One more month, one more R code club session! This time we gather together 12 R nerds to talk about cool R stuff.

Measuring the citation impact of our data systems

We keep track of the citations

We want to illustrate the impact of the citations that use the VLIZ data systems. We started this month with a brainstorm, with people from different departments: data center policy library

The second brainstorm we want to create a shiny app in which we can select the data system where we show a map with the citations, disciplines in a horizontal barchar. Table with a summary. Add sections for filters.

By creating a new project you can connect to gitlab

We are looking at packages that look at citations to get the impact of the citations etc

We should look at prices for extended api. How to make a relative citation index chart

How to make your own shiny app

Our colleague Lennert S explained the logic behind the shiny apps aplications. Shiny apps require to define inputs and outputs that will be shown, while this content is edited in the server side of the application. Shiny apps require the server executing to run R.

Some examples can be seen with the shiny package: shiny-examples

R shiny applications are reactive: the changes made by the user produces the server to recalculate the outputs

An example was made by our colleague jonas m in order to access a database of flies!

There is a website shinyapps.io that allows to upload a shiny app for free but only 25h a month

Best practices when uploading shiny apps in VLIZ servers

Our colleague Paul from the IT department gave us some insights in best practice to create projects from a Software engineering point of view.

Following documentation

Copy and modify existing projects

Use tooling read R packages book

Read ackages.skeleton() is cool but the template is fixed, customizing is not allowed. An alternative is cookiecutter, for projects templating.

Source or version control: GIT

Git is a version control system. Allows to collaborate with other people. This allows to see what changes where made or recover a version that works if you messed up. Most important, you have a backup of your work.

Work space – start git – cre…

What to commit: app code, test, non-generated documentation, maybe some data files. Avoid generated files, binary files, large data files (more than 1MB), sensitive information like passwords

Ignore: with this we make sure we don’t accidentally add files that we do not want in git.

Commit messages: explain shortly what the commit does.

Branching: create different versions of your code at a certain version.

Dependency management: 1 way: the dependencies are in a central system. The best way from the institution point of view. 2 way: do it by environment, it is more flexible

Testing: write functions to test out

Environment variables: credentials go here.

Continuous integration: we want to build our code in a way that changes form everybody come together.

He showed a demo!!

The IT departments aims to show us good practices for creating good projects.

Newsletter with RMarkdown