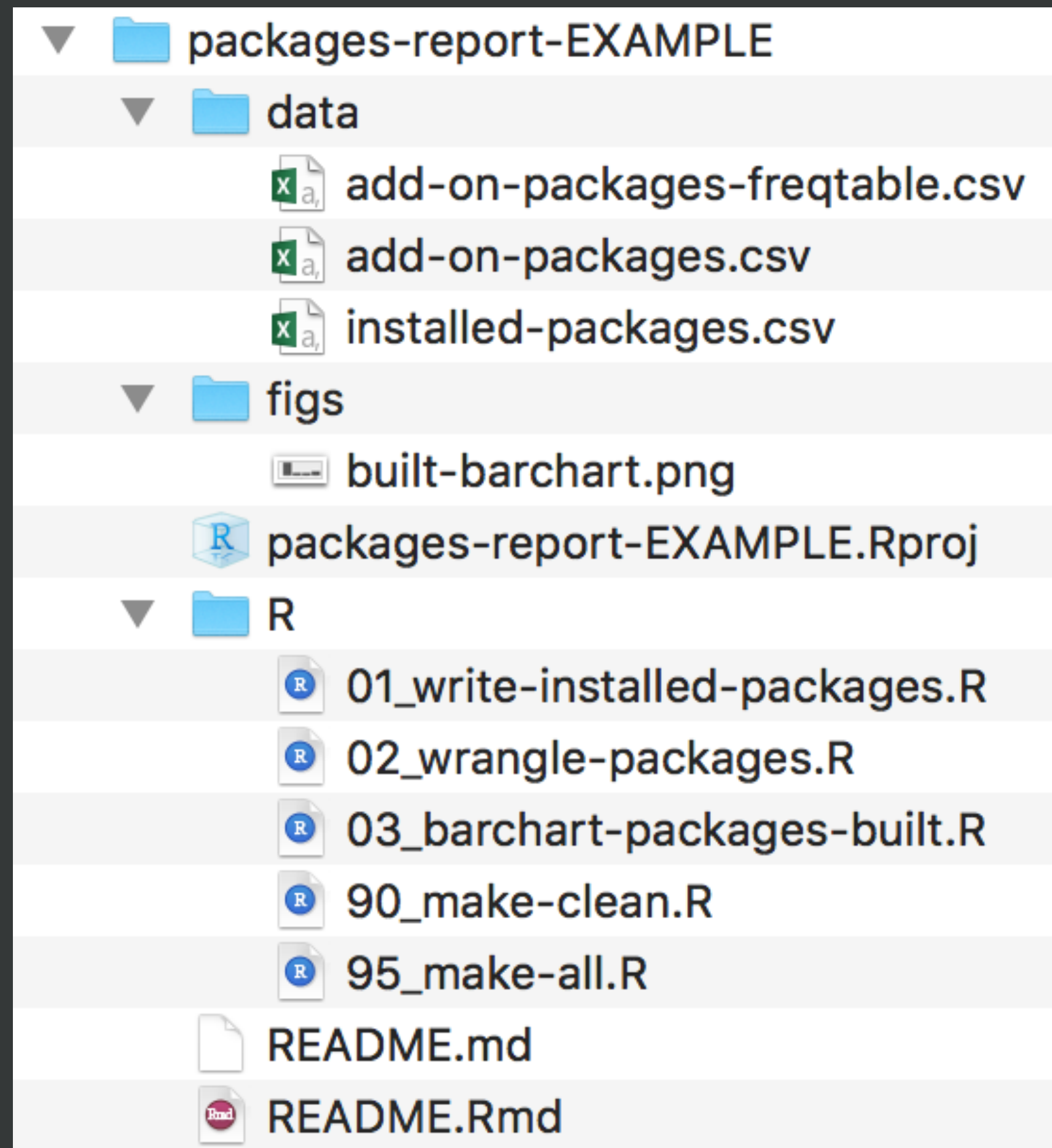
Statistical
analysis

Be organized

do this as you go, not "tomorrow"

but also don't fret over past mistakes

raise the bar for *new* work



>>>

file salad
+ an out-of-date README

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())
```

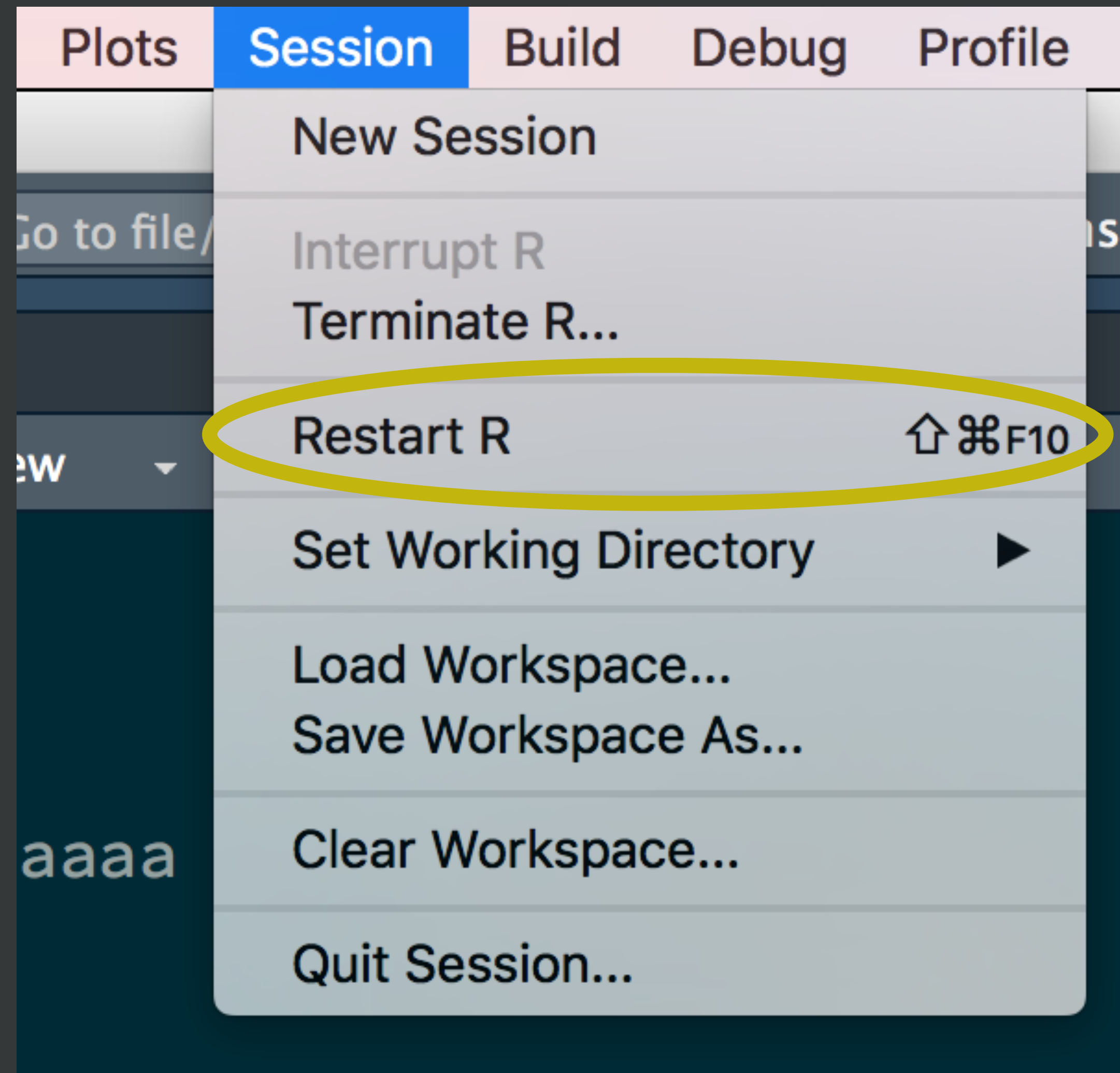
Project-oriented workflow designs this away:

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

What does it mean to be an RStudio Project?

- 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

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

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")`

Practice "safe paths"

Use the [here](#) package to build paths inside a project.

Leave working directory at top-level at all times, during development.

Absolute paths are formed at runtime.

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
```

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

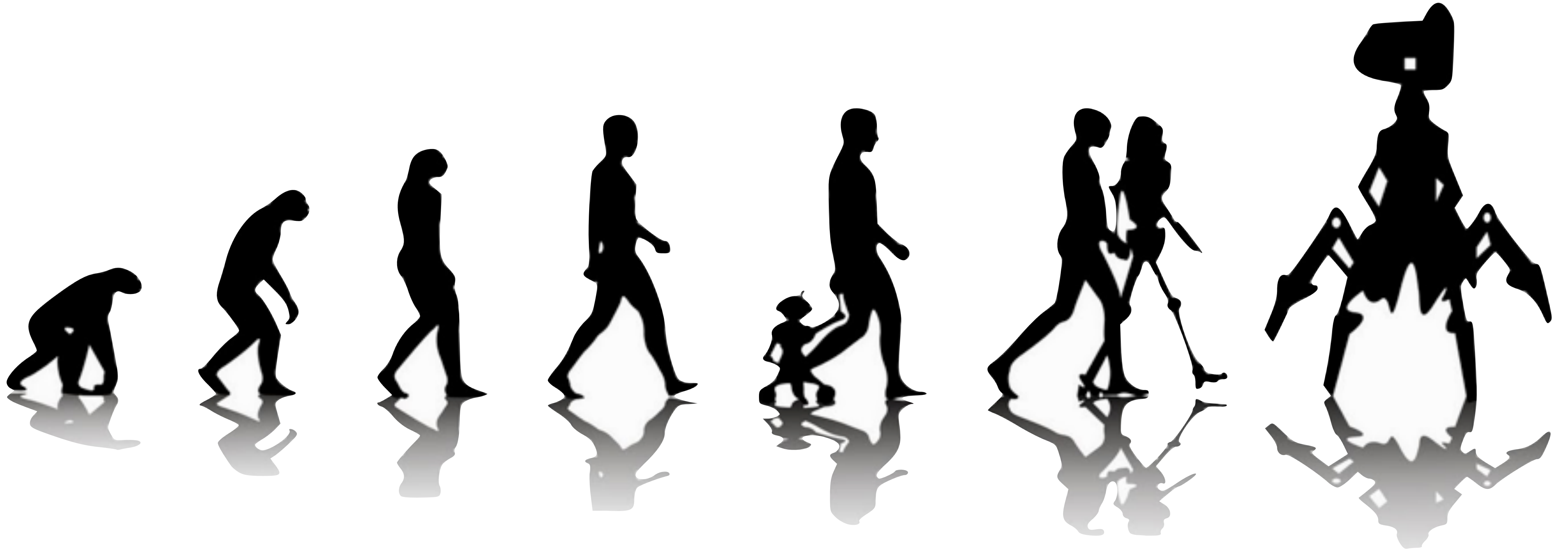
```
> dirs <- dir_ls(path_home("Desktop"), type = "directory")
> (dt <- tibble(dirs = path_file(dirs)))
# A tibble: 2 x 1
  dirs
  <fs::path>
1 day1_s1_explore-libraries
2 day1_s2_copy-files
```

Sorts in the same order as you
experience in real life

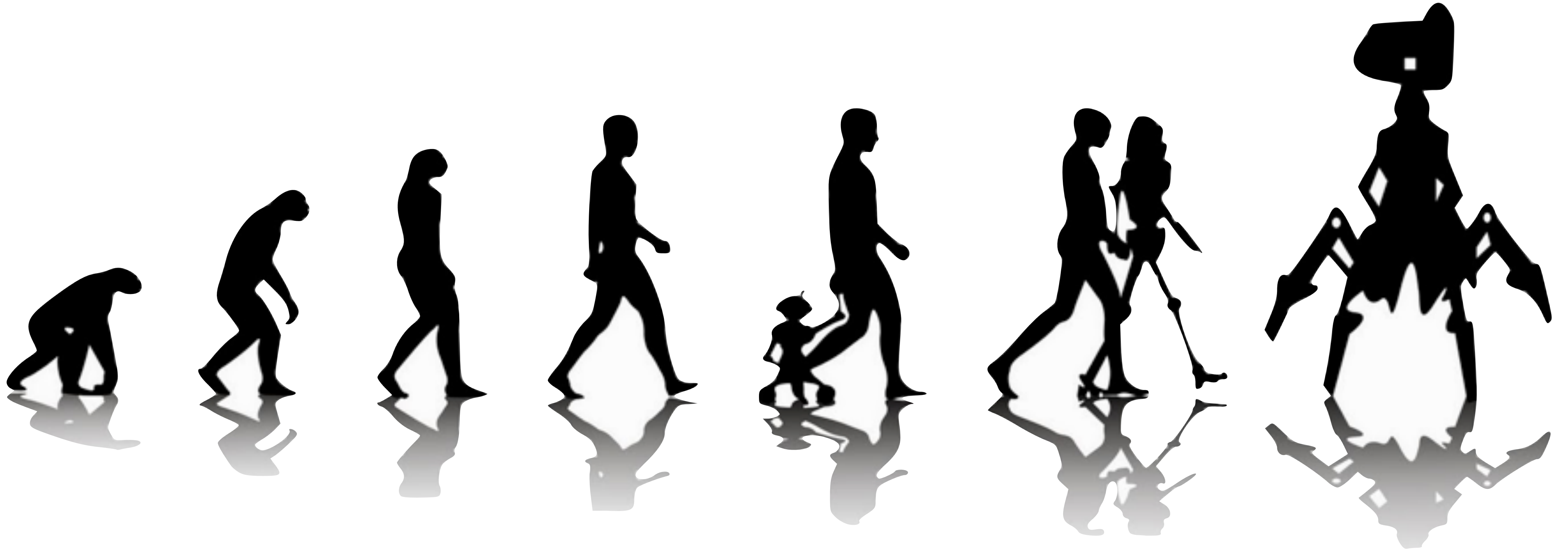
Names matter

easy to implement NOW

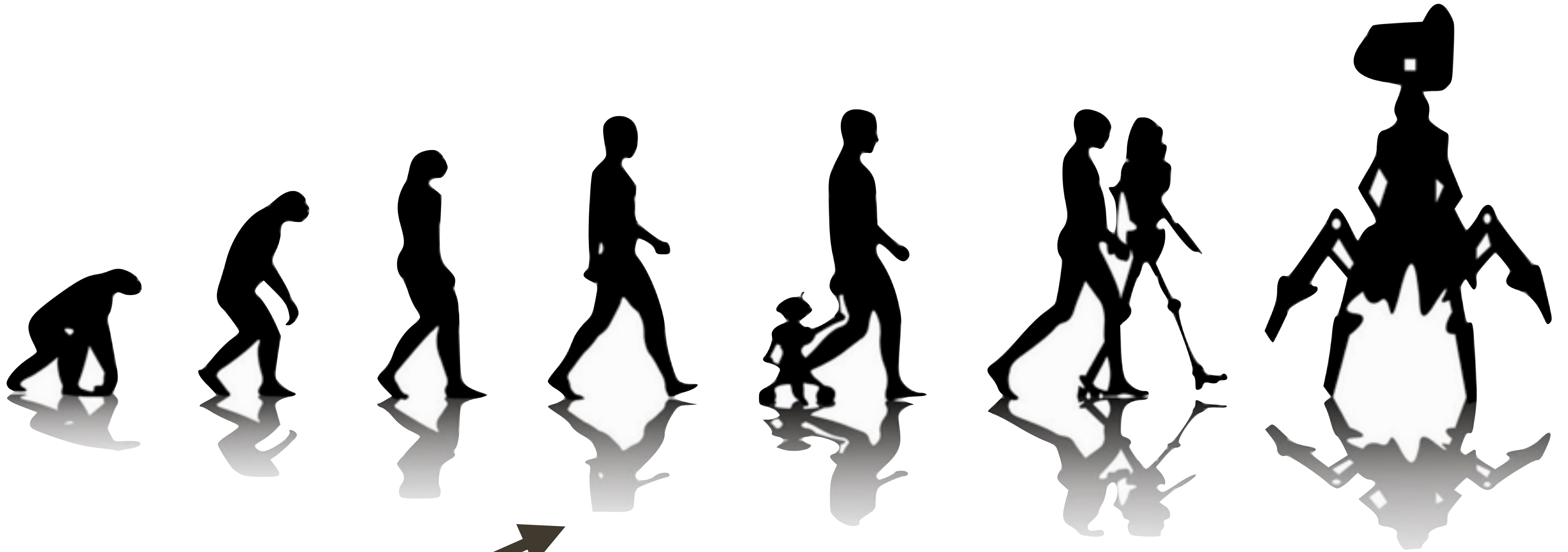
payoffs accumulate as your skills
evolve and projects get more complex



use version control

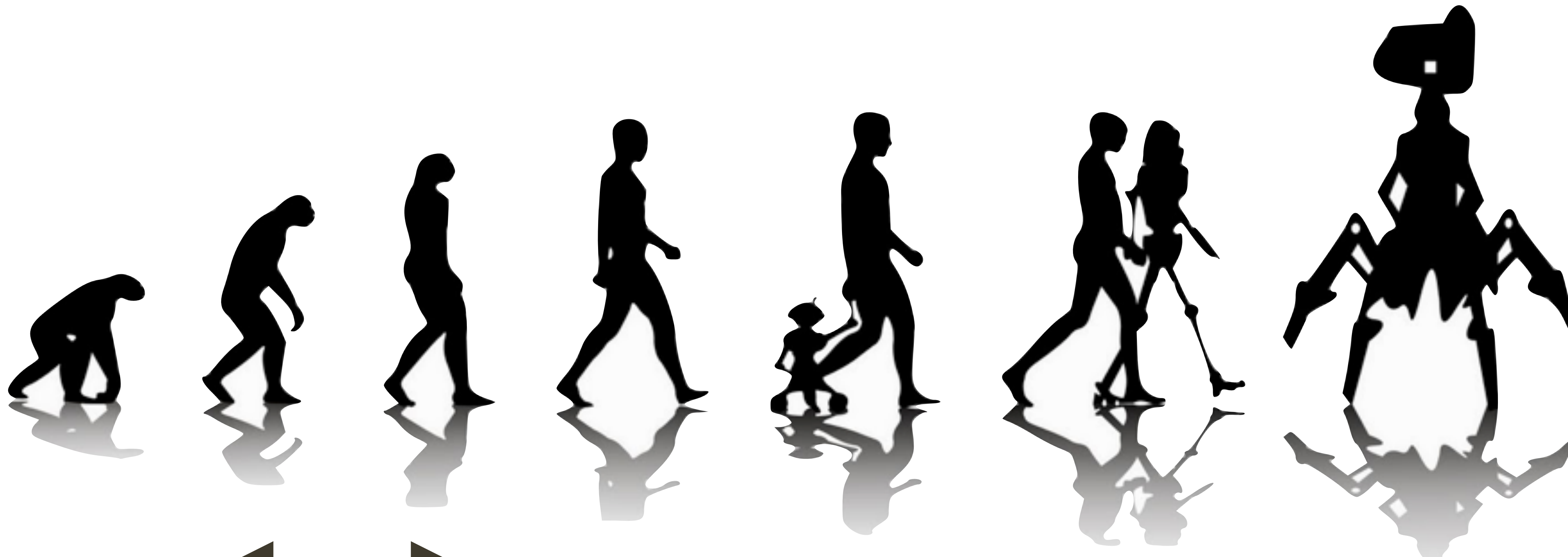


we teach Git + GitHub



"commit"

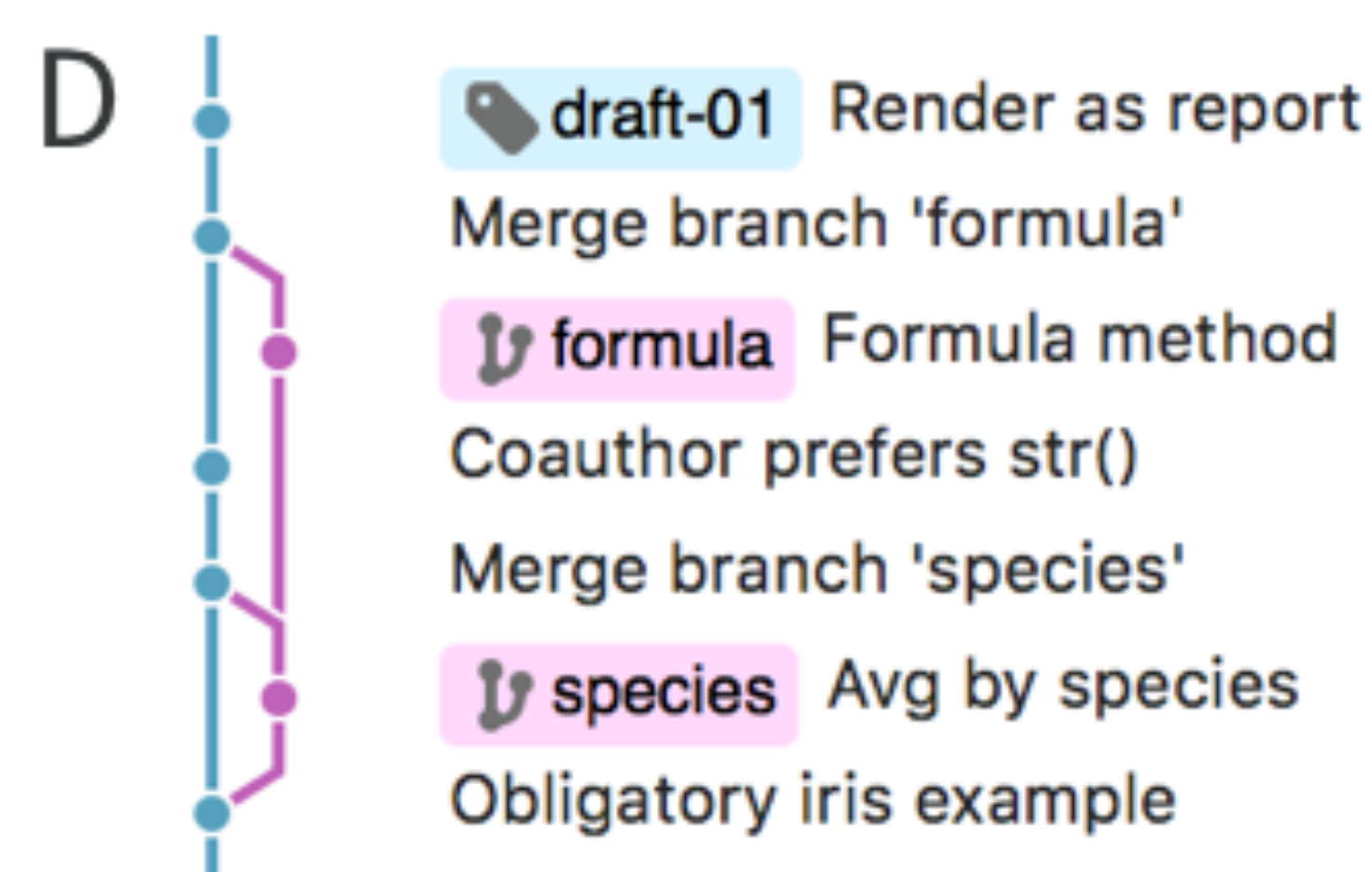
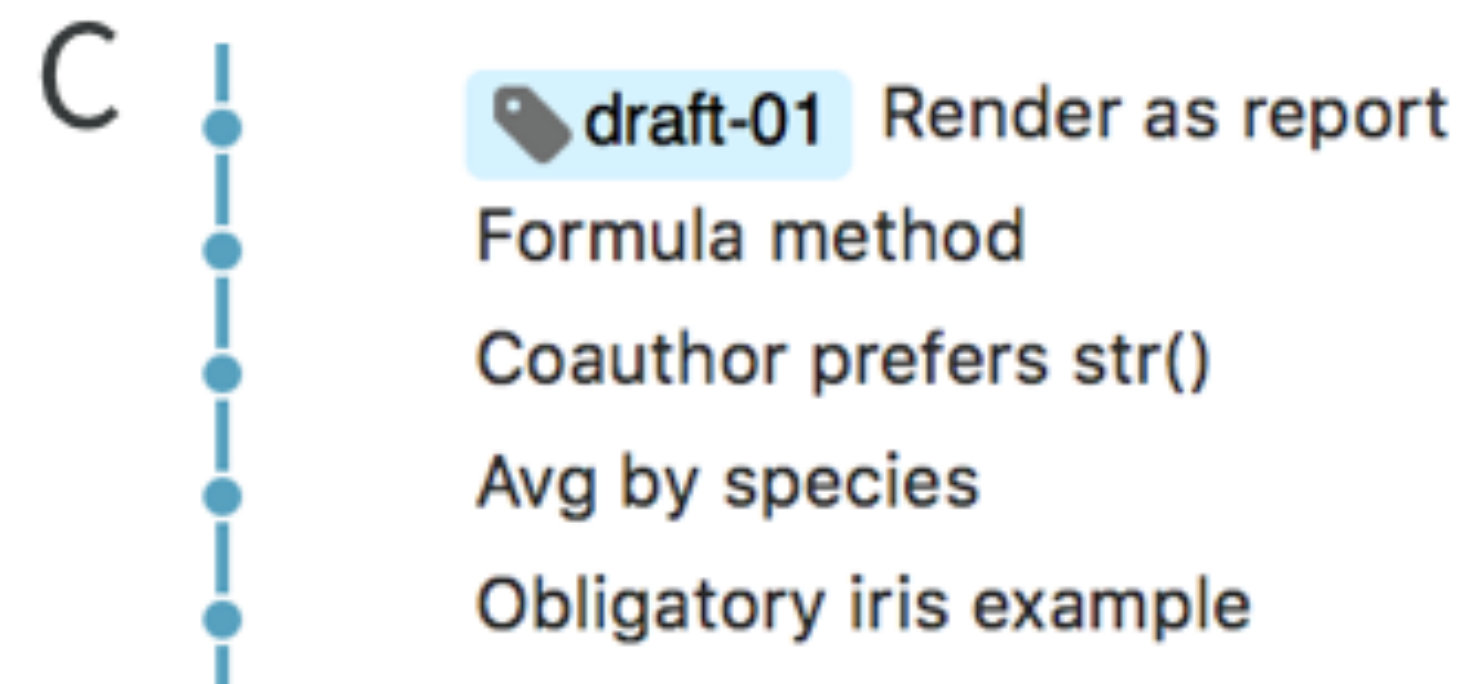
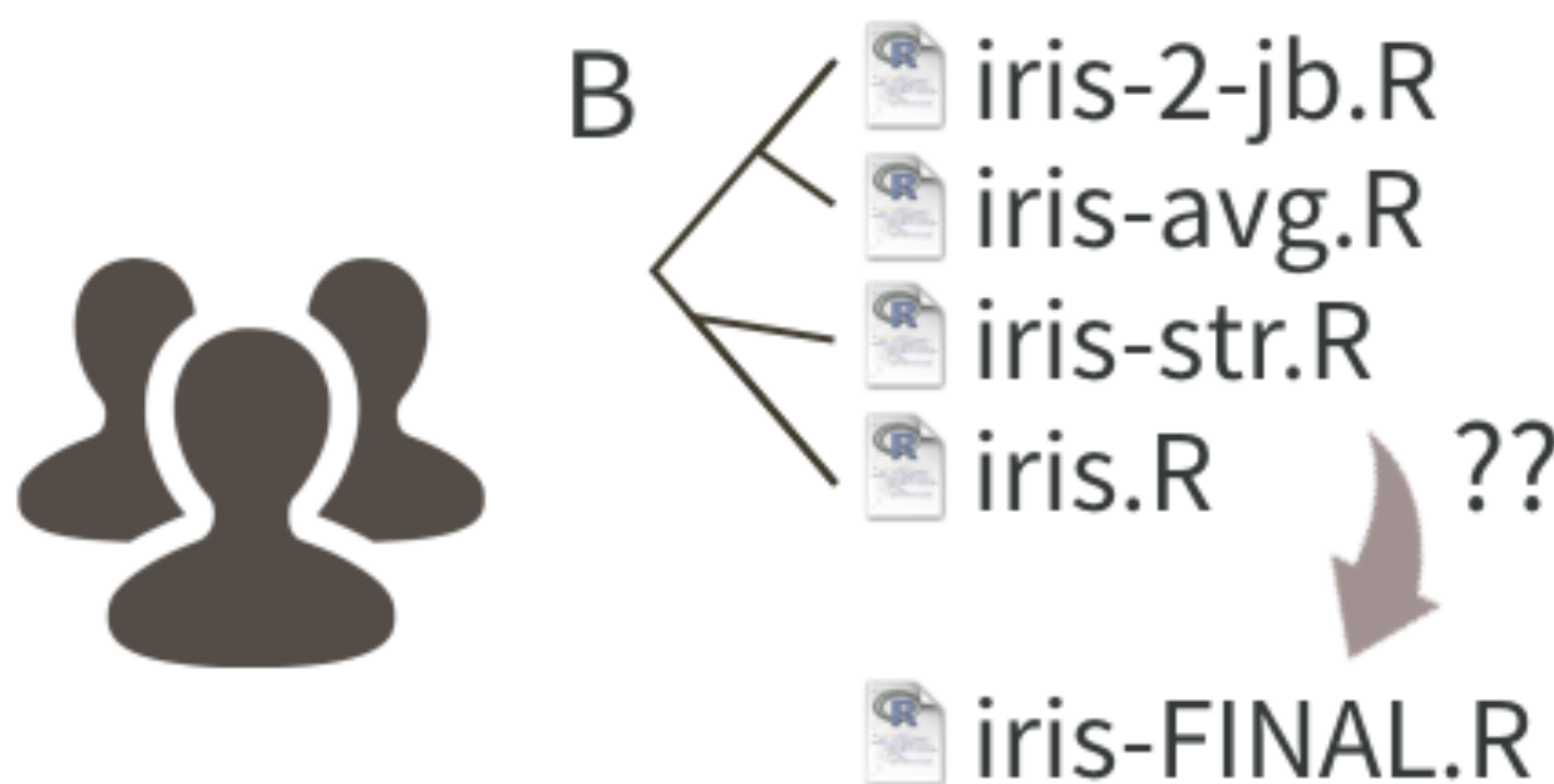
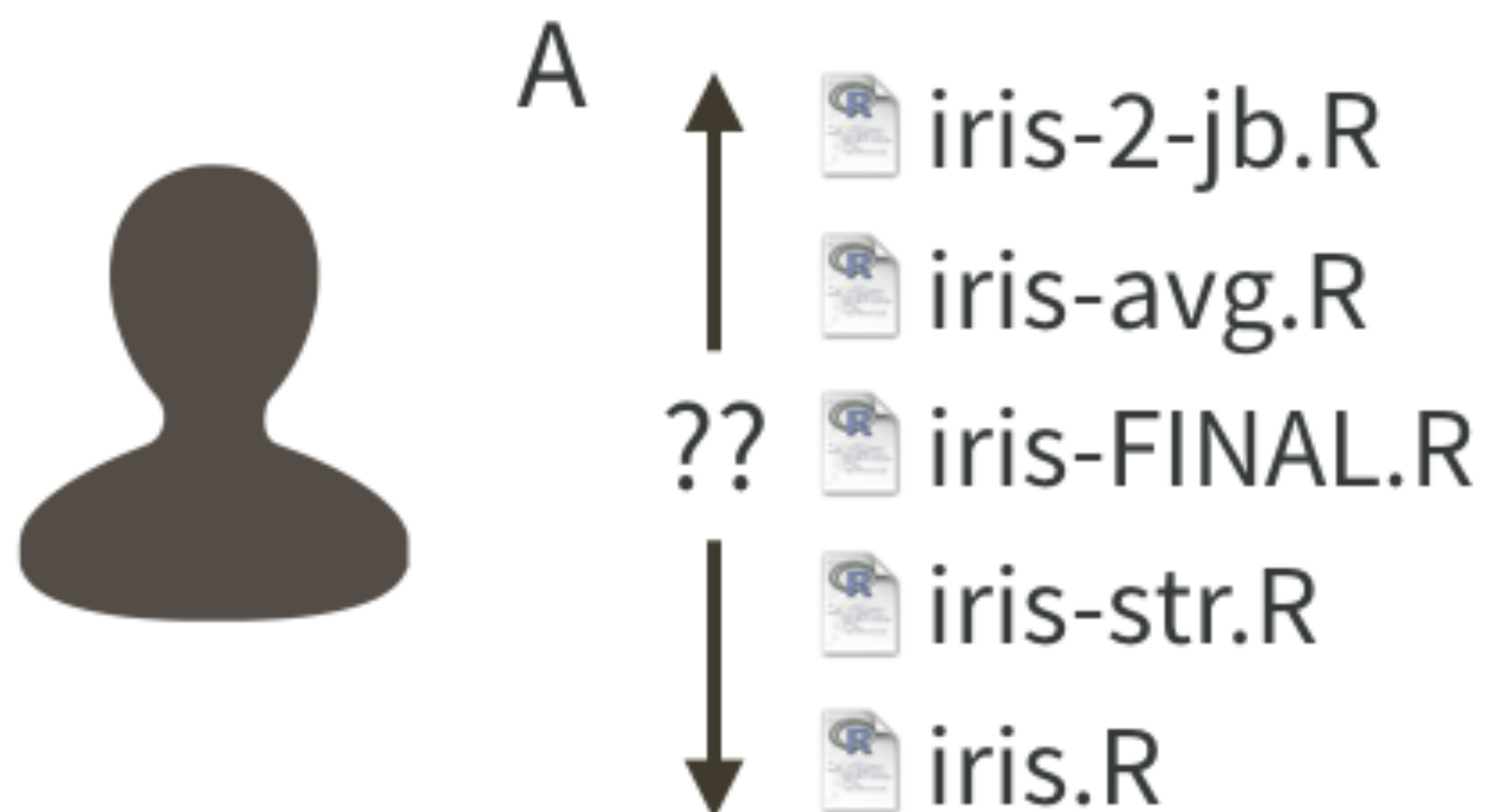
a file or project state that is **meaningful to you**
for inspection, comparison, restoration



Δ

"diff"

What changed here?
Why?



happygitwithr.com



Why version control?

- experiment without fear
- explore cause and effect
- embrace incrementalism
- collaborate
- expose your work

agony reduction

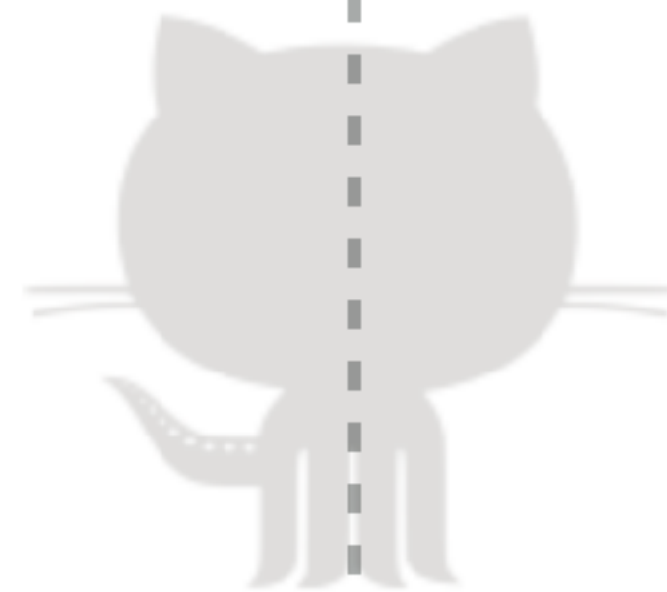
Use a Git client, if you like

No one is giving out Hard-core Git Nerd Badges

I like RStudio + SourceTree

<http://happygitwithr.com/git-client.html>

Them



You

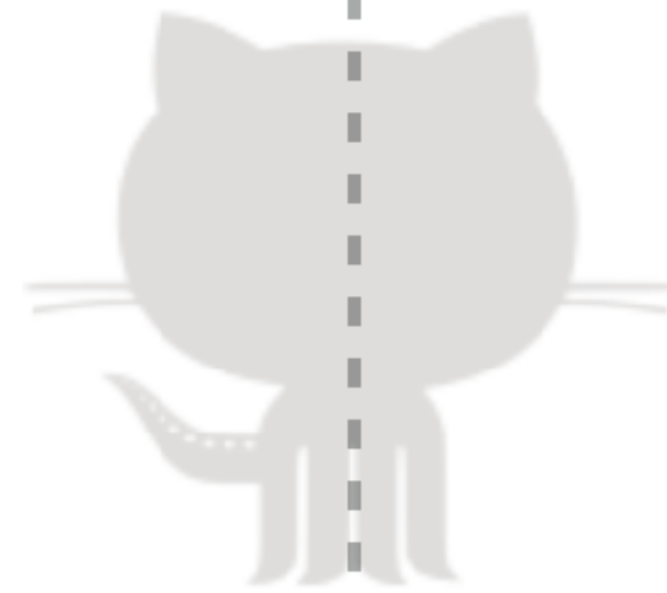


not your
problem



"clone"

Them



You



push



pull



not your
problem

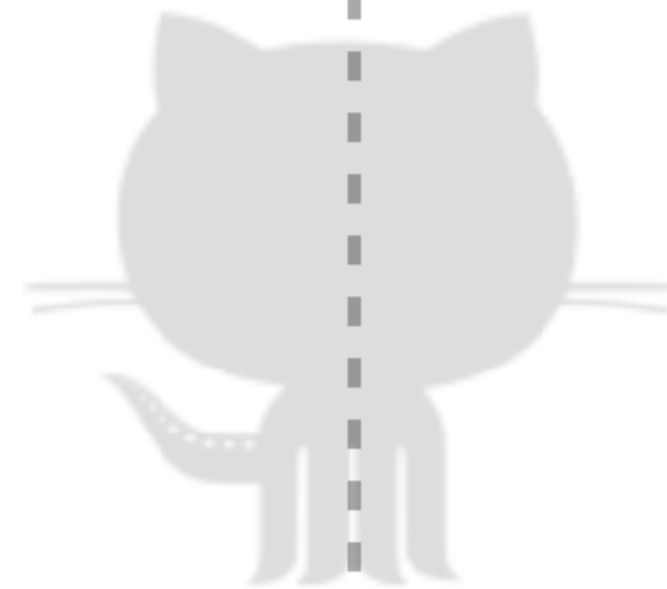


daily work, your stuff

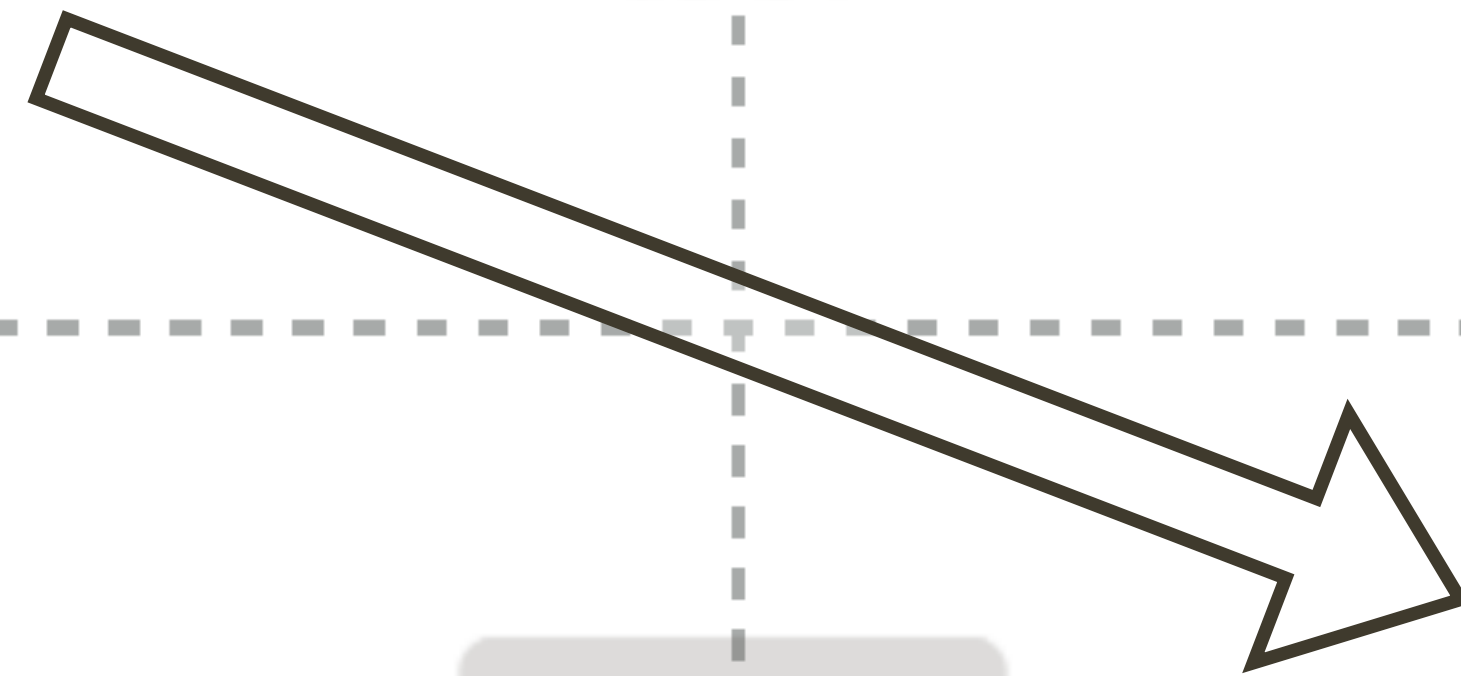
Them



You



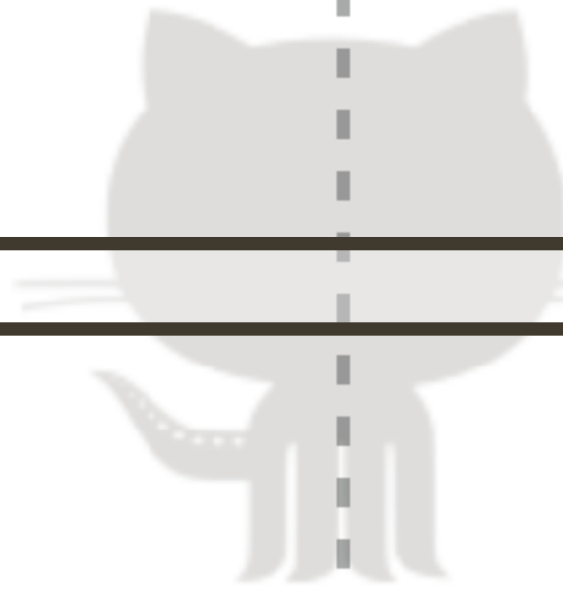
not your
problem



"clone"

*not as useful as you might think

Them



You

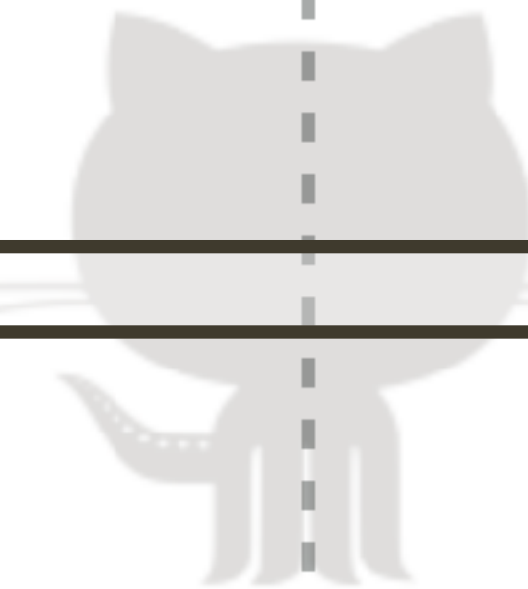


not your
problem

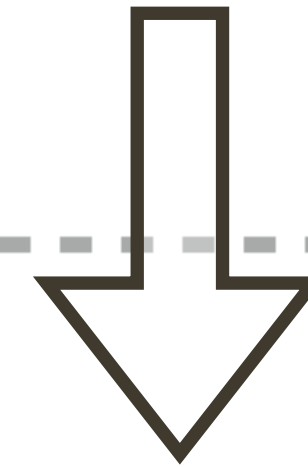


"fork"

Them



You



not your
problem



"fork and clone"

Them

You



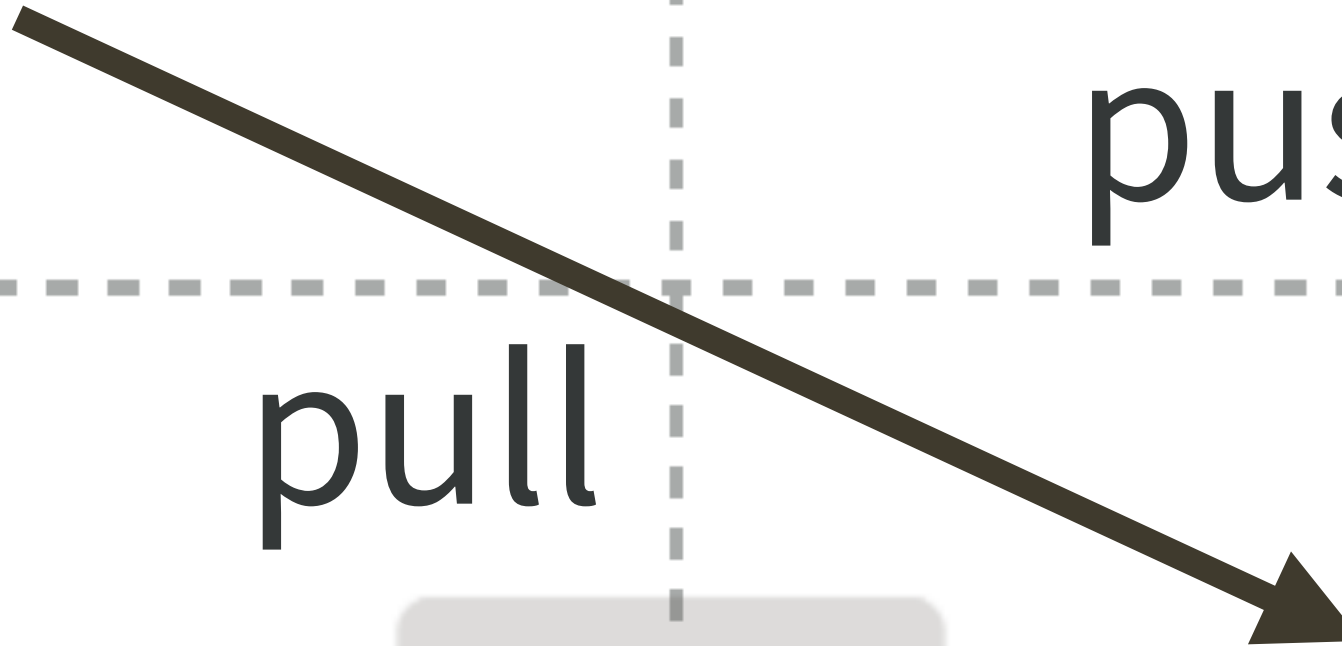
pull request



push



pull

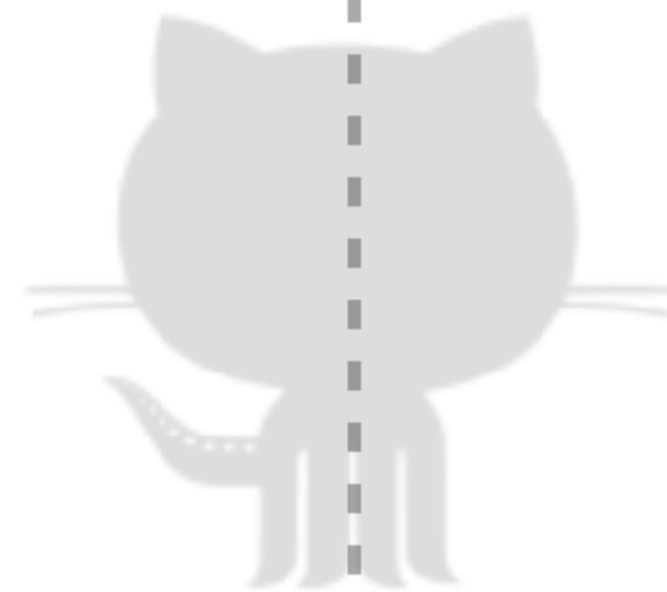


not your
problem



contribute to other people's stuff

Them



You



push



pull

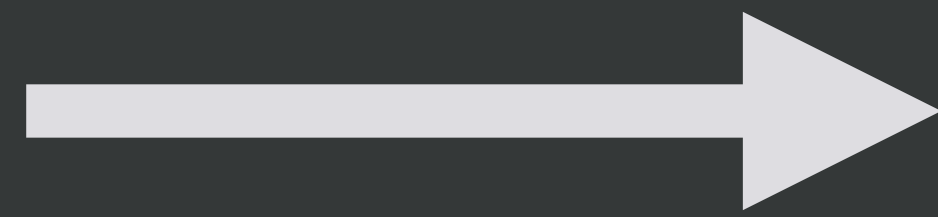


not your
problem



daily work, your stuff

what you
need to write



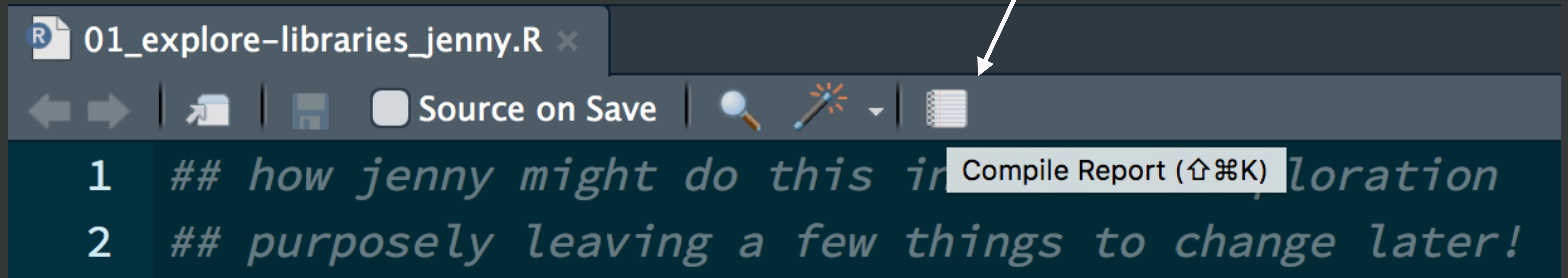
what people
like to read

foo.R
foo.Rmd



foo.md
foo.html

Compile Report



≈ `rmarkdown::render("whatever.R")`

Compile Report from R Script

Create a standalone report that contains the code and output from your R script.

For more information on compiling reports, see the documentation at [Compiling Reports from R Scripts](#)

Report output format:

HTML

Compile

Cancel

Sure, HTML is fine ... for now.

248 lines (201 sloc) | 723 KB

Raw

Blame

History



```
1 <!DOCTYPE html>
2
3 <html xmlns="http://www.w3.org/1999/xhtml">
4
5 <head>
6
7 <meta charset="utf-8" />
8 <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
9 <meta name="generator" content="pandoc" />
10
11
12 <meta name="author" content="jenny" />
13
14
15 <title>01_explore-libraries_jenny.R</title>
16
17
18 <meta name="viewport" content="width=device-width, initial-scale=1" />
19
20 <script src="data:application/x-javascript;base64,LyohCiAqIEJvb3RzdHJhcCB2My4zLjUgKGh0dHA6Ly9nZXRib290c3RyYXAuY29tKQogKiBDdb3B!
21 <script src="data:application/x-javascript;base64,LyoqCiogQHByZXNlc3ZlIEhUTUw1IFNoaXYgMy43LjIgfiCBAYWZhcmthcyBAamRhbHRvbiBAam9!
22 <script src="data:application/x-javascript;base64,LyohIFJlc3BvbWQuanMgdjEuNC4yOiBtaW4vbWF4LXdpZHRoIG1lZGhIHF1ZXJ5IHBvbHlmaWx!
23 <script src="data:application/x-javascript;base64,CgovKioKICogaF1ZXJ5IFBsdWdpbjogU3R5Y2t5IFRhYnMKICoKICogQGF1dGhvc3BwRhbiB!
24 <link href="data:text/css; charset=utf-8,%2Ehljs%2Dliteral%20%7B%0Acolor%3A%20%23990073%3B%0A%7D%0A%2Ehljs%2Dnumber%20%7B%0Aco
```



NO, raw `.html` is NOT immediately useful* on GitHub.

But Markdown = `.md` is useful.

Let's render `.R` to `.md` instead of `.html`!

* it CAN BE useful in actual web publishing workflows

foo.R → foo.html

```
#' ---  
#'| title: "Untitled"  
#'| output: html_document  
#'| ---
```

foo.R → foo.md → foo.html

```
#'| ---  
#'| title: "Untitled"  
#'| output:  
#'|   html_document:  
#'|     keep_md: yes  
#'| ---
```

foo.R → foo.md

```
#'| ---  
#'| output: md_document  
#'| ---
```

foo.R → foo.md

```
#'| ---  
#'| output: github_document  
#'| ---
```

01_explore-libraries_jenny.R

jenny Sat Jan 27 22:46:07 2018



```
## how jenny might do this in a first exploration  
## purposely leaving a few things to change later!
```

Which libraries does R search for packages?

```
.libPaths()
```

```
## [1] "/Users/jenny/resources/R/library"  
## [2] "/Library/Frameworks/R.framework/Versions/3.4/Resources/library"
```

```
## let's confirm the second element is, in fact, the default library  
.Library
```

```
## [1] "/Library/Frameworks/R.framework/Resources/library"
```

This is what I mean by "expose your work".

Take away #1:

Consider putting rendered products on GitHub.

Just because someone can fork, clone, install all necessary packages, then run your code, it doesn't mean they want to or will.

Be kind. Be realistic.

Take away #2:

For consumption on GitHub, Markdown (.md) is vastly more useful than .html, .docx, .pdf, etc.

Binary formats like .docx and .pdf are also a reliable source of merge conflicts. Think carefully before you track them with Git.

Simplest use of GitHub Pages = Project webpage

GitHub Pages

In your repo's Settings

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Source

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more.](#)

master branch ▾

Save

Theme Chooser

Select a theme to publish your site with a Jekyll theme using the master branch. [Learn more.](#)

Choose a theme

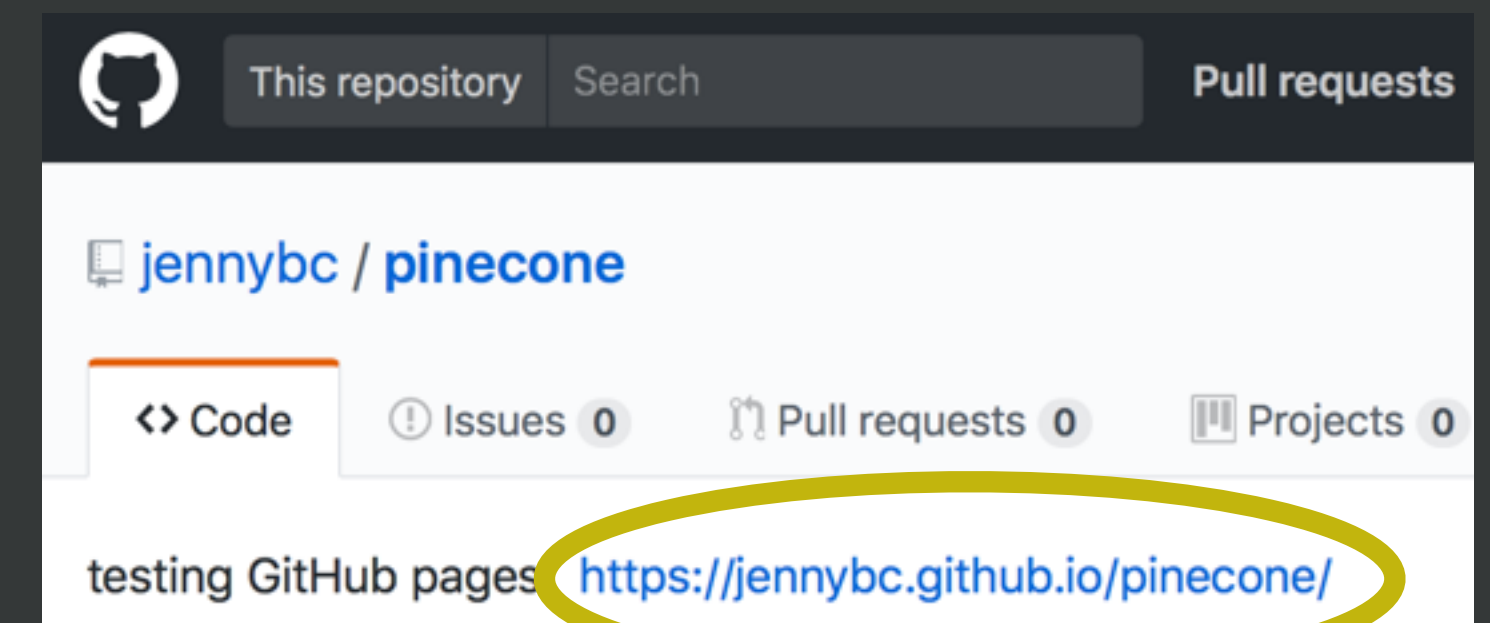
<https://github.com/blog/2289-publishing-with-github-pages-now-as-easy-as-1-2-3>

Hot tips for simple GitHub Pages

README.md becomes index.html, by default

Given that foo.md exists, these internal links work (trial & error):
👍 foo, [foo](foo.md), [foo](foo.html)

Record your site URL as your repo's website



Recovering from Git(Hub) failure

Scenario: You try to push and cannot

What's the problem?

There are changes on GitHub that you don't have.

Pull. If the gods smile upon you, merge works. Now push.

Recovering from Git(Hub) failure

Scenario: You pull and get a merge conflict.

What's the problem?

GitHub can't figure out how to reconcile diffs.

Resolve the conflicts.

Or abort ... and come back later.

<<<<<< HEAD

Wingardium Leviosaaaaaaa

=====

Wing-GAR-dium Levi-0-sa

>>>>>> 33579525d88af071268b0a0c64c54f357712589a

Git inserts **markers** at each locus of conflict and shows you both versions.

You must form a consensus version and delete the markers, at each locus. Commit. Push. Carry on.

Recovering from Git(Hub) failure

Scenario: You have a huge mess you cannot fix.

Official answer: git reset.

Unofficial answer: burn it all down 🔥

🔥 requires you have a remote repo in a decent state!

Commit early, commit often! And push! It's your safety net.

Rename local repo to, e.g. "foo-borked".

Re-clone to a new, clean local repo, "foo".

Copy any files that are better locally from "foo-borked" to "foo".

Commit. Push. Carry on.

smell-test.R

wrangle.R

model.R

>>>

everything.R

make-figs.R

report.Rmd

Import

smell-test.R

Tidy

wrangle.R

Transform

make-figs.R

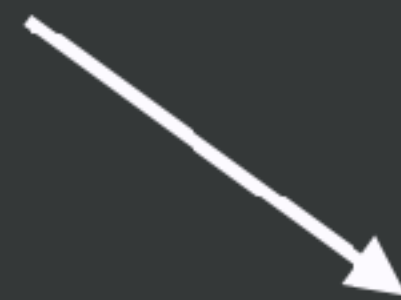
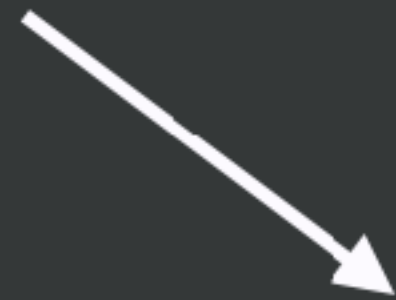
Visualise

Model

model.R

report.Rmd

Communicate



raw-data.xlsx

data.csv

fits.rds

ests.csv

>>>

.Rdata

raw-data.xlsx
Import

Tidy data.csv

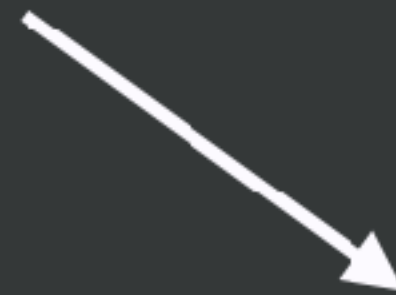
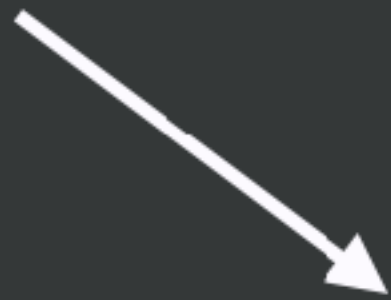
Transform

figs/hist.png
figs/dot.png

Visualise

fits.rds
ests.csv
Model

Communicate



<http://happygitwithr.com/search.html>

<https://github.com/search/>

The `col_type` argument to `read_csv` seems to be used in different ways? How can I see ways people are using it?

What if I wanted results from `read_delim` as well?

How can you filter out python results? Rd results?

read_csv OR read_delim col_type language:r extension:r

What packages are actually attached by the tidyverse package?

R packages define a `.onAttach()` function to run code when the package is attached by `library()`

'**user:**' is used to search both user repos and organization repos.

The **cran** organization is a mirror of all CRAN packages

The **tidyverse** organization has all tidyverse packages

'**repo:**' is used to search a specific repository.

user:cran onAttach

user:tidyverse onAttach

repo:tidyverse/ggplot2 onAttach

```
print(mtcars, print.gap = -1)
```

```
#> Error in print.default(m, ..., quote = quote, right = right) :
```

```
#> 'gap' must be non-negative integer
```

Search on google for error message string

Search in R source for error message

If error translated, look in .po files for english version

repo:wch/r-source "'gap' must be non-negative integer"

lookup package

github.com/jimhester/lookup

Not on CRAN

Powered by GitHub search

Works for *most* functions