Table of contents

Contributing guidelines

This document serves as a starting point to know what and how you might contribute to this project. It starts with covering the most common thing you will encounter as a contributor: Not knowing where to start or what to work on. This includes us too, since this project is largely a side project we work on.

Before getting further into the document, there are a few things you will need to know how to use in order to contribute.

- At a *minimum*, you need to know how to use the GitHub interface for contributing. If you are just starting out, check out the GitHub's Quickstart. Specifically, read these:
  - Get a quick overview of using GitHub with their "Hello World" guide.
  - We follow a popular style of contributing called the "branch-based workflow", described in the guide GitHub flow.
  - We use GitHub for many aspects of communication, described in the guide Communicating on GitHub.
- For larger contributions, we strongly recommend using RStudio with Git (connected to GitHub) to write text or code. More details and resources on this are later in this guide.

Not sure where to start?

Because this is a side project for us, we often work on this maybe once or twice a month. So we often forget where we are, what's been done so far, what's the status of the project, and what we should work on next. So let's break it down into two topics: What's been done so far and what to work on.

Checking what's been done so far

Newly entering or returning to a project after some time is always a bit disorienting. So, if you are in this situation or feel a bit overwhelmed or confused about what's been done or what the status of the project is, review this documents in this order:

1. ROADMAP: This document is the first place to look to see what's been done and what the next steps are. It can help get a sense of where to focus your efforts and time.
2. Commit history of the `main` branch: This will show a list of files changed and messages on what people have done to the project. This is the current state of the project.
3. Open Pull Requests: To see what people are current working on, so you can see what to *not* work on since someone is already doing it.
4. Closed Pull Requests: While you can technically see this in the Commit History, checking the Closed Pull Requests gives a brief overview of what was recently done or worked on.

What to work on next

If you've looked over what's been done so far and especially the ROADMAP, and still feel a bit unsure what to do work on, then follow these steps as a starting point. See the Workflow section below for instructions on how we make changes to files.

1. Review (proofread, edit, revise) in order:
   - `README.md`
   - `CONTRIBUTING.md`
   - `ROADMAP.md`
2. Read over the `doc/protocol.Rmd` file.

Types of contributions

We appreciate ALL types of contributions! Contributions include:

- Adding to discussions on issues or in the Discord server
- Reviewing and, if appropriate, approving open PRs
- Proofreading and fixing typos/grammar in text
- Reviewing code, running it yourself, and fixing bugs/reformatting, adding code comments
- Adding larger amounts of new text or code

Workflow to add text or code

There are two suggested ways of adding text or code: Through the GitHub interface or through RStudio. If you are adding code, we definitely recommend using the RStudio way.

For things like fixing typos, spelling mistakes, or grammatical errors in the documentation, you can edit directly in the GitHub web interface.

For bigger changes, it's a good idea to first file an issue and make sure someone from the team agrees that it's needed.

If you are adding more substantial text or code, this is the process you would use:

- If you are not a member of the Science Collective GitHub organization, you will need to fork the project to your own account. If you are a member, you don't need to fork. Then clone the Git repository onto your computer. If you haven't done this before, we recommend using R and RStudio (with the `{usethis}` package):
  - For non-members: `usethis::create_from_github("science-collective/scoping-review", fork = TRUE)`
  - For members: `usethis::create_from_github("science-collective/scoping-review")`
- Install the project package dependencies, while inside the RStudio R Project (`scoping-review.Rproj`) and in the Console, with:

  ```r
  # install.packages("renv")
  renv::restore()
  ```

  Note: If you ever get frustrated with using renv, you can turn it off with `renv::deactivate()`.

- Create a Git branch for your pull request (PR). We recommend using `usethis::pr_init("brief-description-of-change")` (replace the `"brief-description-of-change"` with your own text). For more details on using the usethis PR helpers, read this webpage.
- If you need to use another package, add the package dependency by using:

  ```r
  use_package("PACKAGENAME")
  ```

- Make your changes, commit to git, and then create a Pull Request by running `usethis::pr_push()`, and following the prompts in your browser. The title of your PR should briefly describe the change. The body of your PR should contain `Fixes #issuenumber` if you were working on a specific Issue.

Installing or updating necessary packages

There are several R package dependencies to this project. Install them by running this code in the Console while in the R Project:

```r
# install.packages("renv")
renv::restore()
```

If you are on Linux, you might need to install some additional Linux packages:

```bash
sudo apt install libcurl4-openssl-dev libxml2 \
    libxml2-dev libtiff-dev libjpeg-dev libpng-dev
```

Re-building the sources

To rebuild all the sources extracted from the different databases like Zenodo or PubMed, run this:

```r
targets::tar_make()
```

This will run a pipeline to re-build and re-run all the code associated with the scoping review.

Code of Conduct

Please note that the scoping-review project is released with a Contributor Code of Conduct. By contributing to this project you agree to abide by its terms.

Bibliography