



Development workflow at ThinkR

PROPRE, DataOps and DevOps workflows

Who am I?

Sébastien ROCHETTE

Data Scientist,  expert,  trainer

- <https://statnmap.com>
- <https://thinkr.fr> (teaching)
- <https://rtask.thinkr.fr> (consulting)
- <https://github.com/ThinkR-open>
- https://twitter.com/thinkr_fr



Reproducible Analytical Pipelines

Share pipelines for others to reproduce outputs

Named PROPRE in French ("PROcessus de Publications REproductibles")

- Allows productions to be reproducible
- Same idea as sharing analyses through publications
- Makes analyses accessible
- Gives users confidence in the production
- Efficiency of workflows with "click" reduction



Note that pipelines can be citable (DOI - [Zenodo](#))

More about PROPRE (FR): https://rdes_dreal.gitlab.io/propre/index.html

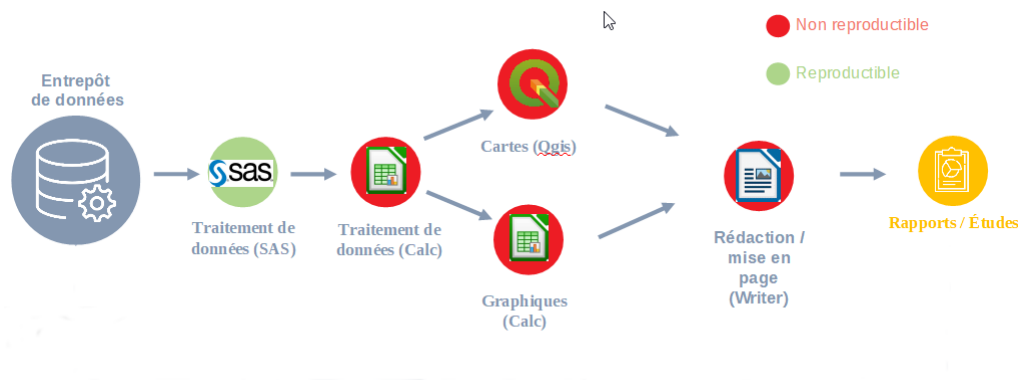
ThinkR uses RAP

Private and open-source projects

- Clients/Users can take over the development after delivery
- Verify if it works as you asked, in every details
- Document features with words for non-coders
- Robust to new fonctionnalités
- Re-usable for new datasets or updates

From data to analyses

An integrated workflow



From data to analyses

An integrated workflow

- Everything is code
- Avoid user guides explaining where to click
 - Give sense to human tasks
- Anyone can launch the process
 - Trainees, PhDs, new collaborators, ...
- Give confidence in the outputs, increase potential use
- Requires stability in the data format and content

Use Case: Process PROPRE

Coaching of a public organization

- Yearly analysis of public data
 - Regional specificities
- DataOps / DevOps approach
 - Data producers
 - R developers
 - Data analysts
 - Users, decision-makers
- Open-source: https://gitlab.com/rdes_dreal/propre.rpls



ThinkR methodology

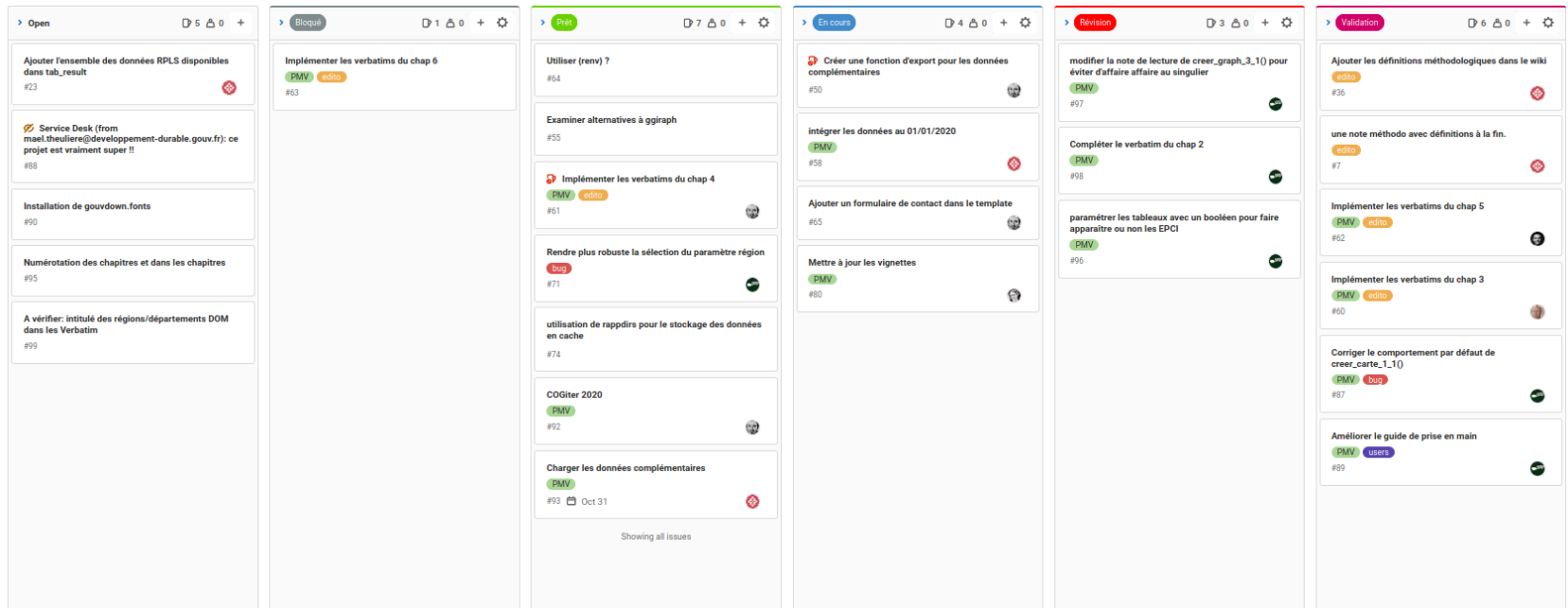
DevOps steps for production

Step	Dev	Ops
1. Infrastructure	Versionning Communication Monitoring	Collaboration
2. Design - UI	Propose output appearance HTML, PDF, Shiny	Define Validate
3. Prototype	Independent core developments Analysis, Graphs, Tables Documentation	Validate each output
4. Build	Combine prototypes and UI	Validate pages / sections
5. Strengthen	Reproducible examples Version control Unit tests Docker	Validate tests
6. Deploy	Send in production	Test complete outputs
Repeat 3:6	Develop, document, test, propose	Test, feedbacks, validation

ThinkR methodology

1. Infrastructure : set collaboration

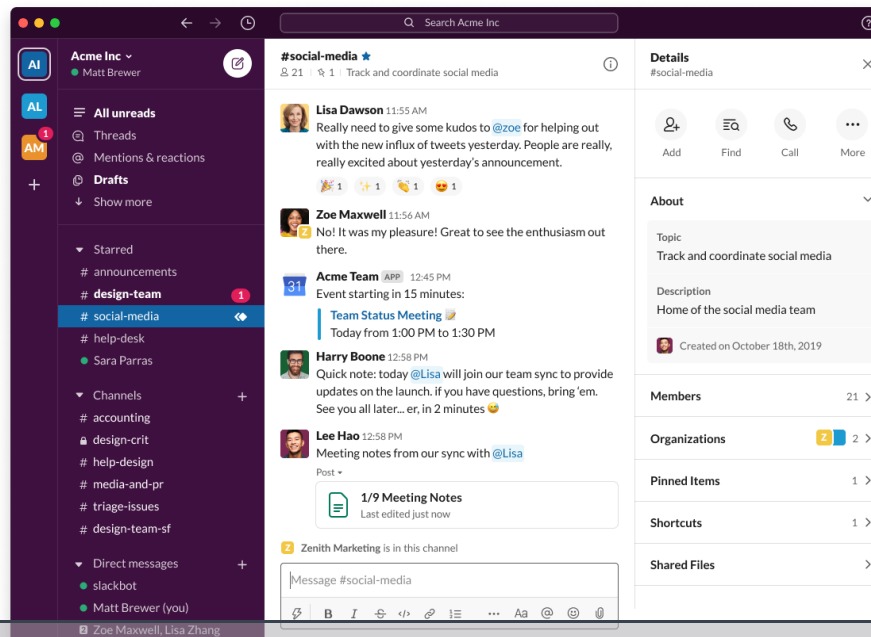
- Roles from data to users
- *git* & GitLab project ([RPLS](#))
- Kanban project monitoring ([RPLS](#))



ThinkR methodology

1. Infrastructure : set collaboration

- Roles from data to users
- *git* & GitLab project (RPLS)
- Kanban project monitoring (RPLS)
- Asynchronous communication (chat, issues)
- Meetings



2. Design - UI : define output formats

Questions

- Who is the public / target?
- What do they do with the outputs?
- What do they need?
 - Text, figures, tables
 - Interactivity
 - Download
 - Print, presentation
 - Monthly, yearly updates
- How important is the appearance vs content?
- How will they participate in the content?
 - Tests
 - Proof reading
 - Texts and analyses
 - Word, markdown, git

Propositions

- Data analysis reports
 - HTML page, HTML book, website, PDF document, ...
- Shiny application
 - One page, Multiple pages, Dashboard, ...

ThinkR methodology

2. Design - UI : define output formats

Data analysis reports with random text and images

Title of the report

A template for PDF reports

STATNMAP, THINKR

ThinkR

ThinkR
R report

Created on September 29, 2018

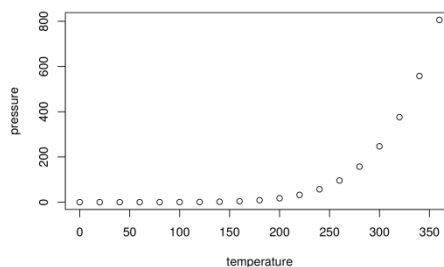


Figure 1: Pressure of cars

Contents

1 Section 1	1
1.1 R Markdown	1
1.2 Including Plots	1
1.3 A few extra	2

1. Section 1

1.1. R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed      dist
##  Min.   : 4.0    Min.   : 2.00
##  1st Qu.:12.0    1st Qu.: 26.00
##  Median :15.0    Median : 36.00
##  Mean   :15.4    Mean   : 42.98
##  3rd Qu.:19.0    3rd Qu.: 56.00
##  Max.   :25.0    Max.   :120.00
```

1.2. Including Plots

You can also embed plots with reference and caption as in Fig. 1.

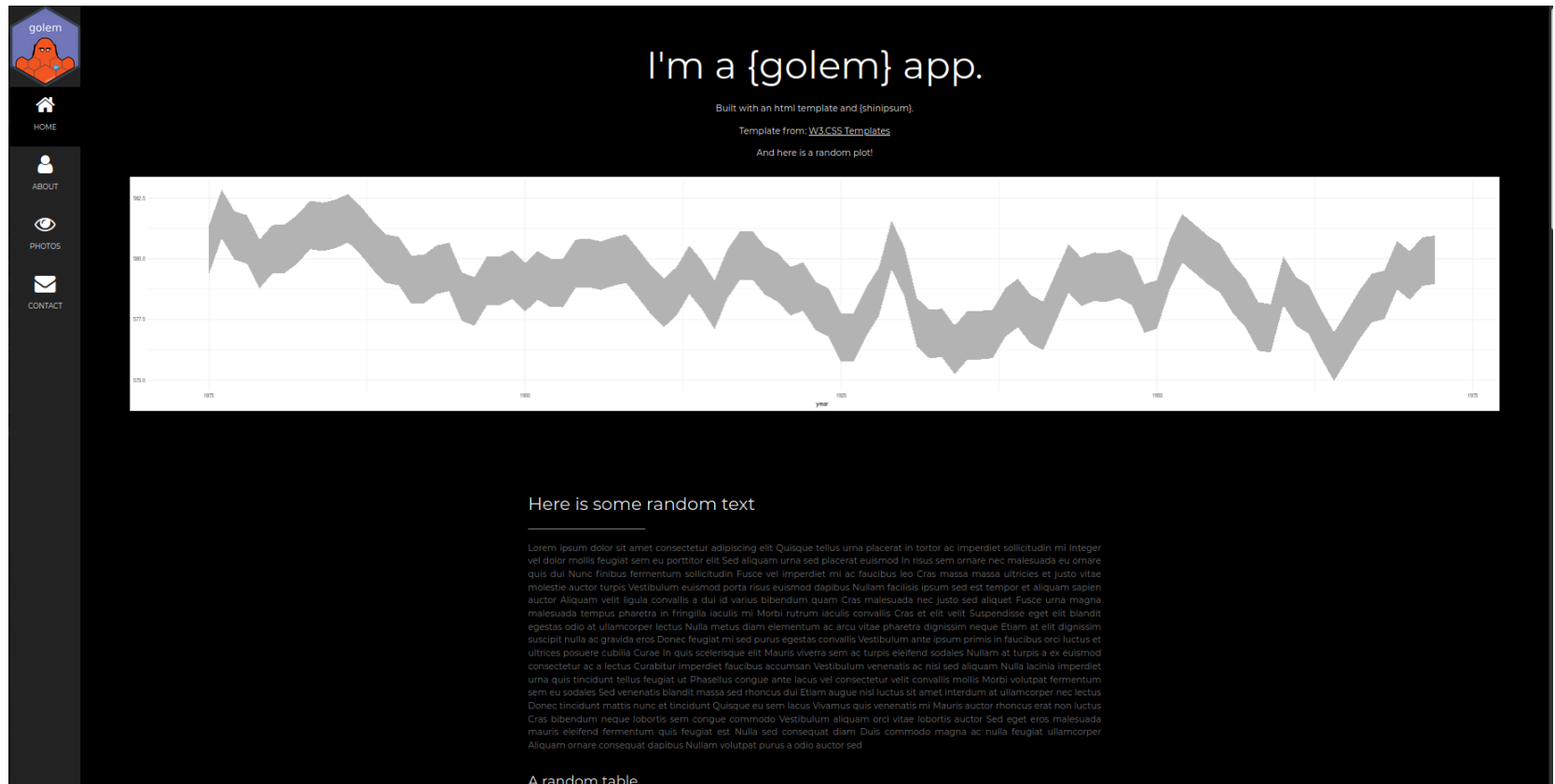
StatnMap

sebastien@thinkr.fr

1 / 2

2. Design - UI : define output formats

Shiny dashboard with random text and images



ThinkR methodology

3. Prototype - Strengthen - R package

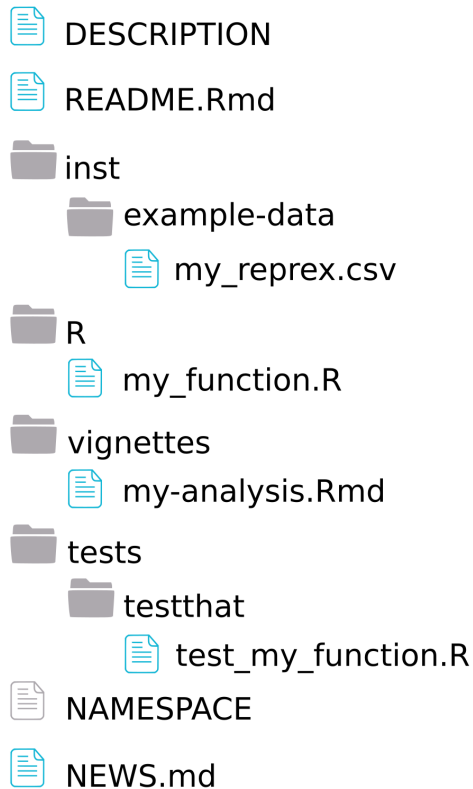
Documentation & tests: the heart of R packages



ThinkR methodology

3. Prototype - Strengthen - R package

Where do we document?



- **DESCRIPTION**: What? Who? How?
- **README.Rmd** : Presented on [GitLab](https://thinkr.fr). What? How to install? How to use (reproducible example) ?
- **R/** : Fonctions with documentation for each parameter and reproducible examples
- **vignettes/** : Developer / User guides with plain text and reproducible examples
- **NEWS.md** : List of main modifications between versions

ThinkR methodology

3. Prototype - Strengthen - R package

How do we document vignettes? "Rmd first" development

What we write

```
my-analysis.Rmd | DATASET.R
16 Load all necessary package for the analysis
17 ```{r, message=FALSE}
18 library(userproject)
19 library(dplyr)
20 library(ggplot2)
21 library(here)
22 library(readr)
23 ```
24
25 ## Read a client database
26
27 Dataset has been built using package {fakir} available on Github with
  'remotes::install_github("ThinkR-open/fakir")'. It is saved as a CSV file
  in this project.
28
29 ```{r, message=FALSE}
30 dataset_path <- system.file("example-data/clients.csv", package =
  "userproject")
31 clients <- read_csv(dataset_path)
32 ```
33
34 ## Table by department
35
36 - Create a function to filter on a particular department
37
38 ```{r}
39 filter_by_dpt <- function(x, dpt) {
40   filter(x, id_dpt == dpt)
41 }
42 # Examples
43 filter_by_dpt(clients, dpt = 11)
44 ```
45
46 - Write a test for function 'filter_by_dpt'
```

What you read

Load packages

Load all necessary package for the analysis

```
library(userproject)
library(dplyr)
library(ggplot2)
library(here)
library(readr)
```

Read a client database

Dataset has been built using package {fakir} available on Github with `remotes::install_github("ThinkR-open/fakir")`. It is saved as a CSV file in this project.

```
dataset_path <- system.file("example-data/clients.csv", package = "userproject")
clients <- read_csv(dataset_path)
```

Table by department

- Create a function to filter on a particular department

```
filter_by_dpt <- function(x, dpt) {
  filter(x, id_dpt == dpt)
}
# Examples
filter_by_dpt(clients, dpt = 11)
```

```
## # A tibble: 10 x 16
##   num_client first last job age region id_dpt departement cb_provider
##   <dbl> <chr> <chr> <chr> <dbl> <chr> <chr> <chr> <chr>
## 1 95 Cora_ Glei_ Hort... NA Langu... 11 Aude VISA 13 di_
## 2 108 Jay Hane_ Seis... 22 Langu... 11 Aude Maestro
## 3 112 Dana Dool_ Scie... 29 Langu... 11 Aude JCB 15 dig_
## 4 232 Marc_ Roma_ Conf... 19 Langu... 11 Aude Discover
## 5 266 Mrs. KiaM_ Air ... 22 Langu... 11 Aude JCB 15 dig_
## 6 303 Demo_ Will_ Admi... 18 Langu... 11 Aude VISA 16 di_
## 7 390 Mira_ Jerde Acco... 40 <NA> 11 Aude Discover
## 8 433 Neve_ Ledn_ Prod...
```


ThinkR methodology

3. Prototype - Strengthen - R package

How do we document functions? {roxygen2} and reproducible example

What we write

```
my-analysis.Rmd x dev_history.R x filter_by_dpt.R x
Source on Save Run
1 #' Filter by department
2 #'
3 #' @param x dataframe with column named id_dpt
4 #' @param dpt department number
5 #'
6 #' @return a filtered dataframe
7 #'
8 #' @importFrom dplyr filter
9 #' @export
10 #'
11 #' @examples
12 #' dataset_path <- system.file("example-data/clients.csv",
13 #'   package = "userproject")
14 #' clients <- readr::read_csv(dataset_path)
15 #' filter_by_dpt(clients, dpt = 11)
16 filter_by_dpt <- function(x, dpt) {
17   filter(x, id_dpt == dpt)
18 }
```

What you read

Filter by department

Filter by department

```
filter_by_dpt(x, dpt)
```

Arguments

x dataframe with column named id_dpt

dpt department number

Value

a filtered dataframe

Examples

```
dataset_path <- system.file("example-data/clients.csv",
  package = "userproject")
clients <- readr::read_csv(dataset_path, col_types = "dcccccccccTddcdc")
filter_by_dpt(clients, dpt = 11)
#> # A tibble: 10 x 16
#>   num_client first last job age region id_dpt departement cb_provider
#>   <dbl> <chr> <chr> <chr> <dbl> <chr> <chr> <chr> <chr>
#> 1 95 Cora_ Glei_ Hort_ NA Langu.. 11 Aude VISA 13 di...
#> 2 108 Jay Hane_ Seis_ 22 Langu.. 11 Aude Maestro
#> 3 112 Dana Dool_ Scie_ 29 Langu.. 11 Aude JCB 15 dig...
#> 4 232 Marc_ Roma_ Conf_ 19 Langu.. 11 Aude Discover
#> 5 266 Mrs. KiaM_ Air _ 22 Langu.. 11 Aude JCB 15 dig...
#> 6 303 Demo_ Will_ Admi_ 18 Langu.. 11 Aude VISA 16 di...
#> 7 390 Mira_ Jerde Acco_ 40 NA 11 Aude Discover
#> 8 433 Neve_ Ledn_ Prod_ 26 Langu.. 11 Aude Discover
#> 9 224 Desh_ Hodk_ Scie_ NA Langu.. 11 Aude Maestro
#> 10 371 Darr_ Terr_ Educ_ 25 Langu.. 11 Aude Mastercard
#> # ... with 7 more variables: name <chr>, entry_date <dtm>,
#> # fidelity_points <dbl>, priority_encoded <dbl>, priority <chr>,
#> # entry_year <dbl>, age_class <chr>
```

Contents

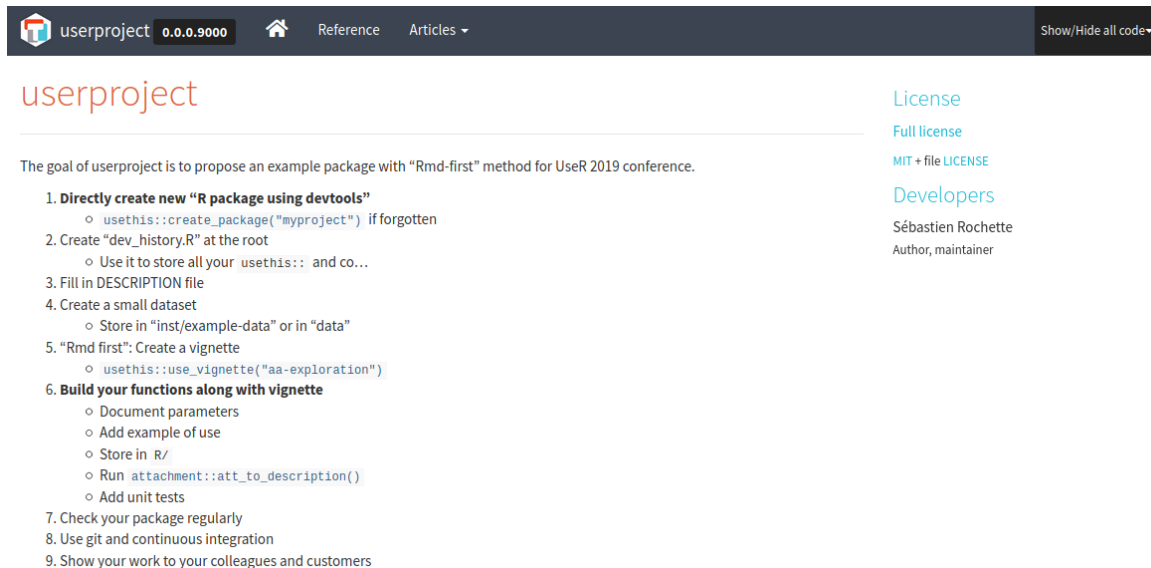
Arguments

Value

Examples

3. Prototype - Strengthen - R package

How do we share documentation? {pkgdown} website



userproject 0.0.0.9000 Reference Articles Show/Hide all code

userproject

The goal of userproject is to propose an example package with "Rmd-first" method for UseR 2019 conference.

1. **Directly create new "R package using devtools"**
 - `usethis::create_package("myproject")` if forgotten
2. Create "dev_history.R" at the root
 - Use it to store all your `usethis::` and co...
3. Fill in DESCRIPTION file
4. Create a small dataset
 - Store in "inst/example-data" or in "data"
5. "Rmd first": Create a vignette
 - `usethis::use_vignette("aa-exploration")`
6. **Build your functions along with vignette**
 - Document parameters
 - Add example of use
 - Store in R/
 - Run `attachment::att_to_description()`
 - Add unit tests
7. Check your package regularly
8. Use git and continuous integration
9. Show your work to your colleagues and customers

[License](#)
[Full license](#)
[MIT + file LICENSE](#)
[Developers](#)
Sébastien Rochette
Author, maintainer

3. Prototype - Strengthen - R package

How do we share documentation? {pkgdown} website

- 1 topic = 1 vignette = 1 validation (e.g. [RPLS](#))

The screenshot shows the pkgdown website for the `propre.rpls` package. The top navigation bar includes links for 'propre.rpls 0.2.0', a home icon, 'Reference', 'Articles', and 'Changelog'. The main content area is divided into several sections:

- propre.rpls**: A section with a status bar showing 'dev: pipeline running coverage 99.00%'. Below it, it states the website presentation and provides links for the master version (<https://rpls.thinkr.fr/>), the development version, and the common code coverage.
- Utilisation**: A section with a list of instructions for creating a new Rstudio project, selecting parameters, completing author and date fields, optionally adding a list of episodes, launching the compilation, and integrating comments and analyses.
- Articles**: A dropdown menu listing various articles, including 'Prise en main', 'Preparation des donnees', 'Caption des illustrations', 'Discretisation de variables numeriques', and several chapters (Chapitre 1 to 6) covering topics like Tableau, Graphe, Carte, Verbatim, and Creation du projet de publication.
- License**: A section with a 'Full license' link and a 'MIT + file LICENSE' link.
- Developers**: A list of authors, including Fabio Dos Santos Pereira, Juliette Engelaere-Lefebvre, Daniel Kalioudjoglou, Murielle Lethrosne, Mael Theuliere, Marouane Zellou, and others.

Plus de détail sur la prise en main du package au niveau de la vignette "Prise en main"

ThinkR methodology

3. Prototype - Strengthen - R package

How do we document the development process? **ThinkR** tip

Classical package (RPLS)

dev_history.R

```
# Hide dev_history from build
usethis::use_build_ignore("dev_history.R")

# Add a raw dataset
usethis::use_data_raw()

# Licence
usethis::use_mit_license("ThinkR")

# Use git
usethis::use_git_ignore("*.Rproj")
usethis::use_git()
usethis::use_readme_md()

# Create Rmd file
usethis::use_readme_rmd("my-analysis")
```

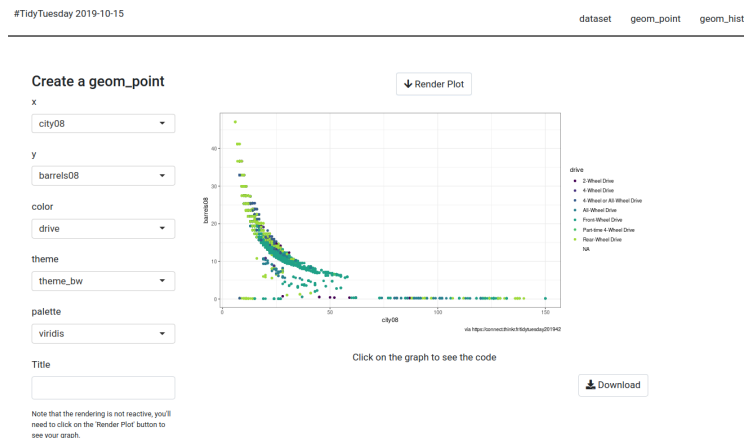
{golem} package ({tidytuesday201942})

```
#> dev
#> └─ 01_start.R
#> └─ 02_dev.R
#> └─ 03_deploy.R
#> └─ deliverables.R
#> └─ run_dev.R

## Fill the DESCRIPTION ----
## Add meta data about your application
golem::fill_desc(
  pkg_name = "my.shiny.app", # The Name
  of the package containing the App
  pkg_title = "The Shiny App", # The
  Title of the package containing the App
  pkg_description = "A Shiny application
  to explore data.", # The Description of
  the package containing the App
```

4. Build - Produce desired output

- Data analysis reports
 - Include validated functions in the desired report
 - Present completed document for validation (e.g. [Engineering Production-Grade Shiny Apps](#))
- Shiny applications
 - Include validated functions in the backend
 - Link backend with frontend
 - Present the interface on our platform (e.g. [tidytuesday201942](#))



ThinkR methodology

5. Strengthen - Test robustness - Unit tests

- Test that each function returns what it is supposed to (RPLS)
- Test that new functionalities do not break earlier ones

What we write

```
test_that("dataprep fonctionne", {  
  
  indicateurs_rpls <-  
  dataprep(nom_reg="Pays de la Loire",  
    epci_list = c("244400404", "244400644"))  
  
  testthat::expect_is(indicateurs_rpls,  
    "data.frame")  
  testthat::expect_true(ncol(indicateurs_rpls)  
    == ncol(tab_result) + 1 + 1 + 2)  
})
```

What you read

propre.rpls coverage - 99.31%

Files		R/creer_verbatim_2.R					
File	Lines	Relevant	Covered	Missed	Hits / Line	Coverage	
R/creer_verbatim_2.R	94	54	54	0	3	100.00%	
R/creer_tableau_3_1.R	105	54	54	0	1	100.00%	
R/creer_tableau_1_1.R	84	40	40	0	1	100.00%	
R/creer_tableau_4_1.R	83	38	38	0	1	100.00%	
R/creer_graphe_3_1.R	86	37	37	0	1	100.00%	
R/creer_graphe_3_2.R	78	35	35	0	1	100.00%	
R/creer_graphe_6_1.R	85	32	32	0	1	100.00%	
R/creer_verbatim_1.R	73	32	32	0	1	100.00%	
R/creer_graphe_6_2.R	76	32	32	0	1	100.00%	
R/creer_graphe_2_1.R	78	32	32	0	1	100.00%	
R/creer_tableau_5_1.R	80	31	31	0	1	100.00%	

ThinkR methodology

5. Strengthen - Test robustness - Continuous integration

- Docker container for automatic check
- Docker container for development with RStudio server and {renv} : [{devindocker}](#)

What we write

```
usethis::use_gitlab_ci()
usethis::use_github_actions()
```

```
image: rocker/geospatial
```

stages:

- build
- test
- pkgdown
- pkgdown-move
- deploy

building:

```
stage: build
```






















script:

```
- Rscript -e 'remotes::install_deps(dependencies = TRUE)'
```

```
- Rscript -e 'devtools::check()'
```

Development workflow at ThinkR

What you read

Status	Pipeline	Triggerer	Commit	Stages
 passed	#203712532 latest		P' 96-param-t... -> b402fce6 maj du news.md	 00:16:23 1 day ago
 passed	#203711429		P' 96-param-t... -> ad449bc1 Ajout du parametre epci dans L...	 00:15:36 1 day ago
 passed	#203691863		P' 96-param-t... -> 6003f21e Update NAMESPACE - suppre...	 00:16:37 1 day ago
 passed	#203685421 latest		P' 97-modifie... -> e272d878 Update cb-ch3-verbatim.Rmd ...	 00:14:28 1 day ago
 passed	#203684713 latest		P' 98-complet... -> e7d9c9f9 correction du verbatim du cha...	 00:16:24 1 day ago
 passed	#203678829 latest		P' dev -> 6003f21e Update NAMESPACE - suppre...	    00:46:52 1 day ago

6. Deploy - Send in production

- Data analysis reports
 - Deliver final report
 - Possibility to deliver proofs of quality: `{testdown}`, `{gitdown}`, `{pkgdown}`
- Shiny applications
 - Deliver Docker container
 - Possibility to deliver production steps
 - Possibility to install on your server

DevOps steps for production

Reminder of clients / users mission

Step	Dev	Ops
1. Infrastructure	Versionning Communication Monitoring	Collaboration
2. Design - UI	Propose output appearance HTML, PDF, Shiny	Define Validate
3. Prototype	Independent core developments Analysis, Graphs, Tables Documentation	Validate each output
4. Build	Combine prototypes and UI	Validate pages / sections
5. Strengthen	Reproducible examples Version control Unit tests Docker	Validate tests
6. Deploy	Send in production	Test complete outputs
Repeat 3:6	Develop, document, test, propose	Test, feedbacks, validation