R package building workshop

(for neuroscientists)

Dominik Krzemiński

Software Sustainability Institute

Background

- R is an open source (originally) statistical software, which was the answer to the commercial S (created at Bell Labs)
- The first official release came in 1995.
- Can be easily extended by making new packages.
- The Comprehensive R Archive Network (CRAN) was officially announced 23 April 1997 with 3 mirrors and 12 contributed packages.

source : en.wikipedia.org

Background

What is package?

In R, a package is the fundamental unit of shareable code. It bundles together code, data, documentation, and tests, and is easy to share with others. As of June 2019, there were over 14,000 packages available on CRAN.

What is CRAN?

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN mirror nearest to you to minimize network load.

Source: https://cran.r-project.org/index.html

Background

`install.packages(packageName)`- command for installing packages

`library(packageName)` - command for loading a package

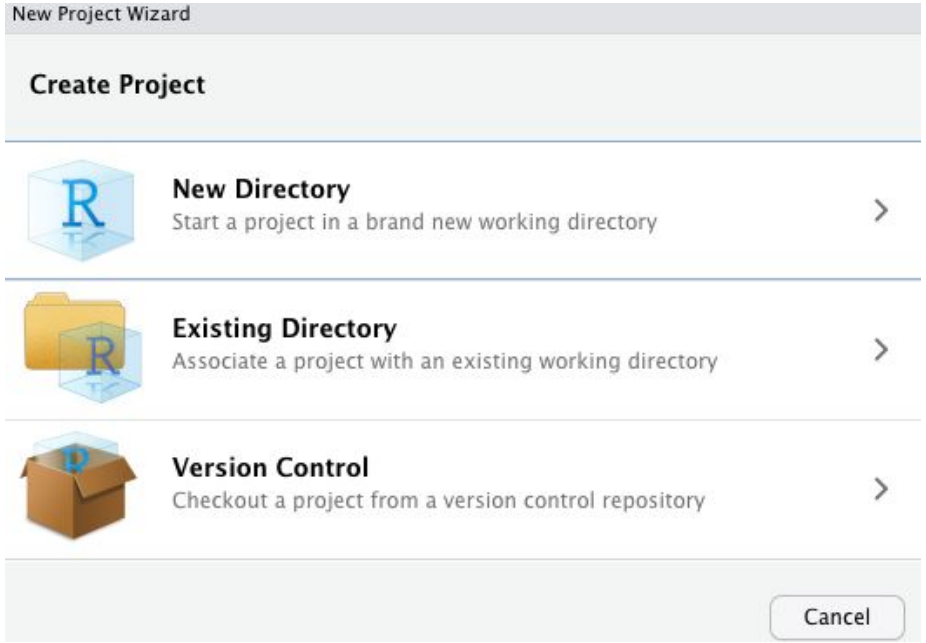
`installed.packages()` - lists all installed packages

`packageVersion("nat")` - checks package version

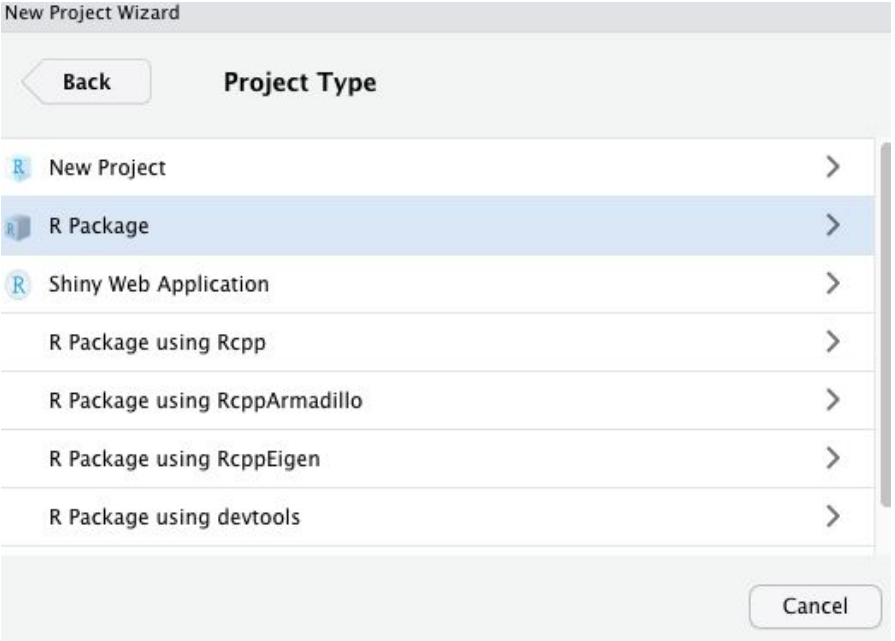
`sessionInfo()` - lists packages loaded to namespace

`.libPaths()` - this is where your packages are located

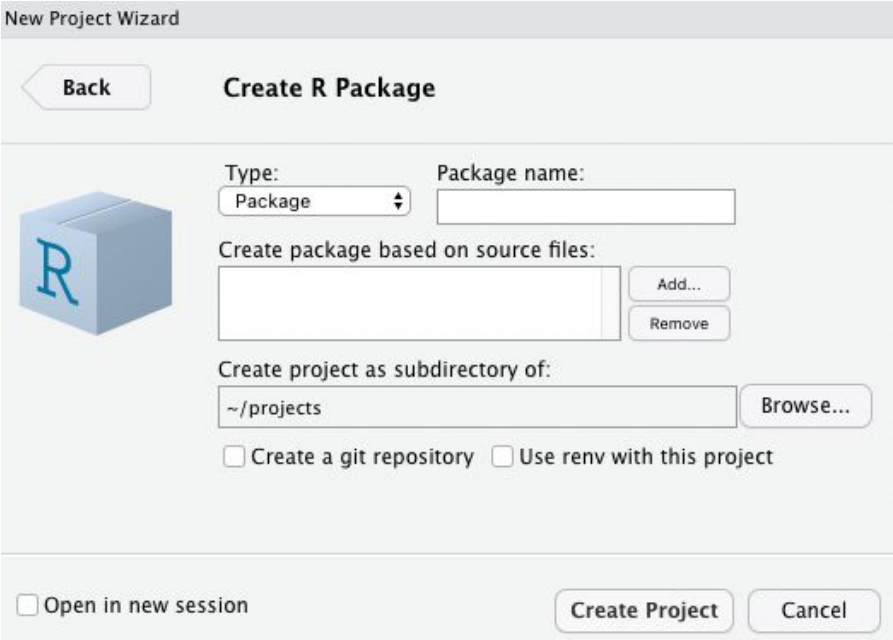
RStudio package building start



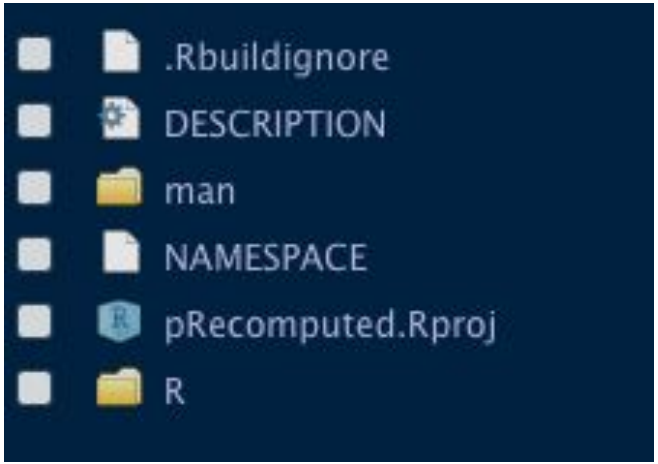
RStudio package building start



RStudio package building start



Package structure



Package structure

- R: contains R code files
- data: contains data files
- man: contains documentation files (.Rd)
- src: contains C, C++, or FORTRAN source.
- tests: with tests
- inst: all extra data with subdirectories copied recursively to the installation directory.

The two most important files

- DESCRIPTION: manual file for the users.
- NAMESPACE

Less important files

- LICENSE
- Readme

DESCRIPTION

```
Package: pRecomputed
Type: Package
Title: What the Package Does (Title Case)
Version: 0.1.0
Author: Who wrote it
Maintainer: The package maintainer <yourself@somewhere.net>
Description: More about what it does (maybe more than one line)
    Use four spaces when indenting paragraphs within the Description.
License: What license is it under?
Encoding: UTF-8
LazyData: true
```

- Title contains no more than 65 characters
- Version must be always a sequence of integers
- Author can be string, but multiple authors have a specific format Author@R

DESCRIPTION

```
Package: nat
Type: Package
Title: NeuroAnatomy Toolbox for Analysis of 3D Image Data
Version: 1.10.2.9000
Authors@R: c(
   person("Gregory","Jefferis", email="jefferis@gmail.com",  role = c("aut", "cre"), comment = c(ORCID =
"0000-0002-0587-9355")),
…
   person("Dominik", "Krzeminski", role = c("ctb"), comment = c(ORCID = "0000-0003-4568-0583"))
   )
URL: https://github.com/natverse/nat, https://natverse.org/
BugReports: https://github.com/natverse/nat/issues
Description: NeuroAnatomy Toolbox (nat) enables analysis and visualisation of 3D
   biological image data….
Depends:
   R (>= 2.15.1),
   rgl (>= 0.98.1)
Imports:
   nabor,
   igraph (>= 0.7.1),
   methods,
   filehash (>= 2.3), …
Suggests:
   spelling,
   testthat,
   Httr, …
License: GPL-3
LazyData: yes
RoxygenNote: 7.1.1
Encoding: UTF-8
VignetteBuilder: knitr
Language: en-GB
```

NAMESPACE

```
# Generated by roxygen2: do not edit by hand

export(build_markdown_report)
export(extract_markers_to_md)
export(todor)
export(todor_file)
export(todor_file_addin)
export(todor_package)
export(todor_package_addin)
export(todor_project_addin)
import(rex)
import(rstudioapi)
import(utils)
```

Code organisation

Organising principle	Source file	Comments
One function	tidyr/R/uncount.R	Defines exactly one function, `uncount()`, that's not particulary large, but doesn't fit naturally into any other `.R` file
Main function plus helpers	tidyr/R/separate.R	Defines the user-facing `separate()` (an S3 generic), a `data.frame` method, and private helpers
Family of functions	tidyr/R/rectangle.R	Defines a family of functions for "rectangling" nested lists (`hoist()` and the `unnest()` functions), all documented together in a big help topic, plus private helpers

Source: https://r-pkgs.org/r.html

Example: `fancr` package

```
) ✗ ls -l R

 839  4 Nov 11:46 cave.R
2301 15 Oct 15:20 cloudvolume.R
 263 15 Oct 15:20 fancr-package.R
4602 15 Oct 15:20 ids.R
 916 15 Oct 15:20 meshes.R
2161  4 Nov 11:46 partners.R
3548 15 Oct 15:20 urls.R
2930  4 Nov 12:12 xform.R
1709 15 Oct 15:20 zetta-api.R
 136 15 Oct 15:20 zzz.R
```

Roxygen

“The premise of roxygen2 is simple: describe your functions in comments next to their definitions and roxygen2 will process your source code and comments to produce Rd files in the man/ directory.”

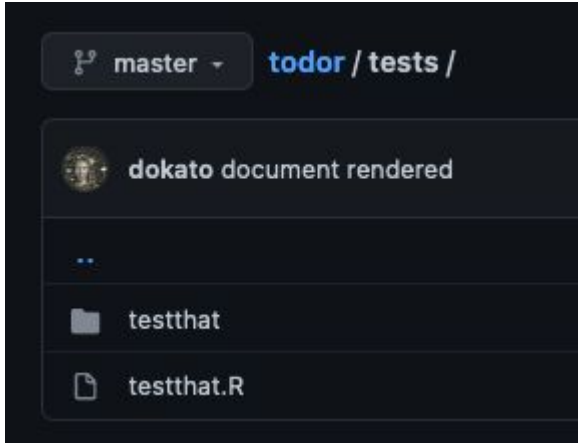
> install.packages("roxygen2")

More: https://roxygen2.r-lib.org/

```
#' The length of a string
#'
#' Technically this returns the number of "code points", in a string. One
#' code point usually corresponds to one character, but not always. For example,
#' an u with a umlaut might be represented as a single character or as the
#' combination a u and an umlaut.
#'
#' @inheritParams str_detect
#' @return A numeric vector giving number of characters (code points) in each
#'    element of the character vector. Missing string have missing length.
#' @seealso [stringi::stri_length()] which this function wraps.
#' @export
#' @examples
#' str_length(letters)
#' str_length(NA)
#' str_length(factor("abc"))
#' str_length(c("i", "like", "programming", NA))
str_length <- function(string) {
}
```

Testing

`Testthat` makes testing as easy as possible.




```r
context("internal functions")

test_that("test process_file function", {
  to_detect <- c("BUG")
  p <- process_file("demo.R", to_detect)
  expect_equal(length(p), 1)
  to_detect <- c("BUG", "TODO")
  p <- process_file("demo.R", to_detect)
  expect_equal(length(p), 2)
})
```

```
> devtools:: test()
```

Devtools

The aim of devtools is to make package development easier by providing R functions that simplify and expedite common tasks.

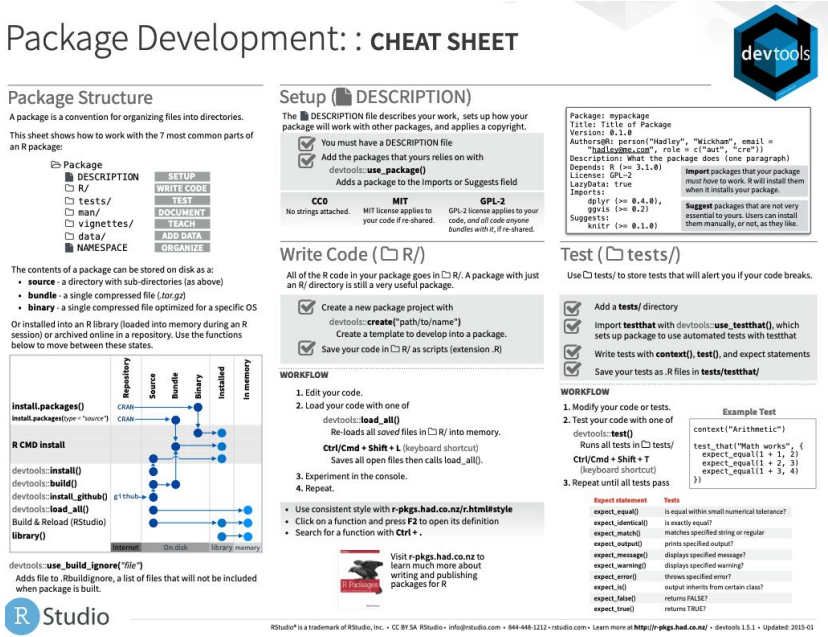
Install devtools from CRAN

```
> install.packages("devtools")
```

Most useful command:

```
> devtools::document() # creates documentation and updates NAMESPACE file
> devtools:: test() # runs tests with (typically) testthat
> devtools::check() # performs CRAN checks, check options for more advanced settings
```

https://rawgit.com/rstudio/cheatsheets/main/package-development.pdf



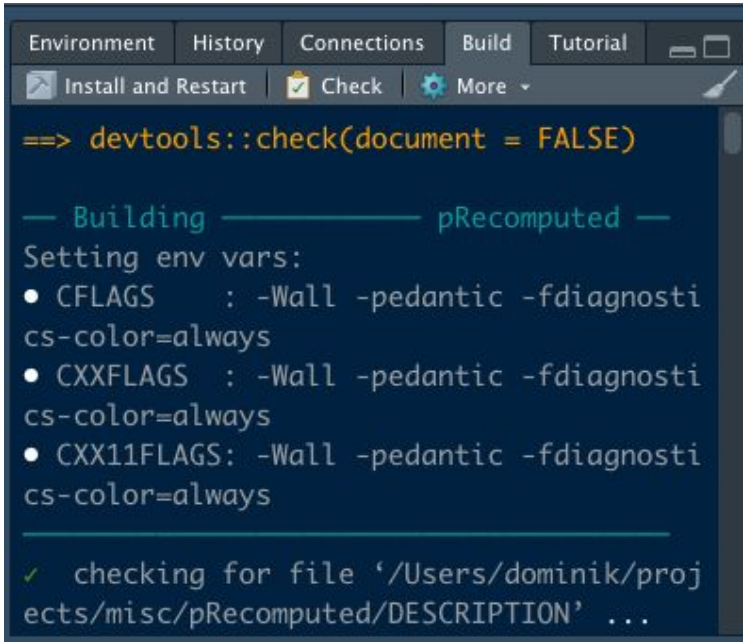
Building a package

Command line:

```
$ R CMD build packageFolder

$ R CMD INSTALL
packageName_version.tar.gz

$ R CMD check packageName_version.tar.gz
```



Submit to CRAN

Submit package to CRAN

Step 1 (Upload)	Step 2 (Submission)	Step 3 (Confirmation)

Your name*:

Your email*:

Package*: Browse... No file selected.
(*.tar.gz files only, max 100 MB size)

Optional comment:

*: Required Fields

Before uploading please ensure the following:

- The package contains a DESCRIPTION file.
- DESCRIPTION file contains valid maintainer field "NAME <EMAIL>".
- You submit a tar.gz created with R CMD build.
- You are familiar with the rest of the CRAN policies

Upload package

Links

https://github.com/flyconnectome/pRecomputed

https://gist.github.com/dokato/3c1f5d322b4cfad8332836f10a675dac

Book on package building: https://r-pkgs.org/

Package development cheatsheet: https://rawgit.com/rstudio/cheatsheets/main/package-development.pdf

Thank you!