Write your first R package

STAT 547M (= second half of STAT 545A)

web companion: STAT 545 > All the package things

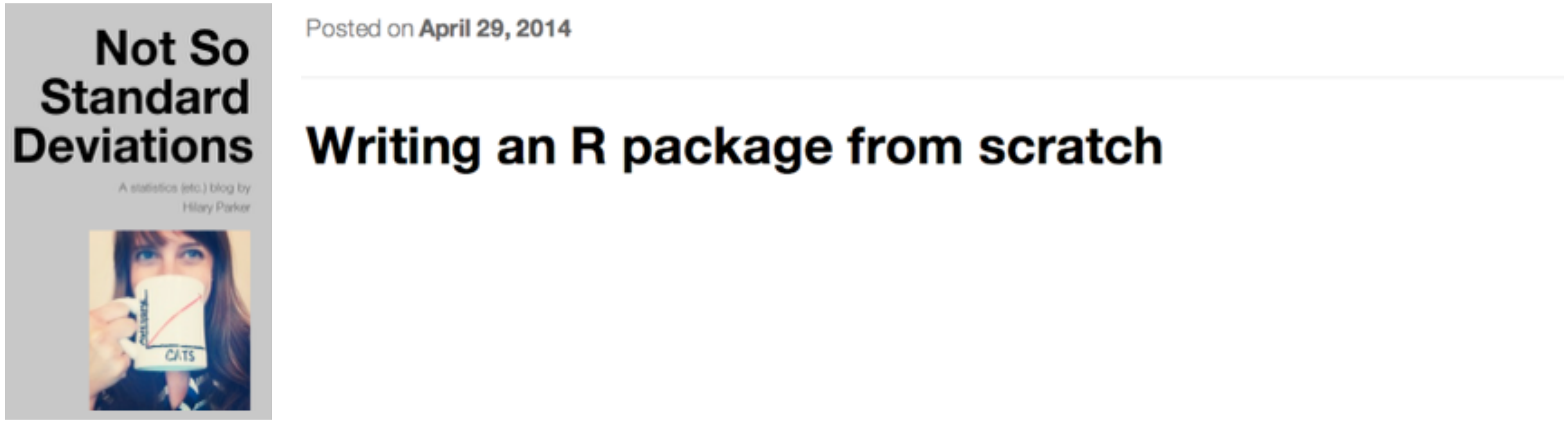
Dr. Jennifer (Jenny) Bryan
Department of Statistics and Michael Smith Laboratories
University of British Columbia

jenny@stat.ubc.ca
https://github.com/jennybc
http://www.stat.ubc.ca/~jenny/
@JennyBryan ← personal, professional Twitter

https://github.com/STAT545-UBC
http://stat545-ubc.github.io
@STAT545 ← Twitter as lead instructor of this course

R packages

http://hilaryparker.com/2014/04/29/writing-an-r-package-from-scratch/



Not So Standard Deviations

A statistics (etc.) blog by Hilary Parker

Posted on April 29, 2014

Writing an R package from scratch

I wish I could go back in time and create the package the first moment I thought about it, and then use all the saved time to watch cat videos because that really would have been more productive.

Disclaimer:

These slides aren’t meant to stand alone.

They are a companion to 3 hours of hands-on activity in which *we actually write an R package*.

Our attention bounced between these Big Ideas + technical details and hands-on work.

R packages are the fundamental unit of R-ness

14 base packages

- functions in these pkgs are what you think of as "base R"

15 Recommended packages

- ship w/ all binary dist'ns of R; no need to install
- use via, e.g., `library(lattice)`

CRAN has > 6K more packages

- e.g., `install.packages("dplyr")`
- e.g., `library(dplyr)`

And then there's Github ...

- e.g., `devtools::install_github("hadley/dplyr")`
- e.g., `library(dplyr)`

What are R packages good for?

- provide functions and datasets for use

Why better than just `source()`ing functions, `read.table()`ing data?

- standard structure facilitates distribution
- help pages, vignettes
- optionally, incorporate non-R code
- tests to ensure code works and stays that way
- checking package as a whole

What are R scripts good for?

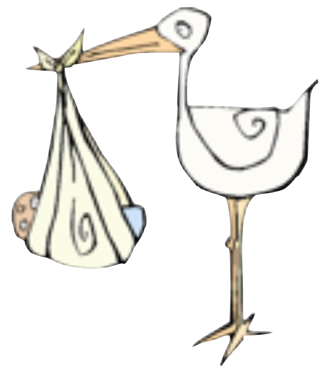
- e.g., executing a series of data manipulations

You will need both in your data analytical life.

Up 'til now in this course, we've focused on writing our own R scripts and using packages developed by other people.

NOW we'll talk about developing our own R packages.

Where do installed packages come from?



Hint: it’s not the stork!

Where do installed packages come from?

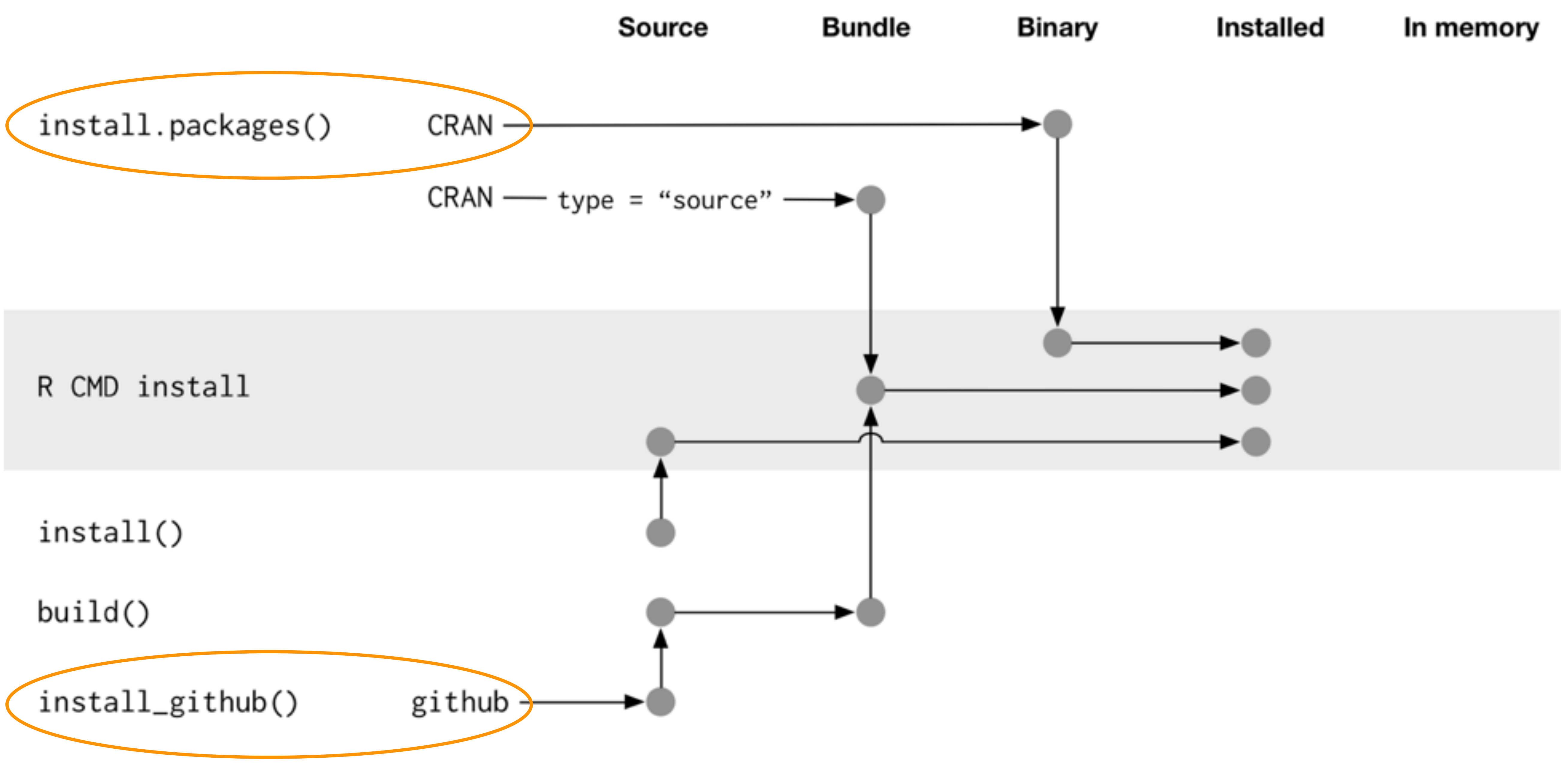


Figure from Hadley Wickham's book, R packages
http://r-pkgs.had.co.nz
https://github.com/hadley/r-pkgs/blob/master/diagrams/installation.png

You've installed packages from CRAN and maybe from GitHub.
Where do they live on your computer?

By default, in your default library.

Get to know your R installation

** your set up is probably different from mine*

```
> R.home()
[1] “/Library/Frameworks/R.framework/Resources"

> .Library
[1] “/Library/Frameworks/R.framework/Resources/library"

> .libPaths()                                            symlinked
[1] "/Users/jenny/resources/R/library"
[2] “/Library/Frameworks/R.framework/Versions/3.2/Resources/library"

> readLines("~/.Renviron")
[1] "R_LIBS=~/resources/R/library"
[2] “GITHUB_TOKEN=????????????????????????????????????????”
[3] “GITHUB_PAT=??????????????????????????????????????????”
[4] "NOT_CRAN=true"
```

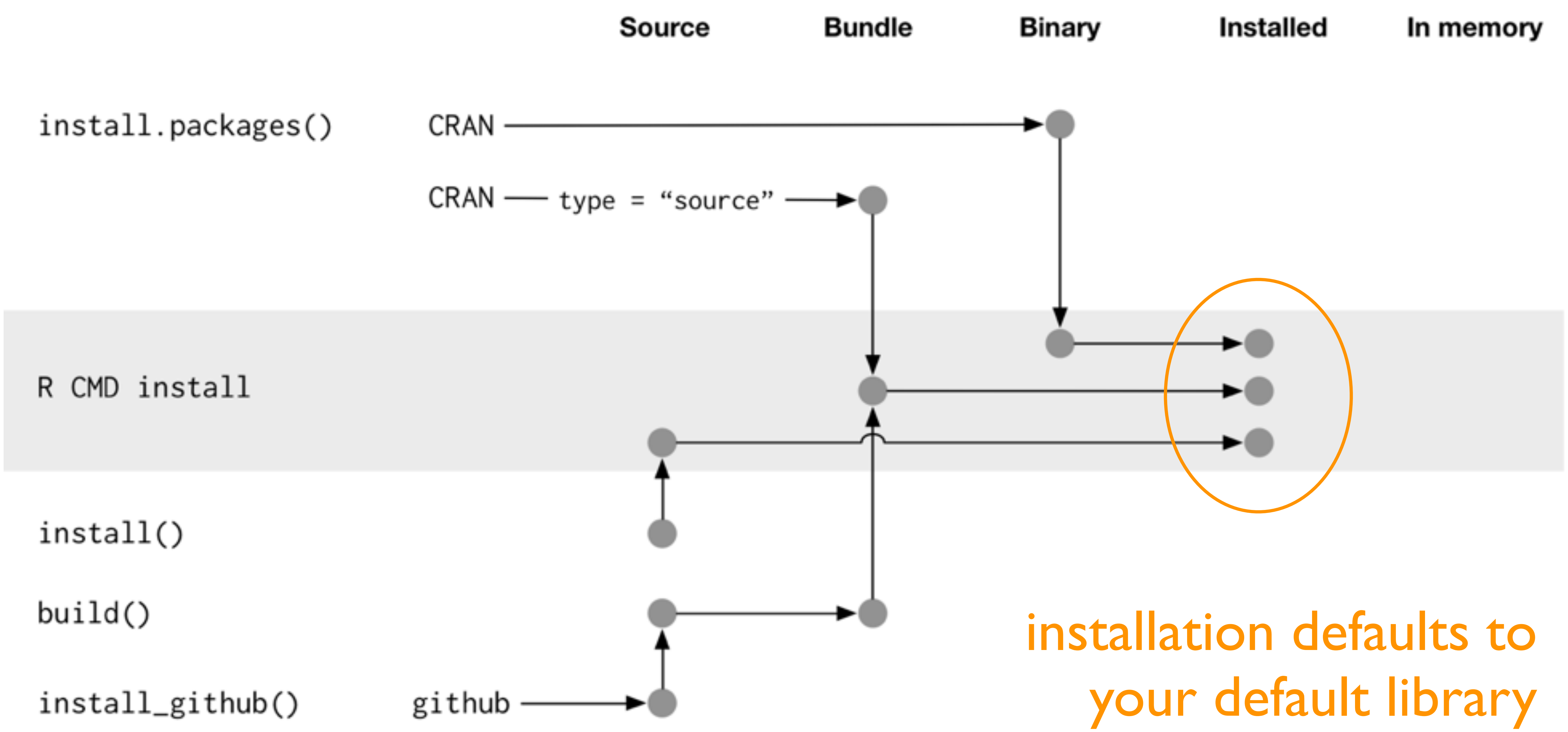
```
> R.home()
[1] "/Library/Frameworks/R.framework/Resources"

> .Library
[1] "/Library/Frameworks/R.framework/Resources/library"

> .libPaths()
[1] "/Users/jenny/resources/R/library"
...
```

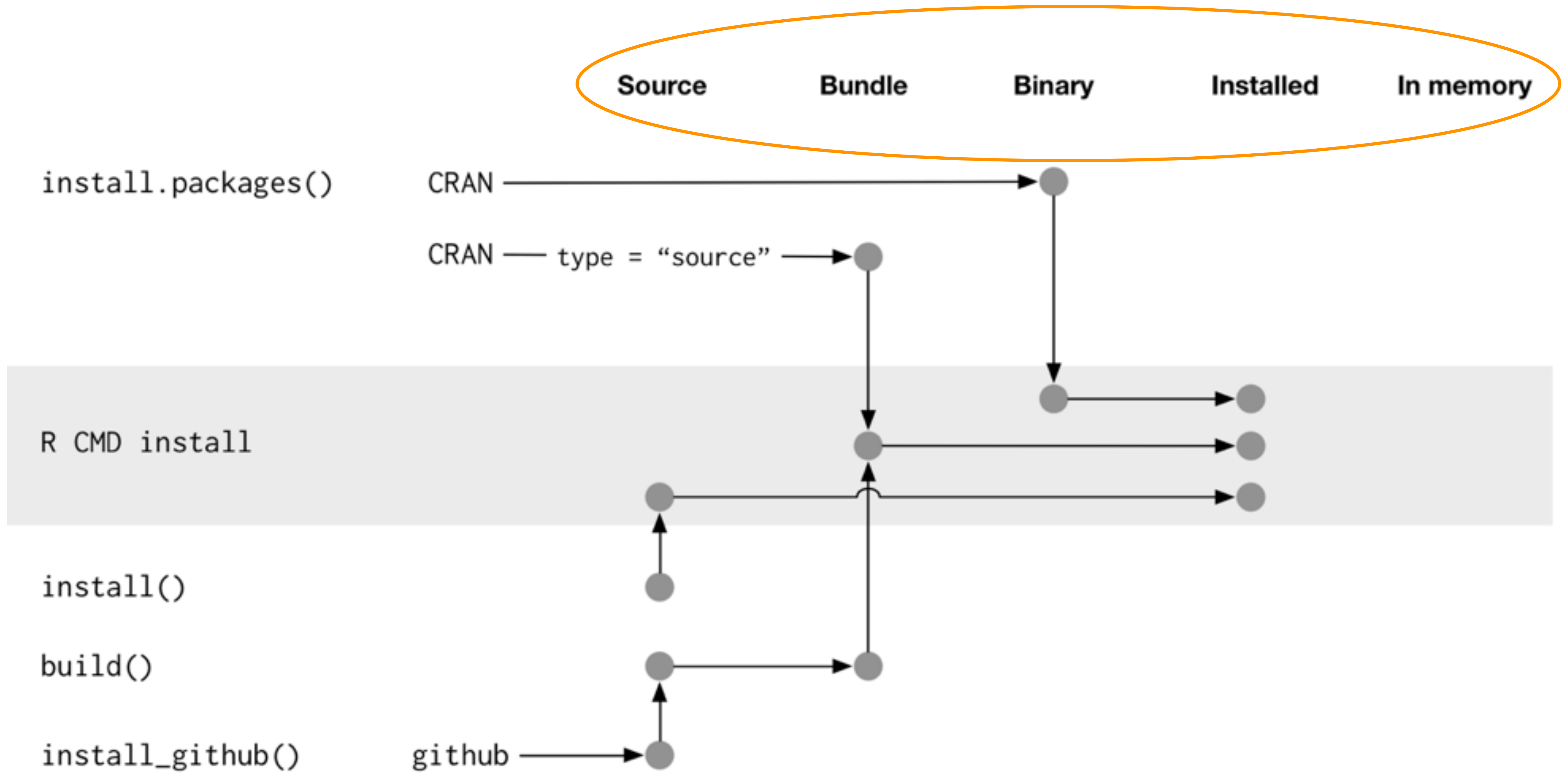
functions like `old.packages()`, `install.packages()`, `update.packages()`, `library()` operate, by default, on the first library listed in `.libPaths()` = your default library

for you, probably same as .Library



installation defaults to
your default library

Note the various "developmental stages" of an R package



Exercise (maybe for homework?)

Take a package we've used in class

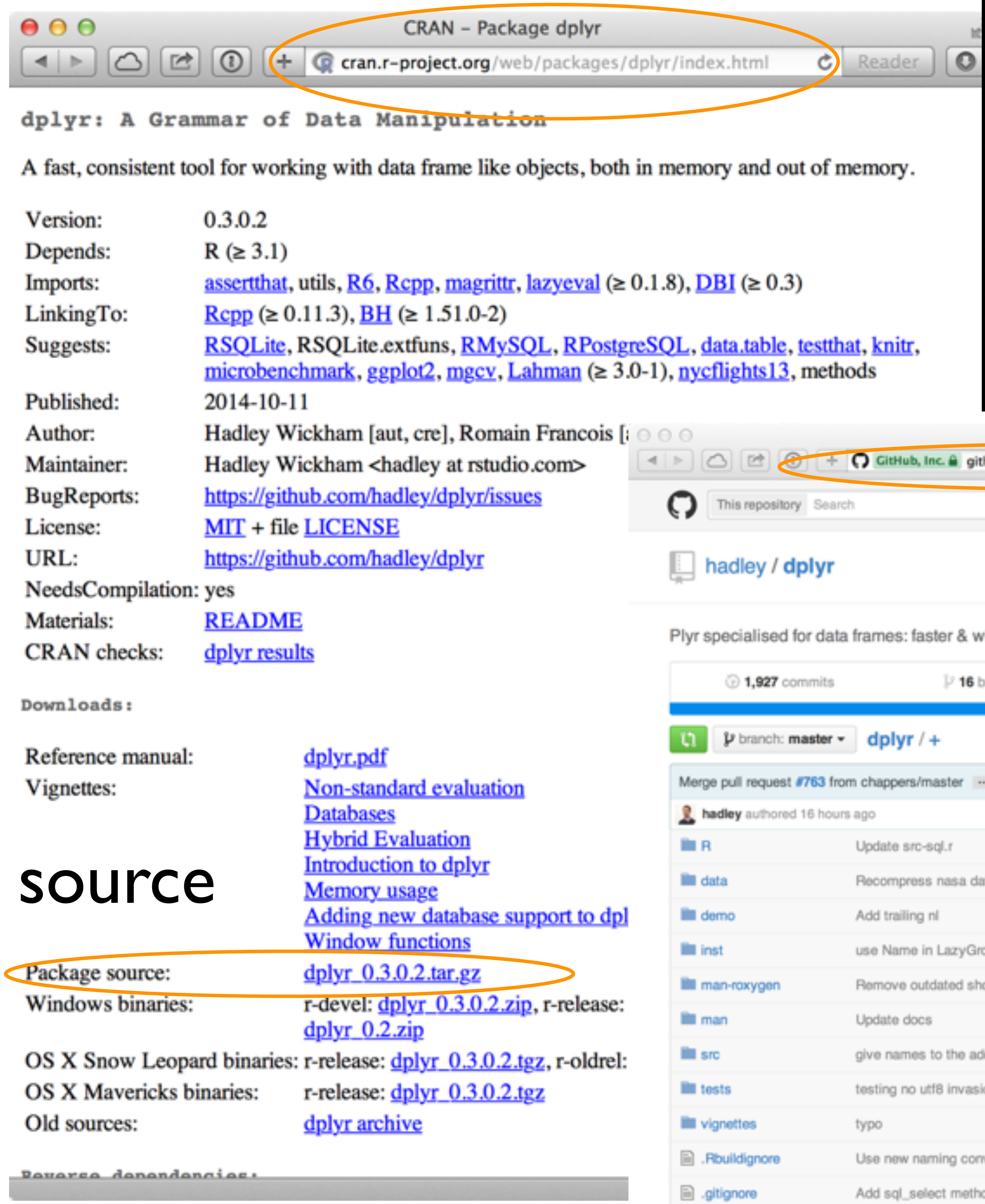
Systematically compare the files and directories of the package when it exists in ...

  source form

    vs

  installed form

consult GitHub or CRAN for source
consult your local library for installed form



CRAN – Package dplyr

cran.r-project.org/web/packages/dplyr/index.html

dplyr: A Grammar of Data Manipulation

A fast, consistent tool for working with data frame like objects, both in memory and out of memory.

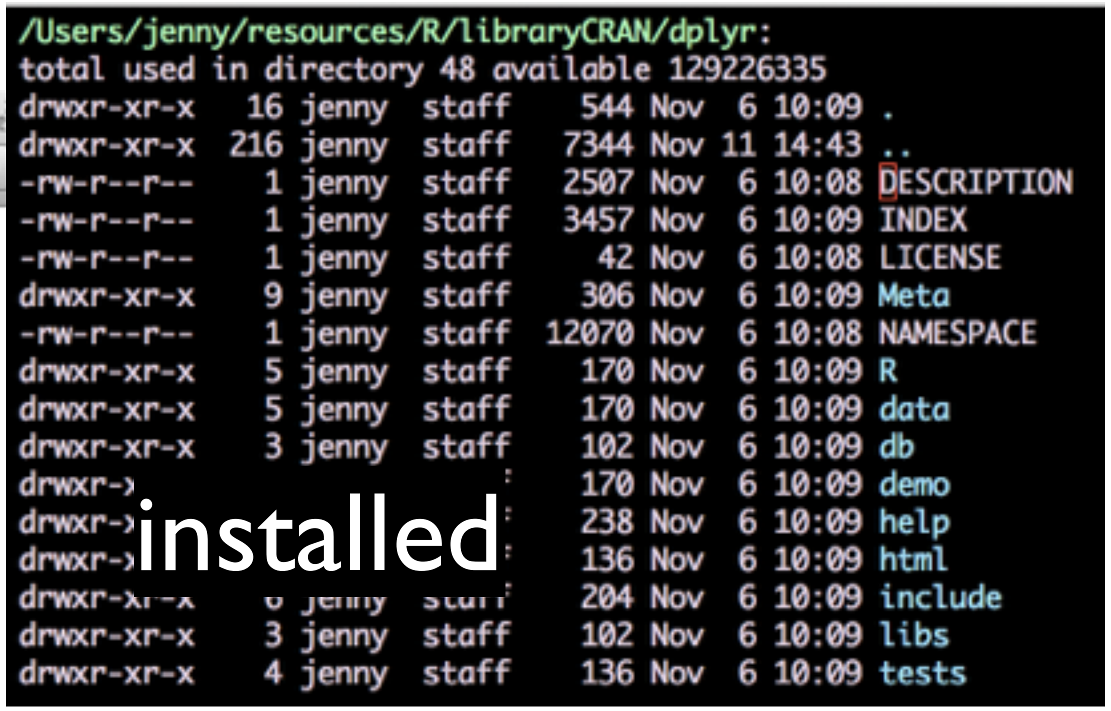
Version:	0.3.0.2
Depends:	R (≥ 3.1)
Imports:	assertthat, utils, R6, Rcpp, magrittr, lazyeval (≥ 0.1.8), DBI (≥ 0.3)
LinkingTo:	Rcpp (≥ 0.11.3), BH (≥ 1.51.0-2)
Suggests:	RSQLite, RSQLite.extfuns, RMySQL, RPostgreSQL, data.table, testthat, knitr, microbenchmark, ggplot2, mgcv, Lahman (≥ 3.0-1), nycflights13, methods
Published:	2014-10-11
Author:	Hadley Wickham [aut, cre], Romain Francois [
Maintainer:	Hadley Wickham <hadley at rstudio.com>
BugReports:	https://github.com/hadley/dplyr/issues
License:	MIT + file LICENSE
URL:	https://github.com/hadley/dplyr
NeedsCompilation:	yes
Materials:	README
CRAN checks:	dplyr results

Downloads:

Reference manual:	dplyr.pdf
Vignettes:	Non-standard evaluation Databases Hybrid Evaluation Introduction to dplyr Memory usage Adding new database support to dpl Window functions
Package source:	dplyr_0.3.0.2.tar.gz
Windows binaries:	r-devel: dplyr_0.3.0.2.zip, r-release: dplyr_0.2.zip
OS X Snow Leopard binaries:	r-release: dplyr_0.3.0.2.tgz, r-oldrel:
OS X Mavericks binaries:	r-release: dplyr_0.3.0.2.tgz
Old sources:	dplyr archive

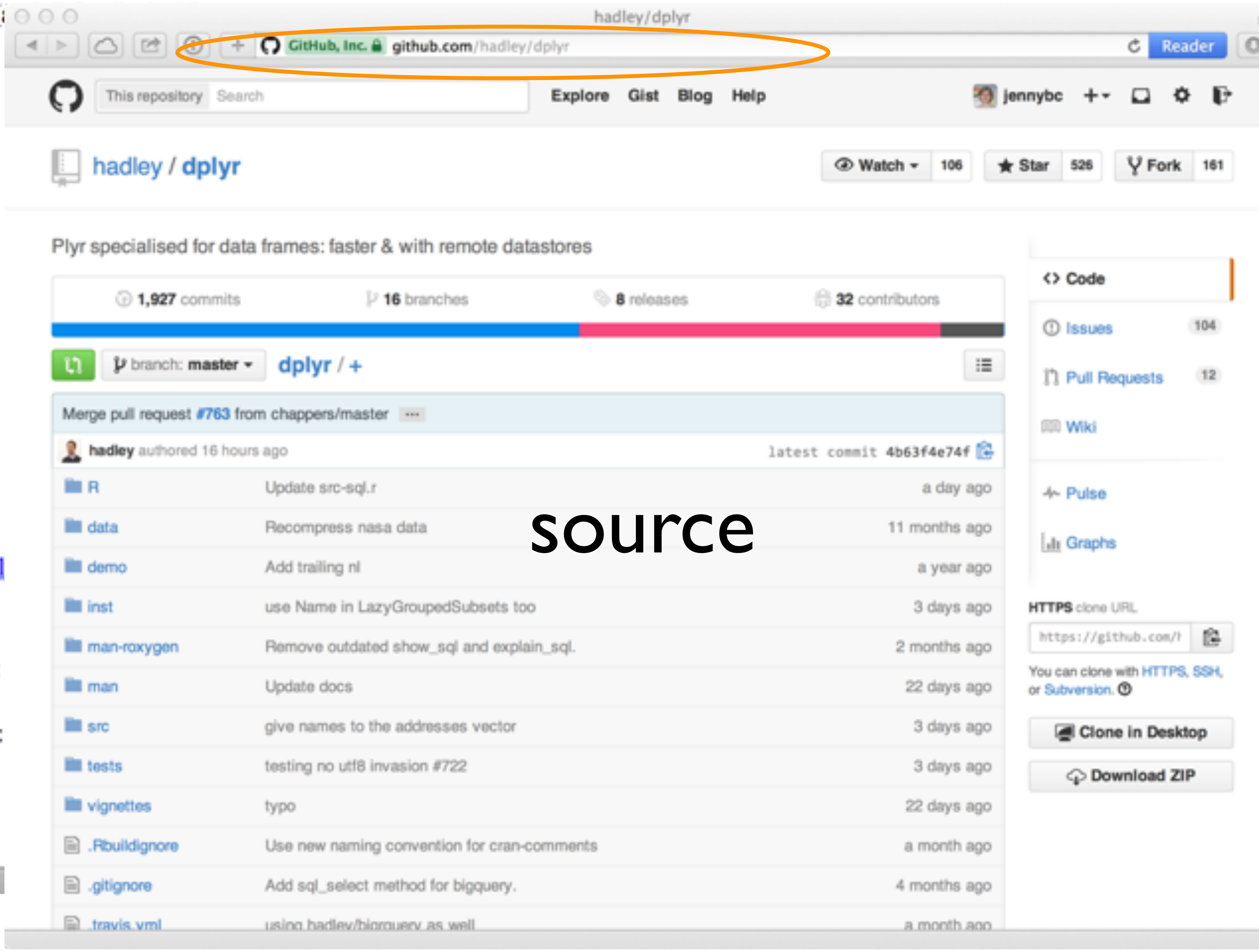
Reverse dependencies:

source



```
/Users/jenny/resources/R/libraryCRAN/dplyr:
total used in directory 48 available 129226335
drwxr-xr-x   16 jenny  staff    544 Nov  6 10:09 .
drwxr-xr-x  216 jenny  staff   7344 Nov 11 14:43 ..
-rw-r--r--    1 jenny  staff   2507 Nov  6 10:08 DESCRIPTION
-rw-r--r--    1 jenny  staff   3457 Nov  6 10:09 INDEX
-rw-r--r--    1 jenny  staff     42 Nov  6 10:08 LICENSE
drwxr-xr-x    9 jenny  staff    306 Nov  6 10:09 Meta
-rw-r--r--    1 jenny  staff  12070 Nov  6 10:08 NAMESPACE
drwxr-xr-x    5 jenny  staff    170 Nov  6 10:09 R
drwxr-xr-x    5 jenny  staff    170 Nov  6 10:09 data
drwxr-xr-x    3 jenny  staff    102 Nov  6 10:09 db
drwxr-x                         170 Nov  6 10:09 demo
drwxr-x                         238 Nov  6 10:09 help
drwxr-x                         136 Nov  6 10:09 html
drwxr-xr-x    6 jenny  staff    204 Nov  6 10:09 include
drwxr-xr-x    3 jenny  staff    102 Nov  6 10:09 libs
drwxr-xr-x    4 jenny  staff    136 Nov  6 10:09 tests
```

installed



hadley/dplyr

GitHub, Inc. github.com/hadley/dplyr

hadley / dplyr

Plyr specialised for data frames: faster & with remote datastores

1,927 commits · 16 branches · 8 releases · 32 contributors

branch: master · dplyr / +

Merge pull request #763 from chappers/master

hadley authored 16 hours ago · latest commit 4b63f4e74f

R	Update src-sql.r	a day ago
data	Recompress nasa data	11 months ago
demo	Add trailing nl	a year ago
inst	use Name in LazyGroupedSubsets too	3 days ago
man-roxygen	Remove outdated show_sql and explain_sql.	2 months ago
man	Update docs	22 days ago
src	give names to the addresses vector	3 days ago
tests	testing no utf8 invasion #722	3 days ago
vignettes	typo	22 days ago
.Rbuildignore	Use new naming convention for cran-comments	a month ago
.gitignore	Add sql_select method for bigquery.	4 months ago
.travis.yml	using hadley/bigrquery as well	a month ago

source

Example: devtools package in source form vs binary/installed form

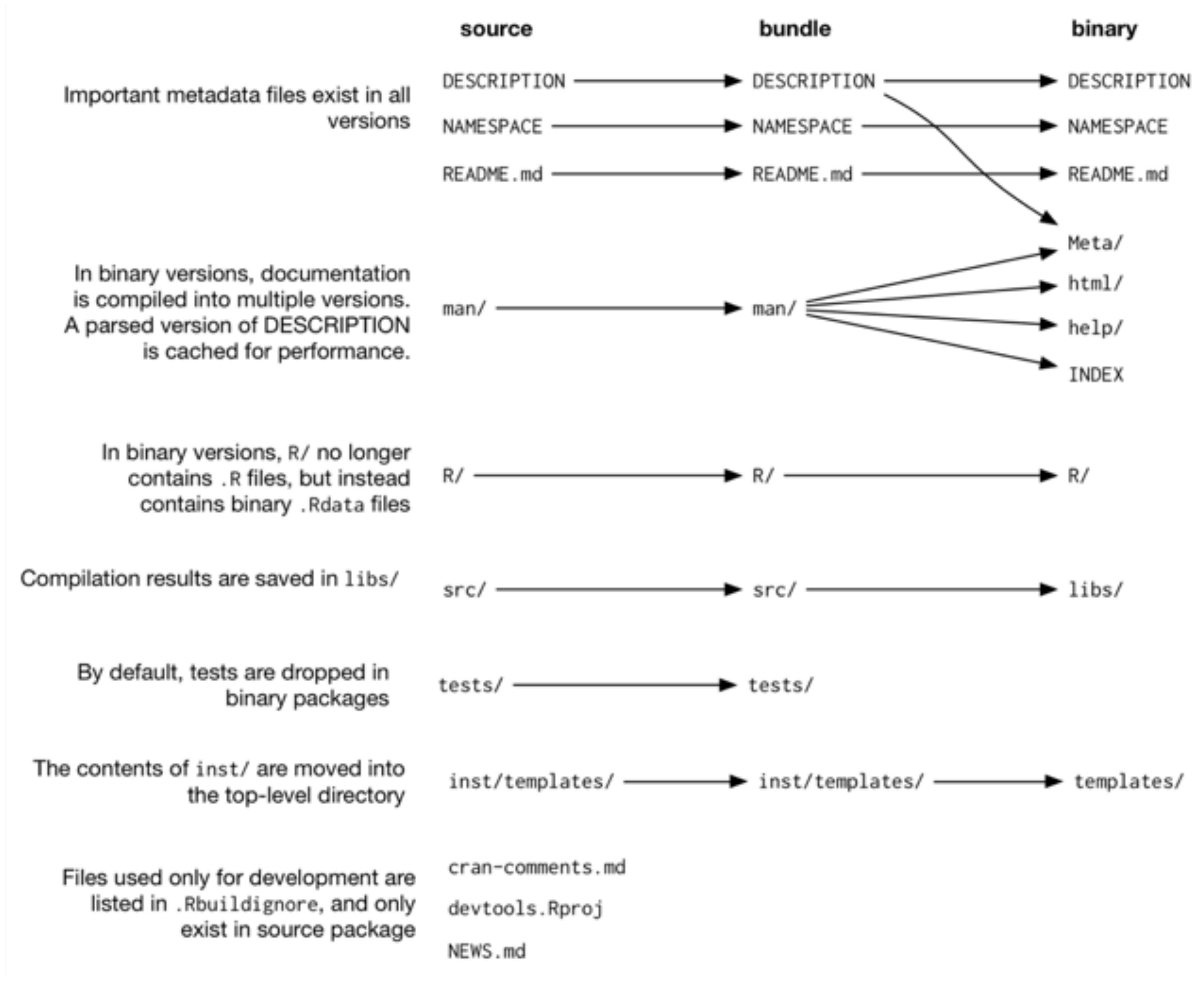
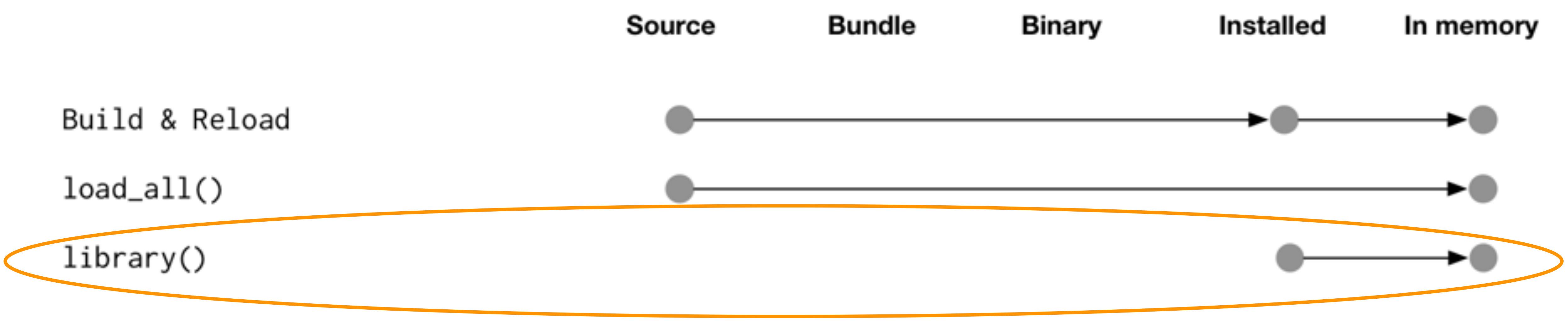

source
bundle
binary
Important metadata files exist in all versions
DESCRIPTION
NAMESPACE
README.md
Meta/
In binary versions, documentation is compiled into multiple versions. A parsed version of DESCRIPTION is cached for performance.
man/
html/
help/
INDEX
In binary versions, R/ no longer contains .R files, but instead contains binary .Rdata files
R/
Compilation results are saved in libs/
src/
libs/
By default, tests are dropped in binary packages
tests/
The contents of inst/ are moved into the top-level directory
inst/templates/
templates/
Files used only for development are listed in .Rbuildignore, and only exist in source package
cran-comments.md
devtools.Rproj
NEWS.md

Figure from Hadley Wickham's book, R packages
http://r-pkgs.had.co.nz
https://github.com/hadley/r-pkgs/blob/master/diagrams/package-files.png

How do installed packages get into memory?



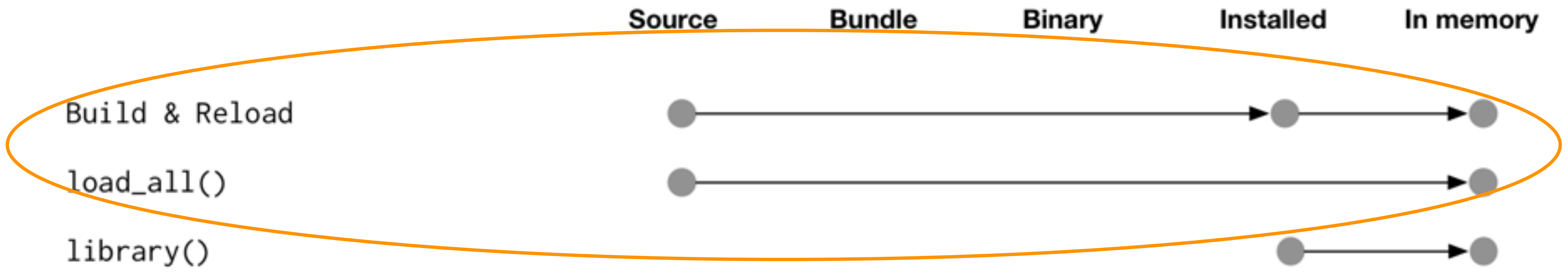
So far, you've only put packages into memory

- that are already installed
- that live in your default library
- using the `library()` function

Figure from Hadley Wickham's book, R packages
http://r-pkgs.had.co.nz
https://github.com/hadley/r-pkgs/blob/master/diagrams/loading.png

If you want to develop your own package, you must

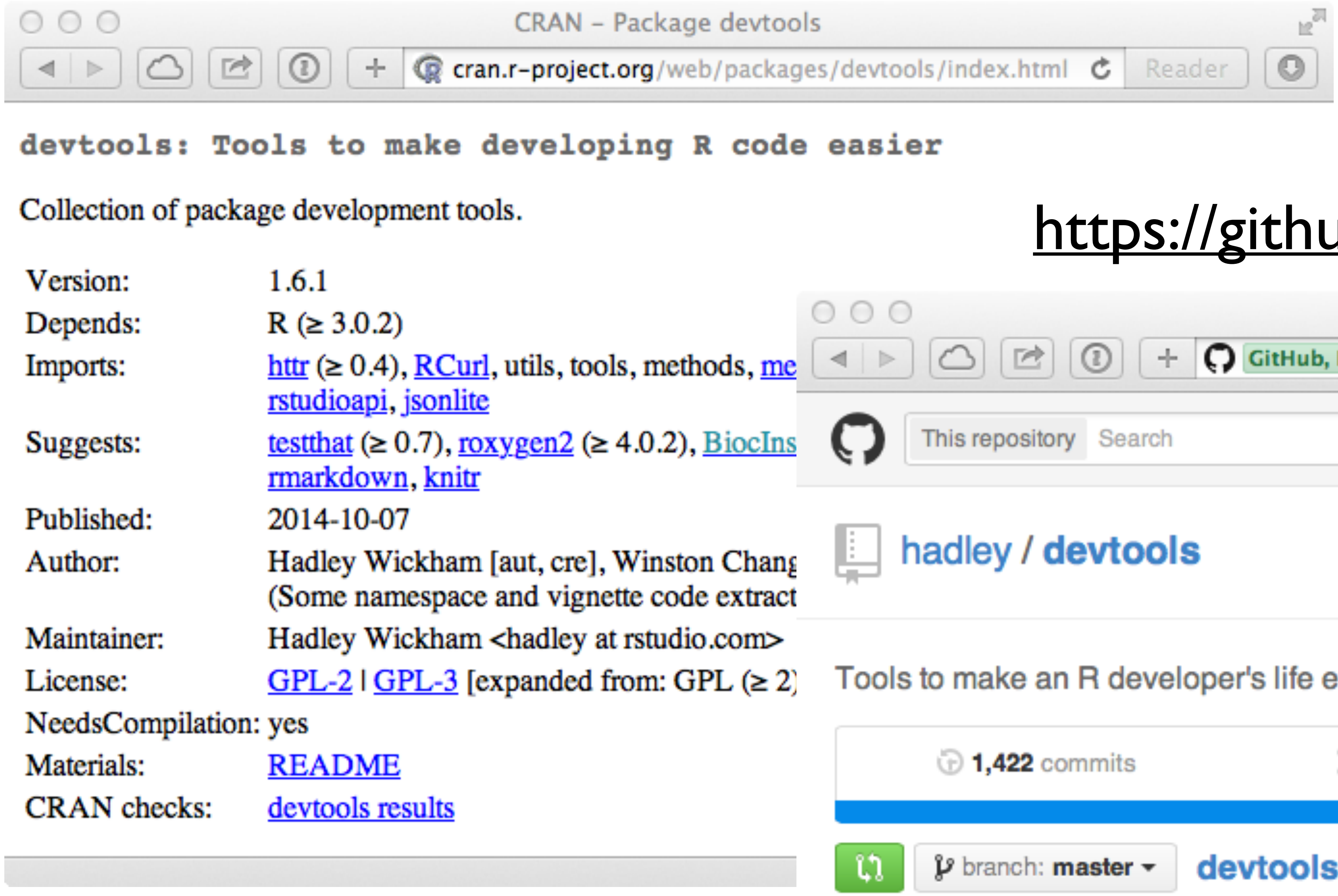
- write package source
- document, test, check it
- install it, load it, use it
- several times in a day



The `devtools` package reduces the agony of this.

RStudio has good (and constantly improving) integration with `devtools`.

http://cran.r-project.org/web/packages/devtools/index.html

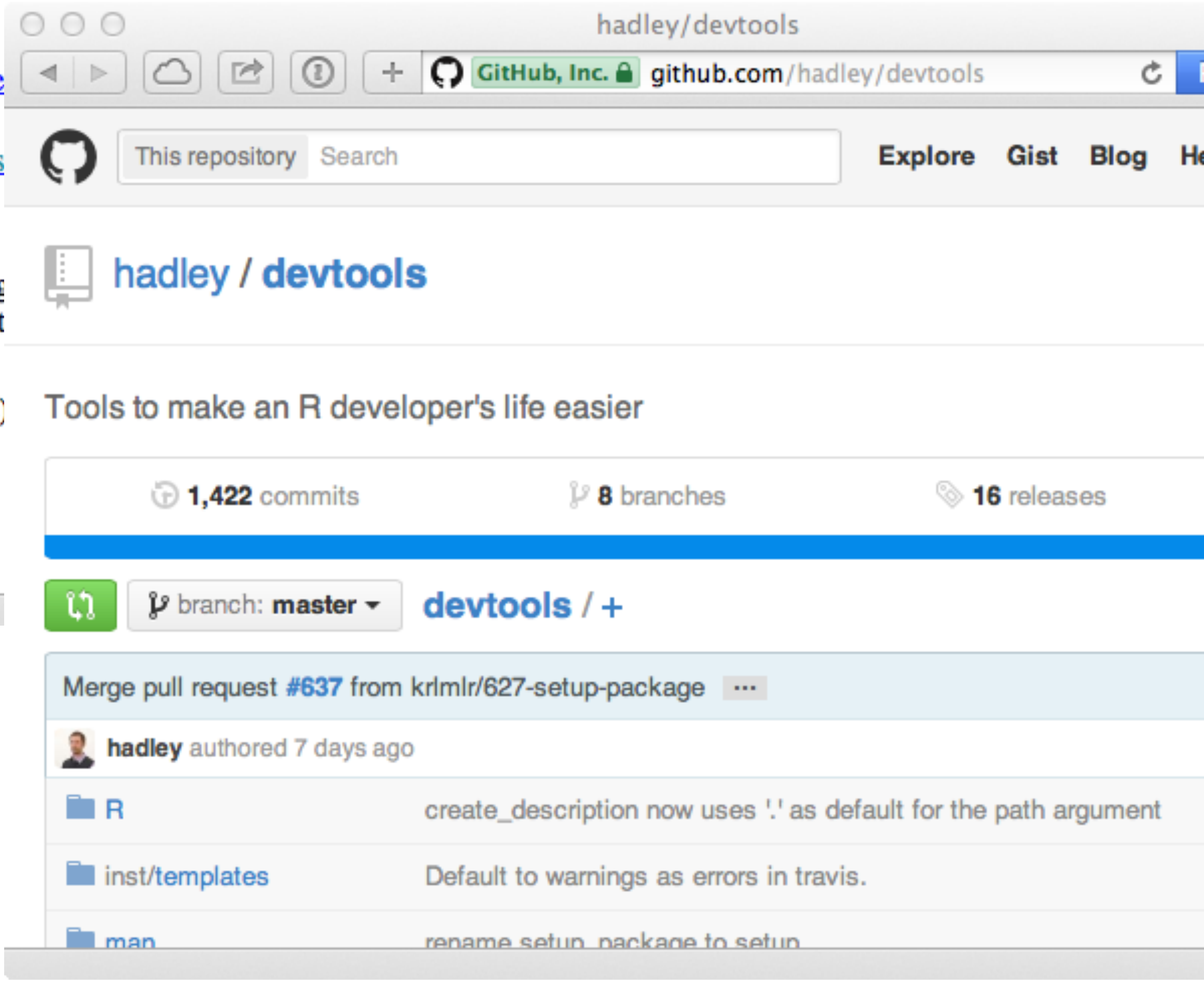


devtools: Tools to make developing R code easier

Collection of package development tools.

Version:	1.6.1
Depends:	R (≥ 3.0.2)
Imports:	httr (≥ 0.4), RCurl, utils, tools, methods, me rstudioapi, jsonlite
Suggests:	testthat (≥ 0.7), roxygen2 (≥ 4.0.2), BiocIns rmarkdown, knitr
Published:	2014-10-07
Author:	Hadley Wickham [aut, cre], Winston Chang (Some namespace and vignette code extract
Maintainer:	Hadley Wickham <hadley at rstudio.com>
License:	GPL-2 | GPL-3 [expanded from: GPL (≥ 2)
NeedsCompilation:	yes
Materials:	README
CRAN checks:	devtools results

https://github.com/hadley/devtools

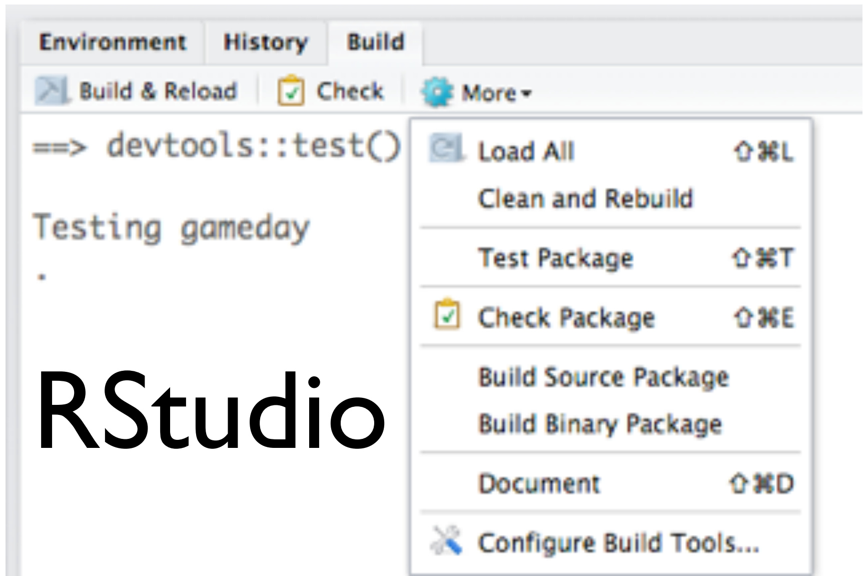


Tools to make an R developer's life easier

Merge pull request #637 from krlmlr/627-setup-package

hadley authored 7 days ago

R	create_description now uses '.' as default for the path argument
inst/templates	Default to warnings as errors in travis.
man	rename setup_package to setup

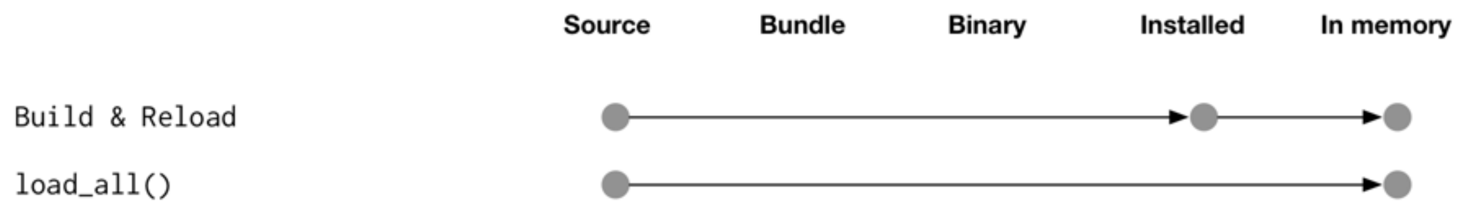


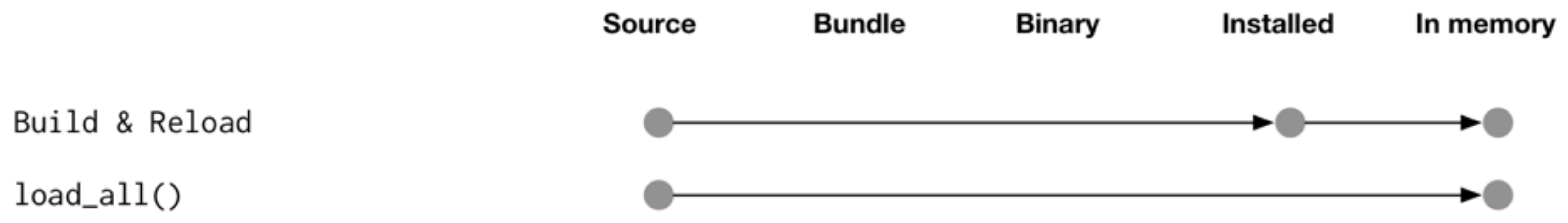
```
==> devtools::test()

Testing gameday
.
```

RStudio

Your first devtools.

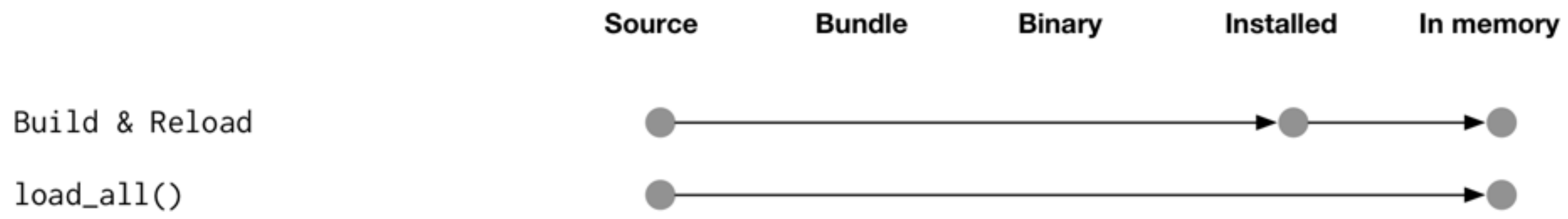
devtools::create()	set up a new package
devtools::document() RStudio > Build> More > Document	wrapper that uses roxygen2 to make formal documentation and NAMESPACE
RStudio Build and Reload	allow you to use your package and see how things are going  Source Bundle Binary Installed In memory Build & Reload load_all()
devtools::load_all() RStudio > Build> More > Load All	



`devtools::load_all()` is to package development

as

interactive “stepping through” code is to script development



RStudio's Build & Reload is to package development

as

`source()` or RStudio's "Source" or `Rscript foo.R` is to script development

You'll go through lots of cycles of editing code, trying it interactively ...

then ... `load_all()` to quickly emulate building and installing ...

oops, something is broken!
... more editing to fix the code, etc ...

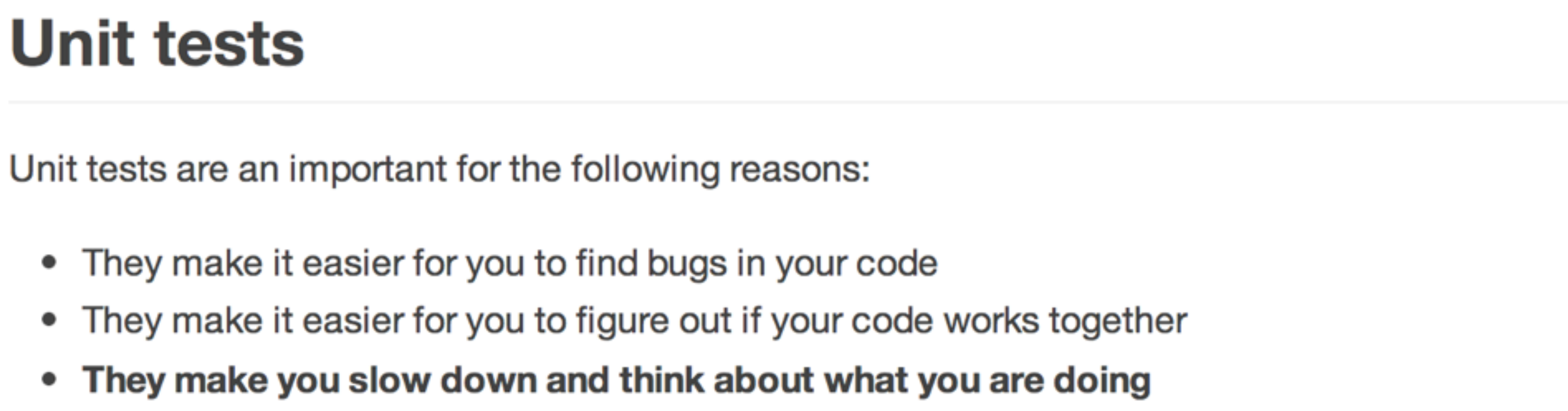
then ... Build and Reload

interleaved with these efforts aimed at adding functionality, you will be doing other crucial work

- keep `DESCRIPTION` and the documentation in the `#'` roxygen comments up-to-date
- periodically run `devtools::document()` to regenerate help files and `NAMESPACE`
- periodically run `R CMD check` to see if your package would pass muster with CRAN
- write and run formal unit tests
- write one or more vignettes

Documentation > Usability > Speed > Statistical superiority

Figure from Jeff Leek's guide to writing R packages
https://github.com/jtleek/rpackages
https://raw.githubusercontent.com/jtleek/rpackages/master/documentation.png



Unit tests

Unit tests are an important for the following reasons:

- They make it easier for you to find bugs in your code
- They make it easier for you to figure out if your code works together
- **They make you slow down and think about what you are doing**

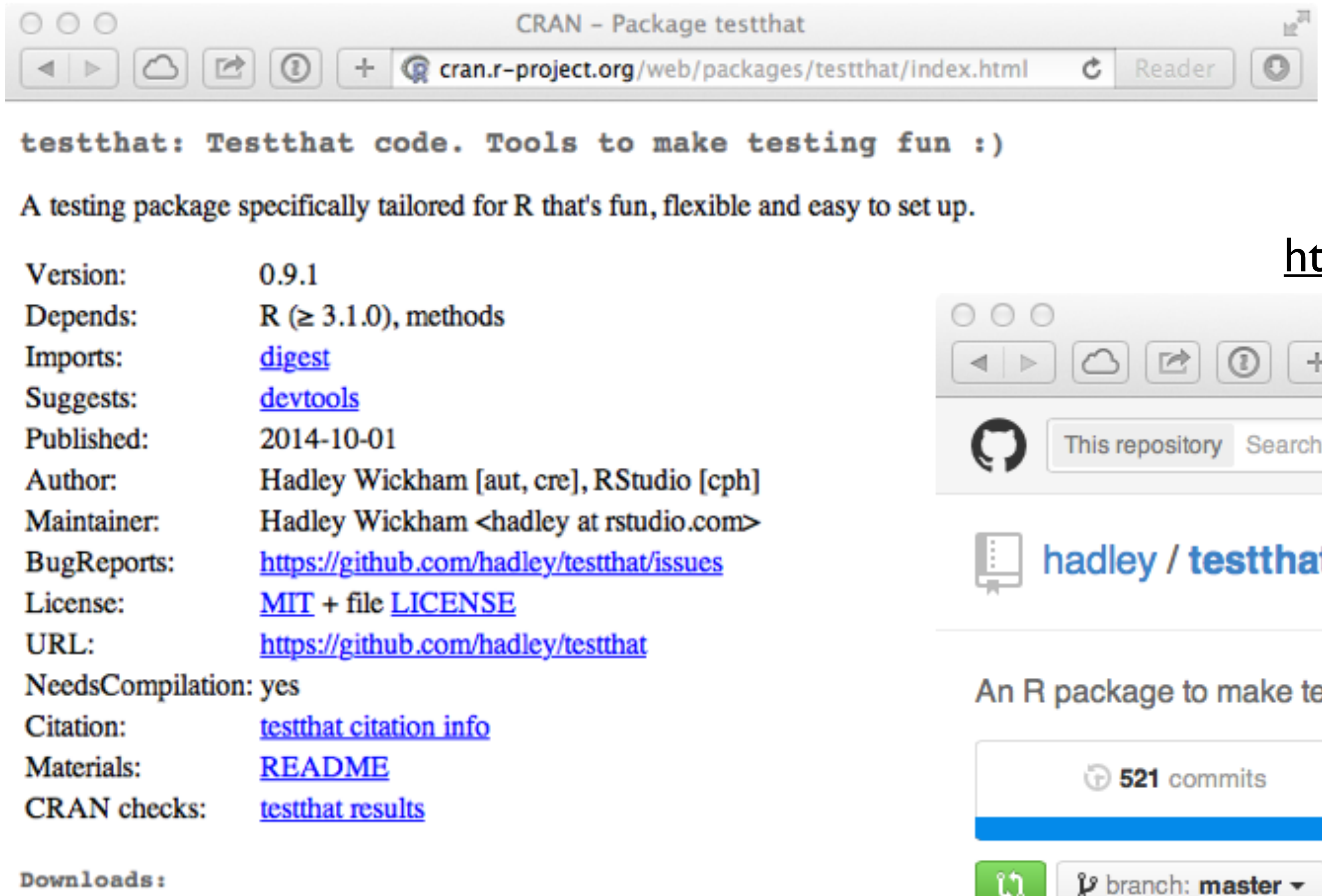
Figure from Jeff Leek's guide to writing R packages
https://github.com/jtleek/rpackages

testthat: Get Started with Testing

by Hadley Wickham The R Journal Vol. 3/1, June 2011

http://journal.r-project.org/archive/2011-1/RJournal_2011-1_Wickham.pdf

http://cran.r-project.org/web/packages/testthat/index.html



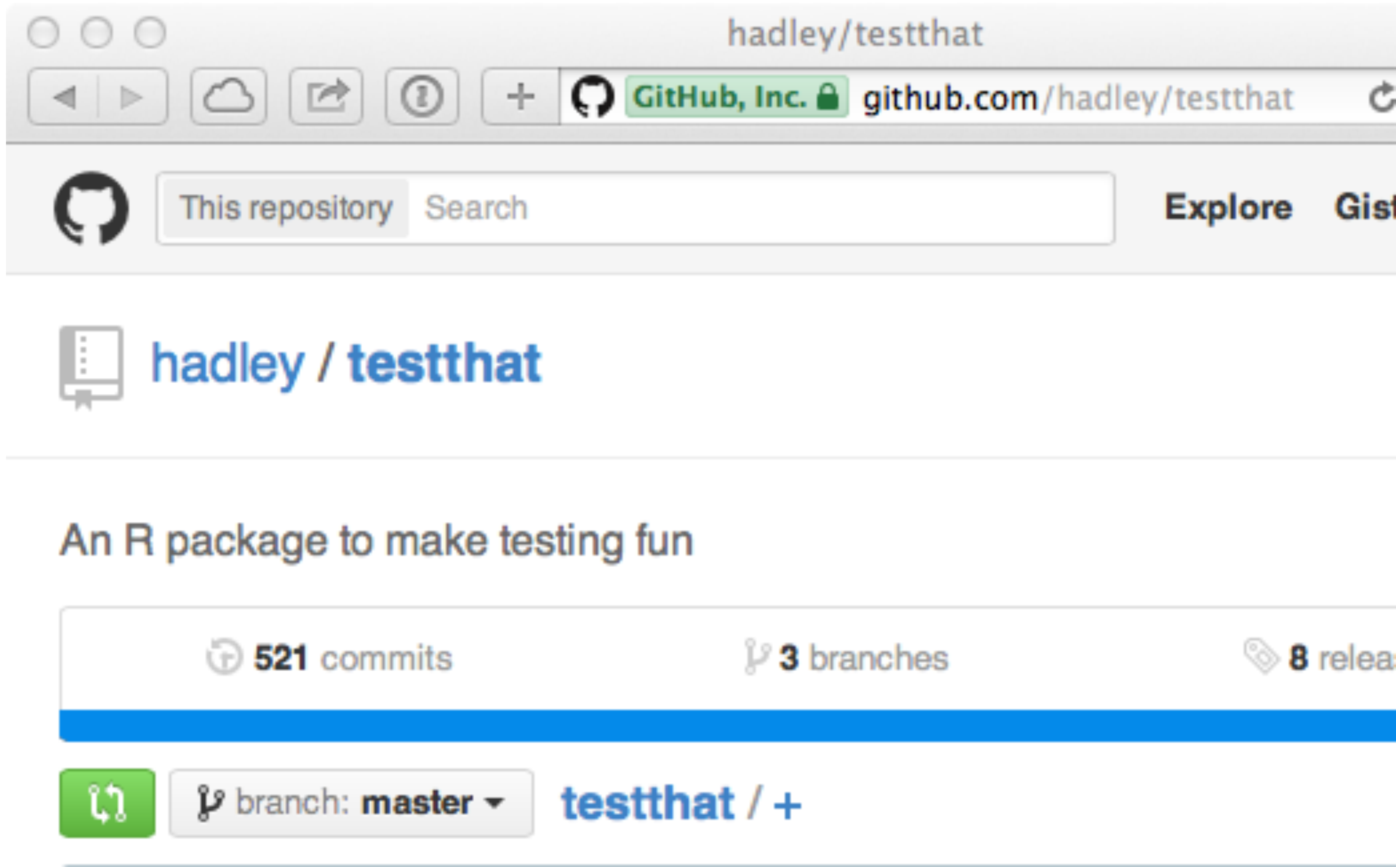
testthat: Testthat code. Tools to make testing fun :)

A testing package specifically tailored for R that's fun, flexible and easy to set up.

Version:	0.9.1
Depends:	R (≥ 3.1.0), methods
Imports:	digest
Suggests:	devtools
Published:	2014-10-01
Author:	Hadley Wickham [aut, cre], RStudio [cph]
Maintainer:	Hadley Wickham <hadley at rstudio.com>
BugReports:	https://github.com/hadley/testthat/issues
License:	MIT + file LICENSE
URL:	https://github.com/hadley/testthat
NeedsCompilation:	yes
Citation:	testthat citation info
Materials:	README
CRAN checks:	testthat results

Downloads:

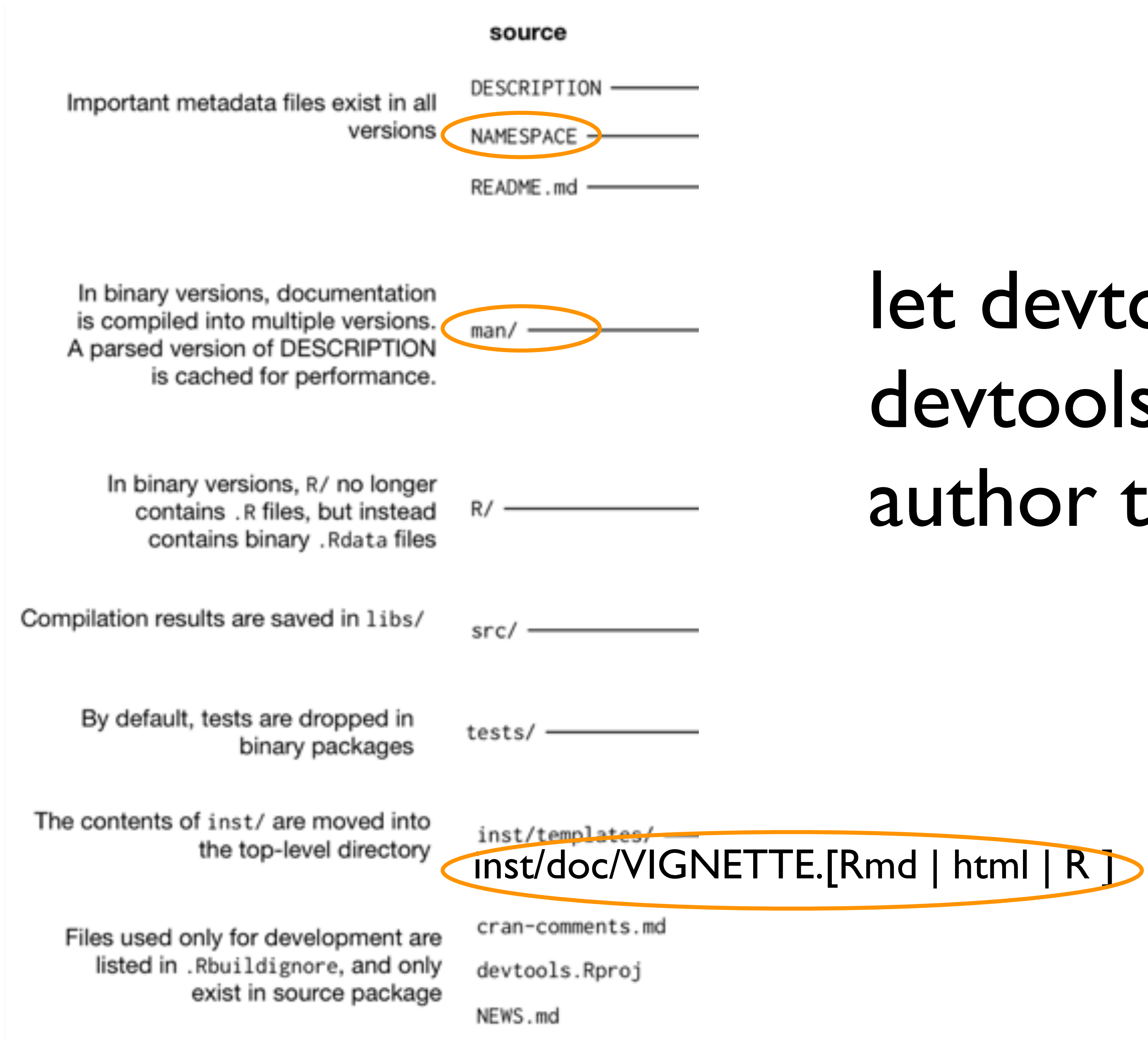
https://github.com/hadley/testthat



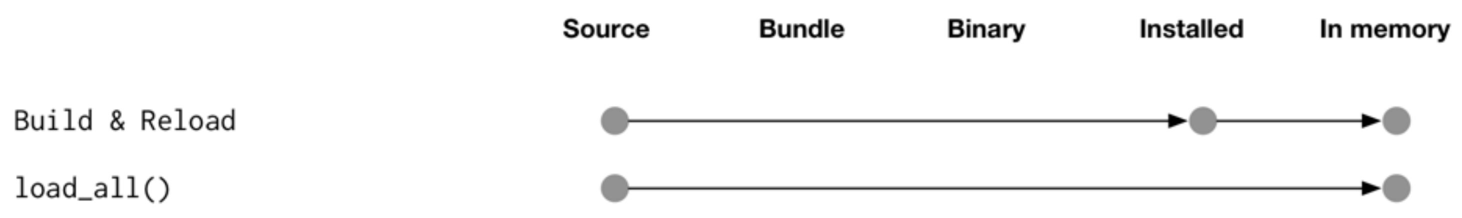
An R package to make testing fun

Which files and directories do you NEVER touch by hand?

at least, in our recommended devtools driven workflow


source
Important metadata files exist in all versions
DESCRIPTION
NAMESPACE
README.md
In binary versions, documentation is compiled into multiple versions. A parsed version of DESCRIPTION is cached for performance.
man/
In binary versions, R/ no longer contains .R files, but instead contains binary .Rdata files
R/
Compilation results are saved in libs/
src/
By default, tests are dropped in binary packages
tests/
The contents of inst/ are moved into the top-level directory
inst/templates/
inst/doc/VIGNETTE.[Rmd | html | R]
Files used only for development are listed in .Rbuildignore, and only exist in source package
cran-comments.md
devtools.Rproj
NEWS.md

let devtools::document() and devtools:: build_vignettes() author these files for you

devtools::create()	set up a new package
devtools::document() RStudio > Build> More > Document	wrapper that uses roxygen2 to make formal documentation and NAMESPACE
RStudio > Build and Reload	 Source Bundle Binary Installed In memory Build & Reload load_all()
devtools::load_all() RStudio > Build> More > Load All	

devtools::use_vignette()
devtools:build_vignettes()
	sets up and renders vignettes, respectively

R CMD check devtools::check() RStudio > Check	see if your package would pass muster with CRAN
devtools::test() RStudio > Build> More > Test Package	wrapper that uses testthat to run formal unit tests