

# how i found your answer

Mara Averick ([@dataandme](#))

Tidyverse Developer Advocate, RStudio

[bit.ly/noreastr](https://bit.ly/noreastr)

by running your code...

A Venn diagram consisting of three overlapping circles. The largest circle on the left is labeled "People who can help you fix your code". The middle circle is labeled "People who can run that code in their head". The smallest circle on the right is labeled "People who will answer without running the code first". The overlapping areas represent the common characteristics of each group.

**People who can help you fix your code**

**People who can  
run that code in  
their head**

**People who will answer  
without running the code first**

# The art of the question

*The most useless problem statement that one can face is “it doesn’t work”, yet we seem to get it far too often.*

- Thiago Maciera



# The newcomer's paradox...

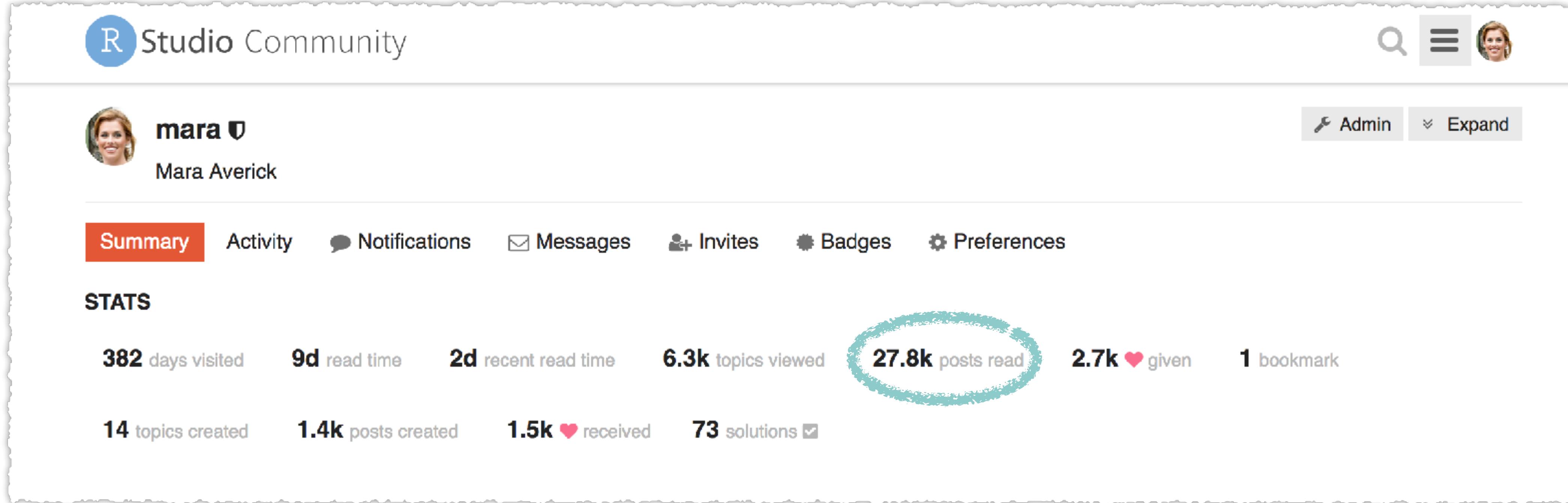


*When you ask for help, some friendly soul will no doubt tell you that "it's easy, just do foo, bar and baz." Except for you, it is not easy, there may be no documentation for foo, bar is not doing what it is supposed to be doing and what is this baz thing anyway with its eight disambiguation entries on Wikipedia?*

— Leslie Hawthorne

"You'll Eventually Know Everything They've Forgotten." In *Open Advice: FOSS: What We Wish We Had Known When We Started*, edited by Lydia Pintscher, 29–32.

# Lessons learned reading ~28k posts...



The screenshot shows the RStudio Community profile summary for user **mara**. The profile picture is a woman with blonde hair. The name **mara** is followed by a verified badge (a blue shield with a white checkmark). Below the name is the full name **Mara Averick**. To the right are buttons for **Admin** and **Expand**. The top navigation bar includes links for **Summary**, **Activity**, **Notifications**, **Messages**, **Invites**, **Badges**, and **Preferences**. The **Summary** tab is active. Below the tabs is a section titled **STATS** with the following metrics:

Metric	Value
days visited	382
read time	9d
recent read time	2d
topics viewed	6.3k
posts read	27.8k
given	2.7k
bookmarks	1
topics created	14
posts created	1.4k
received	1.5k
solutions	73

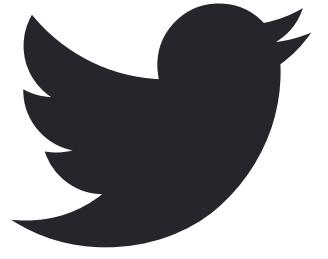
The **posts read** metric is highlighted with a teal oval.

Source: mara profile summary on RStudio Community <https://community.rstudio.com/u/mara/summary>

# Context is key



# Where to ask?



Twitter



RStudio Community

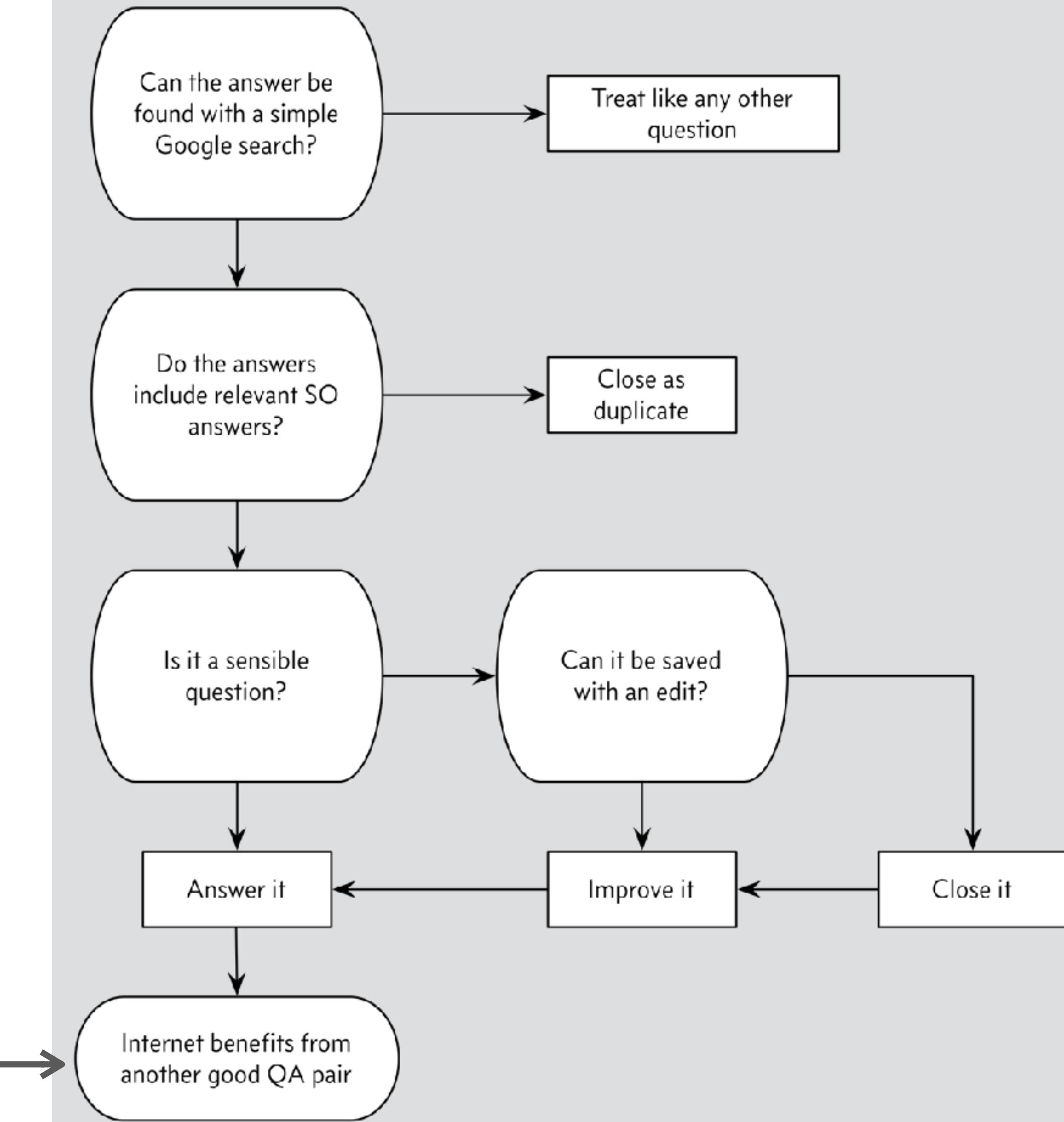


StackOverflow

# How should you respond to RTFM questions?



**goal = good QA pair**





Discourse - Civilized Discussion

https://www.discourse.org

About Features Community Demo Pricing

# Civilized discussion for your community

Try it FREE

**Your Community**

Topic	Replies	Views	Last Post
Net Neutrality Day of Action	4	878	Jul '17
Sharing our experience changing the Discourse UI	6	2.4k	Jul '17
A measure of success	0	763	
Discourse app: just GREAT	0	830	
The community platform of champions	0	890	
Loving Discourse so far	0	807	
Discourse for discussions: superiority to e.g. GitHub	0	1.2k	
Discourse running great on forum with a few hundred users	0	823	
Minimally interesting to anyone today? - [ARCHIVED]	0	1.7k	

Sign Up Log In

**Your Community**

Topic	Replies	Views	Last Post
Discourse is Best Forum Solution in the 2017 CMS Critic Awards	7		Nov '17
Converting, combining, migrating - Yes!	4		Nov '17
Successful Migration	0		Oct '17
Thanks to discourse	6		Oct '17
Post your Discourse T-shirt pics!	45		Oct '17
What an amazon platform...	0		

RStudio Community

https://community.rstudio.com

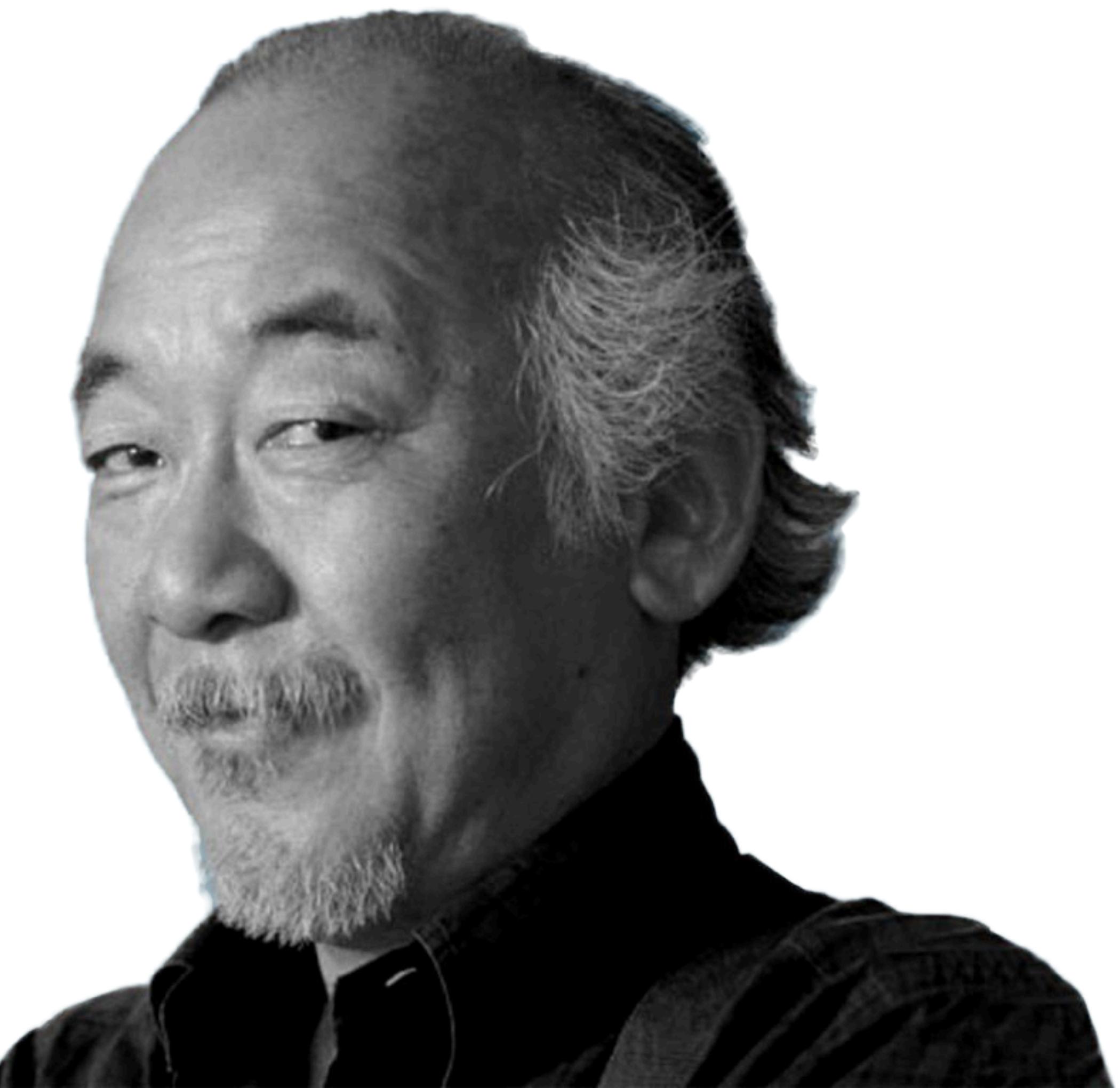
# R Studio Community

Sign Up Log In

Latest Categories Top

Topic	Category	Users	Replies	Views	Activity
<input checked="" type="checkbox"/> Welcome to the RStudio Community! Welcome to community.rstudio.com — we're glad to have you! This welcome page will give you some advice on how to get the most out of the site if you're getting or giving help. We want this to be a friendly, inclusive com... <a href="#">read more</a>	meta		0	2.8k	Jul 22
<input type="checkbox"/> Integrating shiny apps <a href="#">shiny</a>	shiny		1	34	16m
<input type="checkbox"/> Create pdf from Rnw: missing just one font character from eastern European language	R Markdown		4	39	30m
<input type="checkbox"/> UNC pathway when downloading packages <a href="#">ide</a> <a href="#">ide-issue</a>	RStudio IDE		0	6	39m
<input type="checkbox"/> Are packrat bundles really portable across different platforms?	R Admins		12	592	1h
<input type="checkbox"/> /usr/bin/env no such file or directory <a href="#">terminal</a> <a href="#">rstudioserver</a>	R Admins		1	10	1h
<input type="checkbox"/> Deploying APP on shinyapps.io which keeps asking for ggplot2 <a href="#">ggplot2</a> <a href="#">shinyappsi0</a> <a href="#">package-installation</a>	shiny		13	92	1h
<input type="checkbox"/> when I installed R Studio, the packages were visible, but not seeing packages not seeing anymore, how to get back packages in rstudio? <a href="#">ide</a> <a href="#">ide-issue</a>	RStudio IDE		0	5	2h
<input type="checkbox"/> How to read Netcdf files	General		8	58	4h

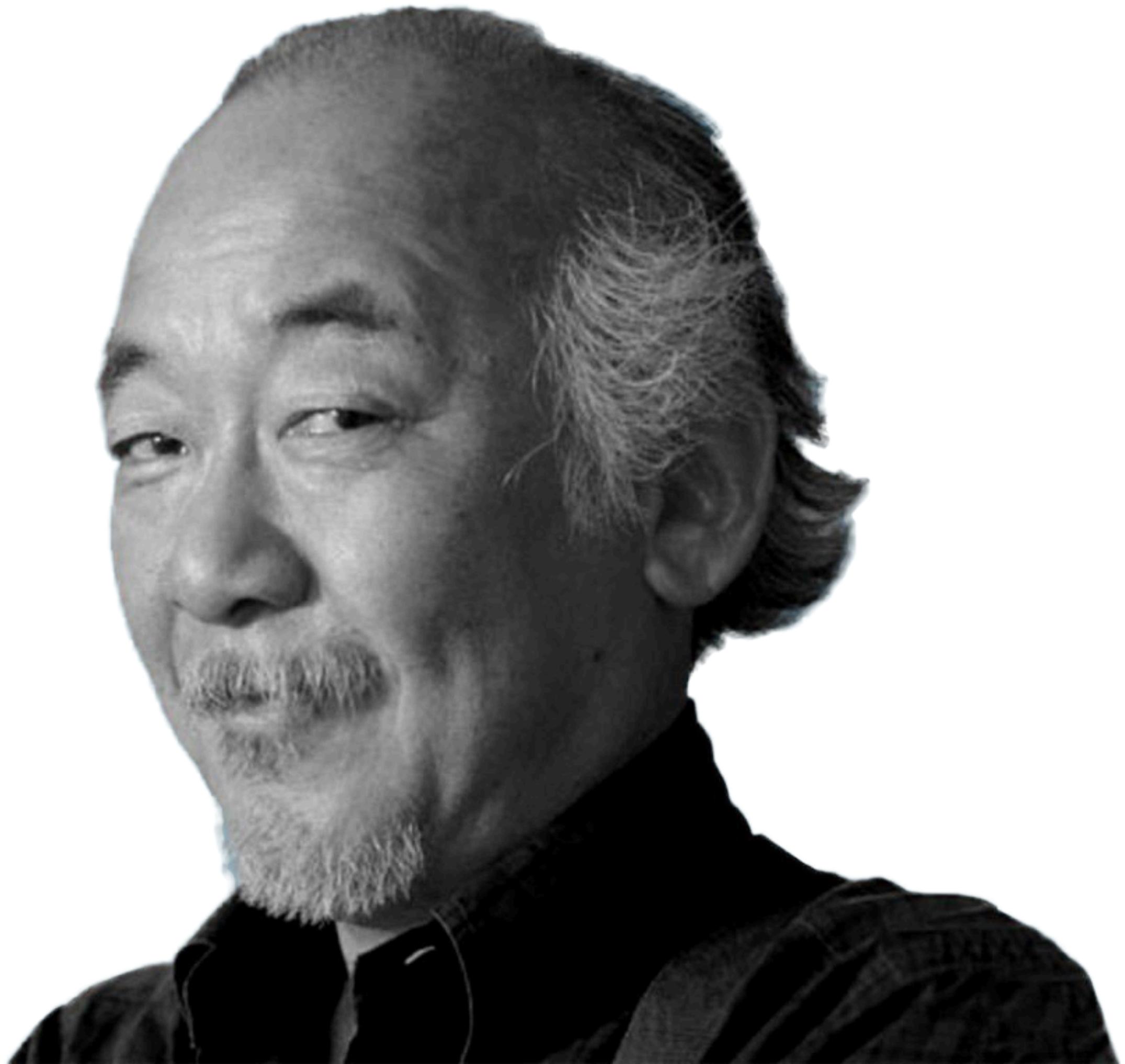
# What's in it for me?



The Karate Kid (1984)

# What's in it for me?

\* mad hot troubleshooting skillz \*



# 10 simple rules for getting help from online scientific communities

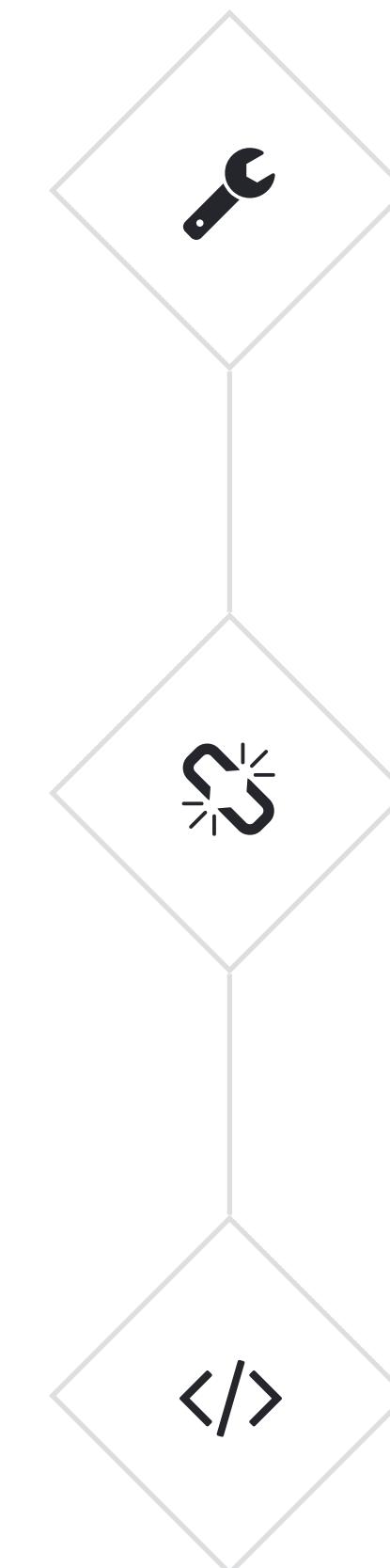
1. Do not be afraid to ask a question
2. State the question clearly
3. New to a mailing list? Learn the established customs before posting
4. Do not ask what has already been answered
5. Always use a good title
6. Do your homework before posting
7. Proofread your post and write in correct English
8. Be courteous to other forum members
9. Remember that the archive of your discussion can be useful to other people
10. Give back to the community

*Dall’Olio, Giovanni M., Jacopo Marino, Michael Schubert, Kevin L. Keys, Melanie I. Stefan, Colin S. Gillespie, Pierre Poulain, et al. 2011. “Ten Simple Rules for Getting Help from Online Scientific Communities.” PLoS Computational Biology 7 (9): 10–12. doi:10.1371/journal.pcbi.1002202.*



ЭМПАТИЯ ДЛЯ ПОЛЬЗУЮЩИХСЯ

# the anatomy of an issue



**PROBLEM DESCRIPTION**

**EXPECTED BEHAVIOUR**

**MINIMAL REPRODUCIBLE EXAMPLE**

# the anatomy of an issue

"Creating an issue template for your repository" - GitHub Help <<https://help.github.com/articles/creating-an-issue-template-for-your-repository/>>

# the anatomy of an issue ~~figuring~~ sh\*t out



PROBLEM DESCRIPTION

EXPECTED BEHAVIOUR

MINIMAL REPRODUCIBLE EXAMPLE

# the anatomy of an issue ~~figuring~~ sh\*t out



# the anatomy of an issue ~~figuring~~ sh\*t out



ANTECEDENTS

BEHAVIOURS

CONSEQUENCES

# the anatomy of an issue ~~figuring~~ sh\*t out

ABC Data Sheet

Record each instance of one behavior, as well as the antecedent (what happened right before the behavior), the consequence (what happened right after the behavior), and what the possible function of that behavior was (what outcome did it achieve for the child/student?).

Date: 4/25 - 4/27 Time of Observation: 10:00 - 10:15

Antecedent	Behavior	Consequence	Possible Function (Attention, Access to items/activities, Escape, Sensory)
Circle Time- Students sing "Days of the Week"	Kicks other student	Time Out	Escape
Circle time- Singing "Days of the Week"	<del>Kicks</del> 2 students	Time Out	Escape

## Error Information: Packages missing

Description of issue - [REDACTED]

Steps taken so far - i tried and tried..

### System Information:

- RStudio Edition: (Desktop or Server)
- RStudio Version:
- OS Version:
- R Version:

### Also:

- RStudio diagnostics report:
- Your sessionInfo():
- RStudio crash report:
- RStudio application log files:

---

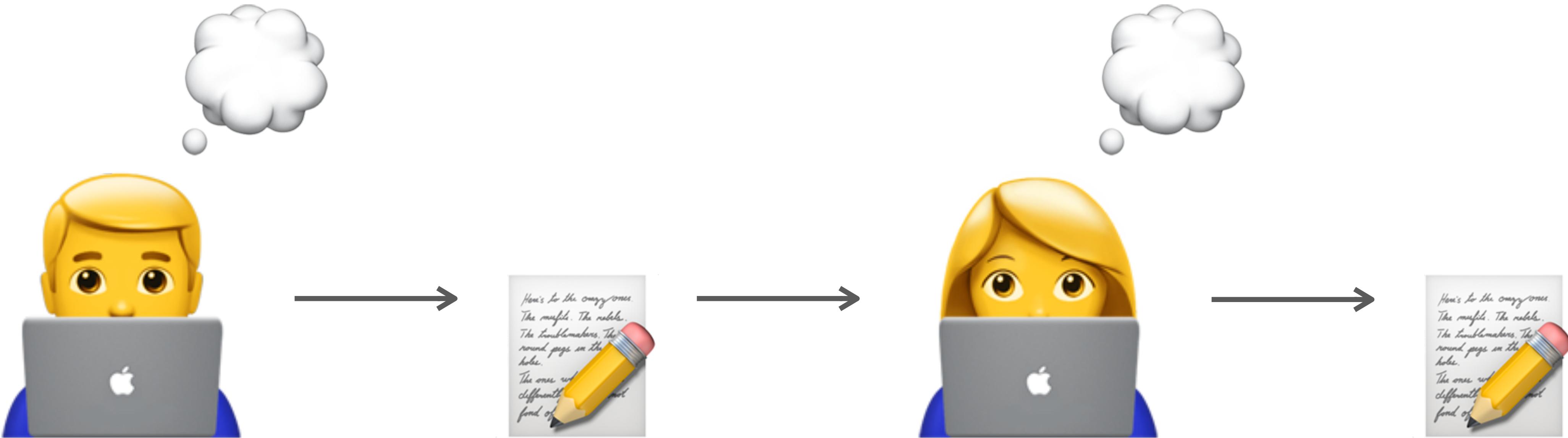
From [Troubleshooting Guide: Using RStudio](#)

They're not *wrong*

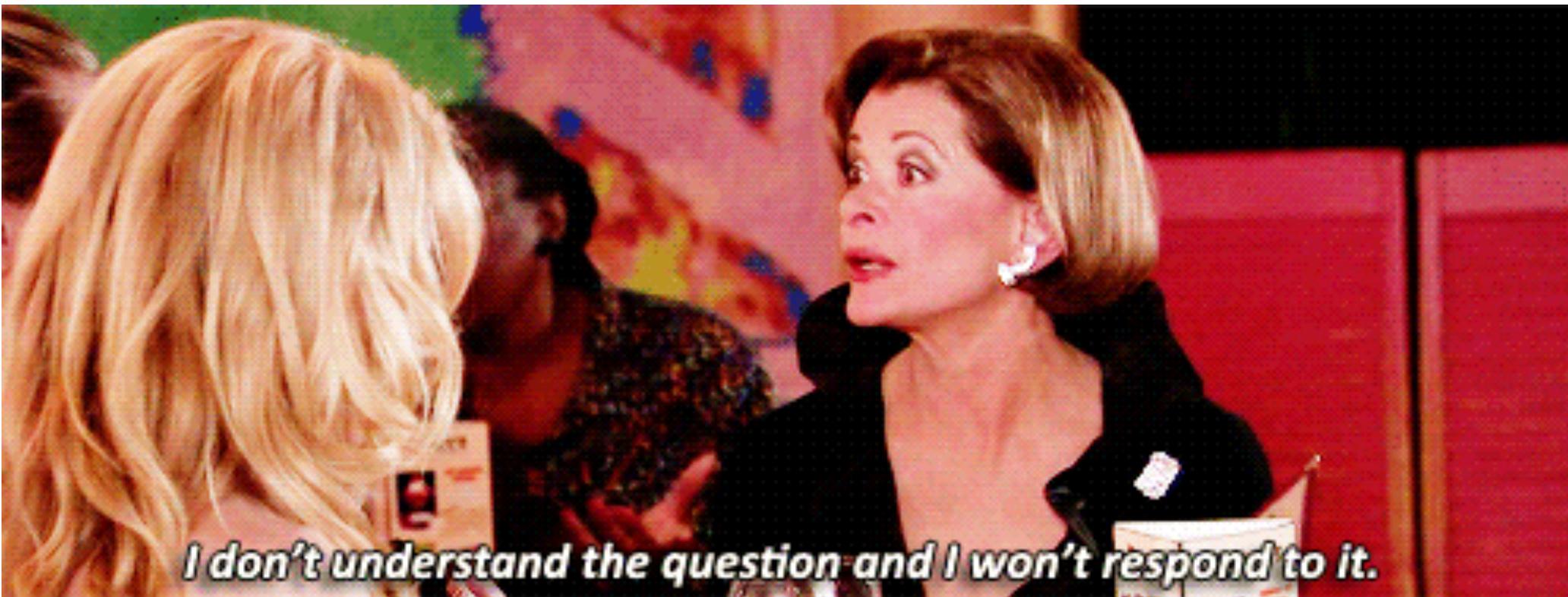
**“It is impossible to speak in such a way that you cannot be misunderstood.”**

– Karl Popper

# Writing prose about code...

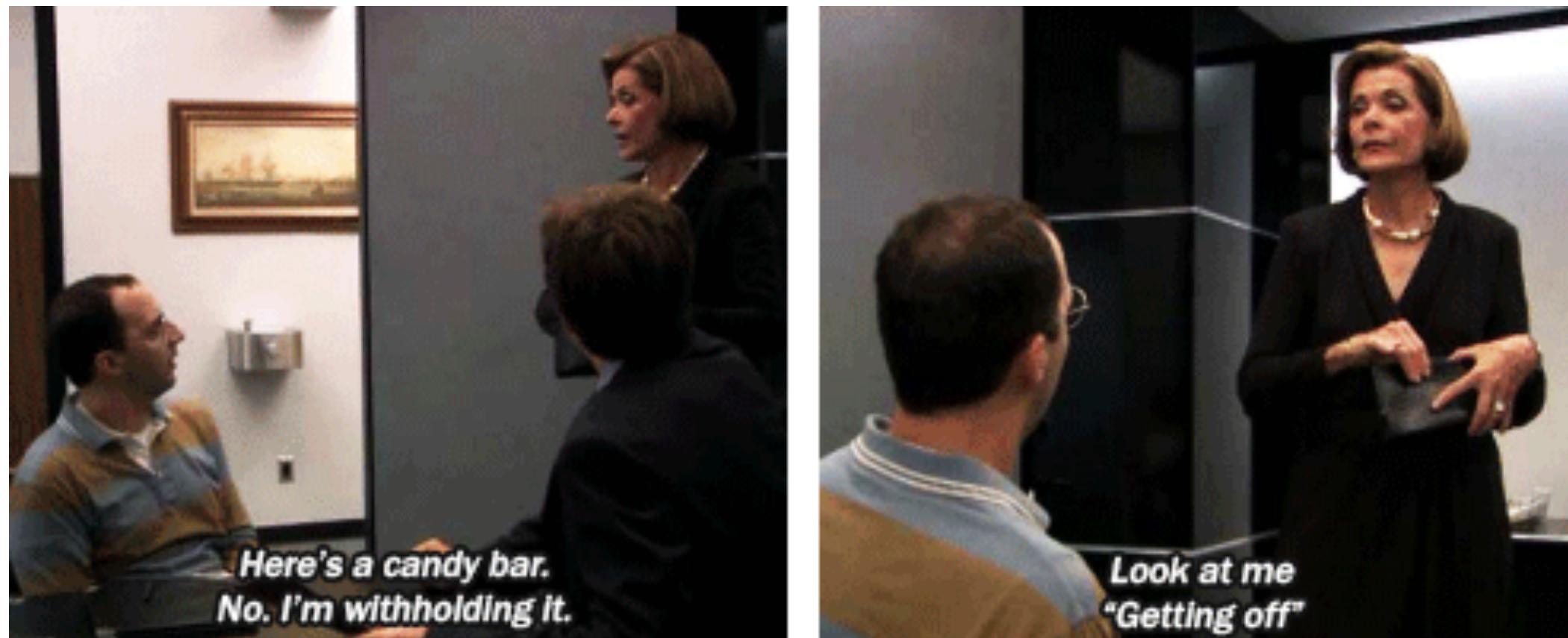


# Writing prose about code...

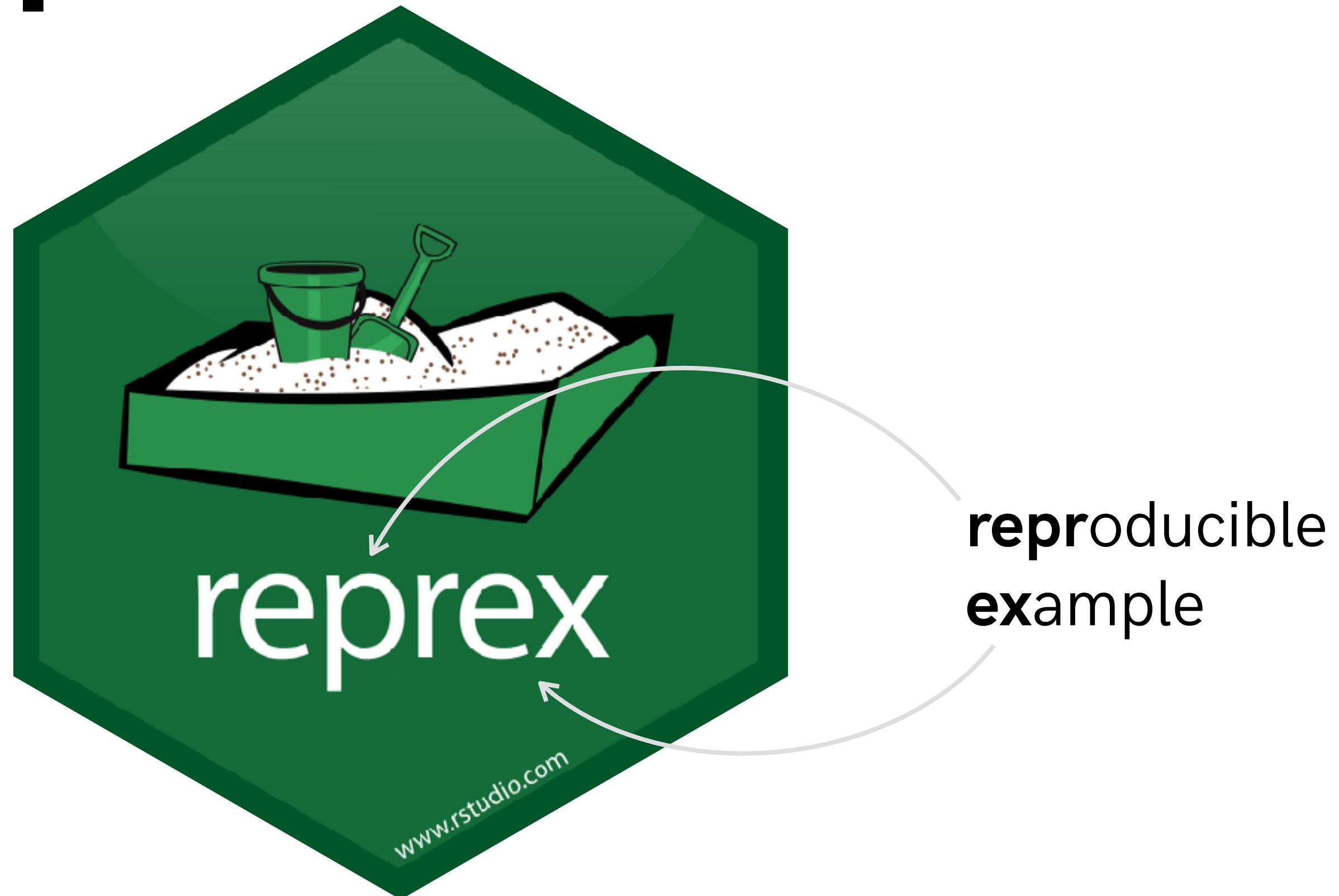


*I don't understand the question and I won't respond to it.*

# Writing prose about code...



# the magic of reprex



# reprex raison d'être



# The reprex request trifecta



## WHAT I'M ASKING YOU TO DO

Make a reproducible example

## WHY I'M ASKING YOU TO DO IT

Help me help you — I need your data to do so

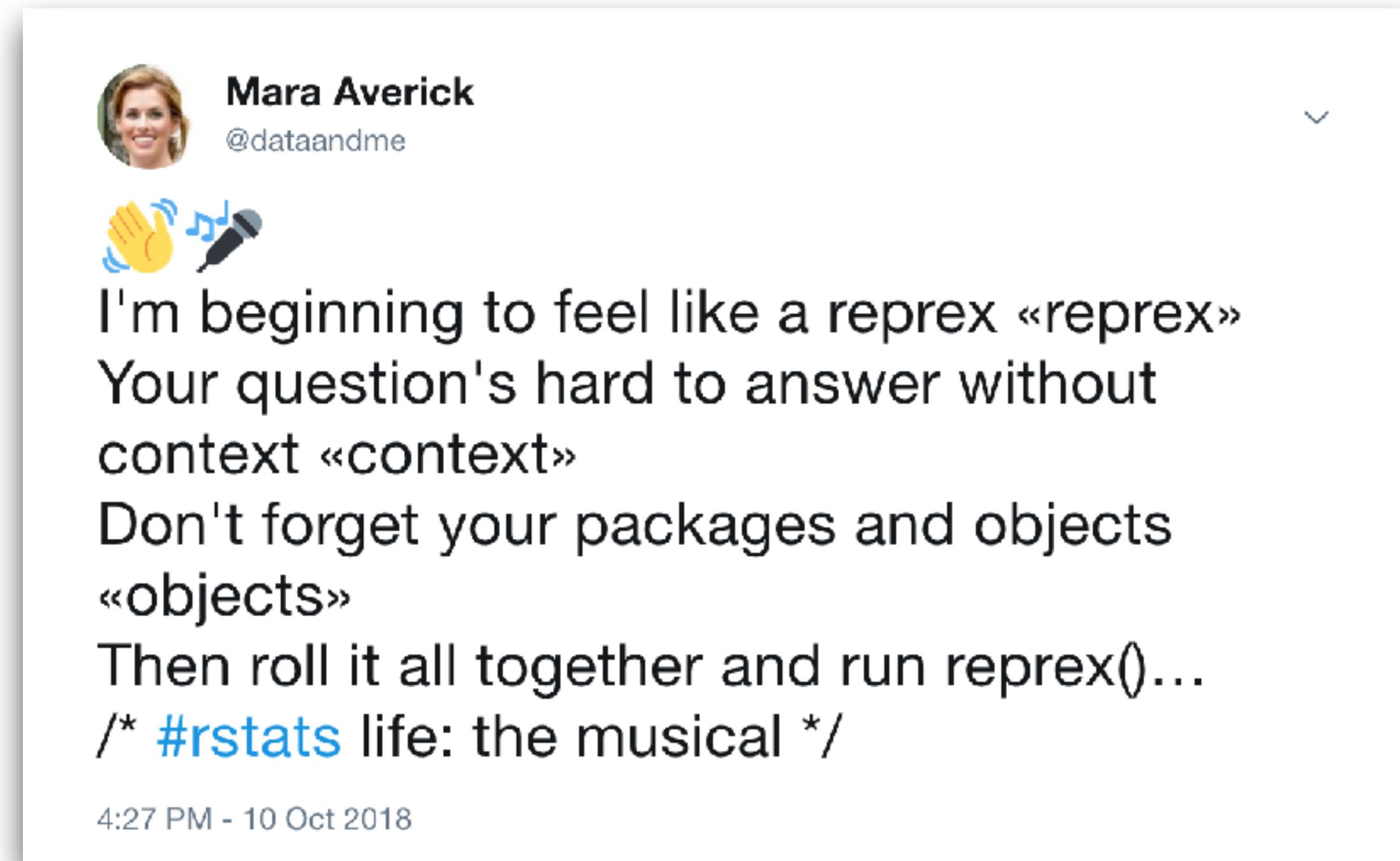
## HOW YOU CAN DO THE THING

Resources, videos, we've got it all...

# Keys to reprex-cellence

- ✓ Code that **actually runs**
- ✓ Code that **doesn't have to be run**
- ✓ Code that **can be easily run**

# Keys to reprex-cellence



Mara Averick  
@dataandme

I'm beginning to feel like a reprex «reprex»  
Your question's hard to answer without  
context «context»  
Don't forget your packages and objects  
«objects»  
Then roll it all together and run reprex()...  
/\* #rstats life: the musical \*/

4:27 PM - 10 Oct 2018

Source: Jenny Bryan, 2017. "reprex: the package, the point." <https://speakerdeck.com/jennybc/reprex-help-me-help-you>

The screenshot shows an RStudio interface with the following components:

- Left Panel (Code Editor):** An R script titled "2017-01-03-reprex-magic.Rmd" containing the following code:

```
1 library(visdat)
2 
3 vis_miss(airquality)
4 
5 library(ggplot2)
6 
7 ggplot(airquality,
8       aes(x = Ozone,
9            y = Solar.R)) +
10      geom_point()
11 
12 library(naniar)
13 
14 ggplot(airquality,
15       aes(x = Ozone,
16            y = Solar.R)) +
17      geom_missing_point()
```
- Top Bar:** Shows tabs for "2017-01-03-reprex-magic.Rmd", "Untitled1\*", and "Untitled2\*".
- Toolbar:** Includes "Run", "Source", and other standard RStudio icons.
- Console:** Displays the output of running the code:

```
> reprex::reprex()
Rendered reprex ready on the clipboard.
> reprex::reprex()
Rendered reprex ready on the clipboard.

Restarting R session...

> reprex::reprex()
Rendered reprex ready on the clipboard.
>
```
- Viewer:** A panel showing the rendered output of the code, which consists of three error messages:

```
vis_miss(airquality)
#> Error in eval(expr, envir, enclos): could not find function "vis_miss"

ggplot(airquality,
       aes(x = Ozone,
            y = Solar.R)) +
      geom_point()
#> Error in eval(expr, envir, enclos): could not find function "ggplot"

ggplot(airquality,
       aes(x = Ozone,
            y = Solar.R)) +
      geom_missing_point()
#> Error in eval(expr, envir, enclos): could not find function "ggplot"
```

Source: Nick Tierney. "Magic reprex." 2017-01-11 <<http://www.njtierney.com/post/2017/01/11/magic-reprex/>>

The screenshot shows the RStudio interface. On the left, the 'Script' tab of the 'Untitled1' file is open, displaying the following R code:

```
1 library(ggplot2)
2
3 df <- data.frame(x = 1)
4
5 ggplot(df, aes(x, x)) + geom_point() +
6   ggtitle("gjpjQ") +
7   theme(plot.title = element_text(size = 10))
8
9 ggplot(df, aes(x, x)) + geom_point() +
10  ggtitle("gjpjQ") +
11  theme(plot.title = element_text(size = 100))
```

The 'Console' tab is selected in the top right, showing the command prompt: > |

The bottom navigation bar shows tabs for Environment, History, and Connections.

Source: Nick Tierney. "Magic reprex." 2017-01-11 <<http://www.njtierney.com/post/2017/01/11/magic-reprex/>>

# But wait, there's more...

The screenshot shows a web browser window displaying a post on the RStudio community forum. The title of the post is "ggplot2: Can I send arguments in a formatter function?". The post was made by a user named mara, who is identified as a Sustainer. The post has 1 like and 1d ago. The post content includes R code demonstrating how to use the `scale_y_continuous` function with the `percent` argument. The code creates a dataset with 5 categories and plots them as bars. The y-axis is labeled "year" and ranges from 1.00% to 7.00%.

ggplot2: Can I send arguments in a formatter function?

mara · Sustainer · 1 · 1d · 2018-10-24 · 2 / 2 · 2018-10-24 · 1d ago

```
library(ggplot2)
library(scales)

set.seed(12)

#Create a dataset with 3-digits precision:
df <- dplyr::data_frame(xvar = LETTERS[1:5],yvar = round(runif(5, min = 0, max = 0.080),
df$xvar <- as.factor(df$xvar)

# The diagram shows percent with 2 decimals, but it should only be 1
ggplot(df, aes(xvar, yvar)) +
  geom_bar(stat = "identity") +
  scale_y_continuous(breaks = pretty_breaks(8), labels = percent)
```

A bar chart with the x-axis labeled "xvar" and the y-axis labeled "year". The y-axis has ticks at 1.00%, 2.00%, 3.00%, 4.00%, 5.00%, 6.00%, and 7.00%. There are five bars corresponding to the categories A, B, C, D, and E. Category A has a value of approximately 6.5%. Category B has a value of approximately 7.5%. Category C has a value of approximately 2.2%. Category D has a value of approximately 1.2%. Category E has a value of approximately 1.5%.

xvar	year
A	~6.5%
B	~7.5%
C	~2.2%
D	~1.2%
E	~1.5%

<https://community.rstudio.com/t/ggplot2-can-i-send-arguments-in-a-formatter-function/16939>

# Meta-help



## Why does head() show 6 rows by default?



hadley RStudio Employee

Nov '17

To answer a question like this, I first start looking through the S books I have on hand (e.g. [The New S Language](#) 11). I don't see `head()` mentioned in the index, so that suggests it's a function introduced by R.

Look in S books

Since it's an R function, I can next search [@winston](#)'s GitHub mirror of the R sources:  
<https://github.com/wch/r-source> 16, finding the source at <https://github.com/wch/r-source/blob/af7f52f70101960861e5d995d3a4bec010bc89e6/src/library/utils/R/head.R> 20

Go to R-source mirror

This includes a comment which suggests we should ask Patrick Burns:

```
### placed in the public domain 2002
### Patrick Burns patrick@burns-stat.com
###
### Adapted for negative arguments by Vincent Goulet
### <vincent.goulet@act.ulaval.ca>, 2006
```

Find relevant bit

But it's worth checking just to make sure it's always used 6. I click on history, and then find the first version:  
<https://github.com/wch/r-source/commit/37271cdbcd7e5d82c79bdb536ef305d93b644ad#diff-941bf47bf09f67538338535bd512d521> 28 - so it has been six from the very beginning.

See the history

So next step, I'll email Patrick and see if he recollects...

Send an email

# Reading error messages

```
> install.packages('openssl')
* installing *source* package ‘openssl’ ...
** package ‘openssl’ successfully unpacked and MD5 sums checked
Using PKG_CFLAGS=
----- ANTICONF ERROR -----
Configuration failed because openssl was not found. Try installing:
* deb: libssl-dev (Debian, Ubuntu, etc)
* rpm: openssl-devel (Fedora, CentOS, RHEL)
* csw: libssl_dev (Solaris)
* brew: openssl@1.1 (Mac OSX)
If openssl is already installed, check that 'pkg-config' is in your
PATH and PKG_CONFIG_PATH contains a openssl.pc file. If pkg-config
is unavailable you can set INCLUDE_DIR and LIB_DIR manually via:
R CMD INSTALL --configure-vars='INCLUDE_DIR=... LIB_DIR=...'
-----
ERROR: configuration failed for package ‘openssl’
* removing ‘/R/R-3.5.0_SL/lib64/R/library/openssl’
```

```
The downloaded source packages are in
  '/tmp/RtmpCM5CD4/downloaded_packages'
Warning message:
In install.packages("openssl") :
  installation of package ‘openssl’ had non-zero exit status
```

Super Useful Information

# Reading error messages

```
> install.packages('openssl')
* installing *source* package ‘openssl’ ...
** package ‘openssl’ successfully unpacked and MD5 sums checked
Using PKG_CFLAGS=
----- ANTICONF ERROR -----
Configuration failed because openssl was not found. Try installing:
* deb: libssl-dev (Debian, Ubuntu, etc)
* rpm: openssl-devel (Fedora, CentOS, RHEL)
* csw: libssl_dev (Solaris)
* brew: openssl@1.1 (Mac OSX)
If openssl is already installed, check that 'pkg-config' is in your
PATH and PKG_CONFIG_PATH contains a openssl.pc file. If pkg-config
is unavailable you can set INCLUDE_DIR and LIB_DIR manually via:
R CMD INSTALL --configure-vars='INCLUDE_DIR=... LIB_DIR=...'
----- 
ERROR: configuration failed for package ‘openssl’
* removing ‘/R/R-3.5.0_SL/lib64/R/library/openssl’
```

The downloaded **source** packages are in  
‘/tmp/RtmpCM5CD4/downloaded\_packages’  
Warning message:  
In `install.packages("openssl")` :  
  installation of package ‘openssl’ had non-zero exit status

Super Useful Information

Per the error message...



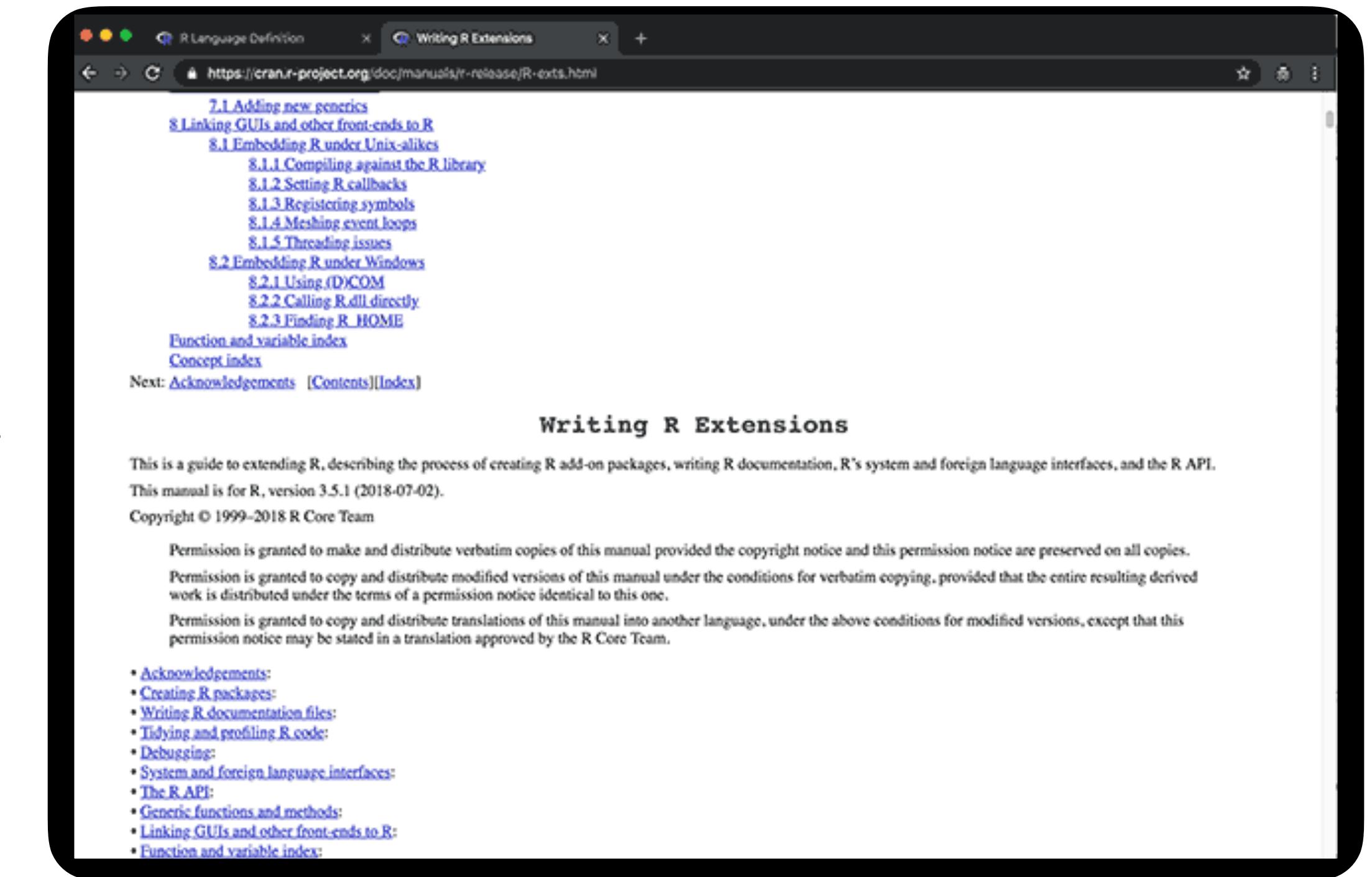
This may literally be the first time they've seen a useful error message

# RTFM



# RTFM

TFM



# The R manuals as bookdown

R manuals

**R Manuals as bookdown**

You'll find here a list of the R manuals converted to bookdown.

Notes: I do not own any of the content written in these manuals — I just turned the epub into a bookdown format!

<https://cran.r-project.org/manuals.html>

- [Intro to R](#)
- [R Data Import/Export](#)
- [R Installation and Administration](#)
- [Writing R extensions](#)
- [The R language definition](#)
- [R Internals](#)

Colin Fay <https://colinfay.me/r-manuals/>

# The R manuals as bookdown

The screenshot shows a web browser window with the title bar "Writing R extensions". The address bar indicates the page is "Not Secure" and points to [colinfay.me/writing-r-extensions/](https://colinfay.me/writing-r-extensions/). The left sidebar contains a table of contents for the manual, listing chapters such as "Creating R packages", "Writing R documentation files", and "The R API: entry points for C code". The main content area displays the first chapter, "Writing R Extensions", which includes the title, author ("R Core Team"), date ("2017-10-18"), and a brief description of the manual's purpose. It also includes copyright information, permission notices, and links to other parts of the manual.

**Writing R extensions**

**R Core Team**

**2017-10-18**

**Writing R Extensions**

This is a guide to extending R, describing the process of creating R add-on packages, writing R documentation, R's system and foreign language interfaces, and the R API.

This manual is for R, version 3.4.2 (2017-09-28).

Copyright © 1999–2016 R Core Team

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions, except that this permission notice may be stated in a translation approved by the R Core Team.

All the R manuals as bookdown :

# **Guide them through the docs**



## Tidyr::separate() at second/last occurrence of character



As you guessed, a regular expression is your best bet here. From the `tidyr separate()` function reference 8

Given either regular expression or a vector of character positions, `separate()` turns a single character column into multiple columns.

workflow all in one place:

<https://www.garrickadenbuie.com/project/regexplain/> 30

Here's how I would do it (which is hacky, but it works); you want to split at the underscore that precedes the day, which is numerical. So, I would use a string `look around` 18 to say, in effect, "*split at the underscore that is followed by a digit.*"

```
suppressPackageStartupMessages(library(tidyverse))
example_data <- data.frame(subject_ID = 1:5,
                           daily_measure_1 = 6:10,
                           daily_measure_2 = 11:15,
                           daily_measure_3 = 16:20,
                           daily_measure_4 = 21:25,
                           daily_measure_5 = 26:30)
example_data %>%
  gather(key = "full_name", value = "value", starts_with("daily_")) %>%
  separate(full_name, c("variable", "day"), sep = "_(?=[:digit:])")
#>   subject_ID      variable day value
#> 1           1 daily_measure  1     6
#> 2           2 daily_measure  1     7
#> 3           3 daily_measure  1     8
#> 4           4 daily_measure  1     9
#> 5           5 daily_measure  1    10
```



## Tidyr::separate() at second/last occurrence of character

tidyverse

tidy

regex



mara Sustainer

May 20

As you guessed, a regular expression is your best bet here. From the [tidyr separate\(\) function reference](#)

[RegExr.com](#) has been my go-to for regular expression testing for a while, but Garrick Aden-Buie recently made an RStudio add-in inspired by the very same, [RegExplain](#), which is a great option for keeping your workflow all in one place:

<https://www.garrickadenbuie.com/project/regexplain/>

Here's how I would do it (which is hacky, but it works); you want to split at the underscore that precedes the day, which is numerical. So, I would use a stringr [look around](#) to say, in effect, "*split at the underscore that is followed by a digit.*"

```
daily_measure_1 = 10:20,
daily_measure_4 = 21:25,
daily_measure_5 = 26:30)

example_data %>%
  gather(key = "full_name", value = "value", starts_with("daily_")) %>%
  separate(full_name, c("variable", "day"), sep = "_(?:[0-9])")
```

subject_ID	variable	day	value	
#> 1	1	daily_measure	1	6
#> 2	2	daily_measure	1	7
#> 3	3	daily_measure	1	8
#> 4	4	daily_measure	1	9
#> 5	5	daily_measure	1	10

# "We don't do that here"

- Question phrasing
- Formatting posts
- Community triage
- Question framing
- Community culture of asking

## "We Don't Do That Here": How Collaborative Editing with Mentors Improves Engagement in Social Q&A Communities

Denae Ford<sup>1</sup>, Kristina Lustig<sup>2</sup>, Jeremy Banks<sup>2</sup>, Chris Parnin<sup>1</sup>

<sup>1</sup>North Carolina State University, Raleigh, NC, USA

<sup>2</sup>Stack Exchange, Inc., New York, NY, USA

{dford3, cjparnin}@ncsu.edu, klustig@stackoverflow.com, \_@jeremy.ca

### ABSTRACT

Online question-and-answer (Q&A) communities like Stack Overflow have norms that are not obvious to novice users. Novices create and post programming questions without feedback, and the community enforces site norms through public downvoting and commenting. This can leave novices discouraged from further participation. We deployed a month-long, just-in-time mentorship program to Stack Overflow in which we redirected novices in the process of asking a question to an on-site *Help Room*. There, novices received feedback on their question drafts from experienced Stack Overflow mentors. We present examples and discussion of various question improvements including: question context, code formatting, and wording that adheres to on-site cultural norms. We find that mentored questions are substantially improved over non-mentored questions, with average scores increasing by 50%. We provide design implications that challenge how socio-technical communities onboard novices across domains.

### ACM Classification Keywords

H.5.3. Group and Organization Interfaces: Computer-supported cooperative work

### Author Keywords

social Q&A, collaborative editing, e-mentoring, programming

### INTRODUCTION

Building and maintaining active online communities is a difficult and well-documented problem across many community types [4, 15, 23]. For prospective community members, barriers such as learning community norms [24], overcoming technical hurdles [31], and resolving conflict [12] can be harmful to participation. In addition, these barriers may significantly affect people in marginalized groups, such as women and people of color, from fully participating in online communities [14]. This is especially pertinent for online programming communities.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada

© 2018 ACM. ISBN 978-1-4503-5620-5/18/04...\$15.00

DOI: <https://doi.org/10.1145/3173574.3174182>

Stack Overflow is the largest online programming community [25]. Each month, over 40 million people visit Stack Overflow, a social Q&A site, to learn about, ask, or answer over 14 million programming questions. Despite great popularity, there is evidence that negative behaviors and malfunctioning community mechanics can have long-term effects on site participation. For example, many questions go unanswered [30], and 90% of accepted answers provided by new users are self-answers. For Stack Overflow, "hostile" criticism and conflict [14, 13] is especially problematic for prospective members. As a result, a user may decide not to ask or answer a question for *fear of negative feedback* [14]. These problems can dissuade novices [31] and women [32] from participating in the community. On the other hand, active community members are interested in preserving community norms: not allowing duplicate questions, off-topic or non-closed questions, or poor quality answers. Community members need a mechanism for helping new users ask better questions, while reducing the hostility and negativity of otherwise well-meaning feedback.

In this paper, we applied theory related to learning and communities of practice to a social Q&A site, by using methods related to mutual engagement and formative feedback to improve novices' questions. Building on design claims for increasing engagement in online communities [19], we created *Help Rooms* with collaborative question drafts to enable novices to receive timely and formative feedback from mentors before posting their questions. Our *Help Rooms* work as follows: when a novice is about to post a question, they are asked if they want additional feedback from a mentor. If the novice responds positively, they are redirected to a room with a mentor who can help them edit their question. The mentor offers advice on how to phrase and ask their question so that it can be well received by the Stack Overflow community.

In a one-month online study, we implemented our mechanism for mentored question asking on Stack Overflow, and enabled 271 novices to receive help with their questions. As a result, we found that mentored questions were substantially improved over non-mentored questions. Average scores increased by 50%, resulting in fewer off-topic, deleted, and poor questions. Overall, for mentored questions, there was an increase in the amount of *good* questions asked, and reduction of *bad* questions asked by Stack Overflow standards. Novices surveyed agreed that they feel more comfortable posting on Stack Overflow after their participation (median = 4 on a 5-point Likert

# "We don't do that here"

- Question phrasing
- title = your chance to reel 'em in
- Formatting posts
- Community triage
- Question framing
- Community culture of asking

## "We Don't Do That Here": How Collaborative Editing with Mentors Improves Engagement in Social Q&A Communities

Denae Ford<sup>1</sup>, Kristina Lustig<sup>2</sup>, Jeremy Banks<sup>2</sup>, Chris Parnin<sup>1</sup>

<sup>1</sup>North Carolina State University, Raleigh, NC, USA

<sup>2</sup>Stack Exchange, Inc., New York, NY, USA

{dford3, cjparnin}@ncsu.edu, klustig@stackoverflow.com, \_@jeremy.ca

### ABSTRACT

Online question-and-answer (Q&A) communities like Stack Overflow have norms that are not obvious to novice users. Novices create and post programming questions without feedback, and the community enforces site norms through public downvoting and commenting. This can leave novices discouraged from further participation. We deployed a month-long, just-in-time mentorship program to Stack Overflow in which we redirected novices in the process of asking a question to an on-site *Help Room*. There, novices received feedback on their question drafts from experienced Stack Overflow mentors. We present examples and discussion of various question improvements including: question context, code formatting, and wording that adheres to on-site cultural norms. We find that mentored questions are substantially improved over non-mentored questions, with average scores increasing by 50%. We provide design implications that challenge how socio-technical communities onboard novices across domains.

### ACM Classification Keywords

H.5.3. Group and Organization Interfaces: Computer-supported cooperative work

### Author Keywords

social Q&A, collaborative editing, e-mentoring, programming

### INTRODUCTION

Building and maintaining active online communities is a difficult and well-documented problem across many community types[4, 15, 23]. For prospective community members, barriers such as learning community norms [24], overcoming technical hurdles [31], and resolving conflict [12] can be harmful to participation. In addition, these barriers may significantly affect people in marginalized groups, such as women and people of color, from fully participating in online communities [14]. This is especially pertinent for online programming communities.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada

© 2018 ACM. ISBN 978-1-4503-5620-5/18/04...\$15.00

DOI: <https://doi.org/10.1145/3173574.3174182>

Stack Overflow is the largest online programming community [25]. Each month, over 40 million people visit Stack Overflow, a social Q&A site, to learn about, ask, or answer over 14 million programming questions. Despite great popularity, there is evidence that negative behaviors and malfunctioning community mechanics can have long-term effects on site participation. For example, many questions go unanswered [30], and 90% of accepted answers provided by new users are self-answers. For Stack Overflow, "hostile" criticism and conflict [14, 13] is especially problematic for prospective members. As a result, a user may decide not to ask or answer a question for *fear of negative feedback* [14]. These problems can dissuade novices [31] and women [32] from participating in the community. On the other hand, active community members are interested in preserving community norms: not allowing duplicate questions, off-topic or non-closed questions, or poor quality answers. Community members need a mechanism for helping new users ask better questions, while reducing the hostility and negativity of otherwise well-meaning feedback.

In this paper, we applied theory related to learning and communities of practice to a social Q&A site, by using methods related to mutual engagement and formative feedback to improve novices' questions. Building on design claims for increasing engagement in online communities [19], we created *Help Rooms* with collaborative question drafts to enable novices to receive timely and formative feedback from mentors before posting their questions. Our *Help Rooms* work as follows: when a novice is about to post a question, they are asked if they want additional feedback from a mentor. If the novice responds positively, they are redirected to a room with a mentor who can help them edit their question. The mentor offers advice on how to phrase and ask their question so that it can be well received by the Stack Overflow community.

In a one-month online study, we implemented our mechanism for mentored question asking on Stack Overflow, and enabled 271 novices to receive help with their questions. As a result, we found that mentored questions were substantially improved over non-mentored questions. Average scores increased by 50%, resulting in fewer off-topic, deleted, and poor questions. Overall, for mentored questions, there was an increase in the amount of *good* questions asked, and reduction of *bad* questions asked by Stack Overflow standards. Novices surveyed agreed that they feel more comfortable posting on Stack Overflow after their participation (median = 4 on a 5 point Likert

# "We don't do that here"

- Question phrasing
- Formatting posts
- Community triage
- Question framing
- Community culture of asking

format code as code

## "We Don't Do That Here": How Collaborative Editing with Mentors Improves Engagement in Social Q&A Communities

Denae Ford<sup>1</sup>, Kristina Lustig<sup>2</sup>, Jeremy Banks<sup>2</sup>, Chris Parnin<sup>1</sup>

<sup>1</sup>North Carolina State University, Raleigh, NC, USA

<sup>2</sup>Stack Exchange, Inc., New York, NY, USA

{dford3, cjparnin}@ncsu.edu, klustig@stackoverflow.com, \_@jeremy.ca

### ABSTRACT

Online question-and-answer (Q&A) communities like Stack Overflow have norms that are not obvious to novice users. Novices create and post programming questions without feedback, and the community enforces site norms through public downvoting and commenting. This can leave novices discouraged from further participation. We deployed a month-long, just-in-time mentorship program to Stack Overflow in which we redirected novices in the process of asking a question to an on-site *Help Room*. There, novices received feedback on their question drafts from experienced Stack Overflow mentors. We present examples and discussion of various question improvements including: question context, code formatting, and wording that adheres to on-site cultural norms. We find that mentored questions are substantially improved over non-mentored questions, with average scores increasing by 50%. We provide design implications that challenge how socio-technical communities onboard novices across domains.

### ACM Classification Keywords

H.5.3. Group and Organization Interfaces: Computer-supported cooperative work

### Author Keywords

social Q&A, collaborative editing, e-mentoring, programming

### INTRODUCTION

Building and maintaining active online communities is a difficult and well-documented problem across many community types[4, 15, 23]. For prospective community members, barriers such as learning community norms [24], overcoming technical hurdles [31], and resolving conflict [12] can be harmful to participation. In addition, these barriers may significantly affect people in marginalized groups, such as women and people of color, from fully participating in online communities [14]. This is especially pertinent for online programming communities.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada

© 2018 ACM. ISBN 978-1-4503-5610-5/18/04...\$15.00

DOI: <https://doi.org/10.1145/3173574.3174182>

Stack Overflow is the largest online programming community [25]. Each month, over 40 million people visit Stack Overflow, a social Q&A site, to learn about, ask, or answer over 14 million programming questions. Despite great popularity, there is evidence that negative behaviors and malfunctioning community mechanics can have long-term effects on site participation. For example, many questions go unanswered [30], and 90% of accepted answers provided by new users are self-answers. For Stack Overflow, "hostile" criticism and conflict [14, 13] is especially problematic for prospective members. As a result, a user may decide not to ask or answer a question for *fear of negative feedback* [14]. These problems can dissuade novices [31] and women [32] from participating in the community. On the other hand, active community members are interested in preserving community norms: not allowing duplicate questions, off-topic or non-closed questions, or poor quality answers. Community members need a mechanism for helping new users ask better questions, while reducing the hostility and negativity of otherwise well-meaning feedback.

In this paper, we applied theory related to learning and communities of practice to a social Q&A site, by using methods related to mutual engagement and formative feedback to improve novices' questions. Building on design claims for increasing engagement in online communities [19], we created *Help Rooms* with collaborative question drafts to enable novices to receive timely and formative feedback from mentors before posting their questions. Our *Help Rooms* work as follows: when a novice is about to post a question, they are asked if they want additional feedback from a mentor. If the novice responds positively, they are redirected to a room with a mentor who can help them edit their question. The mentor offers advice on how to phrase and ask their question so that it can be well received by the Stack Overflow community.

In a one-month online study, we implemented our mechanism for mentored question asking on Stack Overflow, and enabled 271 novices to receive help with their questions. As a result, we found that mentored questions were substantially improved over non-mentored questions. Average scores increased by 50%, resulting in fewer off-topic, deleted, and poor questions. Overall, for mentored questions, there was an increase in the amount of *good* questions asked, and reduction of *bad* questions asked by Stack Overflow standards. Novices surveyed agreed that they feel more comfortable posting on Stack Overflow after their participation (median = 4 on a 5 point Likert

# "We don't do that here"

- Question phrasing
- Formatting posts
- Community triage
- Question framing
- Community culture of asking

is this an appropriate place for your Q?

## "We Don't Do That Here": How Collaborative Editing with Mentors Improves Engagement in Social Q&A Communities

Denae Ford<sup>1</sup>, Kristina Lustig<sup>2</sup>, Jeremy Banks<sup>3</sup>, Chris Parnin<sup>1</sup>

<sup>1</sup>North Carolina State University, Raleigh, NC, USA

<sup>2</sup>Stack Exchange, Inc., New York, NY, USA

{dford3, cjparnin}@ncsu.edu, klustig@stackoverflow.com, \_@jeremy.ca

### ABSTRACT

Online question-and-answer (Q&A) communities like Stack Overflow have norms that are not obvious to novice users. Novices create and post programming questions without feedback, and the community enforces site norms through public downvoting and commenting. This can leave novices discouraged from further participation. We deployed a month-long, just-in-time mentorship program to Stack Overflow in which we redirected novices in the process of asking a question to an on-site *Help Room*. There, novices received feedback on their question drafts from experienced Stack Overflow mentors. We present examples and discussion of various question improvements including: question context, code formatting, and wording that adheres to on-site cultural norms. We find that mentored questions are substantially improved over non-mentored questions, with average scores increasing by 50%. We provide design implications that challenge how socio-technical communities onboard novices across domains.

### ACM Classification Keywords

H.5.3. Group and Organization Interfaces: Computer-supported cooperative work

### Author Keywords

social Q&A, collaborative editing, e-mentoring, programming

### INTRODUCTION

Building and maintaining active online communities is a difficult and well-documented problem across many community types [4, 15, 23]. For prospective community members, barriers such as learning community norms [24], overcoming technical hurdles [31], and resolving conflict [12] can be harmful to participation. In addition, these barriers may significantly affect people in marginalized groups, such as women and people of color, from fully participating in online communities [14]. This is especially pertinent for online programming communities.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada

© 2018 ACM. ISBN 978-1-4503-5620-5/18/04...\$15.00

DOI: <https://doi.org/10.1145/3173574.3174182>

Stack Overflow is the largest online programming community [25]. Each month, over 40 million people visit Stack Overflow, a social Q&A site, to learn about, ask, or answer over 14 million programming questions. Despite great popularity, there is evidence that negative behaviors and malfunctioning community mechanics can have long-term effects on site participation. For example, many questions go unanswered [30], and 90% of accepted answers provided by new users are self-answers. For Stack Overflow, "hostile" criticism and conflict [14, 13] is especially problematic for prospective members. As a result, a user may decide not to ask or answer a question for *fear of negative feedback* [14]. These problems can dissuade novices [31] and women [32] from participating in the community. On the other hand, active community members are interested in preserving community norms: not allowing duplicate questions, off-topic or non-closed questions, or poor quality answers. Community members need a mechanism for helping new users ask better questions, while reducing the hostility and negativity of otherwise well-meaning feedback.

In this paper, we applied theory related to learning and communities of practice to a social Q&A site, by using methods related to mutual engagement and formative feedback to improve novices' questions. Building on design claims for increasing engagement in online communities [19], we created *Help Rooms* with collaborative question drafts to enable novices to receive timely and formative feedback from mentors before posting their questions. Our *Help Rooms* work as follows: when a novice is about to post a question, they are asked if they want additional feedback from a mentor. If the novice responds positively, they are redirected to a room with a mentor who can help them edit their question. The mentor offers advice on how to phrase and ask their question so that it can be well received by the Stack Overflow community.

In a one-month online study, we implemented our mechanism for mentored question asking on Stack Overflow, and enabled 271 novices to receive help with their questions. As a result, we found that mentored questions were substantially improved over non-mentored questions. Average scores increased by 50%, resulting in fewer off-topic, deleted, and poor questions. Overall, for mentored questions, there was an increase in the amount of *good* questions asked, and reduction of *bad* questions asked by Stack Overflow standards. Novices surveyed agreed that they feel more comfortable posting on Stack Overflow after their participation (median = 4 on a 5-point Likert

# "We don't do that here"

- Question phrasing
- Formatting posts
- Community triage
- Question framing
- Community culture of asking

clarity, research of problem, context

## "We Don't Do That Here": How Collaborative Editing with Mentors Improves Engagement in Social Q&A Communities

Denae Ford<sup>1</sup>, Kristina Lustig<sup>2</sup>, Jeremy Banks<sup>2</sup>, Chris Parnin<sup>1</sup>

<sup>1</sup>North Carolina State University, Raleigh, NC, USA

<sup>2</sup>Stack Exchange, Inc., New York, NY, USA

{dford3, cjparnin}@ncsu.edu, klustig@stackoverflow.com, \_@jeremy.ca

### ABSTRACT

Online question-and-answer (Q&A) communities like Stack Overflow have norms that are not obvious to novice users. Novices create and post programming questions without feedback, and the community enforces site norms through public downvoting and commenting. This can leave novices discouraged from further participation. We deployed a month-long, just-in-time mentorship program to Stack Overflow in which we redirected novices in the process of asking a question to an on-site *Help Room*. There, novices received feedback on their question drafts from experienced Stack Overflow mentors. We present examples and discussion of various question improvements including: question context, code formatting, and wording that adheres to on-site cultural norms. We find that mentored questions are substantially improved over non-mentored questions, with average scores increasing by 50%. We provide design implications that challenge how socio-technical communities onboard novices across domains.

### ACM Classification Keywords

H.5.3. Group and Organization Interfaces: Computer-supported cooperative work

### Author Keywords

social Q&A, collaborative editing, e-mentoring, programming

### INTRODUCTION

Building and maintaining active online communities is a difficult and well-documented problem across many community types [4, 15, 23]. For prospective community members, barriers such as learning community norms [24], overcoming technical hurdles [31], and resolving conflict [12] can be harmful to participation. In addition, these barriers may significantly affect people in marginalized groups, such as women and people of color, from fully participating in online communities [14]. This is especially pertinent for online programming communities.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada

© 2018 ACM. ISBN 978-1-4503-5630-5/18/04...\$15.00

DOI: <https://doi.org/10.1145/3173574.3174182>

Stack Overflow is the largest online programming community [25]. Each month, over 40 million people visit Stack Overflow, a social Q&A site, to learn about, ask, or answer over 14 million programming questions. Despite great popularity, there is evidence that negative behaviors and malfunctioning community mechanics can have long-term effects on site participation. For example, many questions go unanswered [30], and 90% of accepted answers provided by new users are self-answers. For Stack Overflow, "hostile" criticism and conflict [14, 13] is especially problematic for prospective members. As a result, a user may decide not to ask or answer a question for *fear of negative feedback* [14]. These problems can dissuade novices [31] and women [32] from participating in the community. On the other hand, active community members are interested in preserving community norms: not allowing duplicate questions, off-topic or non-closed questions, or poor quality answers. Community members need a mechanism for helping new users ask better questions, while reducing the hostility and negativity of otherwise well-meaning feedback.

In this paper, we applied theory related to learning and communities of practice to a social Q&A site, by using methods related to mutual engagement and formative feedback to improve novices' questions. Building on design claims for increasing engagement in online communities [19], we created *Help Rooms* with collaborative question drafts to enable novices to receive timely and formative feedback from mentors before posting their questions. Our *Help Rooms* work as follows: when a novice is about to post a question, they are asked if they want additional feedback from a mentor. If the novice responds positively, they are redirected to a room with a mentor who can help them edit their question. The mentor offers advice on how to phrase and ask their question so that it can be well received by the Stack Overflow community.

In a one-month online study, we implemented our mechanism for mentored question asking on Stack Overflow, and enabled 271 novices to receive help with their questions. As a result, we found that mentored questions were substantially improved over non-mentored questions. Average scores increased by 50%, resulting in fewer off-topic, deleted, and poor questions. Overall, for mentored questions, there was an increase in the amount of *good* questions asked, and reduction of *bad* questions asked by Stack Overflow standards. Novices surveyed agreed that they feel more comfortable posting on Stack Overflow after their participation (median = 4 on a 5-point Likert

# "We don't do that here"

- Question phrasing
- Formatting posts
- Community triage
- Question framing
- Community culture of asking

*You also might want to edit out the “Thank you!” at the end. I know it seems polite, but people object to it on Stack Overflow.*

## "We Don't Do That Here": How Collaborative Editing with Mentors Improves Engagement in Social Q&A Communities

Denae Ford<sup>1</sup>, Kristina Lustig<sup>2</sup>, Jeremy Banks<sup>2</sup>, Chris Parnin<sup>1</sup>

<sup>1</sup>North Carolina State University, Raleigh, NC, USA

<sup>2</sup>Stack Exchange, Inc., New York, NY, USA

{dford3, cjparnin}@ncsu.edu, klustig@stackoverflow.com, \_@jeremy.ca

### ABSTRACT

Online question-and-answer (Q&A) communities like Stack Overflow have norms that are not obvious to novice users. Novices create and post programming questions without feedback, and the community enforces site norms through public downvoting and commenting. This can leave novices discouraged from further participation. We deployed a month-long, just-in-time mentorship program to Stack Overflow in which we redirected novices in the process of asking a question to an on-site *Help Room*. There, novices received feedback on their question drafts from experienced Stack Overflow mentors. We present examples and discussion of various question improvements including: question context, code formatting, and wording that adheres to on-site cultural norms. We find that mentored questions are substantially improved over non-mentored questions, with average scores increasing by 50%. We provide design implications that challenge how socio-technical communities onboard novices across domains.

### ACM Classification Keywords

H.5.3. Group and Organization Interfaces: Computer-supported cooperative work

### Author Keywords

social Q&A, collaborative editing, e-mentoring, programming

### INTRODUCTION

Building and maintaining active online communities is a difficult and well-documented problem across many community types[4, 15, 23]. For prospective community members, barriers such as learning community norms [24], overcoming technical hurdles [31], and resolving conflict [12] can be harmful to participation. In addition, these barriers may significantly affect people in marginalized groups, such as women and people of color, from fully participating in online communities [14]. This is especially pertinent for online programming communities.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada

© 2018 ACM. ISBN 978-1-4503-5630-5/18/04...\$15.00

DOI: <https://doi.org/10.1145/3173574.3174182>

Stack Overflow is the largest online programming community [25]. Each month, over 40 million people visit Stack Overflow, a social Q&A site, to learn about, ask, or answer over 14 million programming questions. Despite great popularity, there is evidence that negative behaviors and malfunctioning community mechanics can have long-term effects on site participation. For example, many questions go unanswered [30], and 90% of accepted answers provided by new users are self-answers. For Stack Overflow, “hostile” criticism and conflict [14, 13] is especially problematic for prospective members. As a result, a user may decide not to ask or answer a question for *fear of negative feedback* [14]. These problems can dissuade novices [31] and women [32] from participating in the community. On the other hand, active community members are interested in preserving community norms: not allowing duplicate questions, off-topic or non-closed questions, or poor quality answers. Community members need a mechanism for helping new users ask better questions, while reducing the hostility and negativity of otherwise well-meaning feedback.

In this paper, we applied theory related to learning and communities of practice to a social Q&A site, by using methods related to mutual engagement and formative feedback to improve novices’ questions. Building on design claims for increasing engagement in online communities [19], we created *Help Rooms* with collaborative question drafts to enable novices to receive timely and formative feedback from mentors before posting their questions. Our *Help Rooms* work as follows: when a novice is about to post a question, they are asked if they want additional feedback from a mentor. If the novice responds positively, they are redirected to a room with a mentor who can help them edit their question. The mentor offers advice on how to phrase and ask their question so that it can be well received by the Stack Overflow community.

In a one-month online study, we implemented our mechanism for mentored question asking on Stack Overflow, and enabled 271 novices to receive help with their questions. As a result, we found that mentored questions were substantially improved over non-mentored questions. Average scores increased by 50%, resulting in fewer off-topic, deleted, and poor questions. Overall, for mentored questions, there was an increase in the amount of *good* questions asked, and reduction of *bad* questions asked by Stack Overflow standards. Novices surveyed agreed that they feel more comfortable posting on Stack Overflow after their participation (median = 4 on a 5 point Likert

# Avoid the "data dump"

what I think I'm doing



# Avoid the "data dump"

what they think I'm doing...

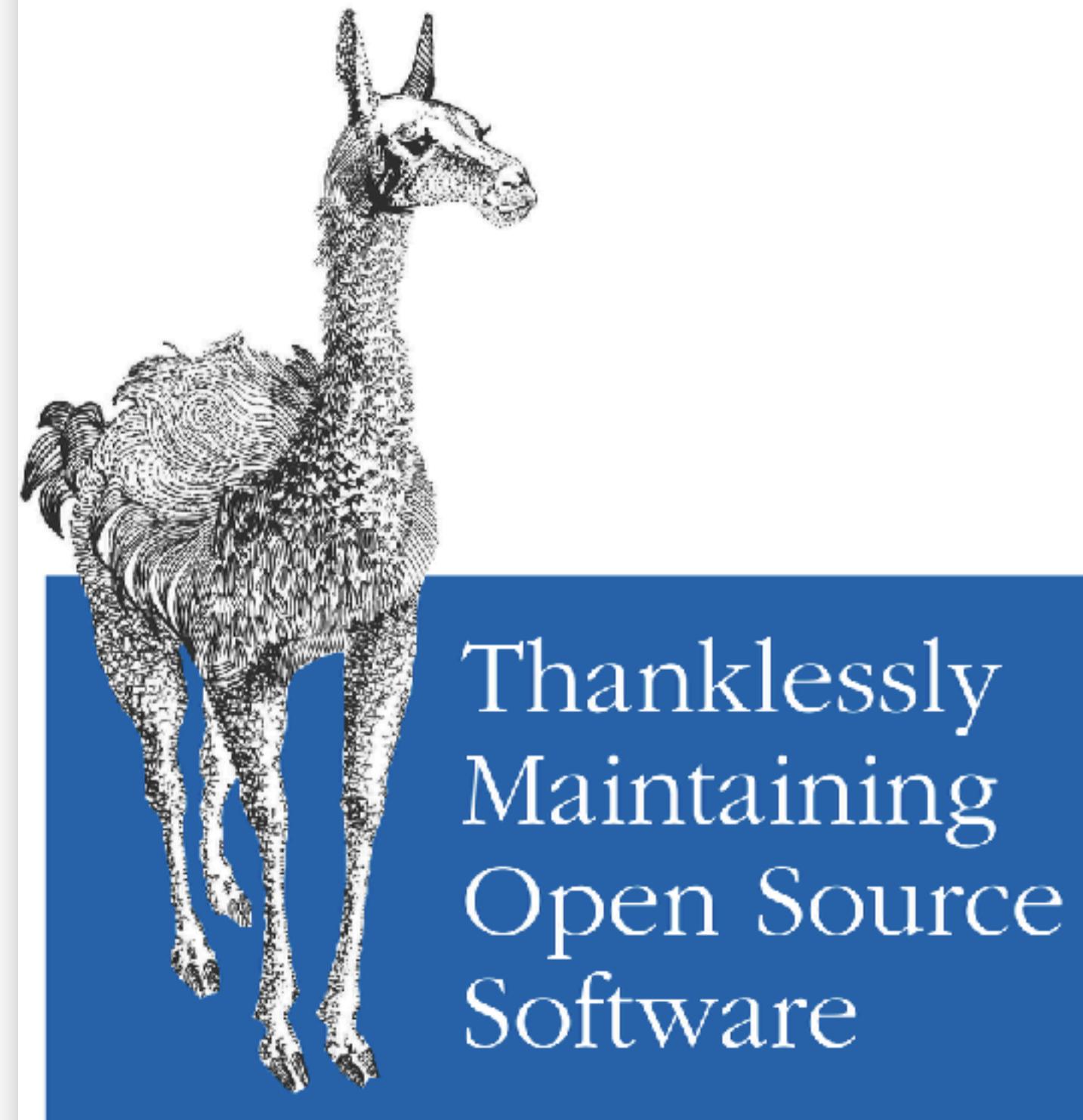


# The source code...



# giving back

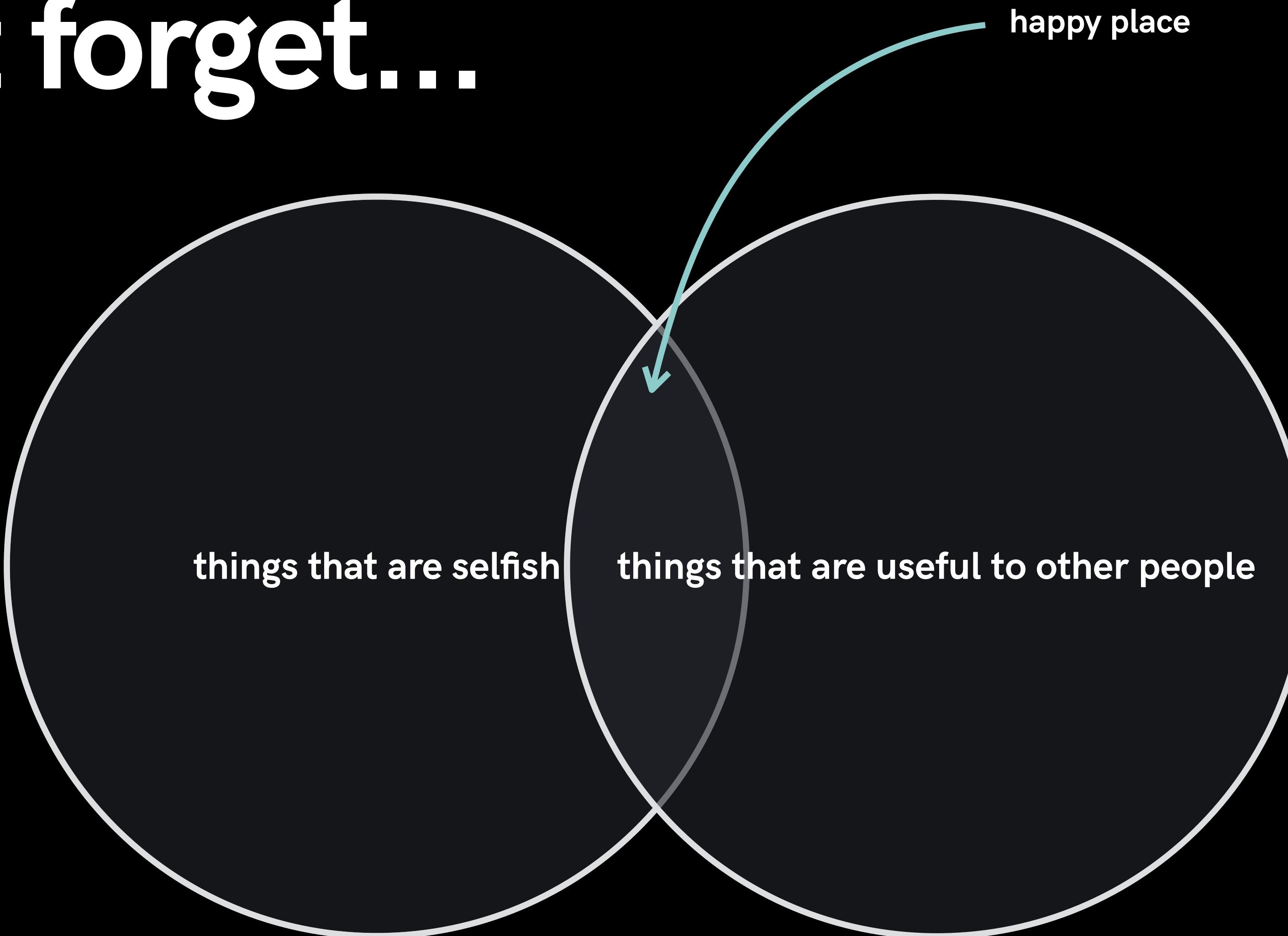
*Acting out of the goodness of your heart, or something*



O RLY?

@ThePracticalDev

# Don't forget...



# Thank You

[bit.ly/noreastr](http://bit.ly/noreastr)



# GitHub surfing links

- <https://help.github.com/articles/about-searching-on-github/>
- <https://help.github.com/articles/searching-code/>
- [http://stat545.com/bit006\\_github-browsability-wins.html](http://stat545.com/bit006_github-browsability-wins.html)
- <https://help.github.com/articles/understanding-the-search-syntax/>
- <https://help.github.com/articles/searching-on-github/>



# Presentation matters

Thanks, Colin Fay

and R Core, obviously

<http://colinfay.me/r-language-definition/>

<http://colinfay.me/writing-r-extensions/>

<http://colinfay.me/r-data-import-export/>

<http://colinfay.me/intro-to-r/>