Rozwój oprogramowania w R: dokumentacja i testy

Mateusz Staniak

Rozwój oprogramowania w R

Instytut Matematyczny UWr, semestr letni 2020

Uniwersytet Wrocławski

Dokumentacja pakietu

Typy dokumentacji

Uniwersytet Wrocławski

Odbiorca:

1. Programista
2. Użytkownik

Cel:

	Tutorials	How-to guides	Reference	Explanation
oriented to	learning	a goal	information	understanding
must	allow the newcomer to get started	show how to solve a specific problem	describe the machinery	explain
its form	a lesson	a series of steps	dry description	discursive explanation
analogy	teaching a small child how to cook	a recipe in a cookery book	a reference encyclopaedia article	an article on culinary social history

https://documentation.divio.com/introduction/

Dokumentacja pakietów

Uniwersytet Wrocławski

Wewnętrzna:

- Dokumentacja obiektów (*?filter*) - krótka forma dokumentacji
- Winietki - długa forma dokumentacji

Zewnętrzna:

- Strony pkgdown (używają dokumentacji wewnętrznej)
- Tutoriale / przewodniki (np. https://mlr.mlr-org.com/, https://msstats.org)

Jak dokumentować pakiety w R?

Uniwersytet Wrocławski

1. Podstawowe narzędzie: pakiet roxygen2
2. Roxygen comment: rozpoczyna linię dokumentacji - #'
3. Roxygen tag: @ - określa sekcję w dokumentacji
4. devtools::document() zamienia komentarze roxygen2 na dokumentację (w folderze *man/*)
5. Obiekty do dokumentowania:
   - cały pakiet (?`dplyr-package`, ?dplyr),
   - funkcje (w tym metody),
   - klasy (S3, S4, R6) - na zajęciach na temat OOP,
   - zbiory danych.

Funkcje

Uniwersytet Wrocławski

- Dokumentacja funkcji:
  1. Tytuł
  2. Krótki opis
  3. @details - szczegóły
  4. @param – opisy parametrów funkcji
  5. @return – opis obiektu zwracanego przez funkcję
  6. @examples – linijki przykładowego kodu (ew. +\donotrun{})

Dla zwięzłości:

- @inheritParams - powtórzenie parametrów z innej funkcji
- @rdname, @describeIn – kilka funkcji w jednym miejscu
- @seealso, @family - powiązane funkcje (linki w dokumentacji)

Pakiet

Uniwersytet Wrocławski

Dokumentacja całego pakietu:

- np. w pliku nazwapakietu.R można umieścić dokumentację dla całego pakietu (?nazwapakietu). Pomocne tagi:
  - @section – akapit
  - @docType package
  - @name nazwapakietu
- Pod dokumentacją należy umieścić *NULL*

Dane

Uniwersytet Wrocławski

Dokumentacja zbioru danych:

- tytuł i dłuższy opis – tradycyjnie w komentarzach roxygen
- ogólny opis zbioru danych (liczba wierszy i kolumn + komentarz) - @format
- opis kolumn: \describe{\item{nazwa}{opis}}
- źródło danych: @source (link, cytowanie...)
- pod opisem należy umieścić napis będący nazwą obiektu.

Winietki i pkgdown

Uniwersytet Wrocławski

- Długa format dokumentacji
- Najprostsze utworzenie winietki: *usethis::use_vignette()*
- Przykłady winietek: np. dokumentacja data.table
- Więcej: poprzednie zajęcia, zajęcia z shiny
- pkgdown: *usethis::use_pkgdown()*, *pkgdown::build_site()* + konfiguracja (poprzednie zajęcia).

Więcej

Uniwersytet Wrocławski

- https://r-pkgs.org/man.html#man-workflow (zwłaszcza https://r-pkgs.org/man.html#text-formatting)
- https://documentation.divio.com/introduction/
- https://cran.r-project.org/web/packages/roxygen2/vignettes/rd.html (i reszta dokumentacji roxygen2)

Testy jednostkowe

Rodzaje testów

Uniwersytet Wrocławski

1. Sposób testowania
   - testy ręczne / manualne (np. przeklikanie naszej aplikacji shiny i sprawdzenie, że wykresy się generują, gdy zmieniamy zakresy parametrów)
   - testy automatyczne (np. check robiony przez travisa)
2. Cel:
   - testy jednostkowe: sprawdzają małe funkcjonalności metod / klas
   - testy integracji: sprawdzają, czy kilka modułów współpracuje poprawnie (np. połączenie z bazą danych)
   - testy funkcjonalne: sprawdzają zgodność wyników z wymaganiami "biznesowymi"
   - end-to-end: sprawdzają działanie oprogramowania w jego normalnym środowisku
   - testy akceptacji: sprawdzają spełnianie wymagań "biznesowych" na produkcji
   - testy wydajności (performance): działanie systemu pod dużym obciążeniem
   - *smoke tests:* sprawdzenie podstawowych funkcjonalności

https://www.atlassian.com/continuous-delivery/software-testing/types-of-software-testing

Zalety testów

Uniwersytet Wrocławski

- Testy jednostkowe są podstawą ciągłej integracji,
- Pozwalają łatwo znajdować błędy w kodzie
- Stanowią dobre przykłady wywołania kodu, więc pomagają go zrozumieć
- Wymuszają jasne określenie obiektów zwracanych przez funkcje i podział funkcji na odpowiednio małe zadania

Testy w R

Uniwersytet Wrocławski

- testy jednostkowe są wymagane przy publikowaniu pakietów na CRAN-ie / Bioconductorze
- pakiet testthat pozwala na łatwą budowę testów (alternatywa: np. tinytest)
- automatyzacja testów i sprawdzania pakietu jest łatwa z Github+TravisCI
  (poprzednie zajęcia)

Testy w R z testthat

Uniwersytet Wrocławski

- *usethis::use_testthat()* - dodaje folder tests/ i całe środowisko do testowania,
- *usethis::use_test()* - dodaje konkretny plik z testami
- testy znajdują się w plikach test_*.R w folderze testthat,
- plik tests/testthat.R odpowiada za wywołanie wszystkich testów,
- dodatkowo w folderze testthat mogą znajdować się pliki setup-xyz.R i teardown-xyz.R, które są wywoływane (odpowiednio) przed i po rozpoczęciu testów
- funkcje setup() i teardown() pozwalają wywołać kod przed/po testach w konkretnym pliku

Struktura testów w testthat

Uniwersytet Wrocławski

- testy opierają się na *expectations*
- funkcja test_that opisuje zestaw fukcjonalności do przetestowania:
  test_that("opis jako napis", {wyrazenie z testami})
  (jest to pojedynczy test)
- wewnątrz funkcji test_that używa się funkcji expect_*, które zawsze przyjmują dwa argumenty: wywołanie funkcji do przetestowania i oczekiwaną wartość

Więcej

Uniwersytet Wrocławski

- https://r-pkgs.org/tests.html
- https://www.r-bloggers.com/automated-testing-with-testthat-in-practice/
- Clean code (znowu)
- https://github.com/markvanderloo/tinytest (!)
- http://dirk.eddelbuettel.com/papers/useR2015_docker.pdf - Docker pozwala tworzyć własne, wyizolowane środowiska do testów (bardziej złożonych i czasochłonnych niż testy jednostkowe)
- https://codecov.io/
- https://r-pkgs.org/r-cmd-check.html#r-cmd-check
- https://testthat.r-lib.org/articles/custom-expectation.html