

# Best Practices in R

2022-08-26

developed by Emil Hvitfeldt

# Welcome!



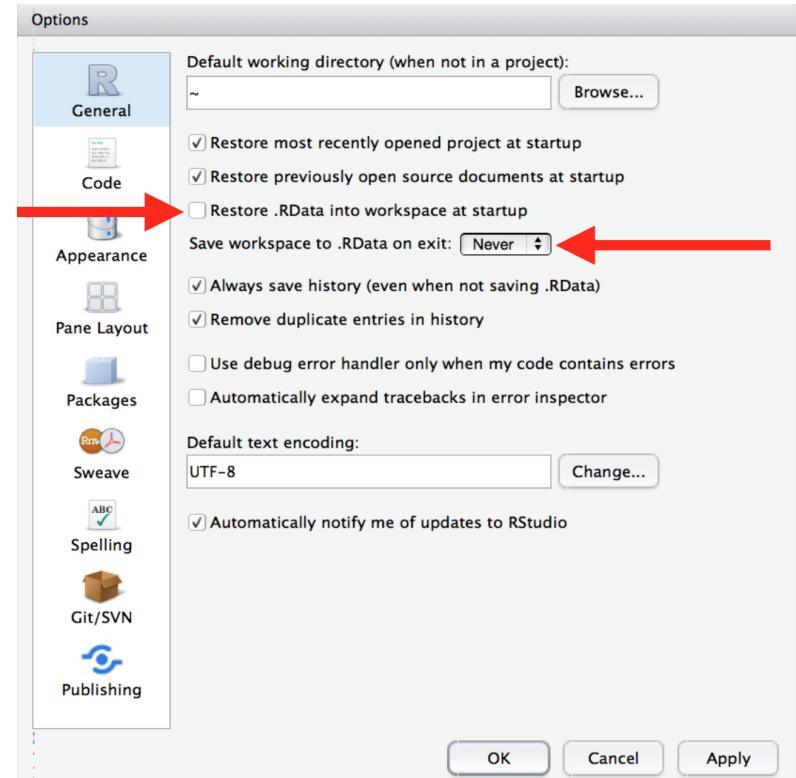
# Change Settings

**Keyboard shortcut to open settings**

⌘ + , in Mac OS,  
ctrl + , in Windows

**✓ - Uncheck "Restore .RData into work space at start up"**

**✓ - Set "Save work space to .Rdata on exit" to "Never"**



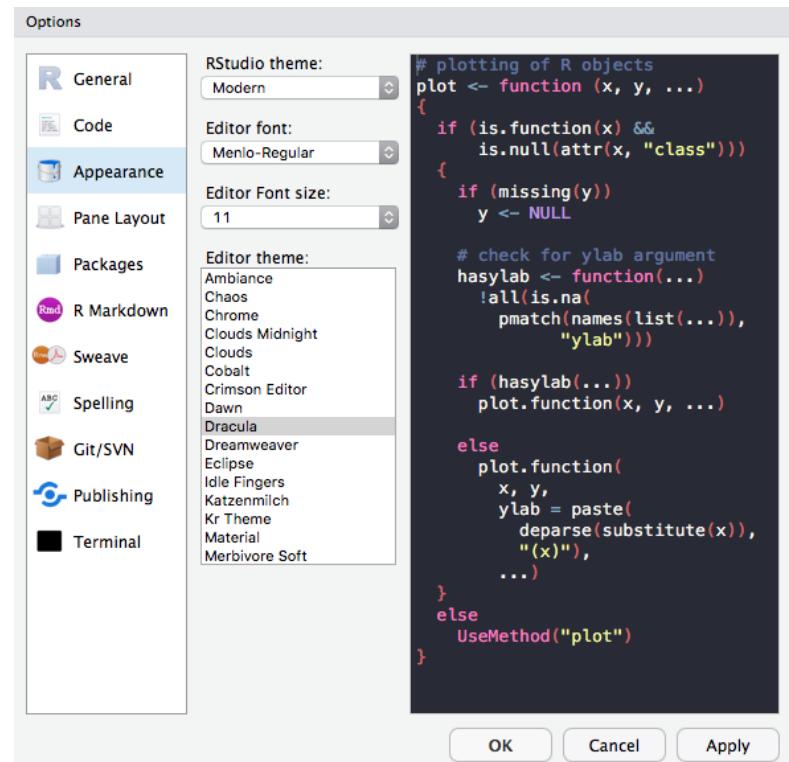
# Change Appearance

## RStudio themes

## Fonts

## Font Sizes

## Editor Themes



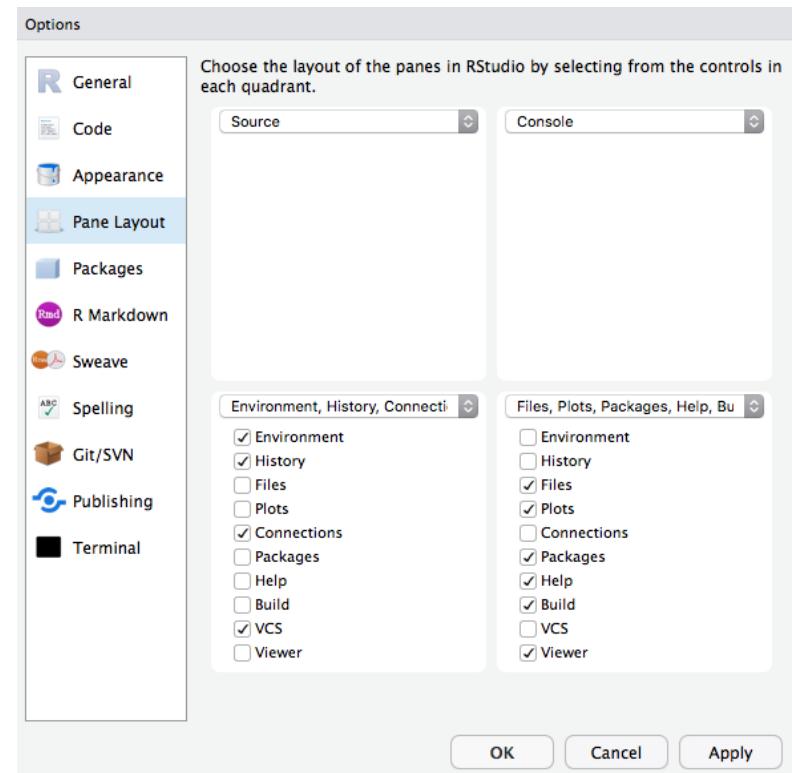
# Pane layouts

Change the layout  
of the panes

Source on top?

Source down to  
the right?

It's all up to you!



# Pane layouts

Some like having both source and console open

The screenshot shows the RStudio interface with a split-pane layout. The left pane is a code editor displaying R Markdown code related to pane layouts. The right pane is a terminal window showing the execution of the code, including pandoc command-line arguments and the resulting HTML output file.

```
~/Documents/Academia/Github/oRganized-talk - master - RStudio
2018-10-29_getting-organized-with-r.Rmd
66 .pull-left[
67 `* + ,` in macOS,
68 `ctrl + ,` in Windows
69 ]
70 Uncheck "Restore .RData into workspace at startup"
71 And Set "Save workspace to .Rdata on exit" to "Never"
72 ]
73
74 .pull-right[! [Settings window](images/settings-general.png)]
75
76 ---
77
78 ## Change appearance
79
80 .pull-left[
81
82 ]
83
84 .pull-right[! [Settings window](images/settings-appearance.png)]
85
86 ---
87
88 ## Panel layouts
89
90 .pull-left[
91 |
92 ]
93 .pull-right[! [Settings window](images/settings-pane-layout.png)]
94
95 ---
96
97
98 ## Use R projects
99

~/Documents/Academia/Github/oRganized-talk / 
|..... ordinary text without R code | 20%
|..... label: setup (with options) List of 1 $ include: logi FALSE | 40%
|..... ordinary text without R code | 60%
|..... label: unnamed-chunk-1 | 80%
|..... ordinary text without R code | 100%

output file: 2018-10-29_getting-organized-with-r.knit.md

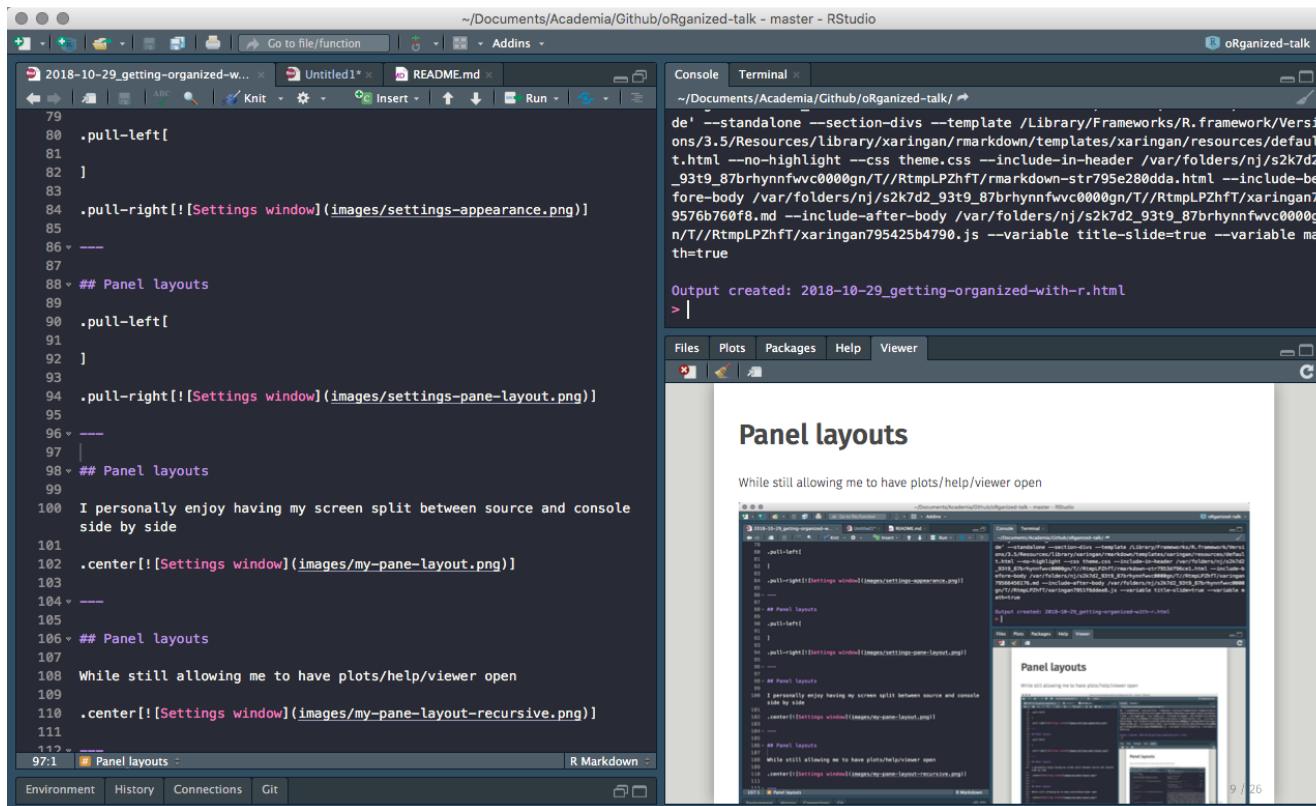
/usr/local/bin/pandoc +RTS -K512m -RTS 2018-10-29_getting-organized-with-r.utf8.
md --to html4 --from markdown+autolink_bare_uris+asci_identities+tex_math_sing
le_backslash+smart --output 2018-10-29_getting-organized-with-r.html --email-obf
uscation none -V 'mathjax-url=https://cdn.bootcss.com/mathjax/2.7.1/MathJax.js?c
onfig=TeX-MML-AM_CHTML' -V 'title-slide-class=center, middle, inverse, title-sli
de' --standalone --section-divs --template /Library/Frameworks/R.framework/Versi
ons/3.5/Resources/library/xaringan/rmarkdown/templates/xaringan/resources/default.
html --no-highlight --css theme.css --include-in-header /var/folders/nj/s2k7d2_
93t9_87brhynnfwc000gn/T//RtmpLPZhft/rmarkdown-str7956f2c0896.html --include-b
efore-body /var/folders/nj/s2k7d2_93t9_87brhynnfwc000gn/T//RtmpLPZhft/xaringan
79540adcae.md --include-after-body /var/folders/nj/s2k7d2_93t9_87brhynnfwc000
gn/T//RtmpLPZhft/xaringan7953cdcbfb1.js --variable title-slide=true --variable m
ath=true

Output created: 2018-10-29_getting-organized-with-r.html
> |
```

Environment History Connections Git Files Plots Packages Help Viewer

# Pane layouts

...while still allowing to have viewer open



The screenshot shows the RStudio interface with a split-pane layout. The left pane contains R code:

```
~/Documents/Academia/Github/oRganized-talk - master - RStudio
2018-10-29_getting-organized-w... Untitled1* README.md
79
80 .pull-left[
81
82 ]
83
84 .pull-right![! [Settings window](images/settings-appearance.png)]
85
86 -->
87
88 ## Panel layouts
89
90 .pull-left[
91
92 ]
93
94 .pull-right![! [Settings window](images/settings-pane-layout.png)]
95
96 -->
97
98 ## Panel layouts
99
100 I personally enjoy having my screen split between source and console
101 side by side
102 .center![! [Settings window](images/my-pane-layout.png)]
103
104 -->
105
106 ## Panel layouts
107
108 While still allowing me to have plots/help/viewer open
109
110 .center![! [Settings window](images/my-pane-layout-recursive.png)]
111
112 -->
97:1 Panel layouts R Markdown
Environment History Connections Git
```

The right pane shows the output of the R code, which generates an HTML file:

```
~/Documents/Academia/Github/oRganized-talk/
de' --standalone --section-divs --template /Library/Frameworks/R.framework/Versions/3.5/Resources/Library/xaringan/markdown/templates/xaringan/resources/default.html --no-highlight --css theme.css --include-in-header /var/folders/nj/s2k7d2_93t9_87brhynfwvc000gn/T/RtmpLPZhf/rmarkdown-str795e280da.html --include-before-body /var/folders/nj/s2k7d2_93t9_87brhynfwvc000gn/T/RtmpLPZhf/xaringan79576b760f8.md --include-after-body /var/folders/nj/s2k7d2_93t9_87brhynfwvc000gn/T/RtmpLPZhf/xaringan795425b4790.js --variable title=slide=true --variable math=true

Output created: 2018-10-29_getting-organized-with-r.html
> |
```

The bottom right corner of the RStudio window shows a preview of the generated HTML page titled "Panel layouts". The page content matches the R code above, including the "While still allowing me to have plots/help/viewer open" section.

# RStudio Projects

**Keep all files from one project together. Use RStudio projects.**

**Self contained**

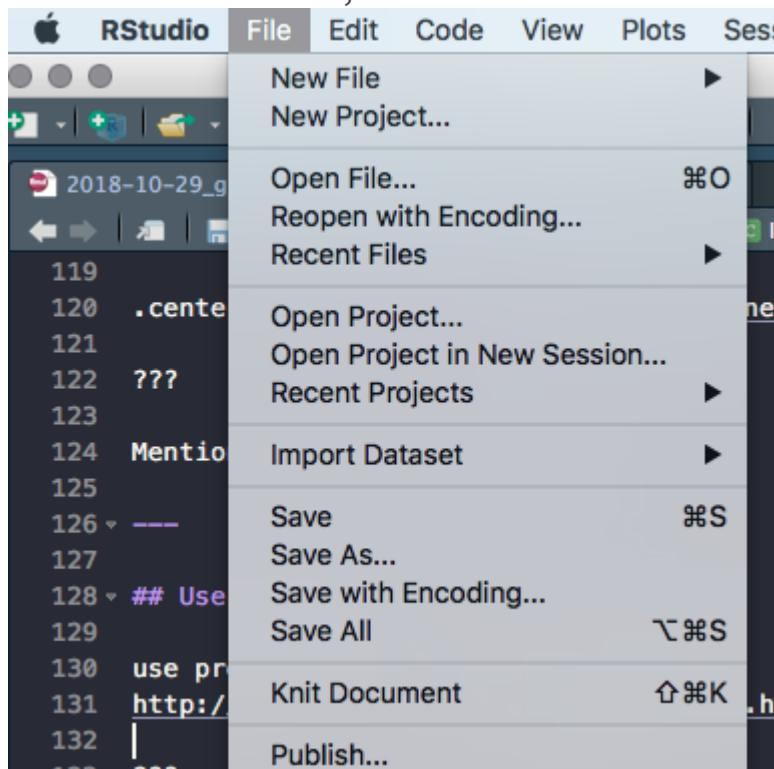
**Project orientated**

# usethis

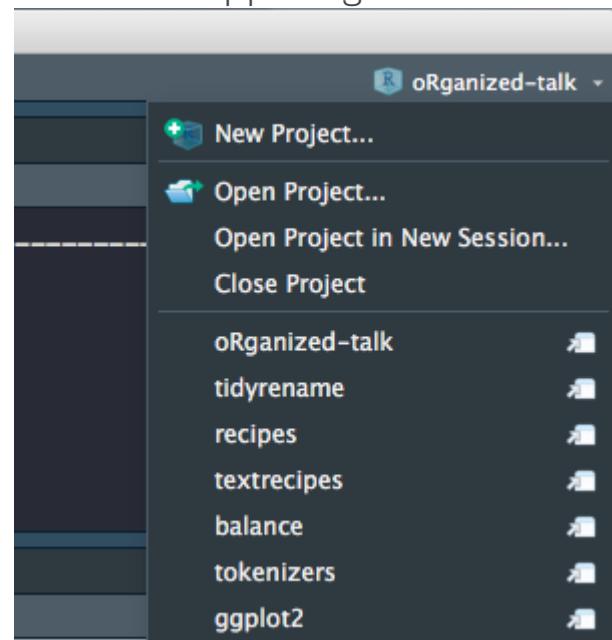
```
usethis::create_project("project_name")
```

# RStudio Projects - Creation 1 / 4

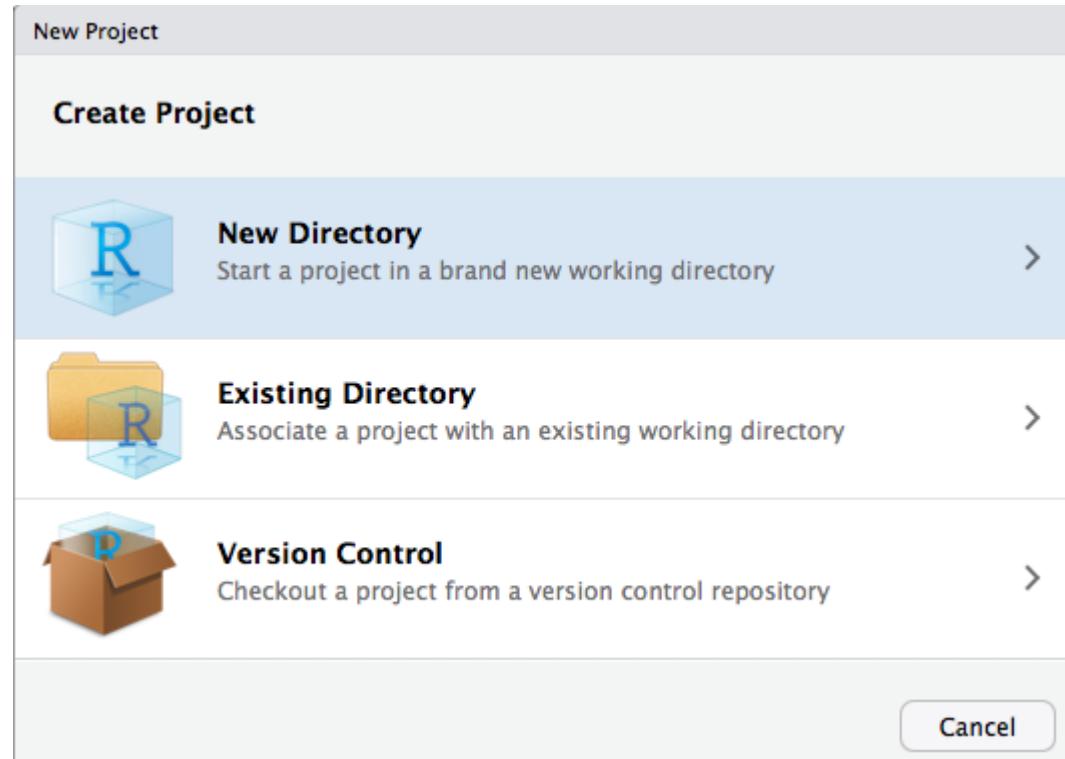
Click File > New Project



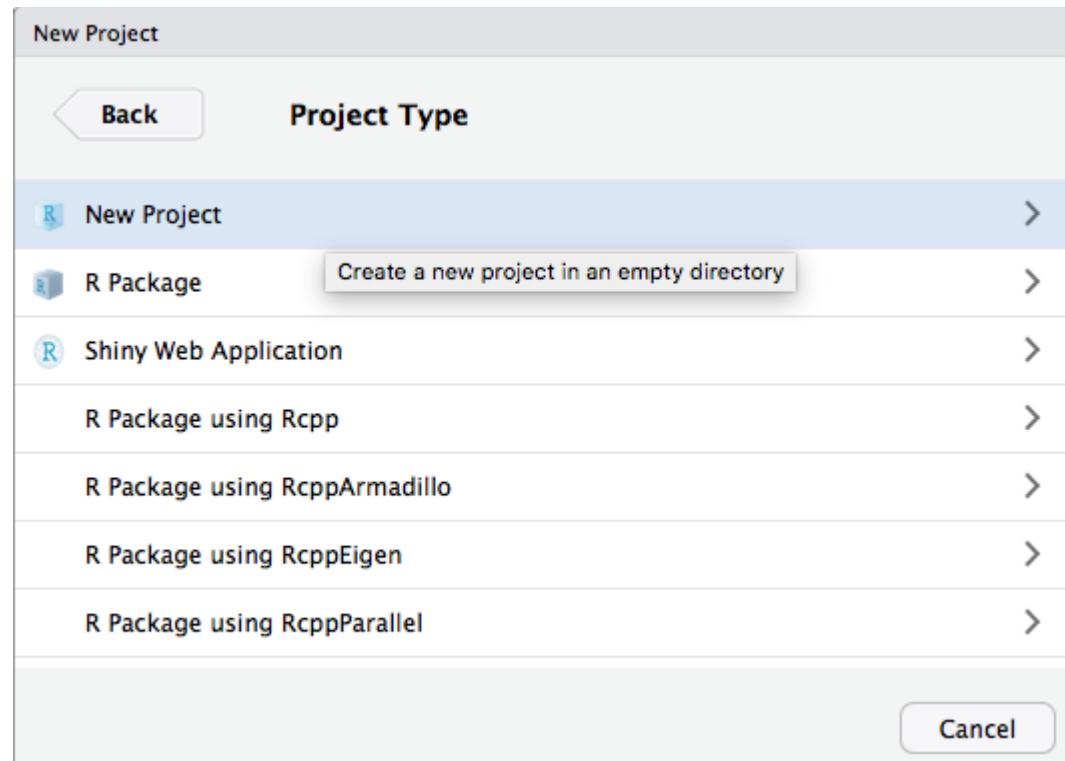
Or click on the upper right



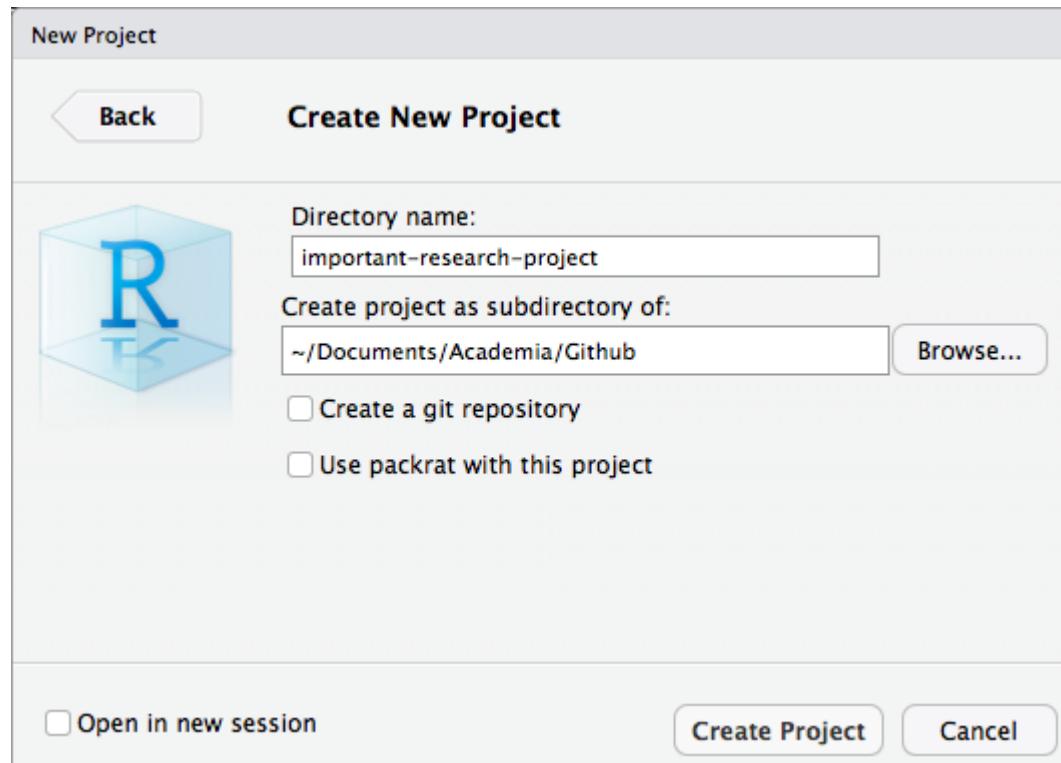
# RStudio Projects - Creation 2 / 4



# RStudio Projects - Creation 3 / 4



# RStudio Projects - Creation 4 / 4



# Folder Structure

```
name_of_project
| --raw_data
|   |--WhateverData.xlsx
|   |--report_2017.csv
| --output_data
|   |--summary2017.csv
| --rmd
|   |--01-analysis.Rmd
| --docs
|   |--01-analysis.html
|   |--01-analysis.pdf
| --scripts
|   |--exploratory_analysis.R
|   |--pdf_scraping.R
| --figures
|   |--weather_2017.png
| --name_of_project.Rproj
| --run_all.R
```

- 1 Raw data separate from cleaned data
- 2 Reports and scripts are separated
- 3 Generated and imported figures has its own place
- 4 Numbered using 2 digits
- 5 Reusable and easily understandable

# Folder Structure

```
library(fs)
folder_names <- c("raw_data", "output_data", "rmd", "docs",
                  "scripts", "figures")

dir_create(fldr_names)
```

# Paths

```
library(tidyverse)

# data import
data <- read_csv("/Users/Emil/Research/Health/amazing_data.csv")
```

# Paths

```
library(tidyverse)  
  
# data import  
data <- read_csv("/Users/Emil/Research/Health/amazing_data.csv")  
  
## Error: '/Users/Emil/Research/Health/amazing_data.csv' does not exist.
```

**Only use relative paths, never absolute paths**

# Introducing the `here` package.

```
library(here)  
here()
```

```
## [1] "/Users/Emil/Research/Health"
```

```
library(here)  
data <- read_csv(here("raw_data", "amazing_data.csv"))
```

# Naming Things



Caitlin Hudon 📱  
@beeonaposy

Follow



Jumping back into code you wrote ages ago  
like

10% luck

20% skill

15% concentrated power of will

5% pleasure

50% pain

100% wishing you used descriptive names

2:34 PM - 6 May 2018

# Naming Things - Files

NO

```
report.pdf  
reportv2.pdf  
reportthisisthelastone.pages  
Figure 2.png  
3465-234szx.r  
foo.R
```

YES

```
2018-10-01_01_report-for-cdc.pdf  
01_data.rmd  
01_data.pdf  
02_data-filtering.rmd  
02_data-filtering.pdf
```

- 1 Avoid spaces, punctuation, special characters and case sensitivity
- 2 Deliberate use of delimiters
- 3 Describe the contents of the file
- 4 Put something numeric first
- 5 Left pad numbers with zeroes
- 6 Use a standard date (YYYY-MM-DD)

# Naming Things - Files

```
library(fs)
dir_ls("data/", regexp = "health-study")
```

```
## 2018-02-23_health-study_power-100_group-A1.csv
## 2018-02-23_health-study_power-100_group-B1.csv
## 2018-02-23_health-study_power-100_group-C1.csv
## 2018-02-23_health-study_power-200_group-A1.csv
## 2018-02-23_health-study_power-200_group-B1.csv
## 2018-02-23_health-study_power-200_group-C1.csv
```

```
stringr::str_split_fixed(x, "[_\\.]", 5)
```

```
##      [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] "2018-02-23" "health-study" "power-100" "group-A1" "csv"
## [2,] "2018-02-23" "health-study" "power-100" "group-B1" "csv"
## [3,] "2018-02-23" "health-study" "power-100" "group-C1" "csv"
## [4,] "2018-02-23" "health-study" "power-200" "group-A1" "csv"
## [5,] "2018-02-23" "health-study" "power-200" "group-B1" "csv"
## [6,] "2018-02-23" "health-study" "power-200" "group-C1" "csv"
```

# Naming Things - Files

```
library(tidyverse)
map_df(dir_ls("data/", regexp = "health-study"), read_csv)

# or

dir_ls("data/", regexp = "health-study") %>%
  map_df(read_csv)
```

# Naming Things - Objects

- 1 Only use lowercase letters, numbers, and \_
- 2 Use names that are not jargony, weight instead of K
- 3 Use informative names

# Naming Things - Objects

*# Bad*

```
df  
e  
tuningVar
```

*# Good*

```
health_data  
error  
tuning_var
```

# What To Avoid - attach()

**Never use attach()**

```
attach(mtcars)  
mean(mpg)
```

```
## [1] 20.09062
```

Loads lots of names into the search path, ambiguous selections.

Try `with()` or `withr` instead

# What To Avoid - attach()

**Never use `rm(list=ls())`**

**Instead, restart the R session**

**CTRL+SHIFT+F10 for Windows**

**CMD+SHIFT+ALT+F10 for Mac OS**

# R Markdown documents versus R scripts

You can use R scripts for simple self contained tasks.

`source()` R scripts into your R Markdown document where you will do analyses, visualizations and reporting.

# R Markdown

- 01-import.R
- 02-clean-names.R
- 03-tidy.R
- etc

Include at the start of R Markdown file

```
{r load_scripts, include = FALSE}
library(here)
source(here("scripts", "01-import.R"))
source(here("scripts", "02-clean-names.R"))
source(here("scripts", "03-tidy.R"))
```

# Naming Chunks

**Names can be placed after the comma**

```
```{r, chunk-label, results='hide', fig.height=4}
```

**or before**

```
```{r chunk-label, results='hide', fig.height=4}
```

In general it is recommended to use alphabetic characters with words separated by - and avoid other characters. - Yihui Xie

- 1 Makes navigating the R Markdown document easier
- 2 Makes your R Markdown easier to understand
- 3 Clarifies error reports or progress of knitting
- 4 Caching when moving chunks around

# Setup Chunk

In a fresh R Markdown document you see this

```
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

The setup chunk is run before another code - use to your advantage

# Setting figure path

```
```{r setup, include=FALSE}
knitr::opts_chunk$set(fig.path = "figures/")
```

# Styling Code

**Use consistent style when writing code**

<http://style.tidyverse.org/>

All about preferences but keep it  
consistent!!!

**Use the styler package to style  
your code for you**

# Keep .Rprofile Clean

Your computer contains a file called .Rprofile.

This file runs first in every session.  
Think of it as configuration file.

```
options(stringsAsFactors = FALSE)
options(max.print = 100)
```

# Keep .Rprofile Clean

**Only** put interactive code in

**Yes**

```
# add this with usethis::use_usethis()
library(usethis)
```

**No**

```
library(tidyverse)
```

# Comment Your Code

## Functions: Arguments and purpose

## Code: What or why, NOT how

```
# Takes a data.frame (data) and replaces the columns with the names
# (names) and converts them from factor variable to character
# variables. Keeps characters variables unchanged.
factor_to_text <- function(data, names) {
  for (i in seq_along(names)) {
    if(is.factor(data[, names[i], drop = TRUE])))
      data[, names[i]] <- as.character.factor(data[, names[i],
                                                drop = TRUE])
  }
  data
}
```

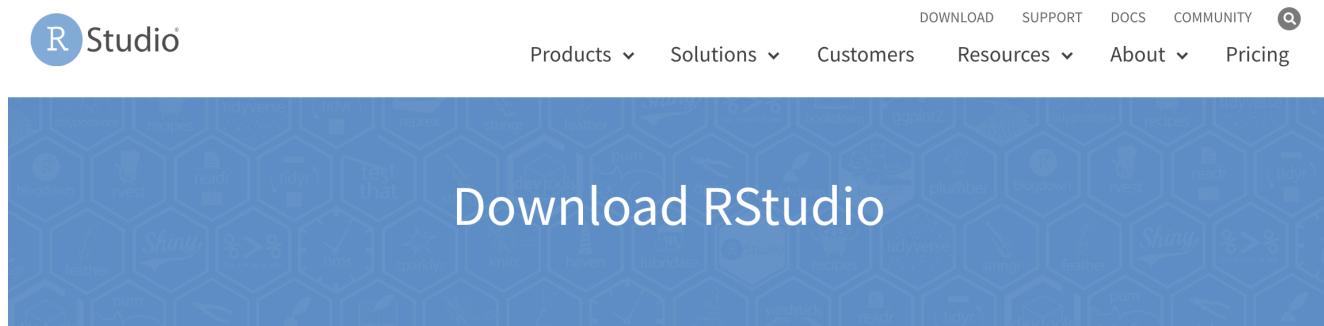
# Updating R and RStudio

The most recent version of R can be downloaded  
from **The Comprehensive R Archive Network (CRAN)**



# Updating R and RStudio

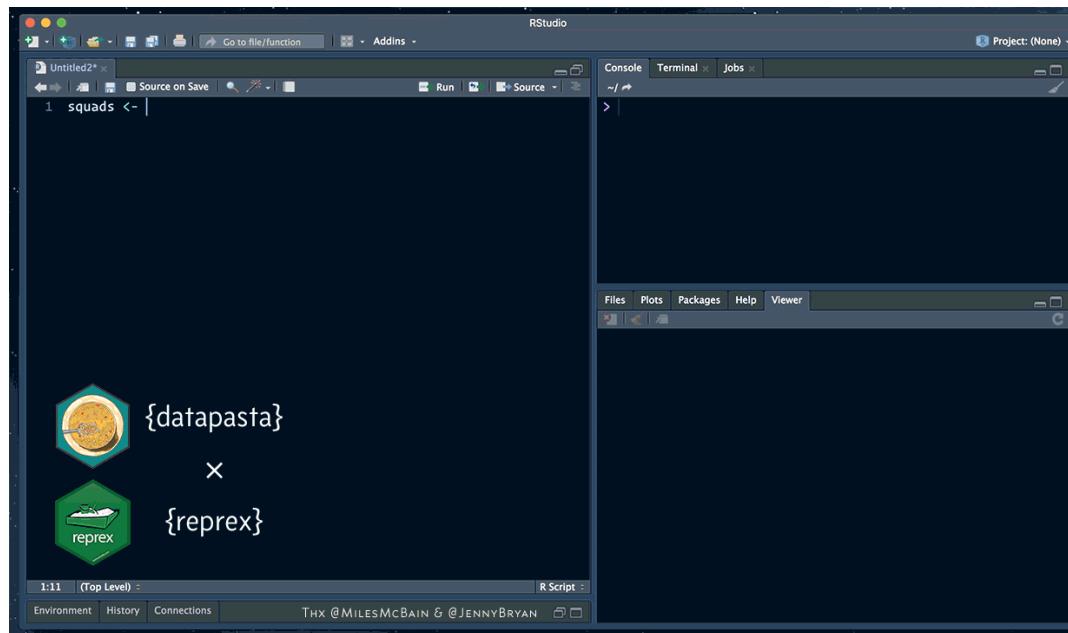
**Download the most recent version of RStudio at their [downloads page](#)**



# How to ask for help (datapasta and reprex)

The reprex package helps you create a reproducible example

datapasta lets you easy copy + paste small samples of data into RStudio



# How to ask for help (reprex)

Check out the **package website** and **RStudio webinar** on creating reproducible examples



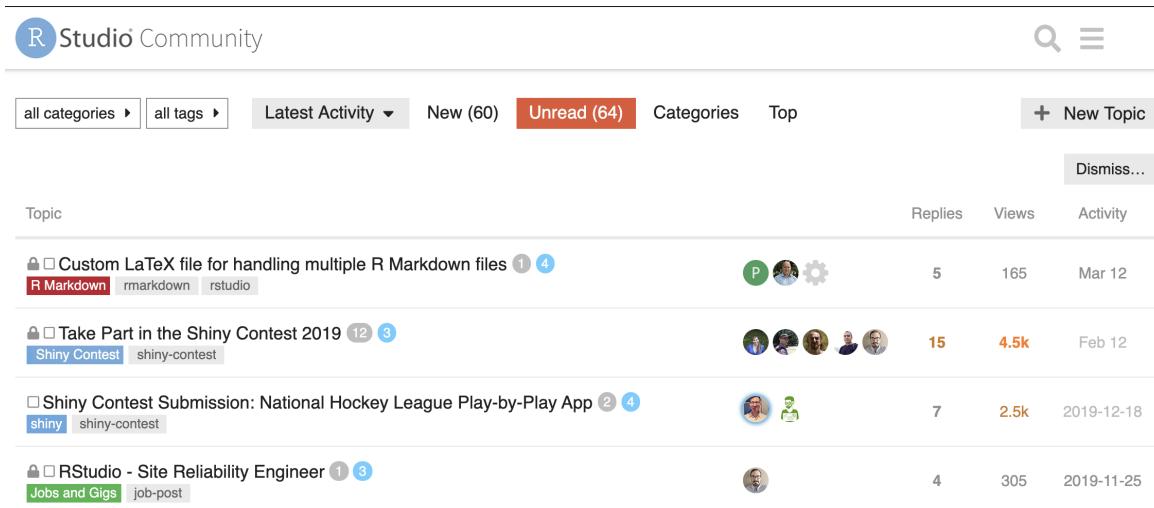
Art by Allison Horst

# Where to get help

**RStudio has a helpful community if you have questions (everyone does!)**

## RStudio Community:

RStudio has a dedicated forum for questions related to R and RStudio:  
<https://community.rstudio.com/>



The screenshot shows the RStudio Community forum interface. At the top, there is a navigation bar with links for 'all categories', 'all tags', 'Latest Activity', 'New (60)', 'Unread (64)' (which is highlighted in red), 'Categories', 'Top', a 'New Topic' button, and a 'Dismiss...' button. Below the navigation bar is a table listing five forum topics. The columns in the table are 'Topic', 'Replies', 'Views', and 'Activity'. Each topic row includes a small user icon, the topic title, its tags, and its last activity date.

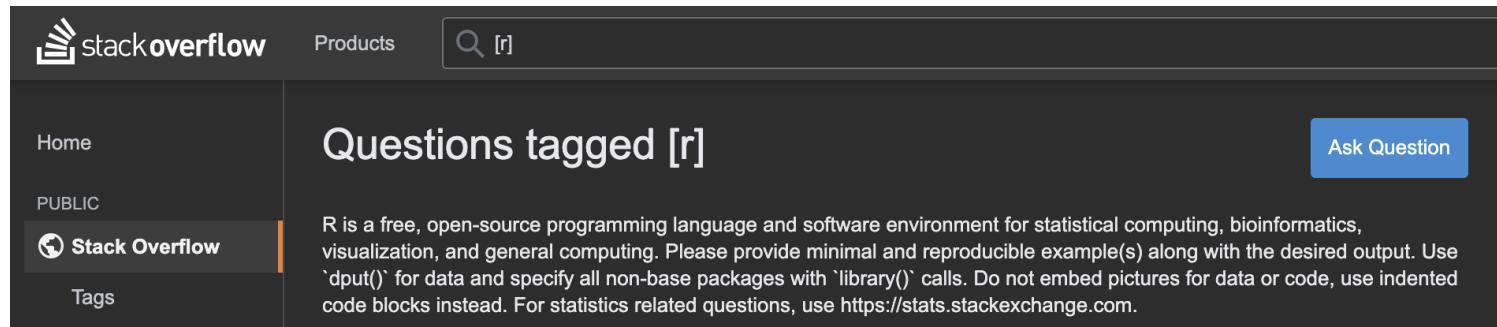
Topic	Replies	Views	Activity
Custom LaTeX file for handling multiple R Markdown files <small>1 4</small> R Markdown   rmarkdown   rstudio	5	165	Mar 12
Take Part in the Shiny Contest 2019 <small>12 3</small> Shiny Contest   shiny-contest	15	4.5k	Feb 12
Shiny Contest Submission: National Hockey League Play-by-Play App <small>2 4</small> shiny   shiny-contest	7	2.5k	2019-12-18
RStudio - Site Reliability Engineer <small>1 3</small> Jobs & Gigs   job-post	4	305	2019-11-25

# Where else to get help

## Stack Overflow

Check out the questions tagged **r** on Stack Overflow:

<https://stackoverflow.com/questions/tagged/r>



The screenshot shows the Stack Overflow homepage with a sidebar on the left. The sidebar includes links for Home, PUBLIC, Stack Overflow (which is highlighted with a blue border), and Tags. The main content area has a search bar at the top with the query '[r]'. Below the search bar, the title 'Questions tagged [r]' is displayed in large text. To the right of the title is a blue 'Ask Question' button. A descriptive text box below the title provides guidelines for asking R-related questions.

Questions tagged [r]

R is a free, open-source programming language and software environment for statistical computing, bioinformatics, visualization, and general computing. Please provide minimal and reproducible example(s) along with the desired output. Use `dput()` for data and specify all non-base packages with `library()` calls. Do not embed pictures for data or code, use indented code blocks instead. For statistics related questions, use <https://stats.stackexchange.com>.

# #rstats on Twitter

If you have a Twitter account, check out #rstats: <https://twitter.com/hashtag/rstats>



Art by Allison Horst