rOpenSci Community Call - Set Up Your Package to Foster a Community

This doc - **http://bit.ly/ropensci-commcall-pkg-community** - is a place for collaborative note taking. It includes:

- Agenda
- Attendees list. Please add yourself
- Questions from attendees. All are encouraged to ask and answer in this doc.
- Info about The Labelathons!
- Resources list. Add your favorites.

Information on today's topic, with speaker bios:
https://ropensci.org/commcalls/apr2021-pkg-community/

Information on our first social label-athon:
https://ropensci.org/events/labelathon01/

Register for the label-athon (recommended)
https://form.jotform.com/210906095013043

This **call will be recorded** and posted along with any other resources at https://ropensci.org/commcalls.

Agenda

1. Welcome and introductions, Stefanie Butland (3 min)
2. Maëlle Salmon, slides(14 min)
3. Hugo Gruson, slides (12 min)
4. Steffi LaZerte, slides (8 min)
5. Q & A moderated by Stefanie (20 min)
6. Wrap-up, Stefanie (2 min)

You all have expertise to share! Everyone is encouraged to add your own questions by typing them below, and sharing your thoughts by answering others' questions. Together we can make this a rich resource for all of us.

In attendance (please add yourself here):

- *Name; Affiliation; Country*

1. Stefanie Butland (she/her); rOpenSci; Canada
2. Maëlle Salmon; rOpenSci; France
3. Hugo Gruson; Institut de Recherche pour le Développement; France
4. Steffi LaZerte; steffilazerte.ca; rOpenSci; Canada
5. Jeremy Selva; National University of Singapore; Singapore
6. Matthias Grenié; iDiv (German Center for Integrative Research in Biodiversity) - University of Leipzig; Germany
7. Andy South; afrimapr, Liverpool School of Tropical Medicine; UK
8. Noam Ross; rOpenSci/EcoHealth Alliance; USA
9. Athanasia Mo Mowinckel; LCBC University of Oslo; Norway
10. Ildiko Czeller; Emarsys Technologies; Hungary
11. Anne Treasure; Talarify/afrimapr; South Africa
12. Urmi Thorat; University of Florida; USA
13. Matt Dray; UK Government; UK
14. Sam Albers; British Columbia Provincial Government; Canada
15. Wayne Smith; California State University, Northridge; USA
16. Valentina Costa; Stazione Zoologica Anton Dohrn, Palermo; Italy
17. Janani Ravi; Michigan State University, (R-Ladies East Lansing); USA
18. Kathryn Doering; Caelum Research Corporation; USA
19. Andy Teucher; British Columbia Government; Canada
20. Ella Kaye; University of Warwick; UK
21. Charles Sweetland; Springhead Data; UK
22. Fan Du; University of Texas at Austin; USA
23. Jeff Stevens; University of Nebraska-Lincoln; USA
24. Jeroen Ooms; UC Berkeley; Netherlands
25. Trang Le; University of Pennsylvania; USA
26. Sina Rüeger; Novartis; Switzerland
27. Rob Crystal-Ornelas; Lawrence Berkeley National Lab; USA
28. Anelda van der Walt; Talarify/afrimapr; South Africa
29. Lluís Revilla Sancho; IDIBAPS; Spain
30. Silvia Canelón; University of Pennsylvania; USA
31. Claudia Engel; Stanford, CA; USA
32. Sailesh Vezzu; NJ; USA
33. Eugene Hickey; TU Dublin; Ireland
34. Adam Hughes; Arkansas; USA
35. Charlotte Soneson; FMI; Switzerland
36. Karly Harker; British Columbia; Canada
37. Elin Waring; Lehman College, New York; USA
38. Yu-Fang Yang; University of Wurzburg; Germany
39. Ernest Guevarra; EcoHealth Alliance; UK

41 of attendees at peak (including speakers)

Notes

Share your notes and comments here. There's a spot for resources below.

Emojis as a way of keeping a friendly tone even when you may need to reject a suggestion from a potential contributor. (great example by Hadley W)

Be mindful to different type of users needs (contributors, users, maintainers)

You can fix code style issues from contributors yourself e.g. 'I will make some cosmetic changes to this PR, you don't need to worry about my style preferences ;-)'

Default github repo files for organisations, e.g. code of conduct etc., will then be applied to all repos in the org.default community health files

You can save responses to Github issues for future use saving replies for your account.

I like the github README template in https://github.com/RichardLitt/standard-readme#license . Unfortunately, I do not know how to use Rmarkdown to reach that level…

Getting feedback from users about how to improve the package is really useful for all.

Use multiple platforms to engage with your users -- email, Twitter, pigeons? -- not just GitHub. Use News.md to apprise people of the changes made (acknowledge feedback/contributions) Share roadmap too.

Let people know where you are: push latest version, CRAN submission issue reporting steps... Avoid using 'allow edits from maintainers' akin to pulling away the keyboard from newbie programmers. :D

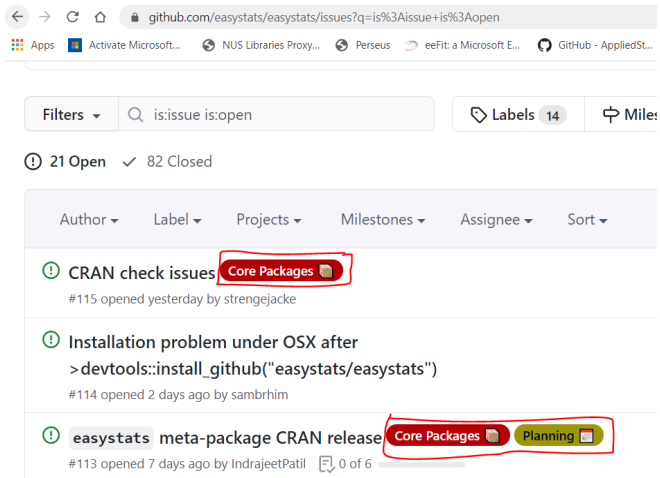
Labeling tips -- even simple ones such as 'Help wanted' or 'Good first issue' can be very helpful.

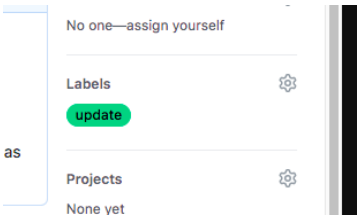
Questions from attendees

Add your questions by typing them below, and share your thoughts by **answering others' questions**. Together we can make this a rich resource for all of us.

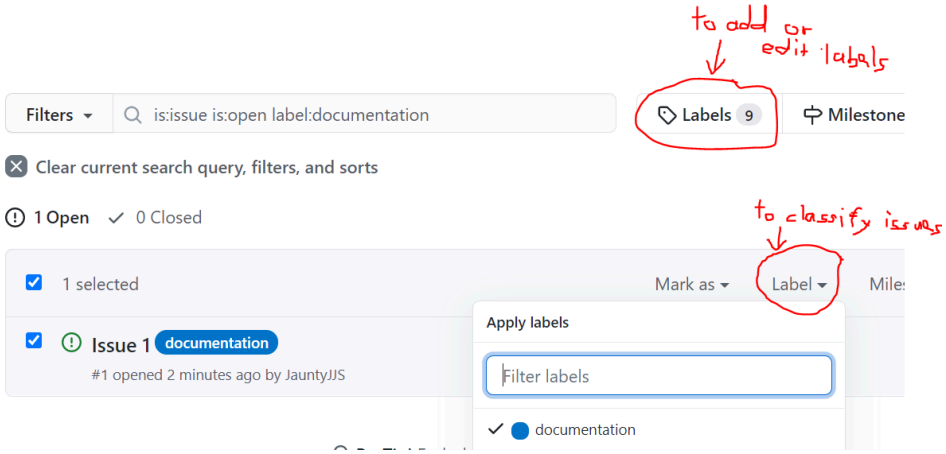
- (Your Name) Your question
- (Rob) I see that many of the files you suggest have standardized names (e.g., CONTRIBUTING.md) does this type of naming enable any features on github or is it more for convention?
    - (Lluís): See the online documentation and there are the other templates for issues and Github Actions.
    - (Noam): Both, as I think these are conventions that GitHub built some features around. For instance, when you open an issue/PR in a repository with CONTRIBUTING.md, GitHub will show a message with a link suggesting you read the contributing guidelines before submitting. I believe it only shows this the first time you open an issue or if the file has changed. (Lluís: Yes)
    - (Rob): Oh, that's interesting about how it shows the message, Noam. Didn't know about that. Thanks for the responses here. And Lluís that's a nice GitHub page with all the descriptions, thanks! :D
- (Mo [Athanasia]) are there any best-practices in terms of formatting NEWS.md files? I find myself struggling to write good version notes.
    - (Stef, note to self) we should put a good example in the resources list
    - (Hugo) The tidyverse team has some guidelines that they use for their projects that might be a good starting point: https://style.tidyverse.org/news.html
    - (Mo) thanks for the link. I'll have a look at that!
- (Silvia) What kind of positive response did Hugo notice after sharing NEWS.md on social media? +1
    - (Hugo) Just general enthusiasm. I'm not a big social media influencer but these tweets are among my most popular ones (in terms of likes/retweets)
- (Silvia) Some packages ask contributors to sign a Contributor License Agreement (CLA) -- when is this recommended/necessary?
    - (Lluís) : I have set up a CLA bot on a repo that was useful once I used https://cla-assistant.io/ . There has been some discussion around it on ggplot2/tidyverse relicensing efforts but don't know the exact issue where this was going on.
    - (Hugo) It might be worth talking about this with RStudio people if you can. They have a CLA on their IDE repository (https://github.com/rstudio/rstudio): https://rstudioblog.files.wordpress.com/2017/05/rstudio_individual_contributor_agreement.pdf
      And I suspect they have the resources to have talked about this with lawyers.

- (Kathryn) Any tips for working with contributors that aren't that comfortable with git/github? My approach has been to be as flexible as possible and allow contributors to contribute via other channels instead of PR (e.g., emailing code that I then can test and add to the repo), but not sure if there are other ideas.+1
    - (Steffi) I agree, be flexible and see what you users are comfortable using. Perhaps include information on beginners guides to git and be clear that you'd be willing to help.
    - Encourage discussions on GitHub Discussions (idea from Sam)
    - (Hugo): this label-athon project from rOpenSci gave me some motivation to try and organise a hack-athon / collaborative hour for pavo (one of the packages I'm maintaining). Just joining zoom at a specific hour and see if users have feedback, guide them through the contribution process if they wish, or just socialise and create this community feeling that is so important. I'll let you know how it goes if I actually go through with it.
- (Lluís): Examples/stories of "help wanted" labels on issues that were taken by users/external contributors? Or they are more of a signal to the community/users that you are open to their input?
    - (Maëlle) https://github.com/ropensci/tinkr/issues/9
- (Mo) - how would you label issues you want help with, but that require a high level of understanding of something complex, that you as a maintainer might not even understand? It would be very intimidating for a "beginner" to look at that and think that this is the level of contribution that is expected.
    - (Hugo) in addition to the `help-wanted` label, GitHub also provides a `good first issue` as default label. It might be useful to signal which issues can be addressed by "beginners".
- (Jeremy) - The easystats R package https://github.com/easystats/easystats has this cool feature to classify issues. Any idea how to do this ?



- (Matt) Jeremy, do you mean the emoji in the labels? You can edit the label names like 'bug :bug:' (emoji name between colons, but note that GitHub doesn't support *all* emoji)
- (Hugo) These are issue labels. If you like them, you should join the label-athon to see how to create them :). In the meantime, you can read more at https://docs.github.com/en/github/managing-your-work-on-github/managing-labels
- (Jeremy) I do not need the emoji yet. I just want to know how to "classify" them. How to put the red box with "Core Package" for example…
- (Matt) There's a 'label' section to the right of each issue where these can be selected (see image). You can edit them with whatever labels/colours you like.
- 

- I got it now thanks.



- (Ella) (Comment, not question) This has been FANTASTIC :) Thank you very very much to all the speakers/organisers. The timing has been perfect for me - a couple of days ago I had an idea for a package that I knew I wanted to have lots of collaboration, and this has been so informative on how to work towards that. I am planning to come along to the label-athon to implement this.
    - (Stefanie) Ella this makes me so happy. Our mission is accomplished ;-)
- (Sina) Maëlle, you mentioned badges for README (in the context of having important information in the README), can you elaborate on that? They will always be on the top, right?
    - (Jeremy) Not necessarily always on the top. The correlation R package https://github.com/easystats/correlation has informative badges but they are placed on top of each header.

(Maëlle) I was thinking of https://github.com/r-lib/cpp11#getting-help-
(Matt) Badges are a good indicator of activity in your repo in general – packages like {badger} can be good for generating these
(Lluís): See ?usethis::use_badge to set up some other badge or use_coverage that sets a new badge for coverage or use_github_action that adds a badge for that github action.

- (Ella) In terms of encouraging users to contribute, do you have any advice on how to leverage twitter to get attention for it? I will shortly be releasing a new package that I really want a great community around, and plan to tweet about it, but I don't have a lot of followers!
    - (Maëlle) making sure your Twitter cards (GitHub, pkgdown) are perfect ;-) As an R-Lady, one case use the #rladies hashtag as well.
    - (Matt) make sure to use the #RStats hashtag – there are some accounts like @icymi_r that retweet these for greater reach
    - (Steffi) be consistent, I don't tweet much, but I tweet every weathercan release and generally pin the tweet at the top of my account, over time my reach gets bigger and bigger. Also, submit to rOpenSci (if it's within scope) and they'll help publicize :D
- (Sina) Now that there will be labelathons in co-working mode, could there be a time where some issues are tackled in co-working fashion, similar to a tidydev day?
    - (Steffi) I'd love this, it'd be really nice! For the first pass we wanted to pick something doable within 1.5hrs, but we're not limited to just labels. Will definitely bear this in mind for future ideas! +1
- (Wayne) Is there a convention for where to put ORCID--readme.md? Citation? or?
    - (Janani) The Distill package to create Rmd -> html has an 'orcid_id' in the yml part -- can be added alongside affiliation -- or social media accounts.
    - (Hugo) The ORCID can be added to the DESCRIPTION file. It is then visible both on the CRAN landing page (e.g., https://cran.r-project.org/web/packages/lightr/index.html) and the sidebar of the pkgdown website (https://docs.ropensci.org/lightr/). As it happens, Maëlle published a blog post about this a couple of year back: https://ropensci.org/blog/2018/10/08/orcid/
        - (Janani) this is very cool, Hugo! thanks!
- (Valentina) Again thank you to all the speakers! I recently released a package that is still a work in progress and someone was starting to contribute. Do you suggest to add them immediately in your contributor list?
    - (Maëlle) yes either in DESCRIPTION or NEWS.md. It sets the tone and it might motivate the contributor in question even more. ;-)

    - (Steffi) I have it set in my contributing guide that anyone who opens a pull request should add themselves as a contributor to the DESCRIPTION as part of the pull request, which helps invite/encourage them (I hope!)
        - (Valentina) Thanks! Definitely gonna do that!
- (Janani) [for anyone, esp. in academia] Is there a good way to indicate in the contributing guidelines that if someone contributes a lot, they will be added as co-authors in the subsequent publication. If the contribution is minor, they will at least be acknowledged.
    - (Steffi) I'm not sure if there's a standard way, but I would just be explicit in the Contributions sections, something like "Contributions are welcome and we will be sure to acknowledge contributors in any future publications. Extensive, above and beyond, contributors will be invited to co-author publications."

The Labelathons!

Question about your timezone below
Do you think the ideas discussed today are great, but it's hard to set aside time to put them into practice? We're following up this Community Call with an experiment!

Join us for 1.5 hours Thursday, April 29, 9 AM Pacific / 16:00 UTC for our first co-working social "Label-athon". Registration encouraged.=😩 🍦

- Meet other package developers and rOpenSci staff in Zoom
- Work independently, implementing some of our recommendations for your own package
- Get answers to your questions

Our experiment includes 4 label-athons (if y'all show up, we can do more) at alternating times to accommodate our community around the world. Thursdays April 29, May 13, May 27, June 10.

Want to join a label-athon, but 9 AM Pacific doesn't work for you?

Tell us your country or timezone + what time? We'll try to set something up

- (Silvia) Is this event best for those that already are maintaining a package or is there potential for involvement from those of us that aren't (yet) package developers/maintainers?
    - (Steffi) I'd say it was originally conceived as time for you to work on making your package contributor-friendly. So, yes best if you have a package (or are working on one). However, we're not going to turn anyone away and I could also see it

being a useful time to explore how you might want to set up future/potential packages/ideas to be contributor friendly or similar themes.
    - (Stefanie) Please do come Silvia. You can work on whatever you want during that time. I'm a huge fan of co-working.
- (Ella) Something a bit earlier would be good. I'm in the UK. 9 AM Pacific is 5pm here and pushing up on toddler dinner-time! Something in working hours here would be great :)
    - (Steffi) Tricky, we'll definitely consider that!
- (Jeremy) Singapore (timezone UTC +8:00) Kuala Lumpur, Singapore (9 AM Pacific is 12 AM the next day for me).
    - (Steffi) We're definitely planning the next (May 13th) to be sometime accessible in your hemisphere (not 100% sure on the timing yet).

Resources

Add your favorites below

Files examples

- wateRinfo contributing guide
- skimr contributing guide, section about scope
- rOpenSci default code of conduct
- ggplot2 governance file
- tidyverse default support file
- cpp11 README section with a badge to a forum
- blogdown interface for new issues, source

GitHub docs

- default community health files
- saving replies for your account
- configuring issue templates for your repository

Communication examples

- Example of a comment about style in a PR (blogdown)
- Example of a nice no in an issue (pkgdown)

Others

- pkgdown (dev version) for customizing what's in the sidebar and footer
- tidyverse style guide
- Beginner contributor resources listed for Hacktoberfest

Recent tweets of Jim Hester's are insightful & very useful

- twitter.com/jimhester_/status/1359584159317000195
- twitter.com/jimhester_/status/1357680037470875649
- twitter.com/jimhester_/status/1356272709345419264
- twitter.com/jimhester_/status/1355195248612356099
- twitter.com/jimhester_/status/1351930803656257544

For tips specific to *maintaining* a package. See the recording, summary post, and collaborative notes from our past Community Call on Maintaining an R Package.

Share your perspectives on the Community Calls we're planning

Ideas for upcoming Community Calls are shared as issues in this public repository. Give a 👍 on a topic, share your perspective by commenting on a topic, or open a new issue if your idea doesn't fit in any others.

Transcript

Stefanie Butland 0:02
Okay, well,

Hello everyone. Welcome. I'm Stefanie Butland. I'm rOpenSci's community manager and I welcome you to today's community call on how to set up your package to foster a community. rOpenSci is a nonprofit initiative founded in 2011 by Karthik Ram, Scott Chamberlain and Carl Boettiger. We try to foster a culture that values open and reproducible research by creating technical infrastructure in the form of carefully vetted staff and community contributed R software tools, and creating social infrastructure infrastructure through a welcoming community of practice, you can find our monthly newsletter at ropensci.org/news. We have a code of conduct that applies to this call. And you can find it in the footer of our website, along with guidelines for reporting and enforcement. The session is being recorded and the video and any other resources, lots of resources associated with it will be posted on our website at ropensci.org/commcalls. And I'll tweet from rOpenSci account when those are up. Today, we're using a shared Google Doc to take collaborative notes. And my colleague, Steffi LaZerte, is has pasted the link to the doc in the zoom chat, I'll ask you to please if you're comfortable with this, add your name, affiliation and your country to the attendees list. And see, there are a couple of places in the doc and I think I will share it. So I can take you through it and see the places that

you can contribute. So we've got the attendees list here. There's a spot here, I'd really encourage folks to add your own notes, as speakers are speaking things that you're hearing that maybe you hadn't really considered before. There's a place at the bottom of the doc to add resources. But here's sort of just you know, quick, add your thoughts and comments. Here's a spot for adding questions as they come up. So if you add your name followed by your question, when I'm moderating during the q&a in the last 20 minutes of the call, I may invite you to unmute yourself and ask the question. Or if you prefer, I could ask the question on your behalf, you can let me know. There's a section here that will explain when Steffi introduces our social labelathon that we're starting next week. That's where you find that information. a spot where if that timing doesn't work for you, tell us what works for you. And then a spot with resources. Many of these are already added to the comm call page. But please do add your own resources at the bottom here.

So stopping sharing.

So today's call the topic I love, particularly because it came out of a discussion in our public repo of community call ideas, and a number of different people added their own comments, added resources, other issues were opened. And so we we put this thing together, but it was like truly a something that was of interest to the community and people came together. And so this is like kind of a culmination of so many different people's ideas. So I think everyone who has, you know, commented ask a question, added a resource because this is a combination of the community's work and I appreciate that. So finally, I should say we're going to have three speakers. And I'll introduce them now and then when we get going, they will just go from one to the other without interruption. So my pleasure to introduce these lovely people by Maëlle Salmon is a research software engineer part time with rOpenSci where she among many other things, maintains the guide we call rOpenSci Packages: Development, Maintenance and Peer Review. She also created the R-hub blog and co wrote the book HTTP testing and R with Scott Chamberlain. Maëlle lives in Nancy France. Hugo Gruson is an evolutionary biologist who fell in love with R and R package development during his PhD. After this, he moved to a research software engineer job in Montpellier, France to work full time on building tools for research in epidemiology. Steffi LaZerte is a consulting R programmer and teacher in Brandon, Manitoba, Canada, and part time very awesome community assistant working with me for rOpenSci. She co authored the rOpenSci community contributing guide with me. And though her background is in behavioral ecology, her love of R drew her away from academia and into a career facilitating science with R. So with that, I would like to turn it over to Maëlle Salmon. There we go.

Maëlle Salmon 5:01
Okay, thank you. So as we share some tips around setting up your package to foster a community. First of all, I wanted to show these pictures of bees working together in a beehive because that's how we might imagine open source developers, a lot of people working together. Now, as I was preparing this talk, I was inspired by the words of an open source professional, my colleague, Scott wrote in his GitHub README profile. Fun fact, about 90% of these species are solitary, not social. And this made me wonder, are package maintainers also solitary, which

will be which will be kind of sad. Well, thankfully, there can be a community around package maintainers. So what is a community of a package? Well, it will be used the maintainer of a package or service, or you could be one of the authors, users and contributors of the of the package. And this can be a really great thing and generate a lot of value. So I wanted to share two quotes from successful projects about community. So in the governance file of the ggplot2 package we can read, people recreate these realisation with ggplot2 as the most important members of the community without these users whose project would have no purpose. And then, Will Landau, when he introduced the targets package successor to the drake package wrote, everyone will contributed to drake is part of targets for years of pull requests issues are open to discussions as to the community past, Stack Overflow threads are materializing in this new suite of tools. So really a lot of activity from on top of different activity, providing value.

So good news about the community of a package is that everyone is this community wants a better well maintained package. Now what I want to also keep in mind is that everyone might want something different from devs themselves. So use a package authors might find some citation of the package some things, users might want their feature wishes implemented, and the contributors need recognition. So that's something to keep in mind when interacting with people in this community. We first sharing some practical tips about files, I wanted to mention the importance of your tone. So firstly, it's important to be kind, it's good to be kind. But also if you are not kind in your interactions in your repository, for instance, people might not come back, or might not come at all because they might have read old answers. It's also important to be graceful because we're all going to make mistakes, like some people might not follow your code style or your route that you want should, and I will emojis and sign up because emojis are compulsory cannot bet. Because it's important to keep in mind that in online interaction, some nuances are lost. And images are a tool that might help convey that something is fun. For instance, I have chosen an example of a kind answer with an emoji to illustrate this idea of tone. And it's an answer I find particularly interesting because it's a "no", so this is Hadley Wickham saying no to a feature request into a package on package. And he wrote, I think if you want this, you could just include it in a vignette. Unfortunately, I don't think it's a good fit for this package. developing good software requires relentless focus, which means that we have to say no to many good ideas. Even though I'm closing this issue, I really appreciate the feedback and hope you'll continue to contribute again in the future. Grinning emoji, so that might be a nice example. So now have to come to the actual tips about files and setup. I like to think of setup for fostering community in three steps. You need to think about the rules and standards you want to implement in your package, you should add them as expected files and then you need to help users and contributors find them. And of course, it may be a bit of repetition like you might change the rules and standards over time. So first thing you need to add in your repository is a code of conduct. So Stef mentions rOpenSci's Code of Conduct at the beginning of this call, for instance, an rOpenSci at an all community call the topic of adding a code of conduct for your lab, for your code, base, and other other things. Now, enter in an R package in our GitHub repository, you would add a code of conduct conduct under dot GitHub slash code of conduct.md. Let's look at an example of a non rOpenSci package. So open the Toronto package by Sharla Gelfand. So it has a code of conduct. And this code of conduct is one that you can get

simply by running there use this function use code of conduct. When another file that might be important, especially for those of you working in academia is a citation file for helping people cite your package it leaves in insert slash citation. And you can start a very simple citation file by running the usethis use citation function. It does not provide much information, so you need to amend it. So let's look at an example from the NASA Power Package by Adam Sparks. So it's a fellow with anything that's, you know, going to read all the lines on on this call. But it has first a part with some information about the data source and saying to cite NASApower, please use, there are actually two citation entries in this file. One of them is a paper, a paper was a package published in the Journal of open source software. And the other entry is a citation of the manual. If you don't have a paper about your package, what you can do to provide a DOI is to archive your repository on zenodo. But this will give you a DOI that you can add to your citation form. For more details and citation files, you can read our blog posts make our package easier to cite. Then another important factor is a contributing guide, that would leave under dot GitHub slash contributing MD. And I'm going to show an example the waterinfo package, but stay stained from UI. And I haven't chosen this example because it's full of emojis, although that's a good a good thing. So this contributing guide has a lot of information from contributors from sharing the love about the package, like telling other people about a package to improving the documentation contributing code, with added some development guidelines. In general, what would you add to your contributing guide? Right? Really, that's up to you. But you could have your style preferences in their infrastructure that your package uses like roxygen2, because not everyone knows about roxygen2 your process preferences. If you want people to open an issue before they open a pull request, maybe something about the scope of your package, which is a case in the skimr contributing guide. And let's see if your package as anything a bit weird a special like if if it interacts with their web API, and that for testing, you can create a sandbox account. How does one do that? Do you use mocking in tests. And for all these things that are a bit more technical, you don't need to re write the docs obvious things, you can links to x link links to XML documentation.

I wanted to make a special note about coding style because you don't want coding style to be a hurdle in contributions to your package. So firstly, maybe to use as if you don't have a strong opinion about coding style, you could use a popular coding style, because it's more likely that people will already use that coding style. And if you use for instance, a tidy roost site, a diverse style guide, when you can link to sections of a style guide. For your contributors, then you might want for a tool to be the one making comments about code style or making fixes to code sell. So you could use the linter and styler packages for various topics. And then you could make a decision to either teach your contributed contributors to your package, because you think that they will come back to the repository, or you could choose to fix it yourself. So I once made a pull request blogdown. And he reassured and wrote, after approving my pull request days away, I will make some cosmetic changes before I merge this pull request. And you don't need to worry about my style preferences. So that's one approach. There are other Community Health files that you could add to your repository, you don't have to GitHub recognizes a security file, I've never seen that use for an R package. So support.md file is for explaining to us how we can get help for your back about your package, which might overlap a bit with the contributing guide.

And sometimes in repositories, you can find a governance file for spelling how decisions are made. But he says no special word for for GitHub. If you find yourself adding the same code of conduct or contributing guide over and over again, in your repositories, you might want to use a GitHub feature of default GitHub community file. So in a doc GitHub repository, you can store vs. vs files and when they apply by default for all repositories, obvious organization, although they can be overridden in individual repositories. So an example would be the code of conduct of rOpenSci, so it lives on our website, but we have it as well as a default GitHub community file. So under the rOpenSci slash dot GitHub repository, there is a code of conduct file, which is just a link to the code of conduct on our website. But it's quite handy to know about this will apply to all repositories. And our code of conduct as noted by Stef is linked from the footer of the rOpenSci website here as well from pkgdown websites. It's very easy to find from different places. Which brings me to my next point, which is that you have this fast contributing guide citation file, but you need to make sure that users and contributors will find their information about contributing. So you should first put links everywhere to their GitHub repository and to pkgdown website so that users that are in the console, know how to find their place. Yes, you should put links to the GitHub repository and pkgdown website in the description file into pkgdown configuration, and in the GitHub repository description. And once you've done that, you can improve your GitHub README. So regarding the citation file, for instance, you have a citation file, but user might not know it exists and might not remember to save the package. So to help with that, you can use some sort of double protection by writing an explicit section in your README about how to cite this package. So Noam Ross wrote in the rOpenSci forum personally, I take a belt and suspenders approach and still put citation things in the readme. And you can apply this idea to having a section about contributing to your package or about getting help when using a package. And you can use badges to draw attention to certain README sections. Then, even if the users have come to your GitHub repository that might miss misinformation is a readme, but you have other ways to communicate information to them. So let's look at an example from the weathercan package by Steffi. If you open an issue in there, you get an issue template, so you can fill an informative issue. And at the bottom, there is a link to the contributing guide of weathercan add to the code of conduct of rOpenSci. And this comes from the issue template comes from the dot GitHub folder in this repository, and our example a bit more complex than the blogdown package if you open an issue in the blogdown package. oops.

You get to this interface, where you can choose to report a bug, request or feature and you have external links to the issue guide, and places to ask question. And this also comes from the dot GitHub folder, where there is to an issue template bug reports and feature requests and one configuration file with their link to external resources. If you do that don't add too many choices because that might overwhelm users. But this, this kind of interface can help them and contributors. which can help users and contributors. And last but not least, you can save replies for your account. So if you find yourself writing the same answer over and over again, you can save it this way. You can use it even when you are in a hurry or getting impatient. Last place where you can put information about your contributing guidelines and your code of conduct is a pkgdown website. So I will show two example one from an rOpenSci packages skimr package.

So by default in a pkgdown a website if you have a contributing guide or a code of conduct in the sidebar there is a community section with the link to the contributing guide. And because their package on the homepage, content comes from the readme, if you have a section is a readme about contributing like skimr, so if I scroll down, I there was a section about contributing and is also visible on the pkgdown website. Now because skimr is an rOpenSci package the footer is rOpenSci website footer, let's look at an example of another package open intro package. So the footer is some text but you can choose to edit in the development version of pkgdown. So if you want you can add text about citing your package to every page of your pkgdown website. So that's all for me the conclusion of this is that you should summarize your rules and we had things surprising things about how your package, you should store them in the standard files and you should help users and contributors find them and the whole time you should watch your tone. Now over to you Hugo

Steffi LaZerte 19:46
Hugo Yeah.

Hugo Gruson 19:55
Okay, you should be able to hear me now and you should see my slides.

Okay, good.

So So hi, everybody, my name is Hugo. And I would like to talk today about my journey from package user to package developer, and how we can, as a community, invite more users to take this journey. And first, I will start with my story and how I got started and fell in love with package development. So my very first contribution to an R package was in a package called pavo that is dedicated to the analysis of color, as seen by your specific receiver. So for example, and then an insect or bird. And at first, I was a regular user of this package, but I was already really excited about the new features that this package would have at each release. And I was always watching out for the news file to see and find out what the what the what was new, and what I could try out in the new version. And one day, I went a little bit further, and I started to browse the GitHub repo. And I noticed an issue that I knew how to solve. So I decided to contact the developers and to offer to submit a pull request. And I would like to emphasize here that at least I'm this was really scary for me, because I didn't know anything about the etiquette or supposed etiquette to contact package developers. And I was afraid they would be offended that I told them that their package was not perfect. But thankfully, which story has a nice ending, because I continue to contribute. And eventually I became an option maintainer of this package. And after this, I started to feel really empowered. And I started working on other R packages. And so the questions that I would like to address in this talk today is how we can facilitate this transition in other users, and how we can make sure that our projects are not just simply open source, but truly community projects. And one of the problem that we have is that many users think that their feedback could not be valued. So example, this is a screenshot of a Twitter conversation where the first person asked us how they would feel how they feel after submitting an issue on GitHub. And one of the answer is someone saying that they never submit anything on GitHub,

because they know that the developer don't actually want to deal eith that shit. And that they feel lucky if they even get a reply. And I think that this is really characteristic of a pattern that we commonly see where the user notice something that is broken or unclear, and they just give up and move on. And what we would like to see is if someone notices something that is broken or unclear, that they report the issue to us, so we can try to fix it. And so I would like based on my experience as a contributor to packages share, to share some tips to make it less scary and more rewarding to give feedback and contribute. And I would like to highlight the fact that these tips are based on my experience, so they might be very subjective. And I would be interested in hearing your feedback about it. And to talk about this with you in the discussion, after I think the very first step is to have confidence in your project and to expect the fact that some people will want to contribute. And I first started to think about it while watching the GitHub issue related to this community call, where someone was sharing their experience about the most active repository. And they wrote that they didn't expect anyone to use this project outside of the class. And but the project was on GitHub, and some people found it and started to use it. And I think it's something that is really common, where we think that our projects are just useful for us, or they are not interesting enough for other people. But actually, if you put something out on GitHub, then it's very likely that someone will find it and will start to use it.

And the problem is that if you don't have this confidence that your project will attract contributors, then you run the risk of missing out on contribution. This, for example, was an answer on a pull request that I submitted. And I got an answer from the developer or almost two years later. And they said that for a very long time, they didn't check pull requests, since they didn't really expect people to be interested enough in this project to want to contribute. And by the time that they saw my request, it was already completely out of date with the project so it was not useful for them. On the contrary, you should show that you are available and that you want to communicate. And if possible, I find that it's useful to be available on many platforms, many channels. For some projects, I noticed that the majority of the feedbacks comes from, come through via email or via Twitter. So I think it's useful to be available on other channels, and just GitHub because not everybody might have an account on GitHub. And it's good to share the recent changes about your package on this platform, for example, the news file each time you make a release, and to open some issues with unresolved questions and possibly tag them with the tag help wanted this, these last two points will create some engagement and a feeling of community. And I've noticed that, for example, sharing the news file, via Twitter, it's something that I didn't do at the beginning. But I noticed that it creates a lot of positive response as well. Then the second point is that you should not let contributors work for nothing or feel ignored. So as soon as you have some feedback, it's good to acknowledge it, even with a short answer if you don't have time, but to say something as soon as possible, even just thank you. And then once the issue is solved, or once you've taken into account, it's nice to mention, again, the name of the person when you share the news, or even to add them to contributors. On the topic of not letting contributors work for nothing, I think it's important to make sure that you always push the most recent version of your work on GitHub, because something that already happened to me is that I submitted a pull request on a project. And then I got an answer saying, look, the version that is on GitHub is not actually the latest version, I have the latest version on my computer. So

the changes that you submitted, are not actually interesting for me because they're already out of date. And this is really demotivating. And you will not want to come back to this project. Once you find out this. On the contrary, you should share upcoming development for your package in the form of a roadmap, because then some users can give their opinion can weigh in on what you're planning to do. Welcome could even solve some problems that you have. And on a related note, it's common for some packages to have some discussion that happens on other channels. So for example, via zoom, or via slack. And it's nice to put a reminder about this on the public repository. So for example, this is a screenshot of a GitHub issue opened by Scott after staff an rOpenSci staff meeting, where he summarized everything that was discussed during the staff meeting. And then it's available for everybody to see and possibly comments. And this reminds me of the really nice sentence that I like in the usethis use release issue documentation, where they say that using this function helps watchers of your package to stay informed about where you are in the process of releasing a new version. So it's the same idea to let people know where you are and what you are going to do to create some engagement and possibly allow them to comment.

You should also make sure that you reduce the workload of possible contributors to make make it easy for them to join. And one point on this topic is to disable failing tests. This is an example of an issue on GitHub where someone tried to fix a link, or can be also fixing it just a typo. And then at this moment is if tests fail for a reason that is completely unrelated to the change. It can be really scary or destabilizing for the user because they start feeling that they did something wrong, or that they broke something when in fact, it's just completely unrelated to the changes. So I think that if your tests are failing for a long time, then it's better to completely disable them rather than keeping them in a broken state for a long time. And On a related note, I think it's good to disable bots that are unnecessary. So for example, I remember one of the first pull request I did to an to an R package. And I got this comment about the codecov bot where it said something about codecov good coverage. I didn't know at the time what code coverage was. And I didn't understand this message message. And I was starting to wonder if I did something wrong if I broke something, how I can fix it. And to be honest, even today, I'm not completely sure how to read how to understand this comment. So I think it's not super helpful, and it's better to disable it. But at the same time, even as you are reducing the workload of possible contributors, it's good to leave some opportunities to learn. And in particular, I think that it's good to avoid using too often, the feature from GitHub, that is allow edits from maintainers. Because if you take the opportunity, from users to fix fix the mistakes themselves, then can be really demotivating. And for me, too, it's akin to the carpentries advice, the advice of not taking the keyboard from a learner, and on the opposite you should just comment and tell them how to fix the possible mistakes that they did. It's really a good tool to learn. And it's really empowering for them. So to summarize, I think you can think about five main ideas to try and facilitate contribution to your projects. The first is to have confidence in your project and expect contribution, then to show that you are available and that you want to communicate, to not let contributors work for nothing or feel ignored to reduce the workload, but at the same time to leave opportunity to learn. And now that I said this, I would like to highlight the fact that we are not perfect ourselves. And even even though in this community called we are giving tips about

what you can do to improve your package or foster community around your package, we don't always follow the steps of ourselves. And what we should do is as a community strive to be better and to develop tools to make this easier for everybody. And with this, I will let the floor to Steffi. Thank you.

Sam Albers 32:30
All right. So I'm not going to chat for too too long. Today, I'm mostly, you've had a really great opportunity to hear from a lot of great tips from Maëlle and Hugo. And now I'm going to talk about just so sort of tackling some low hanging fruit if you're feeling a little overwhelmed as I was, so I have what I thought was a small package weathercan it still is a small package. And I found that my package's community completely took me by surprise. Just a couple of weeks ago, I was just doing some updates. And I suddenly looked at the little badge I had about the number of downloads. And I was shocked to see that was over 20,000. Now there's not much by like a big package standard. But I really thought that weathercan was like a really small, quiet little package. And so suddenly, it occurred to me and like I have a community of users if not have necessarily contributors, but they might become contributors. And so I wanted to think about how I could make my repository more contributor friendly. And one thing I want to talk about today is what I consider like the low hanging fruit. And that's just something as something as simple as labeling the issues and creating issues for your users. Now that might seem overly simplified. So like, why would we bother like with something like that, when it's just you're just putting a label on the issue, can that really help all that much. But I really feel like it can be useful. As we mentioned a couple times already in this call, you know, users can be shy, especially if you're sort of new to R, it can be really intimidating to to sort of either open an issue or to do a pull request or anything like that. And so I find that by labeling issues, specifically is something like Help Wanted, it becomes a really open invitation, it becomes a clear message that I really do want help with these things. And I really do value external input. On a more practical side labels can also be really useful because they can be searched. So for example, this link here brings us to the GitHub search for all open issues in the our open sigh organization. Even more useful is perhaps this one which looks at all or open issues that had been labeled Help Wanted so you can see there's actually one open less than a day ago here, which was labeled as a feature that's helped wanted I opened one A couple of days ago here, that was a vignette that I thought would be really useful. Someone mentioned on Twitter recently that whether cam, which is good for accessing environment, sort of weather data from the Environment and Climate Change Canada website doesn't actually deal well with the fact that some stations get decommissioned, and a new station pops up right at the same place. So you've got sort of two stations or two sets of data, that could be homogenized. And I thought, Well, perhaps someone would be interested in writing a vignette for how we can do that. So I opened a documentation issue. So that also shows you that your when you open issues and you label them, they don't necessarily need to be for coding. So this is a help wanted for a strictly documentation. So a good place to go to get some ideas of what might be good ways of labeling and how to maximize the use of these labels is our contributing guide. So this is rOpenSci's community contributing guide. And we have a section on the in the resources here. So if you go to the resources, we have a section called the issues list. And that talks about this

rOpenSci issues list right here. That was the first link I showed you. And so essentially, it's a dynamic page that shows all open issues in the rOpenSci GitHub organization, we also specify that there might be when you're labeling your issues, a really good way to if you're labeling issues that you want people to search for, it's best to use sort of some of these standard labels such as Help Wanted, good first issue, which is great for inviting new users to our to your package. documentation. Like I said before, they don't have to be coding issues, or feature so adding new features to your package.

One reason why this is particularly important for my package too is because weathercan tends to attract a lot of new users, to our even one user who said that weather can was what brought them to our because essentially, it's one of the easiest ways of getting these book data downloads for the site. And so I really do want my, my package to be friendly and to be, you know, to be welcoming. So, I'm going to go through an example and create an issue here. So like I said in earlier, I created this vignette issue here for teaching people how to homogenize their data. But it also occurred to me that something that might be good is actually to add a function that might identify stations that do need that are continuous. So essentially, if one station was decommissioned, and another station started up right at the same place to sit to connect those two stations together to kind of give opportunities for homogenizing the data. So I've prepared this beforehand, so you guys don't have to see me type everything in. And so here is a new issue on a crate, which is a function to identify potentially continuous stations. And I'm going to add some labels. You see, I've got my label set up already, you can edit the labels here, if you want to add some in I've got my Help Wanted label. And this is a new feature. I've also linked to the issue 116, which is that vignette because they are kind of related. Alright, so I'm going to submit that new issue. All right, there we go. Now at the top of my list, and if we go back to our search here and refresh, we now have a new issue in that help wanted list. And so anyone who's cruising through the help wanted list to see if there's anything they can contribute to, now they can see my issue and see if that's something they want to do. Now, like I said, I really do want to make weathercan friendly, I want to make it open, I want to make it inviting.

But really Who has time for this. And I'm terrible at this, like these two issues. I opened specifically for this call, right? So like Hugo said, you know, we have all these tips and tricks, but we're not always that great about finding the time weathercan is something I maintain on the side. And I'm not great at doing all these things that I would love to do. And so Stefanie and I were talking about that kind of thing. And that's where we came up with the idea of the first rOpenSci social and labelathon. So next week on Thursday, at 1400 UTC, which is 9 Pacific, we're hoping to get together with other developers to work together co working session for an hour and a half to make our packages contributor friendly so we're hoping we'll have a bit of a meet and greet will sort of say hi or socialize as a developer who's often kind of isolated even before COVID. I miss getting together with other people. So kind of excited to kind of have a little bit interaction. And then we'll co-works will work on our own projects on anything that will help us make our packages more contributor friendly, and rOpenSci will tweet all the links to the Help Wanted issues that we can create. I think this is a really great opportunity to learn together, we already found in the labelathon issue that we found some interesting tips and tricks just by

discussing things and sort of working together on some processes here. So we figured out that you can actually use, the usethis package function to gather and get sort of set labels, specific labels for your repositories. This is a social experiment, we've never done this before. So we're hoping to do for these are the dates for the for labelathons, we are going to alternate time so we can accommodate people in different time zones. For more information, here's the link. And because this is an experiment, please register to have give us an idea of what's going on and how many people want to come. On that note as well. We also want to do a zoom poll. And this is because we're just really curious what you guys think about the labelathon. So I'm gonna open up our poll here. It's very simple. And I believe it's anonymous as well. And I'm just asking you, if Yes, no, maybe would you be interested in participating? Also, if you, no matter what your responses, there's a room in the community document to kind of elaborate on your response. If you'd like to do elaborate on your response, that'd be awesome, as well.

Steffi LaZerte 41:41
I'm just gonna let we've got

Sam Albers 41:44
We're going to give a little bit more time for people to kind of trickle in. actually got that as 75% of people who have voted. So that's

Steffi LaZerte 41:53
better than I expected.

Sam Albers 42:07
If you have any questions, we'll be doing our questions soon. And you can add your questions to the community, the collaborative document, or let's, I think, I'll let Stefanie take it from here. She'll explain it.

Steffi LaZerte 42:23
Alright, so

Sam Albers 42:23
I'm going to take up the poll, right before we go on to the questions. So I'm going to end that. And I'm going to share the results. Because we see that most people are interested in coming. There's a lot of maybes, and there's like a couple no. And that's that's actually really encouraging. Thank you, everyone. And like I said, if you would be interested in elaborating perhaps on, especially what you're excited about what you're less excited about are some questions in the collaborative doc that you can use as prompts. Thank you very much.

Stefanie Butland 42:57
Thank you, all

you lovely speaker people.

All right, let

me see. So there's one question. So in the shared doc, there is a section below the notes and below questions that says labelathon explains gives links and below that, you can kind of tell us, you know if the timing works, or doesn't work for you, or if you have any suggestions about what would work for you. We want to hear everything.

So

I am thrilled that we now have a bunch of questions from people. And so I'm going to invite people to unmute themselves if you'd like to ask your question. Or I can ask on your behalf. So I am going to

do

I'm going to jump down first and ask Athanasia. Mo. Do you want to ask one of your questions? whichever one is more burning for you right now?

Athanasia Mowinckel 43:58
Yeah, sure. So which one is more burning? For me? That's a good question. Yeah, I guess, the labeling issue. So like, I have a couple of issues in one of my packages, they're like, pretty high level, super tricky things to solve that I literally have no idea myself how to solve. So I would like that would be a terrible issue for a beginner to see and think that that's like, that's the level of contribution that I'm looking for. Like, how do I signal to them? That I'm happy with them. Like my contribution says, like, I documentation is awesome, do whatever, I'm very happy for everything, but I'm just afraid if they go to the issues and they see this, they're like, Oh, yeah, no, I I don't dare touch this.

Sam Albers 44:48
I think you're just I mean, this is just my opinion, but I think if you're just honest in the issue itself, like I mean, you could get creative with your issue labels and like create one this is tricky or Something like that, you know, it's not a standard, but you know, as your repository you do what you like. But I think also, if you're just honest in the description of the issue being like, I have no idea how to do this, I would really appreciate some help. I don't think it's going to be an easy fix. But you know, even if you have ideas, like, let me know, because then you're kind of inviting people who are like, well, I don't know how to fix it. But I have an idea. And they could like, kind of contribute that way. I would just be honest, I think that's sort of my suggestion

Unknown Speaker 45:27
to everyone else.

Hugo Gruson 45:28

And I would add that beyond the Help Wanted tag GitHub, by default has a tag that is good first issue. So it might be good to use it in combination with the help wanted to signal that some issues are good for beginners or for people that get started with your package or anything. Yes.

Athanasia Mowinckel 45:54
Thank you.

Stefanie Butland 46:00
I, Rob had a question. But now there are a couple of answers from people. So I'm going to pass on that one and throw the next question to Luis.

Lluís Revilla Sancho 46:14
Okay, and well, there are a couple of mentions about various issues, and they're all new contributions to your package. But I haven't never actually seen any issue that with that level that has attracted new users and pull requests. I mean, I don't know if there are some examples or stories or someone has experienced how people start contributing to a package.

Sam Albers 46:39
In my very limited experience, like I said, weathercan it's a tiny little package. So it's it's had one pull request, but it was very specifically around an issue that a user wanted. So it wasn't something that I had requested from users, but it was something that users wanted to contribute. I do think it is that is probably the main reason why people contribute is when they have an idea of what they want the package to do, and they're willing to kind of make it happen. But I still think that what we can do as a community is kind of like just also gently encourage people, even if it's not super common yet to contribute, but I bet Maëlle or Hugo, have a better example.

Maëlle Salmon 47:28
I think it's a last question as well, when it's a small package, maybe not many people will find their way into the repository. But yeah, yeah, that when they come, then it's good for the labels to be there. Like I like seeing Help Wanted issues in repositories where I could contribute, because when I don't feel I am trespassing on someone's territory.

Stefanie Butland 47:51
So Maëlle does that mean that you've actually seen a Help Wanted issue and you have contributed? Because I love this question. It's like we're saying do this for great signaling, does anyone actually respond to help wanted issues if they didn't have the idea in the first place themselves?

Maëlle Salmon 48:05
I think so. Or even the existence of these labels, like the use of these labels in the repository? For me, it's a good sign that people might want contributions. I was even if I don't solve this particular issue,

Stefanie Butland 48:19
may I ask Hugo, do you have a

response to a specific example?

Hugo Gruson 48:23
So Well, I don't have a specific example right now. But what I've seen is in project where if people already wanted to contribute, like someone arrives on the repo and wants to contribute, then there will the help quantity issues are good, because they know which ones like they can start with. I've never seen a case where it brings new people like randomly to the GitHub repo, but it's good for people that arrive and that know that they like this project, and they want to get more involved, then it can it give them a starting point.

Stefanie Butland 49:04
My Community Manager response to this is number one. It's happened a couple of times where I'll see a tweet, and some will say, hey, rOpenSci Do you have any, like issues that need help? And we you know, have a single link that we can share with someone and say, Hey, so these are people, let's say who you know, want to give back to open source. And you know, may have very like deep skills and an area. So for us to be able to share and say have at it. Here's the link. That's pretty powerful. And so one of the things I'm planning to do is if we have people will say for example, in the label athon you may be labeling your issues, maybe somebody people in the labelathon, become aware of it. We're sort of building a community of practice where someone else might want to work on something that you've labeled in that same labelathon if rOpenSci becomes was sort of known for, hey, yeah, we have these. We have Help Wanted issues, we may be able to draw people to actually address it. But I love how good this question is least because nobody has come up with this specific example. I see a raised hand Wayne, but I have a bunch of questions in the doc. So I'm going to go to those. And I'll ask, I'm also going to ask all of the speakers to take a look at the questions that are in the doc and take some time after the call to please answer any questions that you're able to answer there. This doc will

stay available as a resource.

Kathryn, you had a question or two? And I'm wondering if you would like to unmute and ask.

Kathryn Doering 50:50
Can you hear me? Okay,

Stefanie Butland 50:51
I can.

Kathryn Doering 50:52
Excellent. Yeah. So

Lluís Revilla Sancho 50:54
I was just curious, I guess for the packages that I helped maintain. We have a lot of users that aren't super comfortable using version control and GitHub. And so I was curious if anyone had ideas about how to work with them if they want to contribute. So what we've done, I guess, is allow people to like email us code or try to be flexible in terms of workflow. But sometimes it can be kind of tricky figuring out what a person's comfort level is. So just curious if anyone else has ideas about how to deal with that situation.

Sam Albers 51:37
I mean, I guess I also get a lot of questions and issues raised through email simply because my user base and like you said, isn't really on GitHub. I'm never really gotten contributions. But I think I think just in general, this idea of, of being flexible, like it sounds like what you're doing is like, it's a little awkward, but understanding that we accept a bit of awkwardness in an effort to to get more contributions. I'm blanking on the name.

Steffi LaZerte 52:09
And

Sam Albers 52:10
I feel like it was possibly, no, I think it was Mark, actually. So there is a web service that allows you to collaboratively edit, like markdown documents and things like that. So even just like working on collaborative docs or not necessarily coding docs, like even something like Google doc or Dropbox. Like I think that will work. I do do a lot of collaboration through Dropbox. So sorry, I don't know that that was actually helpful for

Stefanie Butland 52:38
answer. I'm wondering if I don't know if Hugo, Hugo or Maëlle have their eyes in the document. But is there a question that you particularly would like to answer

Maëlle Salmon 52:53
that, but I just want to do two servings to get skills, but it's in general, like people don't? Like not knowing their level, like, it's important to make them feel that you are okay with there being a real beginners and offering resources like, you know, being but that's sort of repeat what Steffi said.

Stefanie Butland 53:21
Hugo, is there anything that you particularly want to

address? Otherwise?

Hugo Gruson 53:29
nothing specific.

Stefanie Butland 53:30
Okay. Um,

Sam Albers 53:32
can I just have one? Sorry. I just thought that discussions might be GitHub as that new feature of discussions rolled out. So like, getting someone to log on to or to set up a GitHub account is probably within most people's they'll set. And then you just hop on to discussions, which is sort of a slightly less informal spot, and I know, we'll has it in the targets repo. And it's super useful, because like, not everything merits an issue. And so it's a little bit more of a free flowing conversation. And so that might lower the barrier for intimidation as the GitHub discussions. Thanks, Sam.

Stefanie Butland 54:17
Silvia, do you want to ask one last question, because you've got a couple in there.

Silvia Canelón 54:27
Sure, let me just scroll up. Okay. So my question was, I've seen at least one package where, if you would like to contribute, you're asked to sign a contributor license agreement or a CLA and I was wondering if anybody knows when this is recommended or necessary.

Stefanie Butland 55:00
Okay, does anyone have a strong answer to this question?

Lluís Revilla Sancho 55:08
I have actually set up CLA for one of my repositories. Because it was easy. And it was one of the first things I so I'm trying some automatic setup to to collaborate. But I don't know if it's actually useful. I've seen some discussion about it around ggplot2, with the new relicensing that's going on. And they I think they have something some discussions about it. I don't have a strong counter for that.

Unknown Speaker 55:42
I would, right?

Stefanie Butland 55:43
Yes. Thank you for the add that example into the doc,

Elin Waring 55:47
because I make a comment about that to you. Elin yeah, so I actually like created a CLA for not for our project. We're big, like a big, massive open source project. And the reason we did that, like we had no CLA for a really long time. And the problem is, when you have a truly collaborative project, it's people can actually, if you, if you don't have an agreement, people can come back in a year and say, I'd like my code back. And so if you, you can make a CLA that

doesn't require them to transfer their copyright or to do anything except to promise that they will not come and ask you to remove their code. And it's, it's really, you know, unless someone I mean, if you had to go to court, unless someone is coding a massive new feature for you, as opposed to just modify, like just modifying your files, they're still your file, you still own the copyright to those files. They're just like editing. But if they code a new feature for you, it's their copyright. So I mean, this is in the US, it's different in different countries, which is another reason, like, if you like for GG, if I was RStudio, I would definitely have a CLA because there, there were company, people could sue them, they do have deep pockets. And they have like a huge community that depends on what they do, which is why it's often hard to get code it like it's really hard to get code into WordPress, for example, because even though it's open source, if you don't work for automatic, that it just opens up a whole big can of worms for them. So it's like a similar kind of thing. So personally, I don't think most people need a CLA for our packages, but it's probably not. I mean, you should read I wouldn't sign something without reading it. But most of the time for small projects, they don't, they don't really mean that much.

Stefanie Butland 57:55
Thank you very much Elin. I so much appreciate your attendance and your chiming in here today.

Athanasia Mowinckel 58:01
And I ask I know, I know we're running out of time. Yeah, I had a small follow up to that I'm sorry. So with would that still apply? Even if like your license for the entire repository, clearly states? Like the like, if the entire repository is CC by for instance, wouldn't a contributing code also automatically fall under that?

Elin Waring 58:31
No, because you you own your copyright. So you could say I no longer want my code under cc but so they could you they couldn't like take that anybody who's already received your code, they can't You can't demand they get it back. But you can say stop giving it out. So and CC by by the way is a terrible license code. But GPL or MIT is better. But I have a CC by packet. I'm just gonna say I'm a CC by package because it's 100% data package. And so it makes that's more like content. You know, it's more like written as opposed to

Stefanie Butland 59:15
thank you so much, everybody. I hate to break off the discussion. There. There are lots of questions that I in the doc and I love that people are adding their own answers. That's where this big value comes from. I'm going to leave that doc open for editing for at least another 24 hours and then I'll make it comment only status. It will be linked from the meet the landing page of this community call. I will be adding the video to that page. The video will be posted with closed captions blah blah blah. So I want to close by thanking so much our presenters Maëlle, Hugo, and Steffi, a particular shout out to Hugo because they Keeps it, Hugo, because he has contributed to rOpenSci across so many different axes, you know, ideas and topics and Maëlle is great and receptive and incorporating these things. And so Hugo, shout out to you, and then

especially to Lluis, who's given a lot of comments and questions and feedback and resources on this as well to help us put this thing together. So let me see the labelathon. The first labelathon version one happens at the same time as the community call next Thursday. I love that people are commenting on what times will work better for them. We want to do more of these sort of participatory get together things that aren't very long, help people get together and do the things that end up at the bottom of their priority list. So we will be scheduling more, we'll be doing four and then more if people keep showing up and it works for you and it works for us. Our next community call is may 25. It will be Jeroen Ooms talking about the rOpenSci, R-universe project. You can see all the topics we plan to cover in future community calls. We have a public community call GitHub repository. And it's linked at the bottom of this doc here. So please go there emoji vote your interests, chime in and make comments because you really do influence what it is we talk about who presents etc. So thank you all so much for your attention today. This has been such a pleasure and a true sort of community community call. So on behalf of rOpenSci. I wish you all a happy day or night wherever you are in the world.

Bye everybody.