Package Development in R

R Ladies Nashville

Jacquelyn Neal and Jennifer Morse

September 16, 2019

Sources and Workshop Materials

R Forwards and R Ladies Chicago Workshop in February 2019 in Chicago

All Materials Here: https://github.com/forwards/workshops/tree/master/Chicago2019

Script

One off data analysis

Primarily side-effects

Package

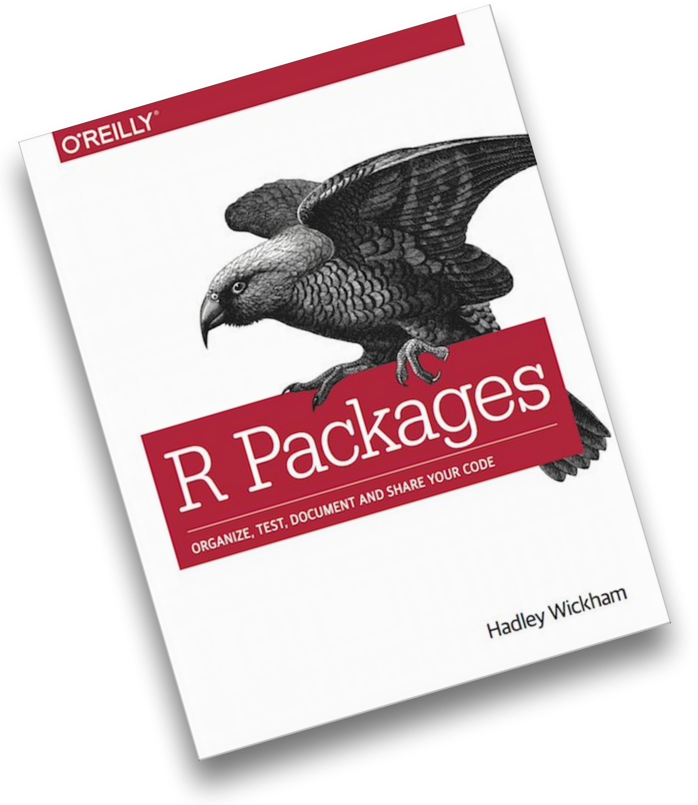
Defines reusable components

No side-effects

Why make a reusable component?

- You want to **test** it
- You want to **generalize** it
- You want to **document** it
- You want to **share** it

This material is mostly drawn from:



http://r-pkgs.had.co.nz

New (draft) version:
https://r-pkgs.org

How is developing a package
same / different
from developing a script?

How same?

Iterate early and often!

Change it, try it, change it, try it, *ad nauseum*

How is developing a package different from writing a script?

Write functions, not “top-level” code.

Dependencies are different,
    no library() calls.

Install & Restart (or simulate that),
    don’t source().

A package is a set of conventions that (with the right tools) makes your life easier

Script

One off data analysis

Primarily side-effects

Package

Defines reusable components

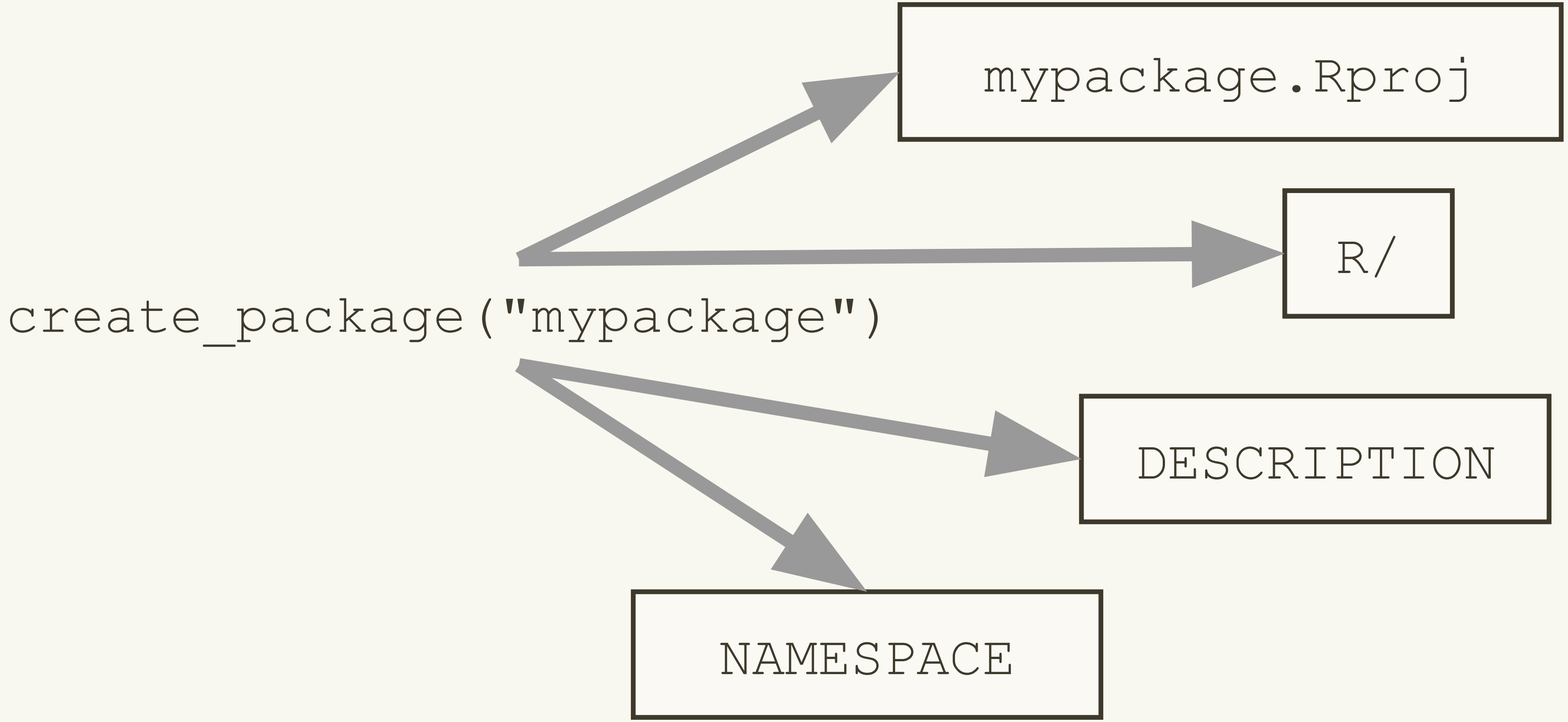
No side-effects

Your First Package

Your turn

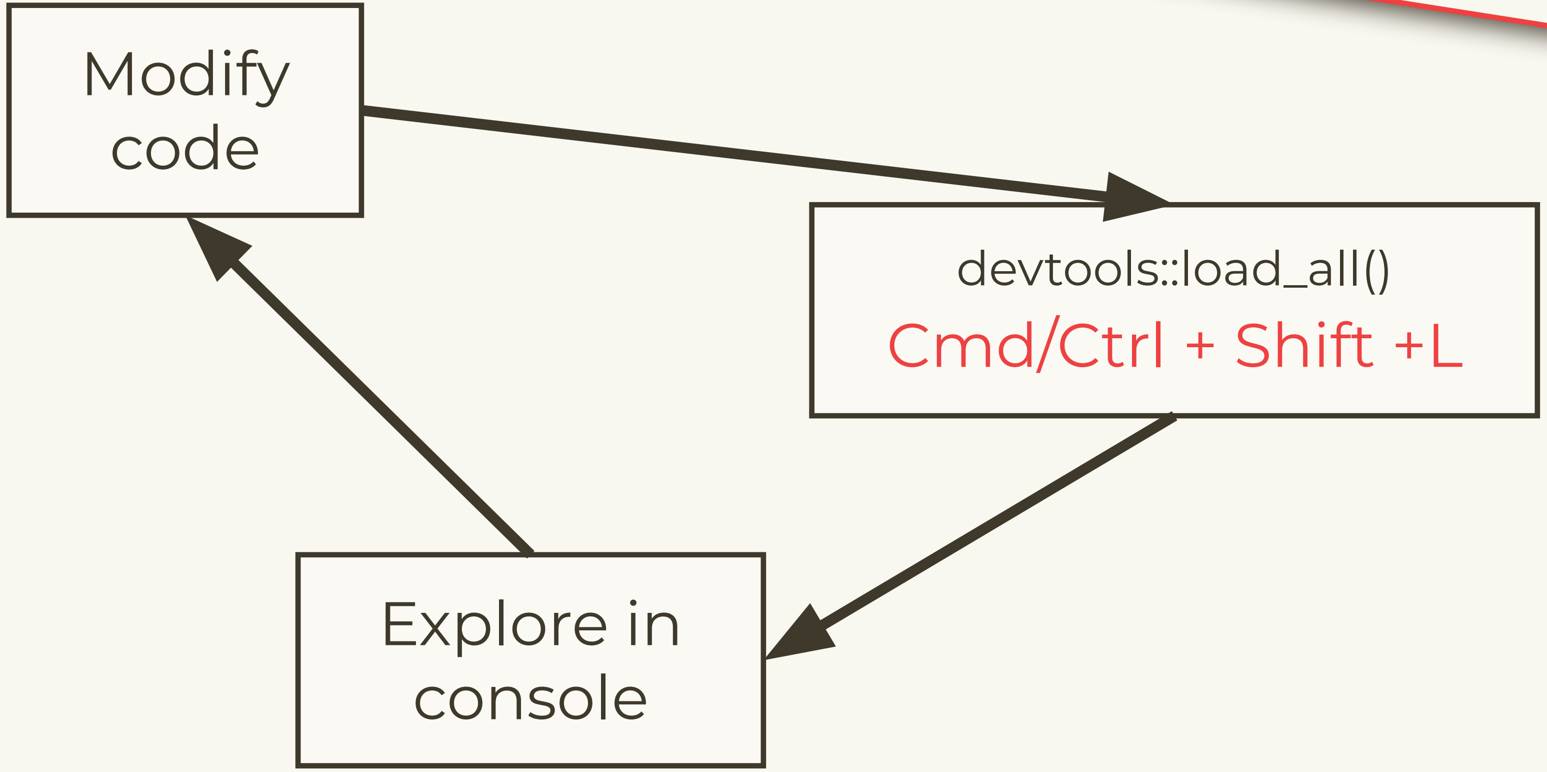
```
# Verify that you can create a package with:
usethis::create_package("~/Desktop/mypackage")

# What other files and directories are created?

# You can also create new project using RStudio
# but it has some slight differences that will
# cause hassles today (but not in general)
```

What happens we run create_package()?



Why bother?

You don't even need to save your code!



What if you need to create a new file?

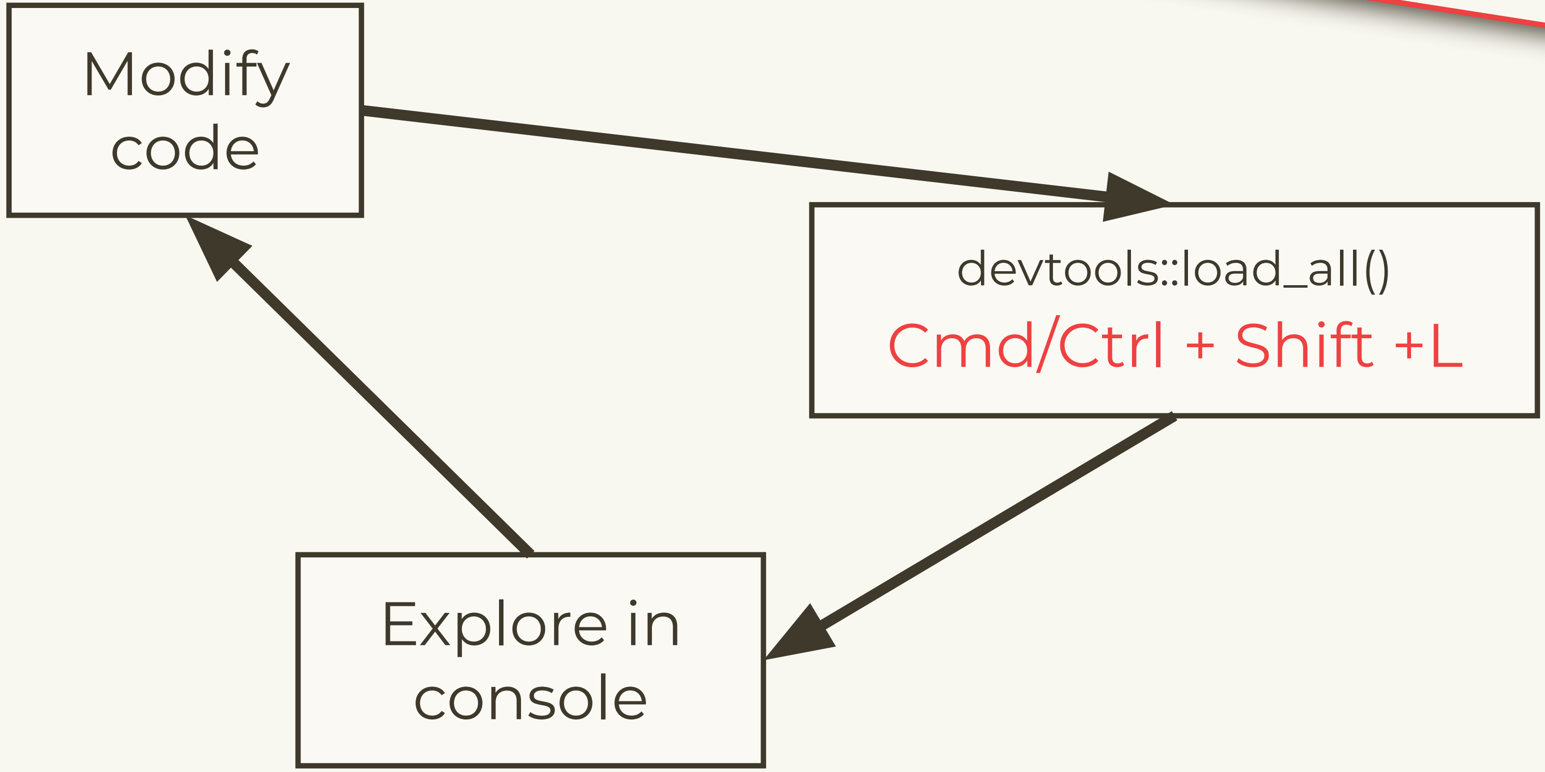
```r
# There's a usethis helper for that too!
usethis::use_r("file-name")

# Organize files so that related code
# lives together. If you can give a file
# a concise and informative name, it's
# probably about right
```

Your turn

- Create a new R file in your package called “zooSounds.R”
  - Or use a personal function you’ve been wanting to put into a package!
- Paste the following code into your script:

```
goToTheZoo <- function(animal, sound){
assertthat::assert_that(
  assertthat::is.string(animal),
  assertthat::is.string(sound))
  glue::glue("The ", animal, " goes ", sound,"!", sep =
" ")
  }
```

Try it out!



You don't even need to save your code!

Your turn

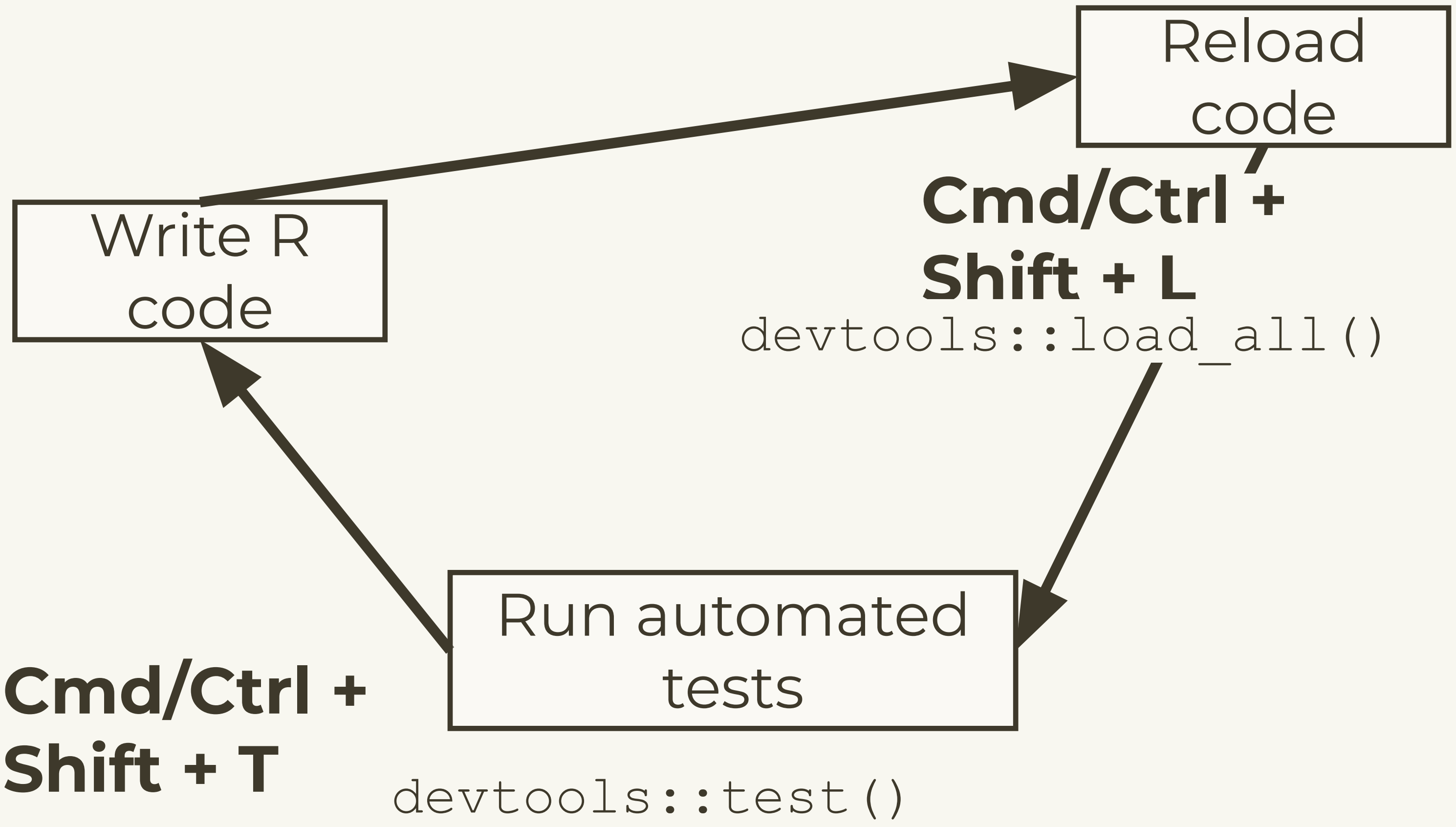
- Change some tiny thing about your function - maybe the animal “says” instead of “goes”?
- Load all with devtools::load_all()
    - You can try checking that it’s ok with devtools::check()

- Try adding yourself as an author to the package in DESCRIPTION, and a fun title and description

Testing Workflow

http://r-pkgs.had.co.nz/tests.html

Testthat gives a new workflow



A sample test

Tests for R/zooSounds.R

```
# In tests/testthat/test-zooSounds.R

library(testthat)

test_that("goToTheZoo produces expected strings", {
  allSounds <- as.character(goToTheZoo("giraffe", "moo"))
  expect_equal(allSounds, "The giraffe goes moo!")
})

test_that("goToTheZoo fails with numbers", {
  expect_error(goToTheZoo(1, 2))
})
```

Four expectations cover 90% of cases

```
expect_equal(obj, exp)
```

```
expect_error(code, regexp)
```

```
expect_warning(code, regexp)
expect_warning(code, NA)
```

```
expect_known_output(code)
```

Organizing Tests

Think about the overall functionality, or “end to end” tests

Test every individual task the function completes separately

Check both for successful situations and for expected failure situations

You can also automate

- GitHub = publish and manage your code online
- Travis or Jenkins = Continuous Integration; run code (like your tests) every time your code changes
  - https://travis-ci.org/
  - https://jenkins.io/
- Codecov = display which functions are tested
  - https://github.com/codecov/example-r

Package Documentation

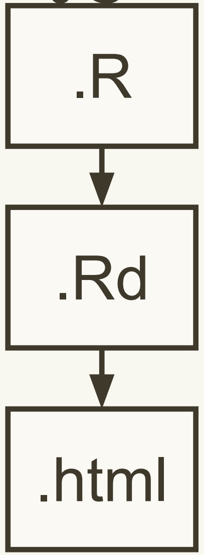
Why?

People need instructions to use new things!

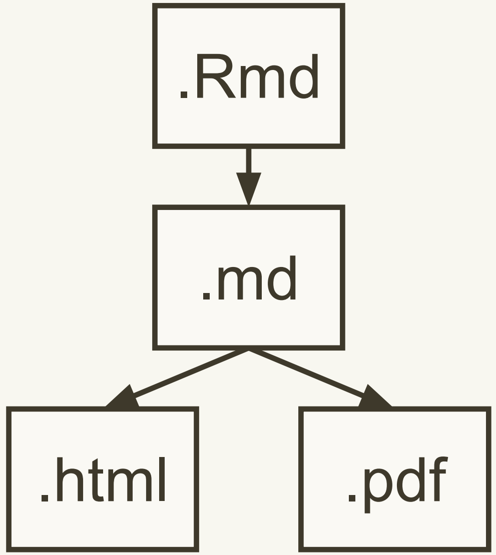
You might want instructions to remind you how your tools work too.

Documentation is the way you preserve the information about your tools.

Function-level with roxygen2



Package-level with rmarkdown

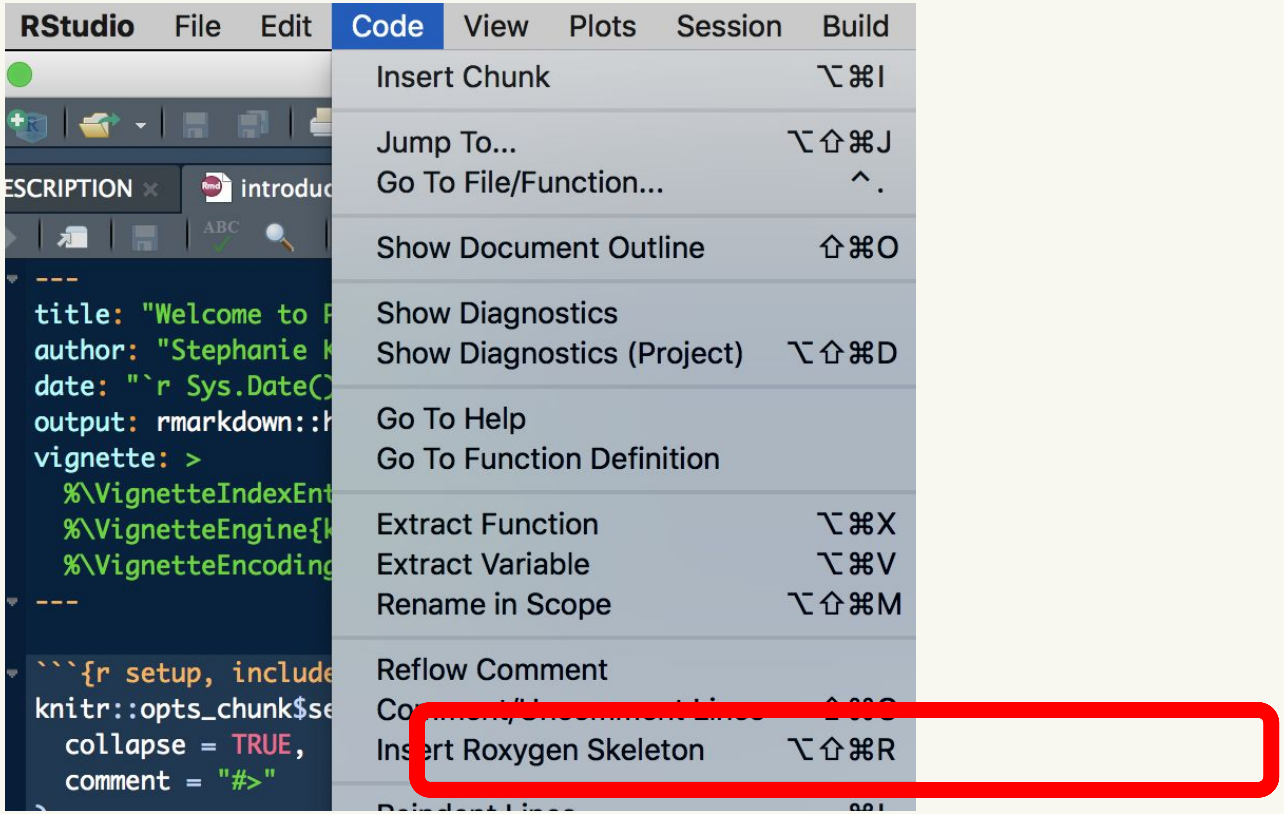


Basic markdown formatting

~~~
# This is a top level heading

This is some text. Make text _italic_ with single  underscores (or
stars). Make it **bold** with double stars (or underscores). Here is a
[link to a markdown guide](http://bit.ly/19fAexE).

* This is a list

* This is another item

```R
# Some R code
f <- function() x + 1
```

## This is a secondary heading

You can also do `inline code`, numbered lists and quotes and more.
~~~

Document Each Function

Roxygen2



~~~
---
title: "Welcome to F
author: "Stephanie K
date: "`r Sys.Date()
output: rmarkdown::h
vignette: >
  %\VignetteIndexEnt
  %\VignetteEngine{k
  %\VignetteEncoding
---

```{r setup, include
knitr::opts_chunk$se
  collapse = TRUE,
  comment = "#>"
~~~

Roxygen allows us to explain the function's parts...

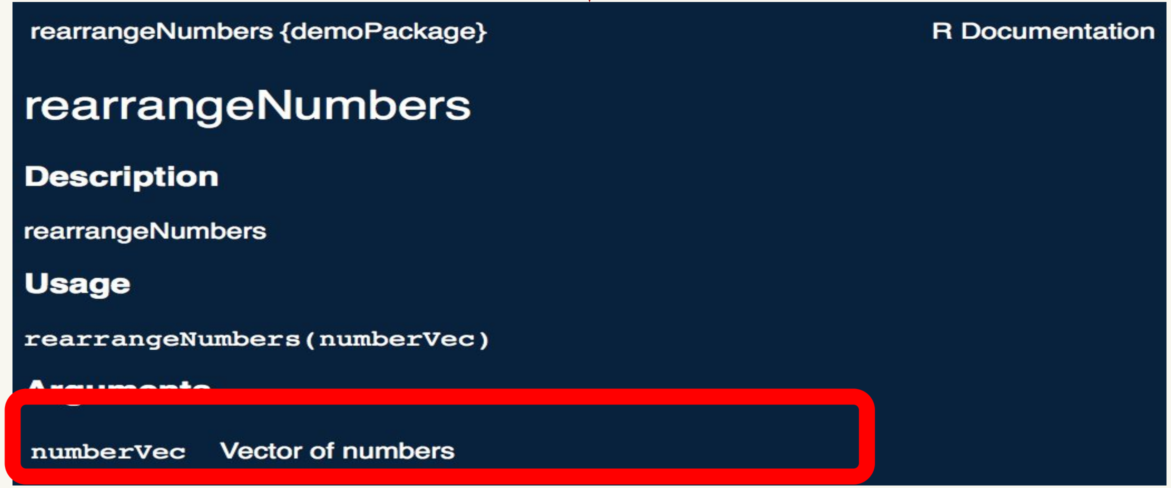
http://r-pkgs.had.co.nz/man.html

You write specially formatted comments in .R

```
#' @param numberVec Vector of numbers
```

Starts with the type of element, then you describe it in reasonable human language

```
devtools::document()
```



rearrangeNumbers {demoPackage} R Documentation

rearrangeNumbers

Description

rearrangeNumbers

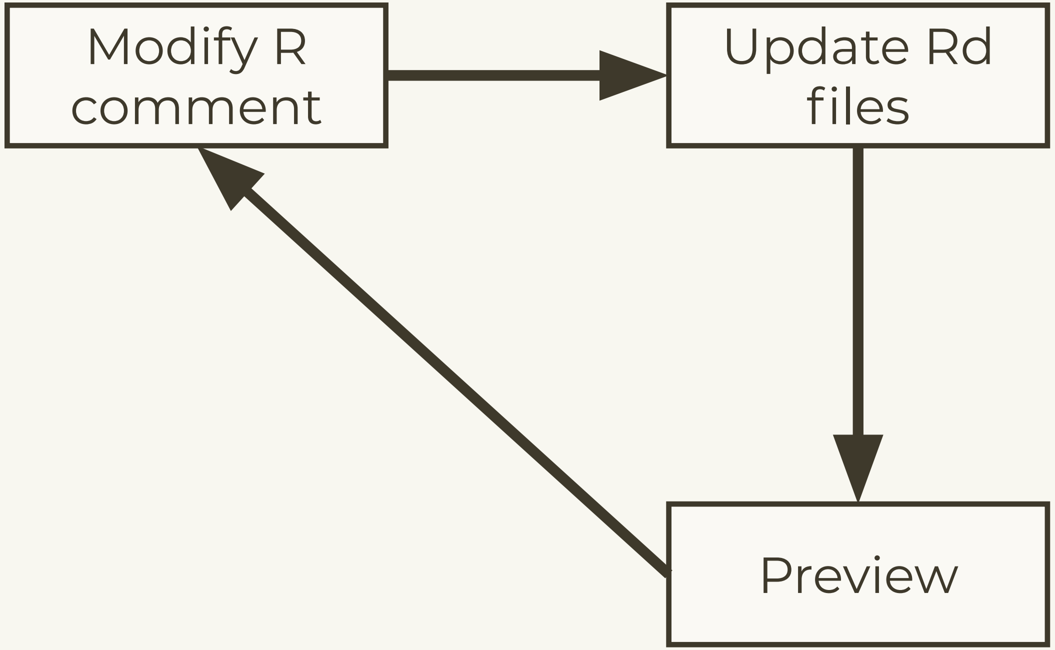
Usage

```
rearrangeNumbers(numberVec)
```

Arguments

`numberVec`	Vector of numbers

Documentation workflow



Cmd/Ctrl + Shift + D

devtools::document()

?topicname

There are five tags you'll use for most functions

Tag	Purpose
@param arg	Describe inputs
@examples	Show how the function works. (Usual RStudio shortcuts work)
@seealso	Pointers to related functions
@return	Describe outputs (value)
@export	Is this a user-visible function?

You can use markdown for formatting

```
# Activate by running
# use_roxygen_md()

**bold**, _italic_, `code`

* [func()]
* [pkg::func()]
* [link text][func()]
* [link text][pkg::func()]
```

roxygen2 - tools to reduce duplication

```
# Document multiple functions in the same file
#' @rdname add_col

# Inherit the parameter descriptions from
# another function
#' @inheritParams add_col

# Inherit everything from another topic
#' @inherit add_col

# Inherit selectively
#' @inherit add_col parameters return references
#'   title description details
#'   sections seealso
```

Read online about how to document other objects

Data

**http://r-pkgs.had.co.nz/data.html#documentin
g-data**

Classes & methods

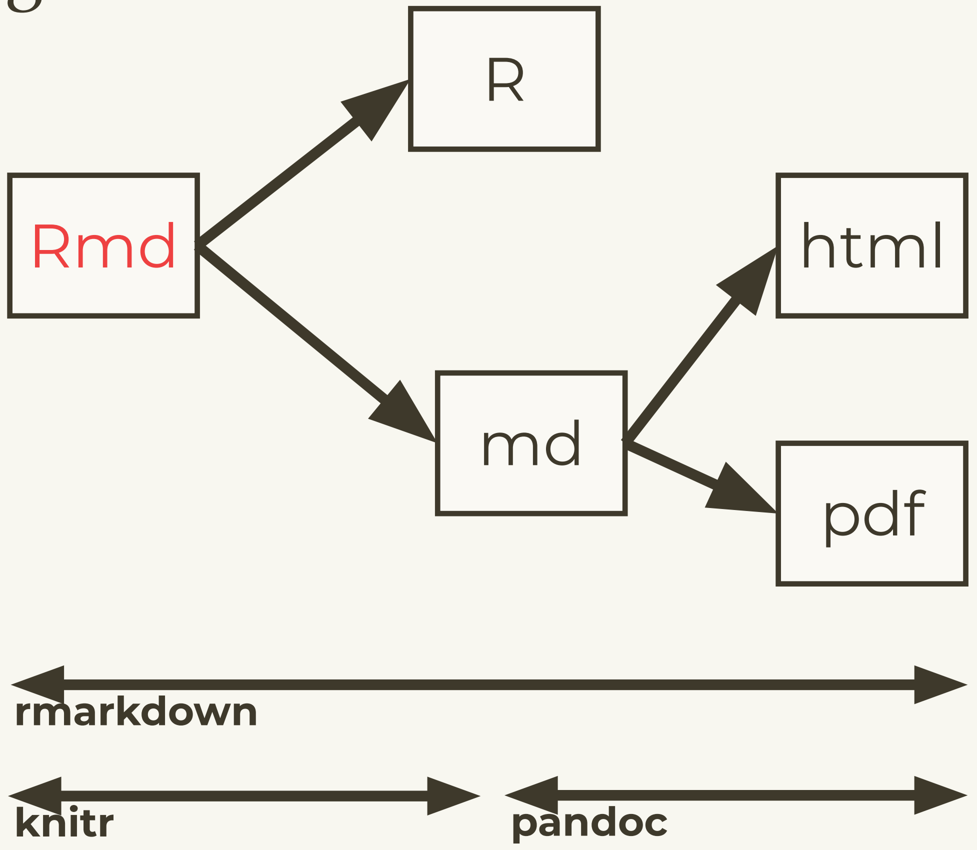
**http://r-pkgs.had.co.nz/man.html#man-c
lasses**

Packages

**http://r-pkgs.had.co.nz/man.html#man-pac
kages**

Document Your Overall Package

Vignettes



Lets you combine prose and code to explain your how you package works.

The hard part is the writing, not the technology!

http://r-pkgs.had.co.nz/vignettes.html

Easiest way to get started is with use_vignette()

```
usethis::use_vignette("name")

# Adds to DESCRIPTION
Suggests: knitr
VignetteBuilder: knitr

# Creates vignettes/
# Drafts vignettes/name.Rmd
```

Vignette = Rmarkdown + special metadata

```
title: "Vignette Title"
author: "Vignette author"
date: "`r Sys.Date()`"
output: rmarkdown::html_vignette
vignette: >
  %\VignetteIndexEntry{Vignette Title}
  %\VignetteEngine{knitr::rmarkdown}
  %\VignetteEncoding{UTF-8}
---

Vignettes are long form documentation commonly included in packages.
Because they are part of the distribution of the package, they need to
be as compact as possible. The `html_vignette` output type provides a
custom style sheet (and tweaks some options) to ensure that the
resulting html is as small as possible. The `html_vignette` format:
```

Special output format for vignettes

Special metadata needed by R

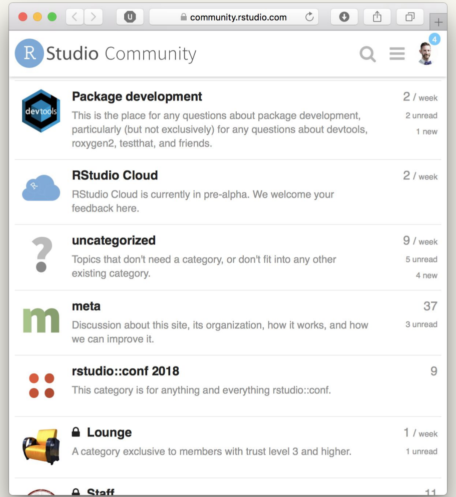
README

If sharing with others, include a README

```
Your choice: but often useful to include
results of running code
usethis::use_readme_md()
usethis::use_readme_rmd()
```

For public projects this should include a brief overview, instructions on how to install, and a few examples. For private projects, this is a great place to jot down notes!

Learning more



community.rstudio.com

RStudio Community

Package development 2 / week, 2 unread, 1 new
This is the place for any questions about package development, particularly (but not exclusively) for any questions about devtools, roxygen2, testthat, and friends.

RStudio Cloud 2 / week
RStudio Cloud is currently in pre-alpha. We welcome your feedback here.

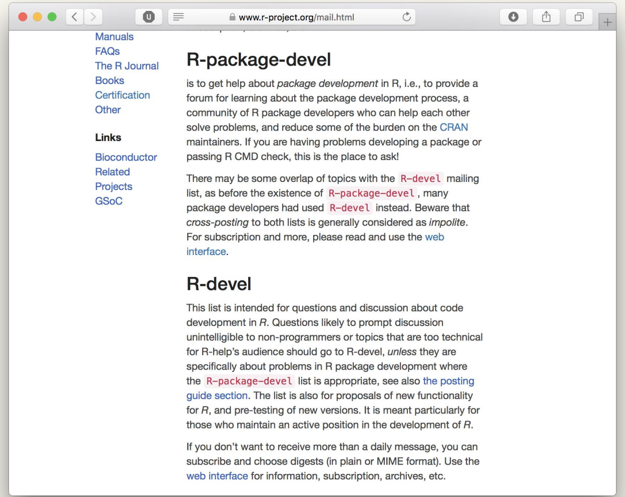
uncategorized 9 / week, 5 unread, 4 new
Topics that don't need a category, or don't fit into any other existing category.

meta 37, 3 unread
Discussion about this site, its organization, how it works, and how we can improve it.

rstudio::conf 2018 9
This category is for anything and everything rstudio::conf.

Lounge 1 / week, 1 unread
A category exclusive to members with trust level 3 and higher.

community.rstudio.com



www.r-project.org/mail.html

Manuals
FAQs
The R Journal
Books
Certification
Other

Links

Bioconductor
Related
Projects
GSoC

R-package-devel

is to get help about *package development* in R, i.e., to provide a forum for learning about the package development process, a community of R package developers who can help each other solve problems, and reduce some of the burden on the CRAN maintainers. If you are having problems developing a package or passing R CMD check, this is the place to ask!

There may be some overlap of topics with the `R-devel` mailing list, as before the existence of `R-package-devel`, many package developers had used `R-devel` instead. Beware that *cross-posting* to both lists is generally considered as *impolite*. For subscription and more, please read and use the web interface.

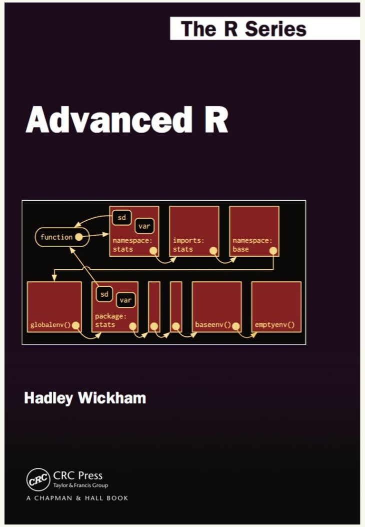
R-devel

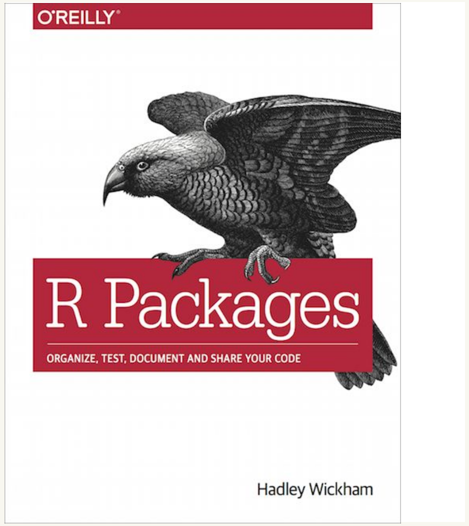
This list is intended for questions and discussion about code development in *R*. Questions likely to prompt discussion unintelligible to non-programmers or topics that are too technical for R-help's audience should go to R-devel, *unless* they are specifically about problems in R package development where the `R-package-devel` list is appropriate, see also the posting guide section. The list is also for proposals of new functionality for *R*, and pre-testing of new versions. It is meant particularly for those who maintain an active position in the development of *R*.

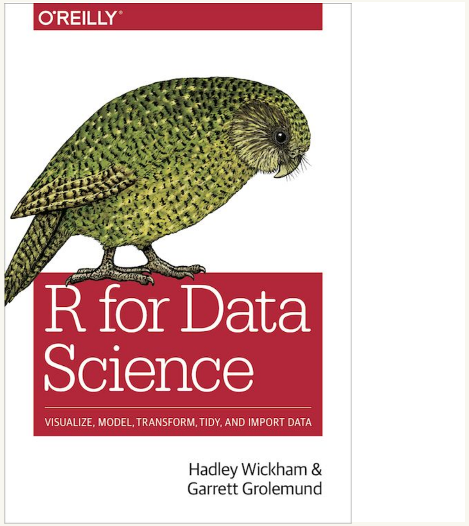
If you don't want to receive more than a daily message, you can subscribe and choose digests (in plain or MIME format). Use the web interface for information, subscription, archives, etc.

R-package-devel mailing list

More details on many topics in books






O'REILLY
R for Data Science
VISUALIZE, MODEL, TRANSFORM, TIDY, AND IMPORT DATA
Hadley Wickham & Garrett Grolemund

http://adv-r.hadley.nz/

http://amzn.com/1466586966

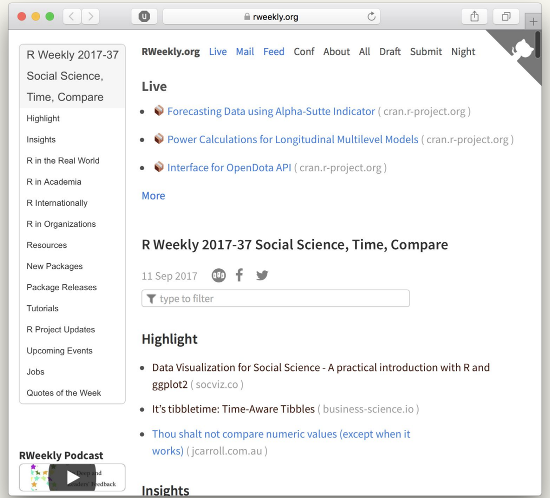
http://r-pkgs.had.co.nz

https://amzn.com/1491910399

http://r4ds.had.co.nz

https://amzn.com/1491910399

rweekly.org


rweekly.org

R Weekly 2017-37 Social Science, Time, Compare
Highlight
Insights
R in the Real World
R in Academia
R Internationally
R in Organizations
Resources
New Packages
Package Releases
Tutorials
R Project Updates
Upcoming Events
Jobs
Quotes of the Week

RWeekly Podcast

RWeekly.org Live Mail Feed Conf About All Draft Submit Night

Live

- Forecasting Data using Alpha-Sutte Indicator (cran.r-project.org)
- Power Calculations for Longitudinal Multilevel Models (cran.r-project.org)
- Interface for OpenDota API (cran.r-project.org)

More

R Weekly 2017-37 Social Science, Time, Compare

11 Sep 2017

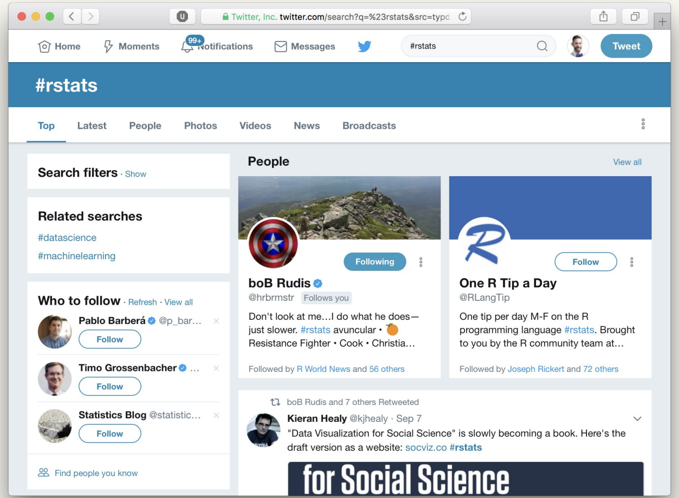
type to filter

Highlight

- Data Visualization for Social Science - A practical introduction with R and ggplot2 (socviz.co)
- It's tibbletime: Time-Aware Tibbles (business-science.io)
- Thou shalt not compare numeric values (except when it works) (jcarroll.com.au)

Insights

#rstats


Twitter, Inc. twitter.com/search?q=%23rstats&src=typd

Home Moments Notifications Messages #rstats Tweet

#rstats

Top Latest People Photos Videos News Broadcasts

Search filters · Show

Related searches
#datascience
#machinelearning

Who to follow · Refresh · View all
Pablo Barberá @p_bar… Follow
Timo Grossenbacher … Follow
Statistics Blog @statistic… Follow

Find people you know

People View all

boB Rudis @hrbrmstr Follows you Following
Don't look at me…I do what he does—just slower. #rstats avuncular • Resistance Fighter • Cook • Christia…
Followed by R World News and 56 others

One R Tip a Day @RLangTip Follow
One tip per day M-F on the R programming language #rstats. Brought to you by the R community team at…
Followed by Joseph Rickert and 72 others

boB Rudis and 7 others Retweeted
Kieran Healy @kjhealy · Sep 7
"Data Visualization for Social Science" is slowly becoming a book. Here's the draft version as a website: socviz.co #rstats

for Social Science

stackoverflow

[r] score:5 is:question closed:no