CT5102: Programming for Data Analytics

Lecture 10: Packages

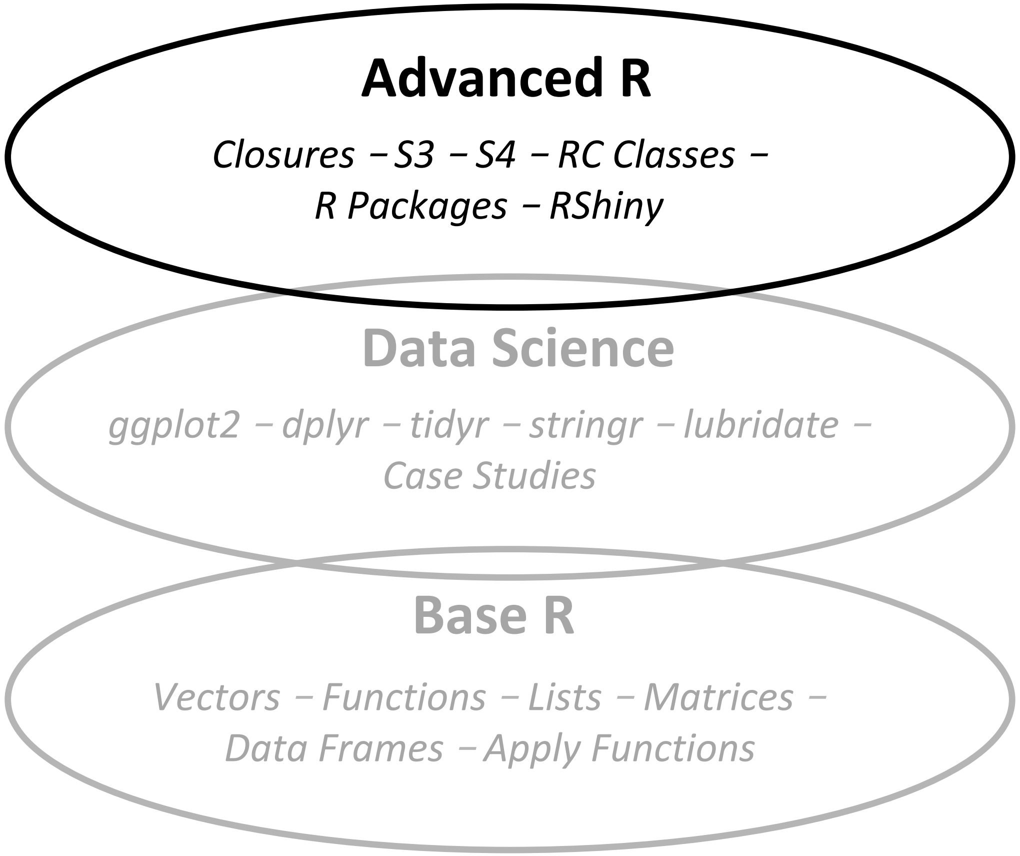
Dr. Jim Duggan,

School of Engineering & Informatics

National University of Ireland Galway.

https://github.com/JimDuggan/CT5102

“Organise, test, document and share your code” (Wickham)



Advanced R

Closures – S3 – S4 – RC Classes – R Packages – RShiny

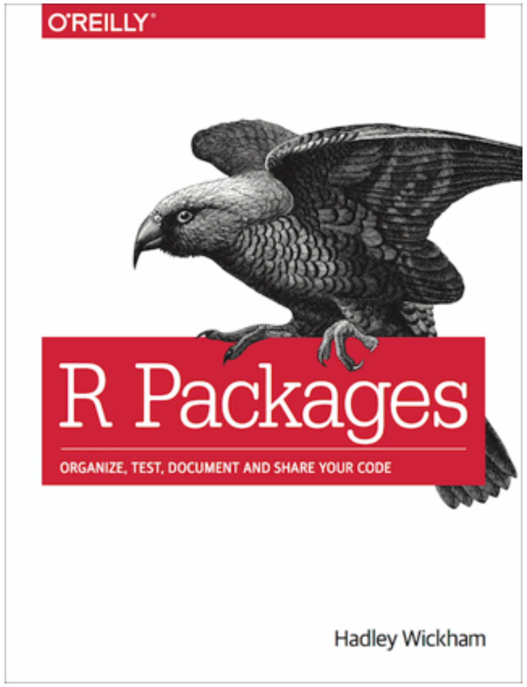
Data Science

ggplot2 – dplyr – tidyr – stringr – lubridate – Case Studies

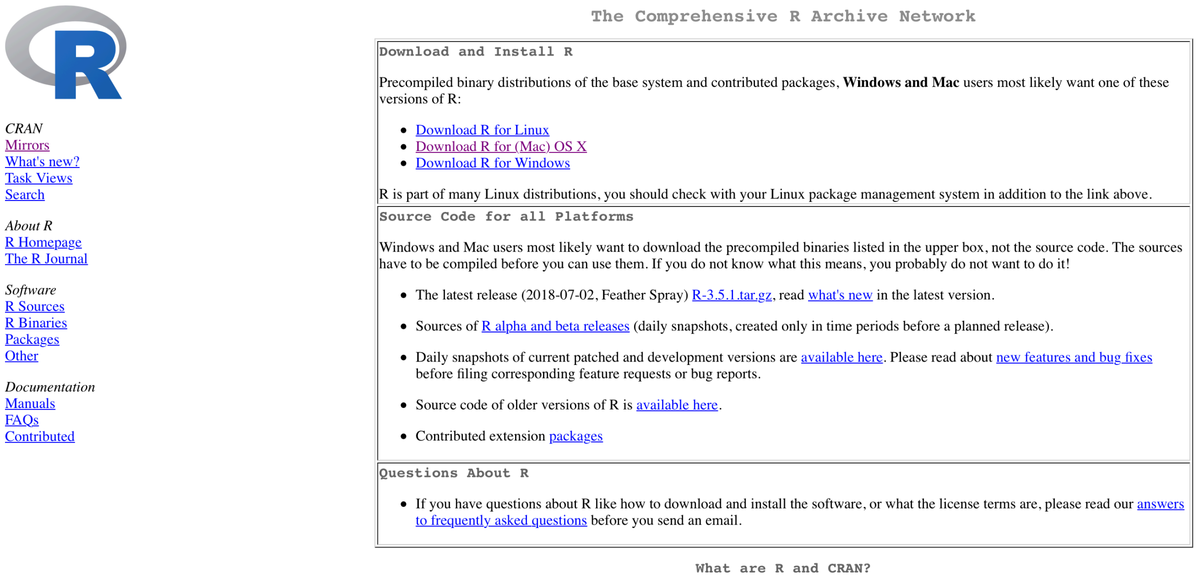
Base R

Vectors – Functions – Lists – Matrices – Data Frames – Apply Functions

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



https://cran.r-project.org



CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

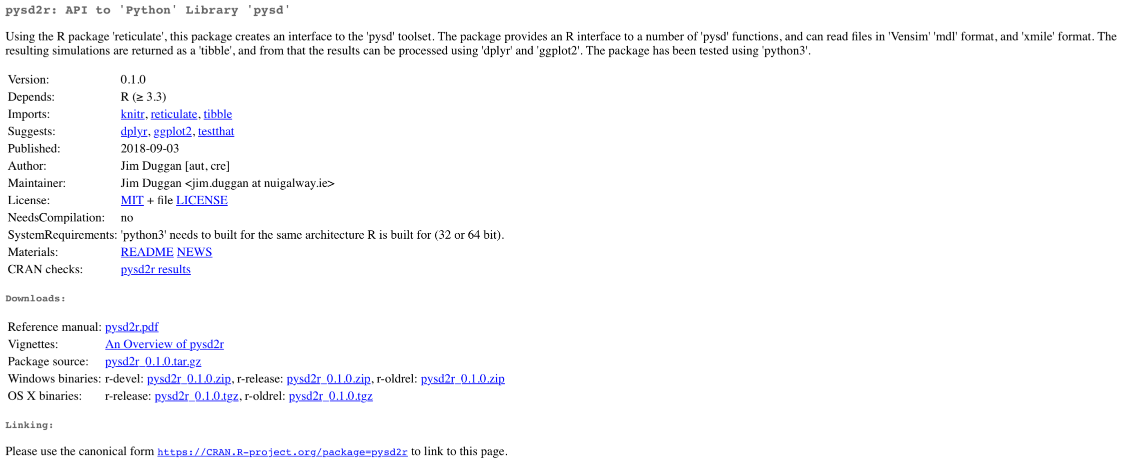
Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2018-07-02, Feather Spray) R-3.5.1.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

What are R and CRAN?



pysd2r: API to 'Python' Library 'pysd'

Using the R package 'reticulate', this package creates an interface to the 'pysd' toolset. The package provides an R interface to a number of 'pysd' functions, and can read files in 'Vensim' 'mdl' format, and 'xmile' format. The resulting simulations are returned as a 'tibble', and from that the results can be processed using 'dplyr' and 'ggplot2'. The package has been tested using 'python3'.

Version:	0.1.0
Depends:	R (≥ 3.3)
Imports:	knitr, reticulate, tibble
Suggests:	dplyr, ggplot2, testthat
Published:	2018-09-03
Author:	Jim Duggan [aut, cre]
Maintainer:	Jim Duggan <jim.duggan at nuigalway.ie>
License:	MIT + file LICENSE
NeedsCompilation:	no
SystemRequirements:	'python3' needs to built for the same architecture R is built for (32 or 64 bit).
Materials:	README NEWS
CRAN checks:	pysd2r results

Downloads:

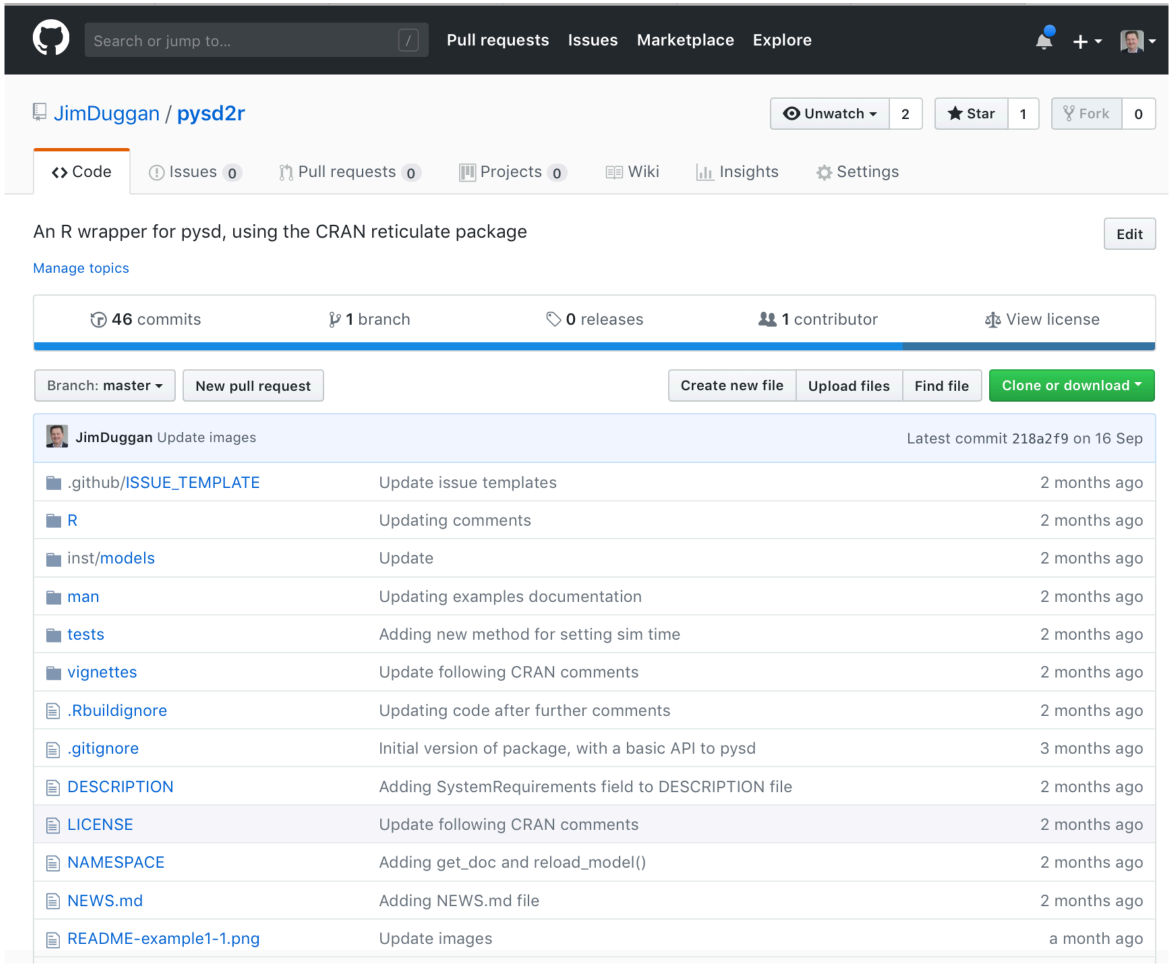
Reference manual:	pysd2r.pdf
Vignettes:	An Overview of pysd2r
Package source:	pysd2r_0.1.0.tar.gz
Windows binaries:	r-devel: pysd2r_0.1.0.zip, r-release: pysd2r_0.1.0.zip, r-oldrel: pysd2r_0.1.0.zip
OS X binaries:	r-release: pysd2r_0.1.0.tgz, r-oldrel: pysd2r_0.1.0.tgz

Linking:

Please use the canonical form `https://CRAN.R-project.org/package=pysd2r` to link to this page.

Packages also on github

https://github.com/JimDuggan/pysd2r



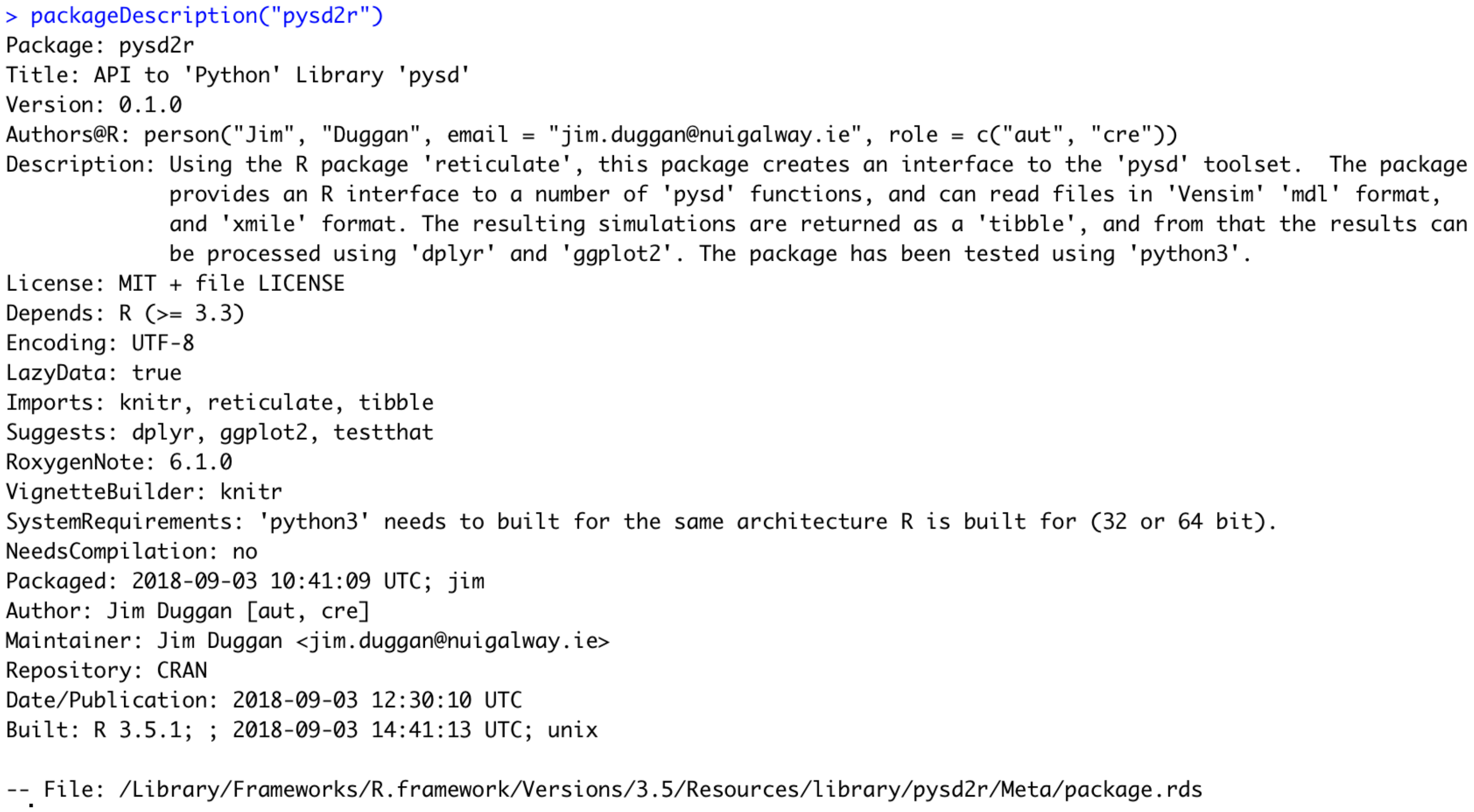
Search or jump to...
Pull requests Issues Marketplace Explore
JimDuggan / pysd2r
Unwatch 2 Star 1 Fork 0
Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings
Edit
Manage topics
46 commits 1 branch 0 releases 1 contributor View license
Branch: master New pull request Create new file Upload files Find file Clone or download

An R wrapper for pysd, using the CRAN reticulate package

JimDuggan Update images — Latest commit 218a2f9 on 16 Sep

.github/ISSUE_TEMPLATE	Update issue templates	2 months ago
R	Updating comments	2 months ago
inst/models	Update	2 months ago
man	Updating examples documentation	2 months ago
tests	Adding new method for setting sim time	2 months ago
vignettes	Update following CRAN comments	2 months ago
.Rbuildignore	Updating code after further comments	2 months ago
.gitignore	Initial version of package, with a basic API to pysd	3 months ago
DESCRIPTION	Adding SystemRequirements field to DESCRIPTION file	2 months ago
LICENSE	Update following CRAN comments	2 months ago
NAMESPACE	Adding get_doc and reload_model()	2 months ago
NEWS.md	Adding NEWS.md file	2 months ago
README-example1-1.png	Update images	a month ago

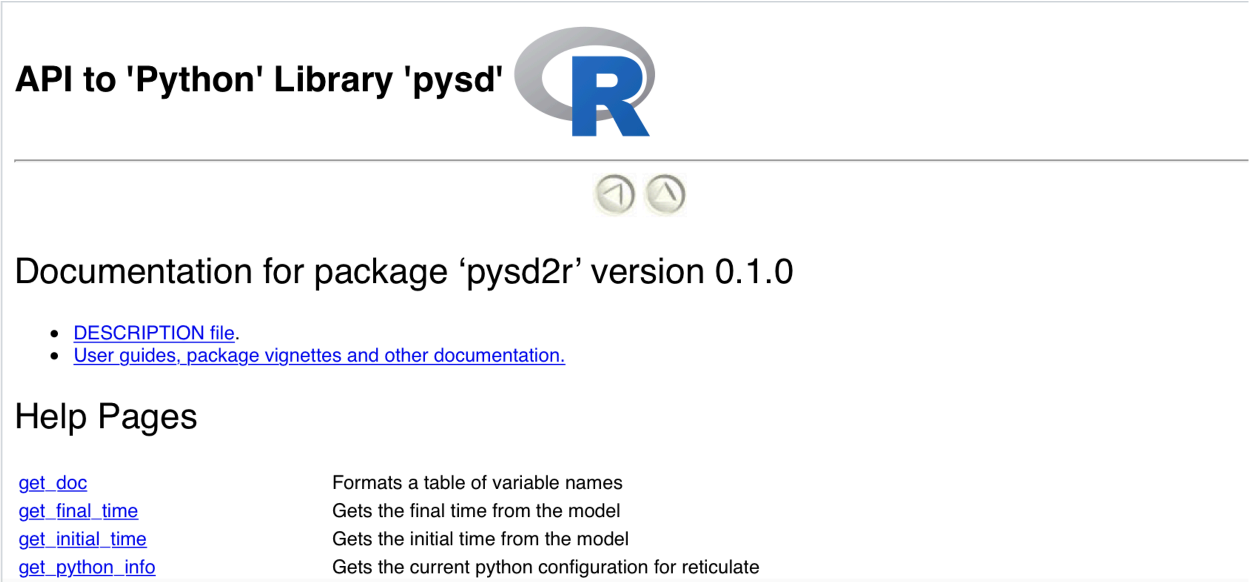
Accessing Package information…



```
> packageDescription("pysd2r")
Package: pysd2r
Title: API to 'Python' Library 'pysd'
Version: 0.1.0
Authors@R: person("Jim", "Duggan", email = "jim.duggan@nuigalway.ie", role = c("aut", "cre"))
Description: Using the R package 'reticulate', this package creates an interface to the 'pysd' toolset.  The package
             provides an R interface to a number of 'pysd' functions, and can read files in 'Vensim' 'mdl' format,
             and 'xmile' format. The resulting simulations are returned as a 'tibble', and from that the results can
             be processed using 'dplyr' and 'ggplot2'. The package has been tested using 'python3'.
License: MIT + file LICENSE
Depends: R (>= 3.3)
Encoding: UTF-8
LazyData: true
Imports: knitr, reticulate, tibble
Suggests: dplyr, ggplot2, testthat
RoxygenNote: 6.1.0
VignetteBuilder: knitr
SystemRequirements: 'python3' needs to built for the same architecture R is built for (32 or 64 bit).
NeedsCompilation: no
Packaged: 2018-09-03 10:41:09 UTC; jim
Author: Jim Duggan [aut, cre]
Maintainer: Jim Duggan <jim.duggan@nuigalway.ie>
Repository: CRAN
Date/Publication: 2018-09-03 12:30:10 UTC
Built: R 3.5.1; ; 2018-09-03 14:41:13 UTC; unix

-- File: /Library/Frameworks/R.framework/Versions/3.5/Resources/library/pysd2r/Meta/package.rds
```

Other useful functions…

```
> ls("package:pysd2r")
 [1] "get_doc"          "get_final_time"    "get_initial_time" "get_python_info"
 [5] "get_timestep"     "pysd_connect"      "read_vensim"      "read_xmile"
 [9] "reload_model"     "run_model"         "set_components"   "set_time_values"
>
>
> help(package=pysd2r)
>
>
> browseVignettes("pysd2r")
```



API to 'Python' Library 'pysd'

Documentation for package ‘pysd2r’ version 0.1.0

- DESCRIPTION file.
- User guides, package vignettes and other documentation.

Help Pages

get_doc	Formats a table of variable names
get_final_time	Gets the final time from the model
get_initial_time	Gets the initial time from the model
get_python_info	Gets the current python configuration for reticulate

Types of package

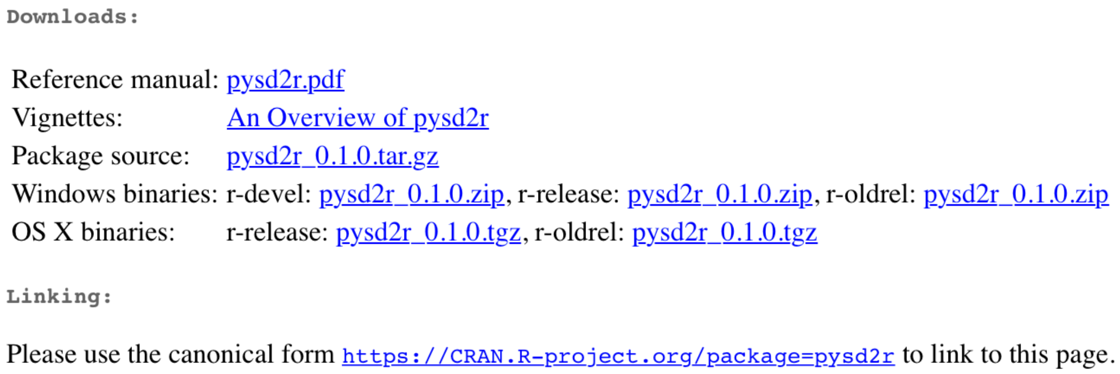
Source
Files in a directory

Bundle
Files in a .tar.gz

Binary
R code cached
C code compiled

Installed
Directory in *library*

In memory
library()



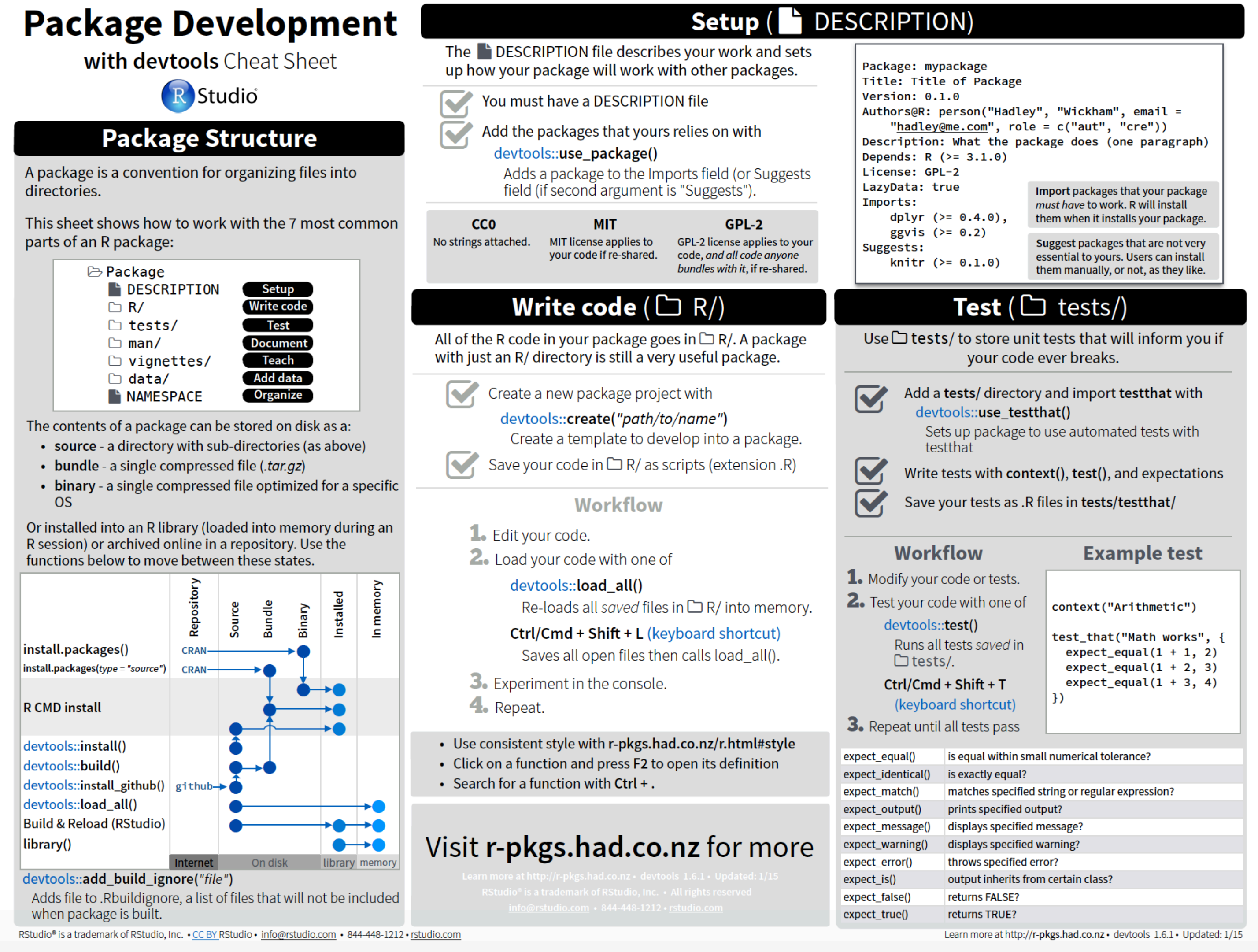
Downloads:

Reference manual:	pysd2r.pdf
Vignettes:	An Overview of pysd2r
Package source:	pysd2r_0.1.0.tar.gz
Windows binaries:	r-devel: pysd2r_0.1.0.zip, r-release: pysd2r_0.1.0.zip, r-oldrel: pysd2r_0.1.0.zip
OS X binaries:	r-release: pysd2r_0.1.0.tgz, r-oldrel: pysd2r_0.1.0.tgz

Linking:

Please use the canonical form `https://CRAN.R-project.org/package=pysd2r` to link to this page.

devtools - a key package



Package Development

with devtools Cheat Sheet

RStudio

Package Structure

A package is a convention for organizing files into directories.

This sheet shows how to work with the 7 most common parts of an R package:

Package	
DESCRIPTION	Setup
R/	Write code
tests/	Test
man/	Document
vignettes/	Teach
data/	Add data
NAMESPACE	Organize

The contents of a package can be stored on disk as a:

- **source** - a directory with sub-directories (as above)
- **bundle** - a single compressed file (*.tar.gz*)
- **binary** - a single compressed file optimized for a specific OS

Or installed into an R library (loaded into memory during an R session) or archived online in a repository. Use the functions below to move between these states.

devtools::**add_build_ignore**("*file*")

Adds file to .Rbuildignore, a list of files that will not be included when package is built.

Setup (DESCRIPTION)

The DESCRIPTION file describes your work and sets up how your package will work with other packages.

☑ You must have a DESCRIPTION file

☑ Add the packages that yours relies on with

devtools::**use_package()**

Adds a package to the Imports field (or Suggests field (if second argument is "Suggests").

CC0	MIT	GPL-2
No strings attached.	MIT license applies to your code if re-shared.	GPL-2 license applies to your code, *and all code anyone bundles with it*, if re-shared.

```
Package: mypackage
Title: Title of Package
Version: 0.1.0
Authors@R: person("Hadley", "Wickham", email =
    "hadley@me.com", role = c("aut", "cre"))
Description: What the package does (one paragraph)
Depends: R (>= 3.1.0)
License: GPL-2
LazyData: true
Imports:
    dplyr (>= 0.4.0),
    ggvis (>= 0.2)
Suggests:
    knitr (>= 0.1.0)
```

Import packages that your package *must have* to work. R will install them when it installs your package.

Suggest packages that are not very essential to yours. Users can install them manually, or not, as they like.

Write code (R/)

All of the R code in your package goes in R/. A package with just an R/ directory is still a very useful package.

☑ Create a new package project with

devtools::**create(**"*path/to/name*"**)**

Create a template to develop into a package.

☑ Save your code in R/ as scripts (extension .R)

Workflow

1. Edit your code.
2. Load your code with one of

   devtools::**load_all()**

   Re-loads all *saved* files in R/ into memory.

   Ctrl/Cmd + Shift + L (keyboard shortcut)

   Saves all open files then calls load_all().
3. Experiment in the console.
4. Repeat.

- Use consistent style with **r-pkgs.had.co.nz/r.html#style**
- Click on a function and press **F2** to open its definition
- Search for a function with **Ctrl + .**

Visit **r-pkgs.had.co.nz** for more

Learn more at http://r-pkgs.had.co.nz • devtools 1.6.1 • Updated: 1/15
RStudio® is a trademark of RStudio, Inc. • All rights reserved
info@rstudio.com • 844-448-1212 • rstudio.com

Test (tests/)

Use **tests/** to store unit tests that will inform you if your code ever breaks.

☑ Add a **tests/** directory and import **testthat** with

devtools::**use_testthat()**

Sets up package to use automated tests with testthat

☑ Write tests with **context()**, **test()**, and expectations

☑ Save your tests as .R files in **tests/testthat/**

Workflow

1. Modify your code or tests.
2. Test your code with one of

   devtools::**test()**

   Runs all tests *saved* in tests/.

   Ctrl/Cmd + Shift + T (keyboard shortcut)
3. Repeat until all tests pass

Example test

```
context("Arithmetic")

test_that("Math works", {
  expect_equal(1 + 1, 2)
  expect_equal(1 + 2, 3)
  expect_equal(1 + 3, 4)
})
```

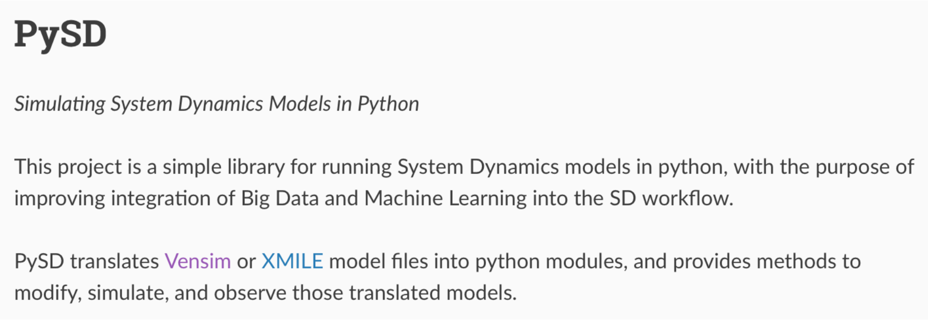
expect_equal()	is equal within small numerical tolerance?
expect_identical()	is exactly equal?
expect_match()	matches specified string or regular expression?
expect_output()	prints specified output?
expect_message()	displays specified message?
expect_warning()	displays specified warning?
expect_error()	throws specified error?
expect_is()	output inherits from certain class?
expect_false()	returns FALSE?
expect_true()	returns TRUE?

RStudio® is a trademark of RStudio, Inc. • CC BY RStudio • info@rstudio.com • 844-448-1212 • rstudio.com

Learn more at http://r-pkgs.had.co.nz • devtools 1.6.1 • Updated: 1/15

(1) Pick a name (Wickham)

- Only lowercase letters and numbers
- Add r (tidyr, stringr)
- Be googleable!
- Be memorable
- Once you have a name, you can create a package

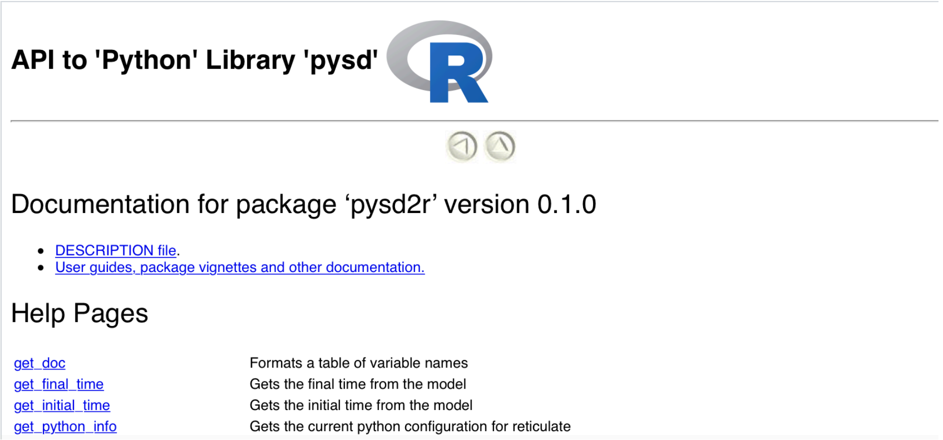


PySD

Simulating System Dynamics Models in Python

This project is a simple library for running System Dynamics models in python, with the purpose of improving integration of Big Data and Machine Learning into the SD workflow.

PySD translates Vensim or XMILE model files into python modules, and provides methods to modify, simulate, and observe those translated models.



API to 'Python' Library 'pysd'

Documentation for package ‘pysd2r’ version 0.1.0

- DESCRIPTION file.
- User guides, package vignettes and other documentation.

Help Pages

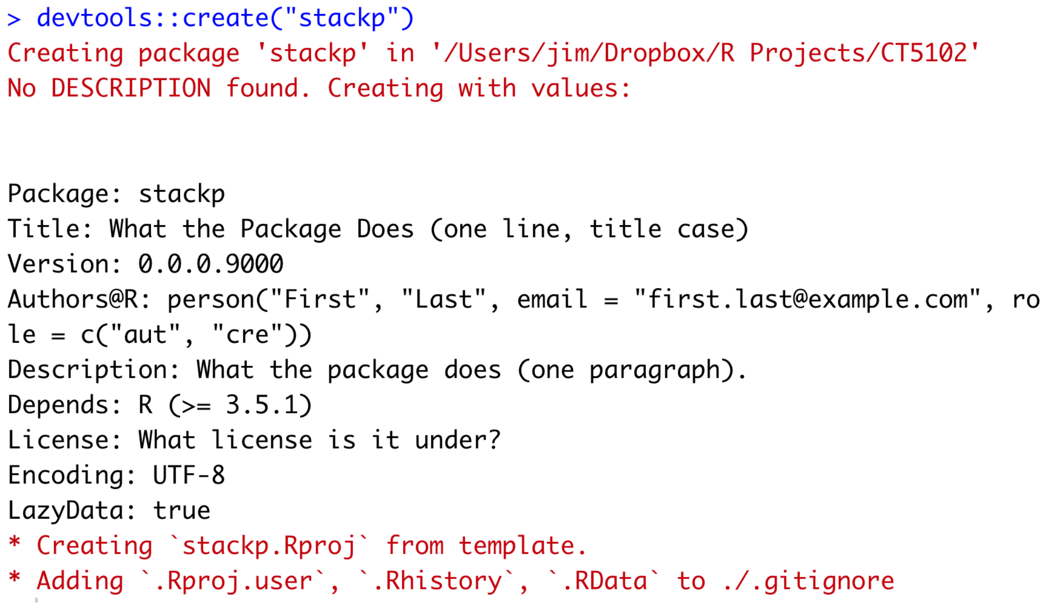
get_doc	Formats a table of variable names
get_final_time	Gets the final time from the model
get_initial_time	Gets the initial time from the model
get_python_info	Gets the current python configuration for reticulate

Our example: stackp

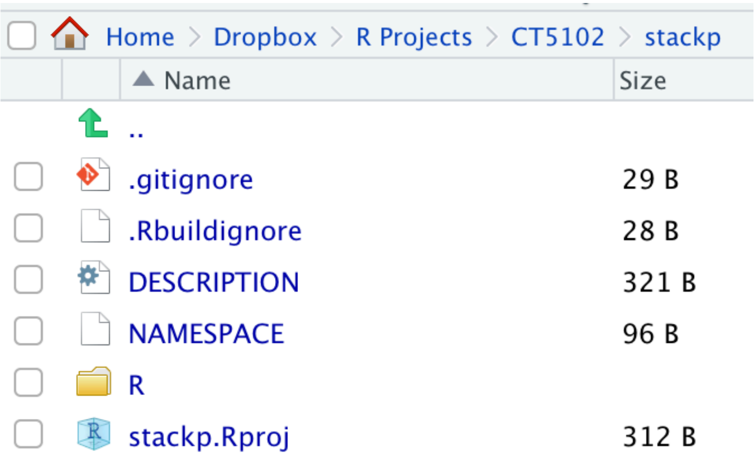
- An S3 stack class as a demo
- Create a tar.gz file, and also install from github
- Can store a stack of any data type (list structure used)

Method	Description
stackp()	Constructor to create the stackp object
push(s,v)	Push the value v onto the stack s (generic)
peek(s)	Get the top value from the stack s (generic)
pop()	Remove the top value from the stack (generic)
summary()	Summarise the stack entries (generic)

(2) Create the package
devtools::create(“stackp”)



```
> devtools::create("stackp")
Creating package 'stackp' in '/Users/jim/Dropbox/R Projects/CT5102'
No DESCRIPTION found. Creating with values:


Package: stackp
Title: What the Package Does (one line, title case)
Version: 0.0.0.9000
Authors@R: person("First", "Last", email = "first.last@example.com", ro
le = c("aut", "cre"))
Description: What the package does (one paragraph).
Depends: R (>= 3.5.1)
License: What license is it under?
Encoding: UTF-8
LazyData: true
* Creating `stackp.Rproj` from template.
* Adding `.Rproj.user`, `.Rhistory`, `.RData` to ./.gitignore
```

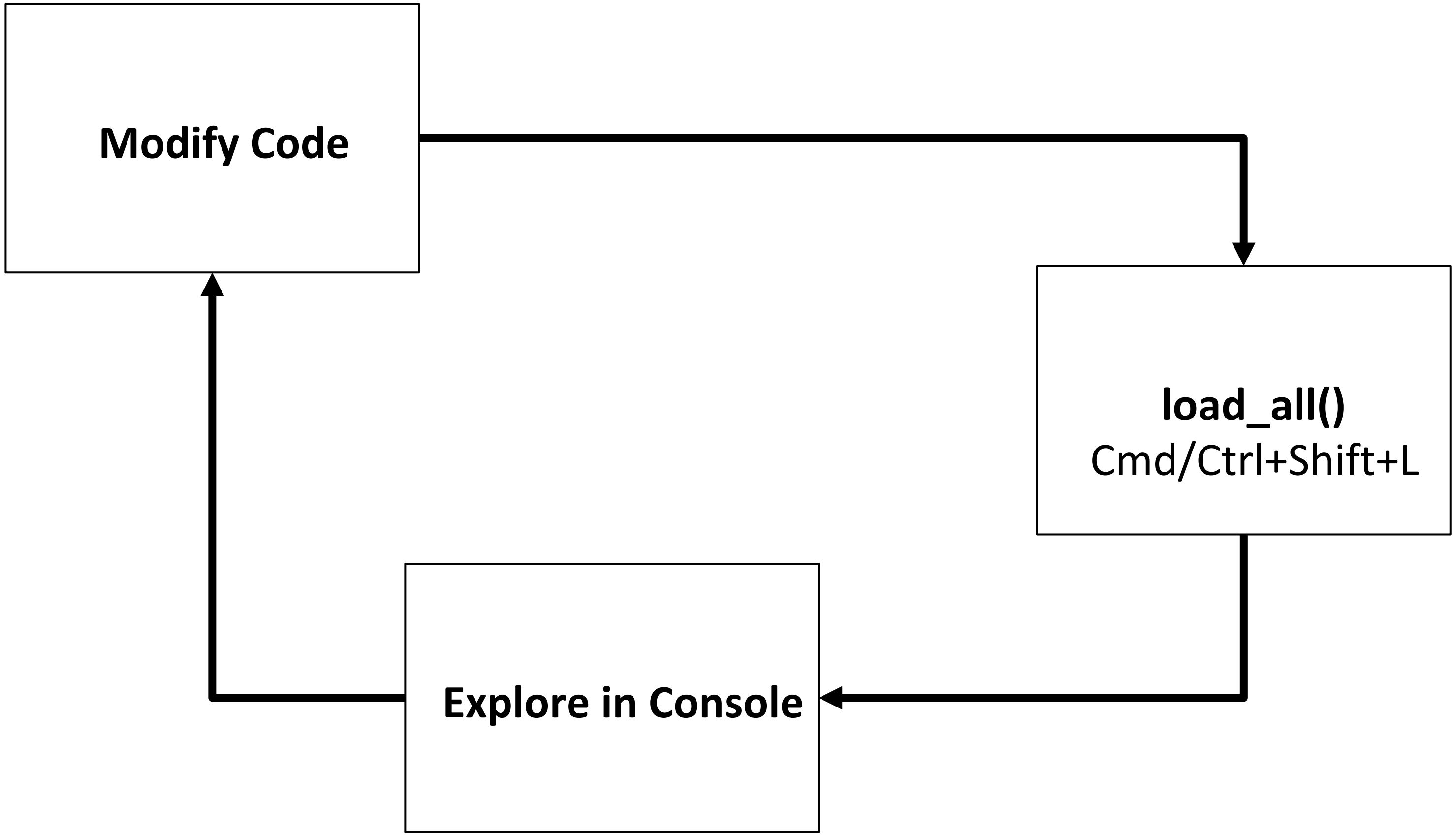


Home > Dropbox > R Projects > CT5102 > stackp

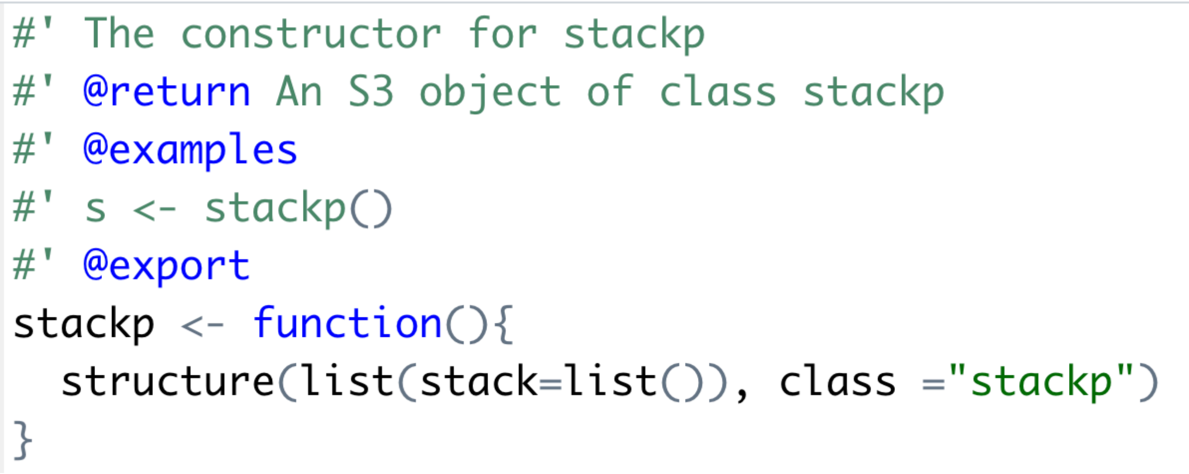
	▲ Name	Size
	..	
□	.gitignore	29 B
□	.Rbuildignore	28 B
□	DESCRIPTION	321 B
□	NAMESPACE	96 B
□	R	
□	stackp.Rproj	312 B

Open the new project stackp in RStudio

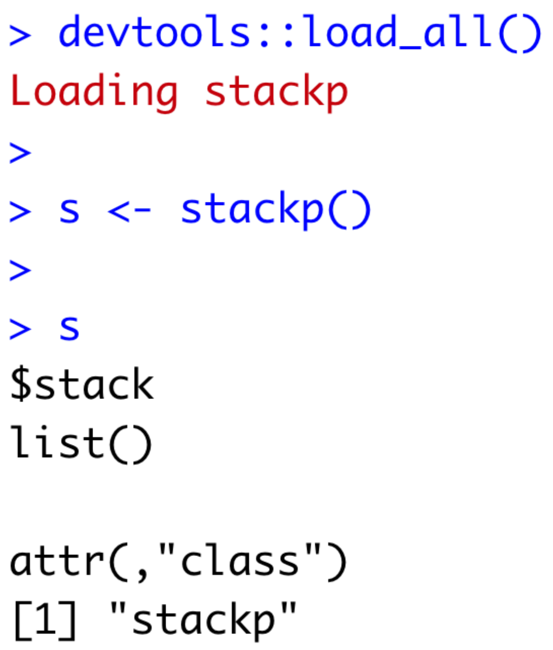
R/ is where your code lives
load_all() allows you to test



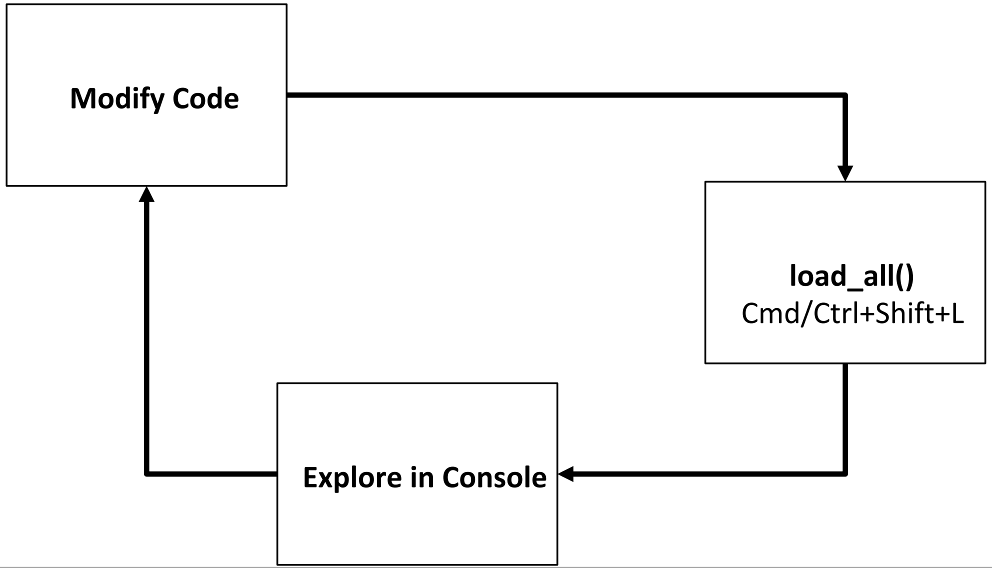
(3) Add code to R/



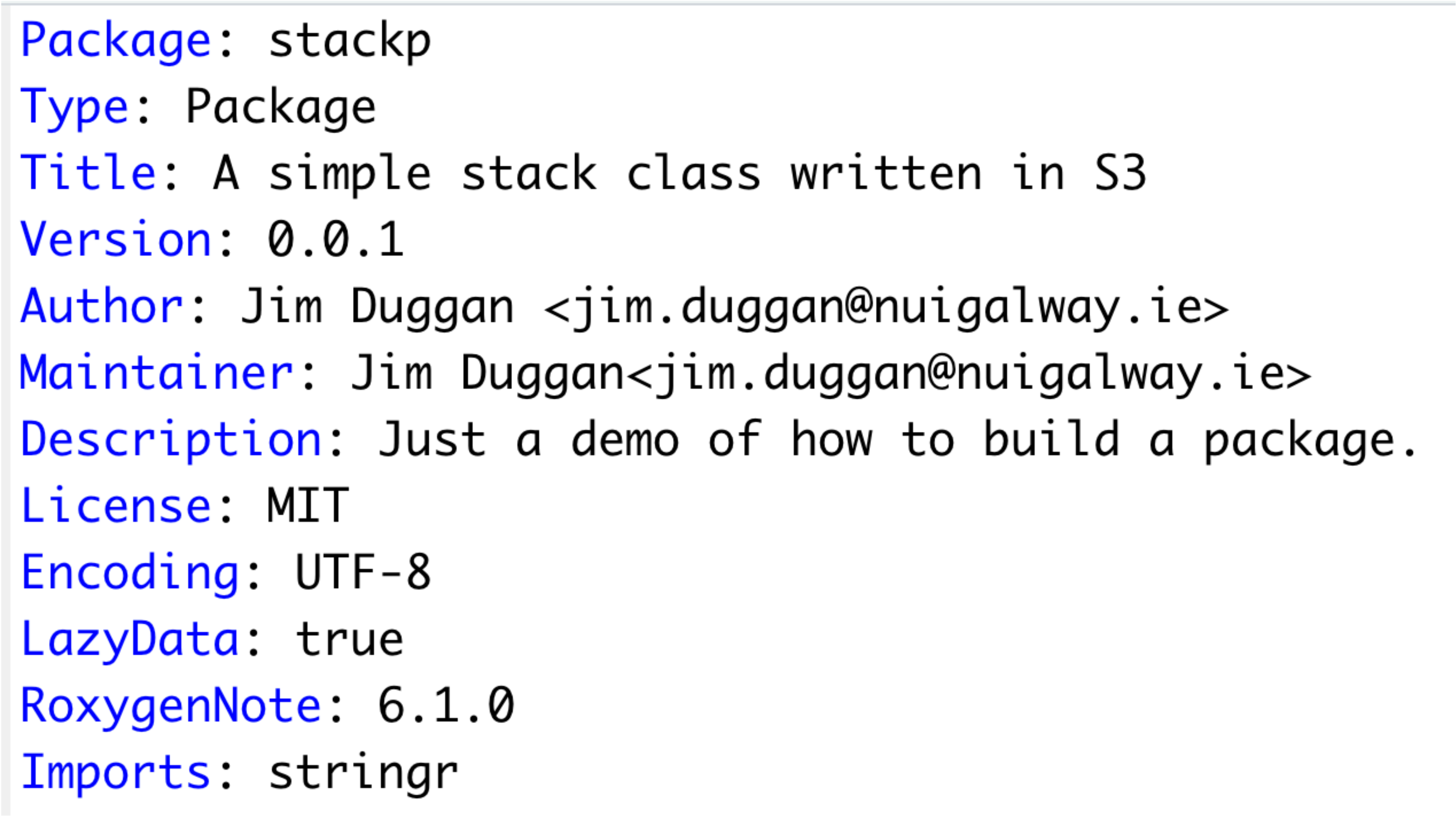
```r
#' The constructor for stackp
#' @return An S3 object of class stackp
#' @examples
#' s <- stackp()
#' @export
stackp <- function(){
  structure(list(stack=list()), class ="stackp")
}
```



```
> devtools::load_all()
Loading stackp
>
> s <- stackp()
>
> s
$stack
list()

attr(,"class")
[1] "stackp"
```



(4) Update default DESCRIPTION file



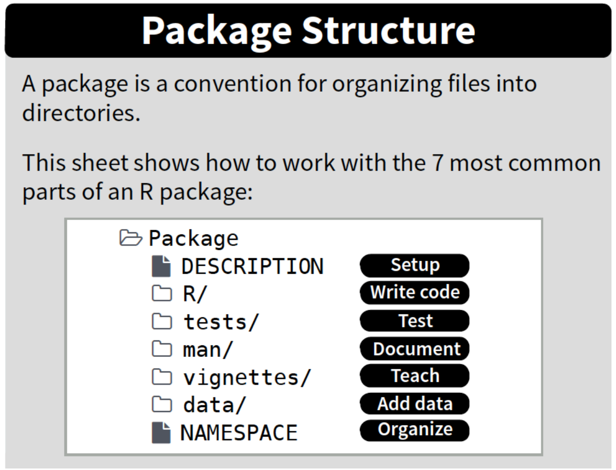
```
Package: stackp
Type: Package
Title: A simple stack class written in S3
Version: 0.0.1
Author: Jim Duggan <jim.duggan@nuigalway.ie>
Maintainer: Jim Duggan<jim.duggan@nuigalway.ie>
Description: Just a demo of how to build a package.
License: MIT
Encoding: UTF-8
LazyData: true
RoxygenNote: 6.1.0
Imports: stringr
```

(5) Create help files

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

- Documentation is one of the most important aspects of a good package.
- Without it, users won’t know how to use your package.
- Documentation is also useful for future-use (so you remember what your functions were supposed to do), and for developers extending your package

Create /man sub-directory

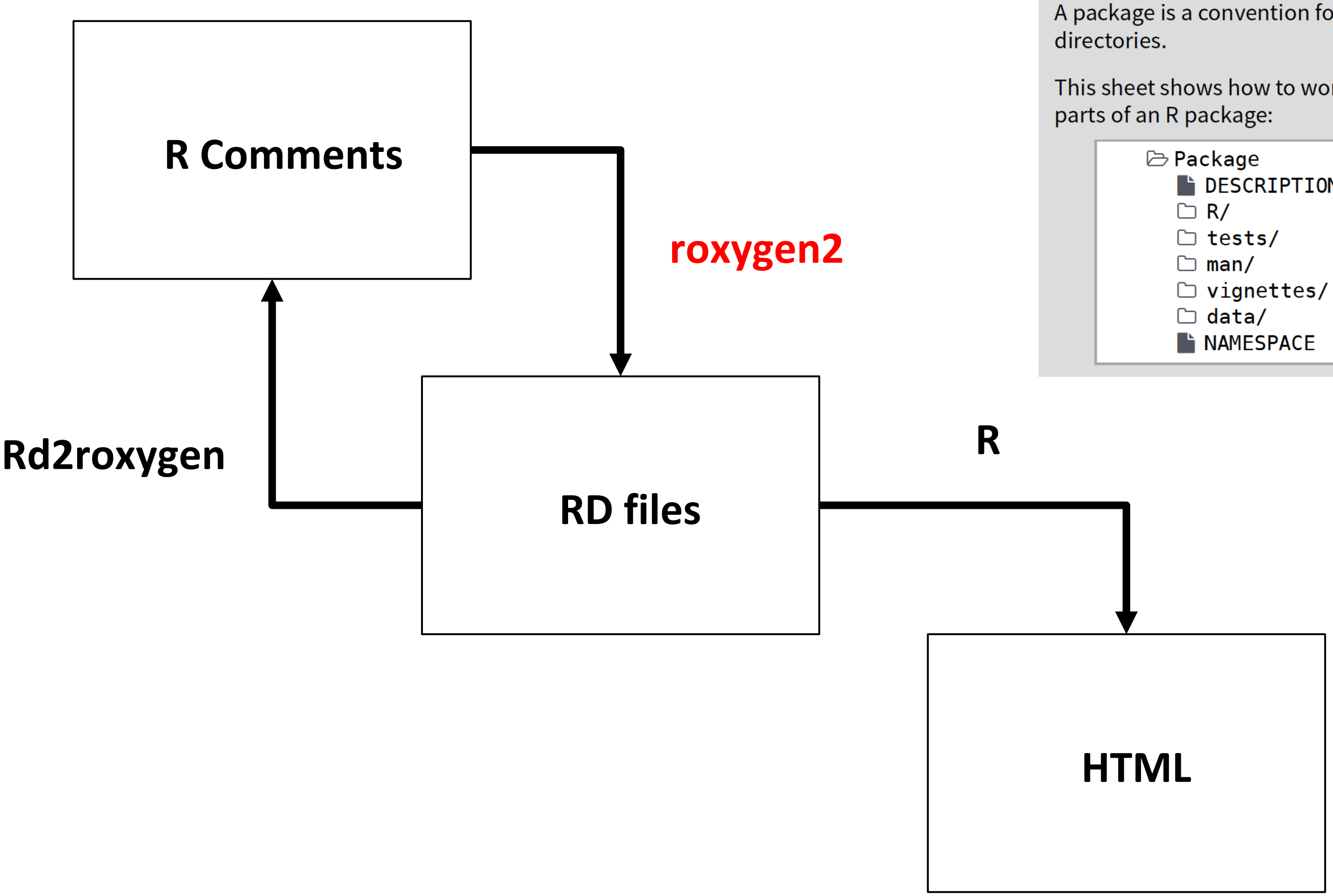


Package Structure

A package is a convention for organizing files into directories.

This sheet shows how to work with the 7 most common parts of an R package:

Package	
DESCRIPTION	Setup
R/	Write code
tests/	Test
man/	Document
vignettes/	Teach
data/	Add data
NAMESPACE	Organize

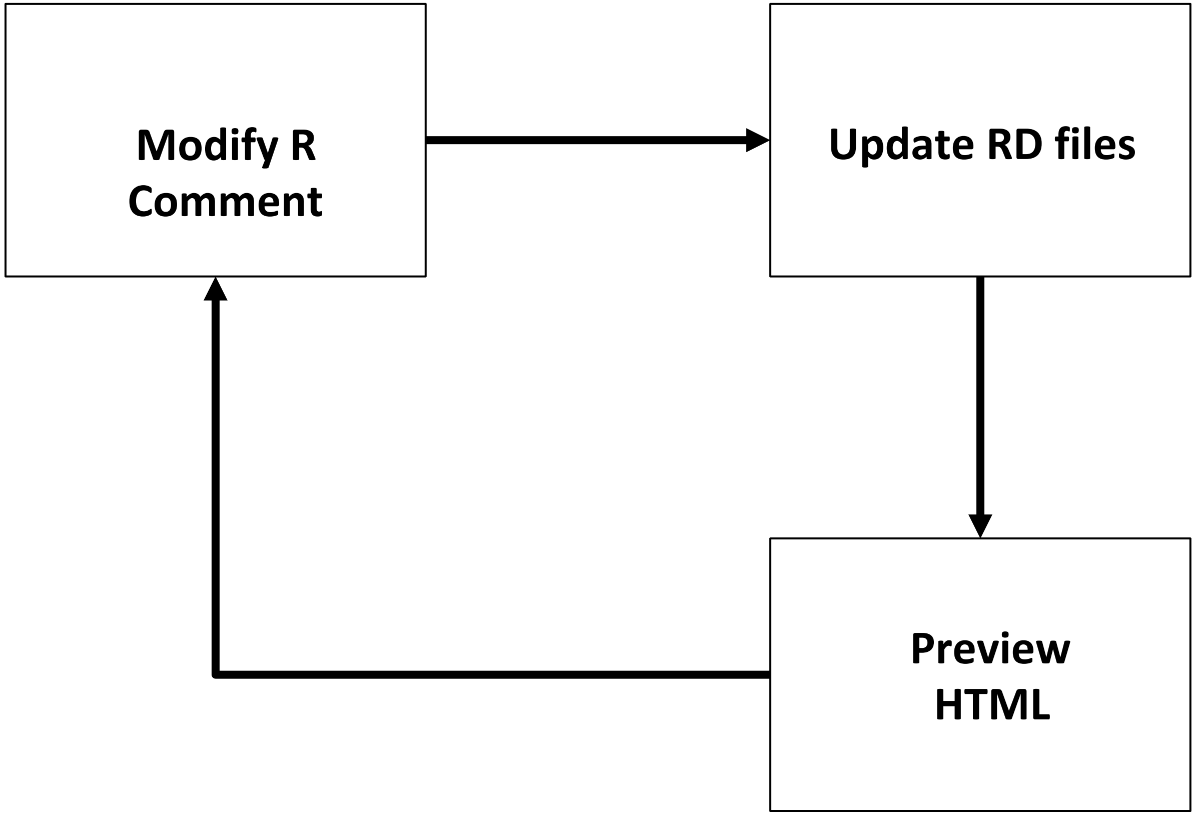


Roxygen

- roxygen lets you write documentation inline in your .R files with a shorthand syntax.
- Add roxygen documentation as comment lines that begin with #'.
- Place comment lines directly above the code that defines the object documented.
- Place a roxygen @ tag (right) after #' to supply a specific section of documentation.
- Untagged lines will be used to generate a title, description, and details section (in that order)

Workflow (Wickham)

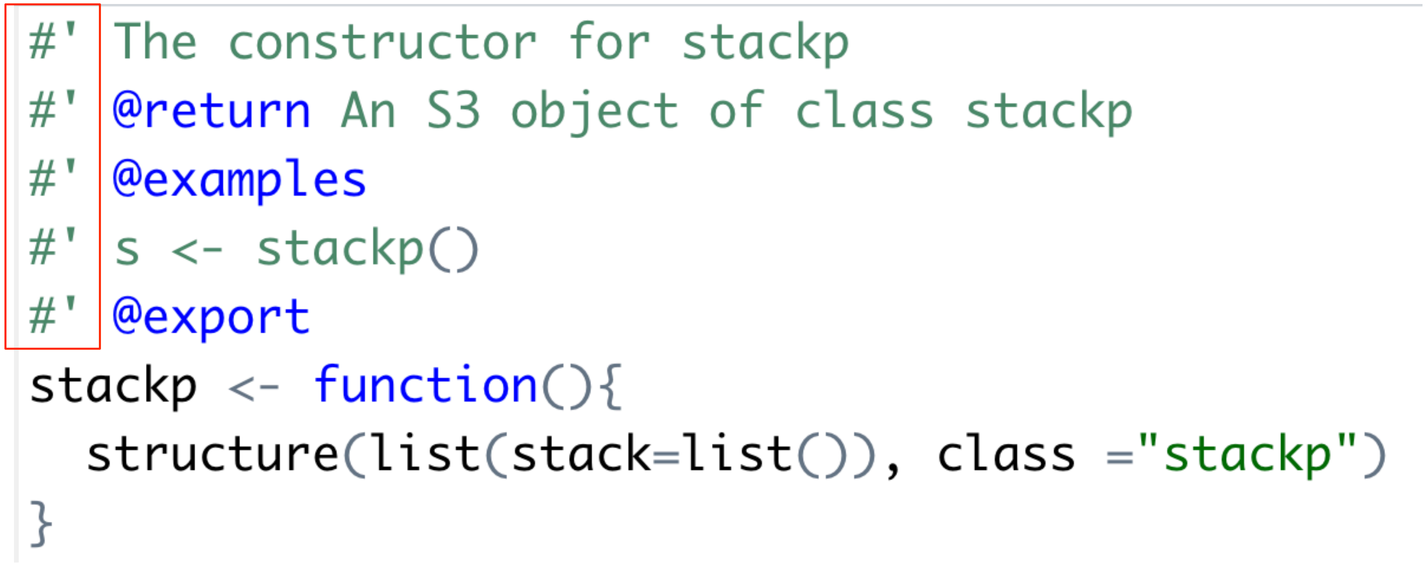
*NB You must have loaded the package with **load_all()** at least once*



devtools::document()
Cmd/Ctrl+Shift+D

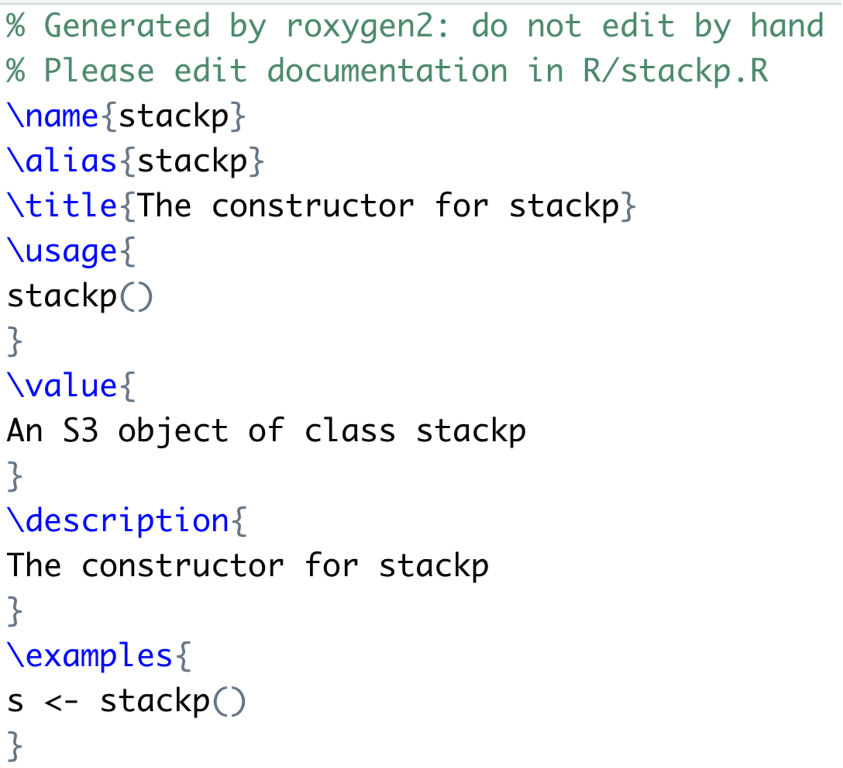
Common Tags

Tag	Purpose
@param arg	Describe function inputs
@examples	Show how the function works
@seealso	Pointers to related functions
@return	Describe outputs (value)
@export	To be discussed

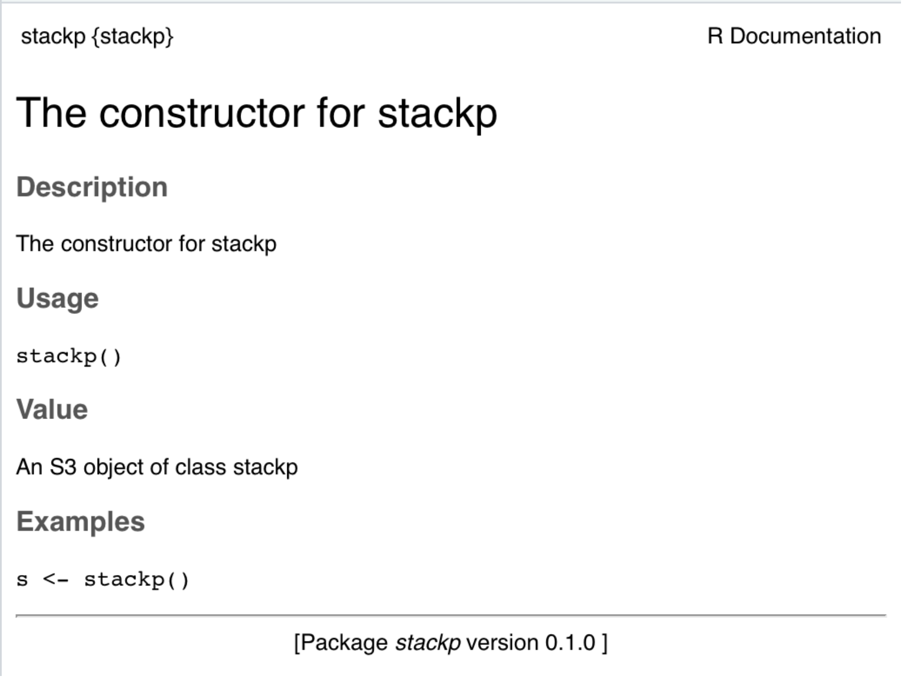


```r
#' The constructor for stackp
#' @return An S3 object of class stackp
#' @examples
#' s <- stackp()
#' @export
stackp <- function(){
  structure(list(stack=list()), class ="stackp")
}
```

RD Files & Preview
?stackp will also work at the console



```
% Generated by roxygen2: do not edit by hand
% Please edit documentation in R/stackp.R
\name{stackp}
\alias{stackp}
\title{The constructor for stackp}
\usage{
stackp()
}
\value{
An S3 object of class stackp
}
\description{
The constructor for stackp
}
\examples{
s <- stackp()
}
```



stackp {stackp} R Documentation

The constructor for stackp

Description

The constructor for stackp

Usage

```
stackp()
```

Value

An S3 object of class stackp

Examples

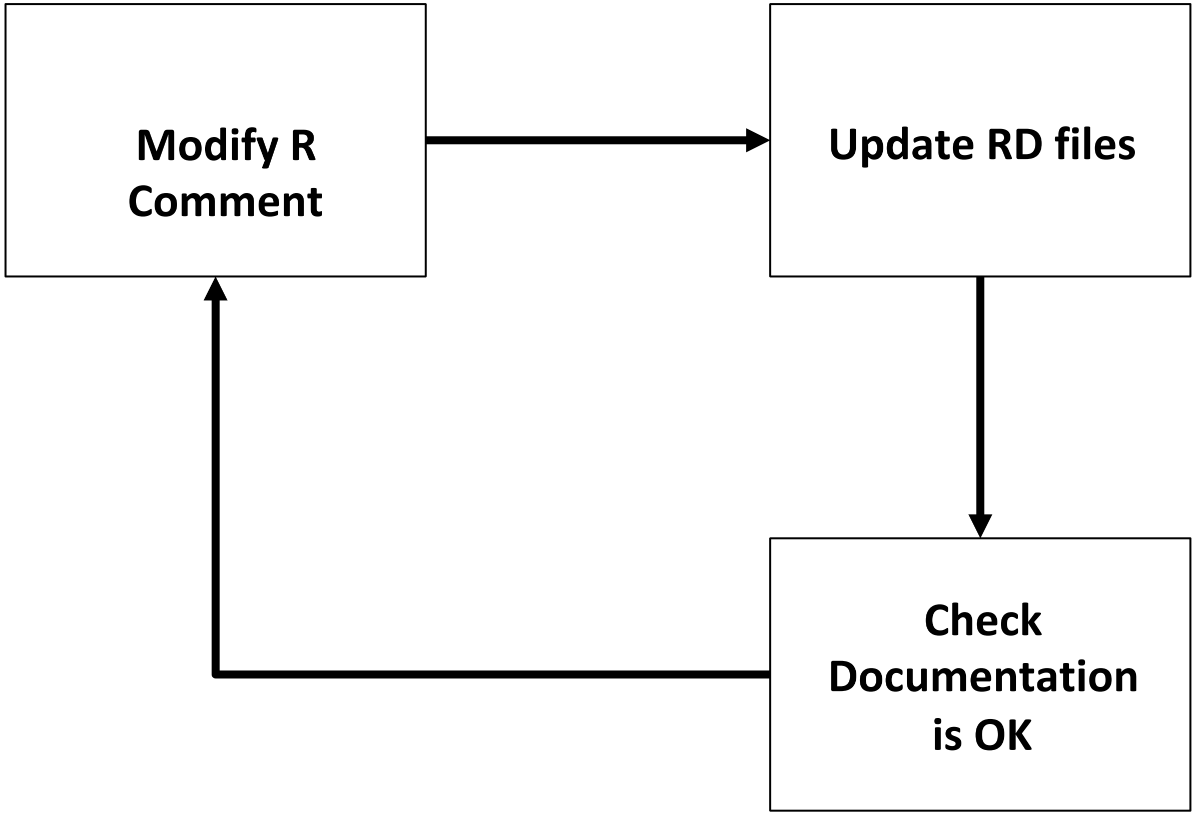
```
s <- stackp()
```

[Package *stackp* version 0.1.0]

Text Formatting with Rd

Tag	Purpose
\code{}	Inline R Code
\eqn{}	Inline equation (latex)
\emph{}	Italic Text
\strong{}	Bold text
\link{foo}	Link to foo in the current package
\link[bar]{foo}	Link to foo in package bar
\url{http://rstudio.com}	Link to website
\email{j@mail.com}	Email address

Revised Workflow



Modify R Comment

Update RD files

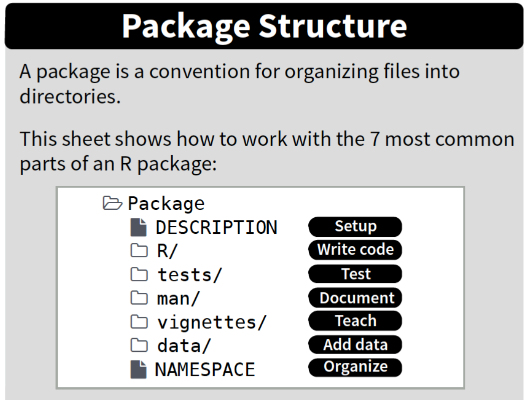
Check Documentation is OK

devtools::document()
Cmd/Ctrl+Shift+D

devtools::check_man()

(6) Vignettes

- Lets you combine code and prose to explain how your package works
- devtools::use_vignette("name")
- Adds to DESCRIPTION

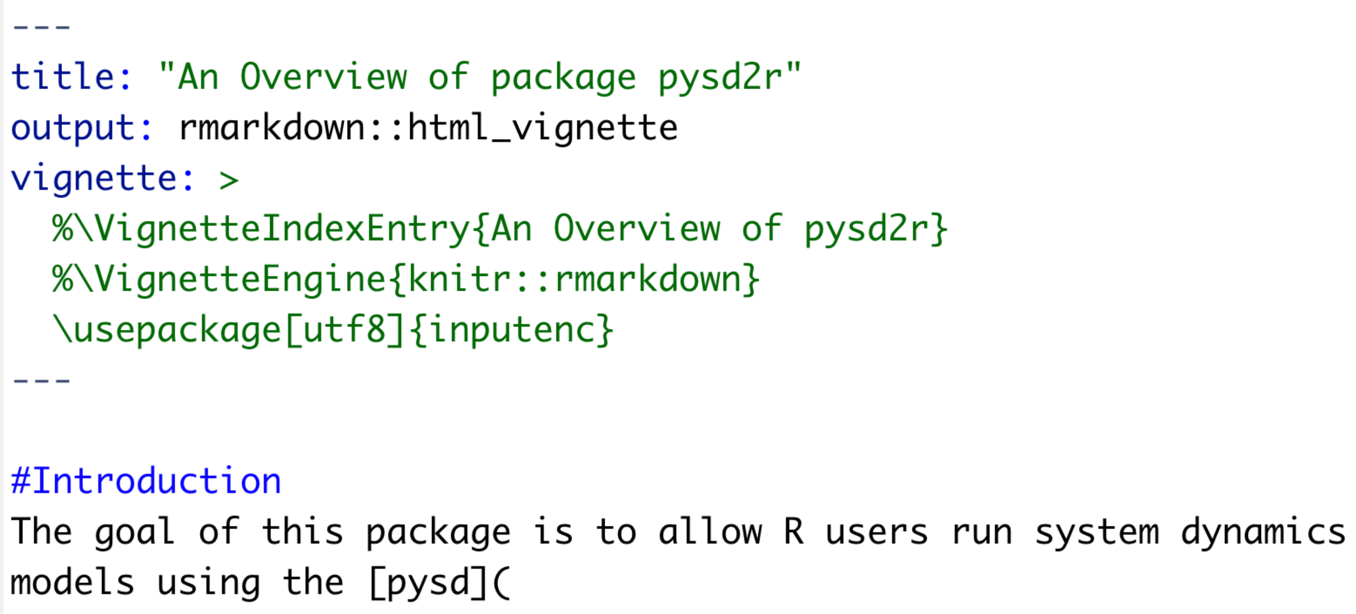


Package Structure

A package is a convention for organizing files into directories.

This sheet shows how to work with the 7 most common parts of an R package:

```
Package
  DESCRIPTION     Setup
  R/              Write code
  tests/          Test
  man/            Document
  vignettes/      Teach
  data/           Add data
  NAMESPACE       Organize
```



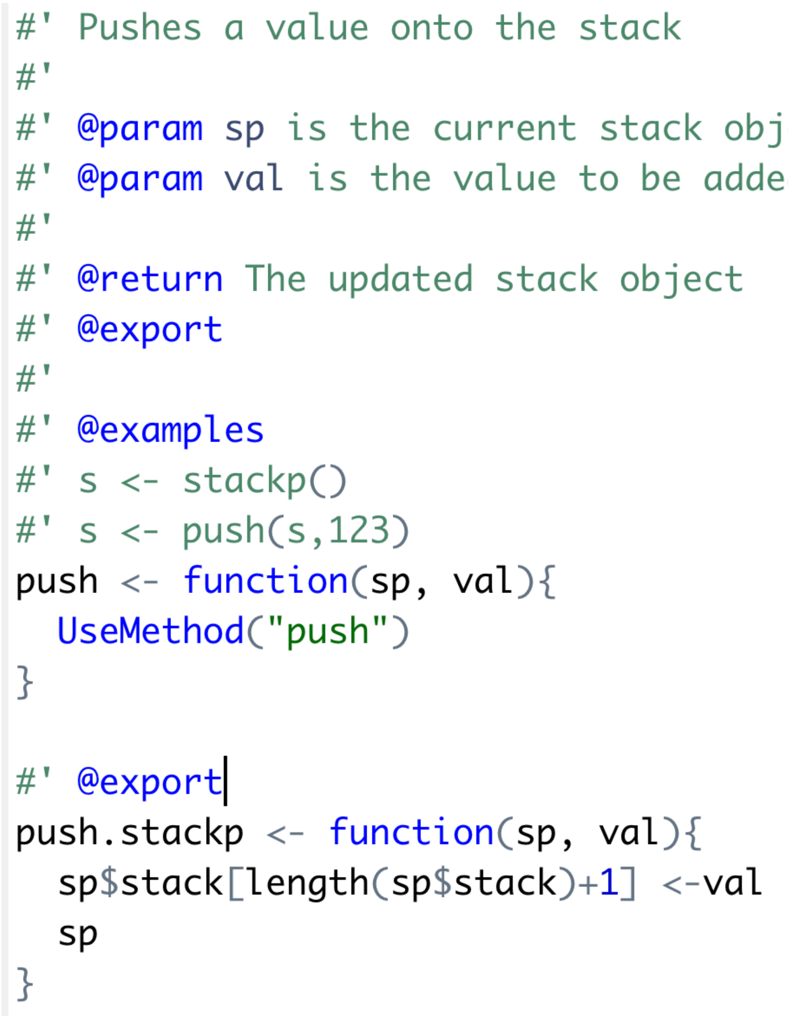
```
---
title: "An Overview of package pysd2r"
output: rmarkdown::html_vignette
vignette: >
  %\VignetteIndexEntry{An Overview of pysd2r}
  %\VignetteEngine{knitr::rmarkdown}
  \usepackage[utf8]{inputenc}
---


#Introduction
The goal of this package is to allow R users run system dynamics
models using the [pysd](
```

(7) NAMESPACE

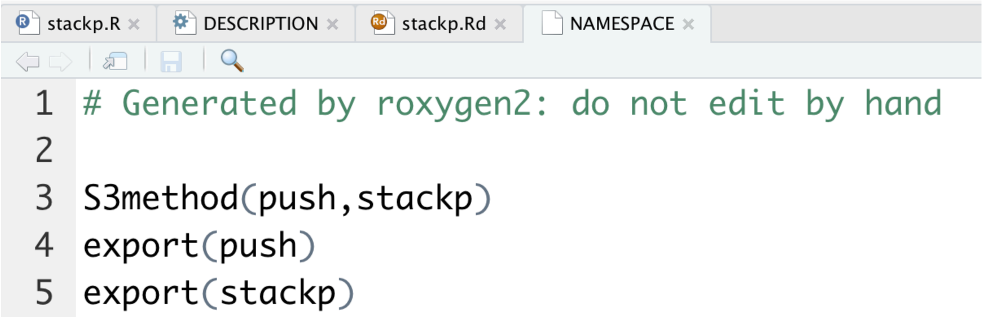
- A NAMESPACE splits a function into two classes
- Default NAMESPACE exports everything – better to export functions explicitly

Internal	External
Only for use within the package	For use by others
Documentation optional	Must be documented
Easily changed	Changing will break other people’s code

devtools::check()



```
#' Pushes a value onto the stack
#'
#' @param sp is the current stack object
#' @param val is the value to be added to the stack
#'
#' @return The updated stack object
#' @export
#'
#' @examples
#' s <- stackp()
#' s <- push(s,123)
push <- function(sp, val){
  UseMethod("push")
}

#' @export
push.stackp <- function(sp, val){
  sp$stack[length(sp$stack)+1] <-val
  sp
}
```



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

S3method(push,stackp)
export(push)
export(stackp)
```

@export generates the right NAMESPACE directive (Wickham)

Object type	Namespace Directive
Function	export()
S3 method	S3method()
S4 class	exportClass()
S4 method	exportMethods()

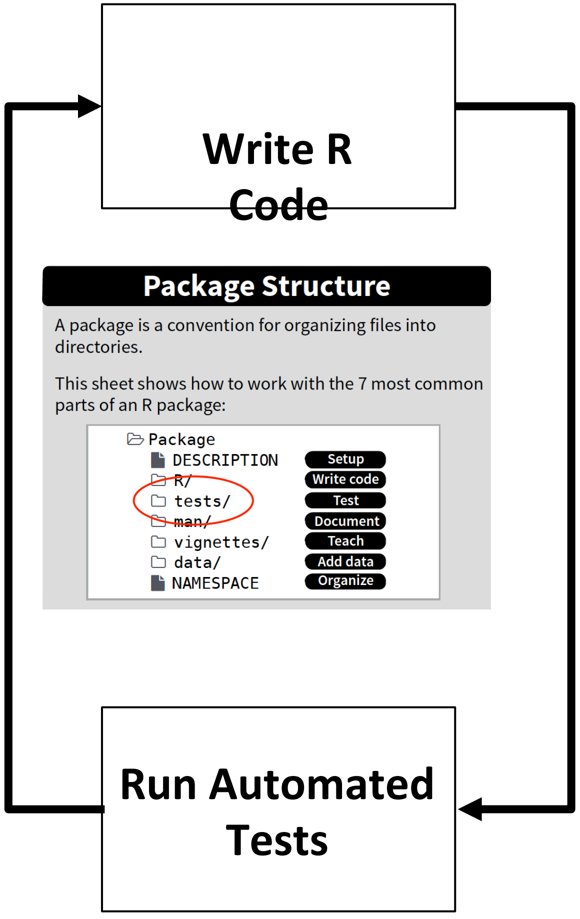
(7) Adding Tests – Why Test?

- **Correctness**
  - Ensure current correctness
  - Interactive feedback
  - Reduce maintenance burden
- **Design**
  - Think adversarially
  - Enable refactoring
  - Force good design
- **Working with others**
  - Guard against future developers
  - Communicate with colleagues
  - Guard against complex interactions
  - Social pressure
  - Establish interfaces

http://matthewrocklin.com/blog/work/2016/02/08/tests

testthat package (Wickham)

- Provides easy transition from informal to formal tests.
- Can be used in wide variety of situations.
- Wide range of expectations/assertions.
- Fun output designed to keep you motivated.
- Currently used by over 1100 packages.



Package Structure

A package is a convention for organizing files into directories.

This sheet shows how to work with the 7 most common parts of an R package:

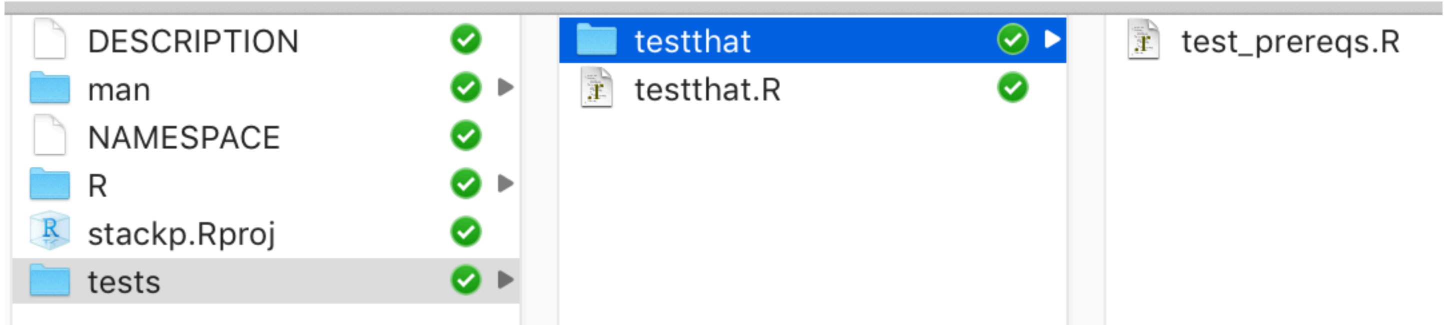
```
Package
  DESCRIPTION     Setup
  R/              Write code
  tests/          Test
  man/            Document
  vignettes/      Teach
  data/           Add data
  NAMESPACE       Organize
```

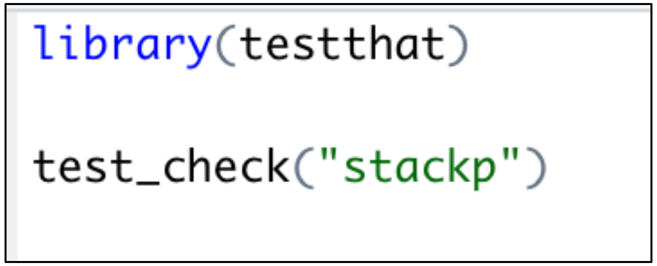
devtools::test()

How to create tests (Wickham)

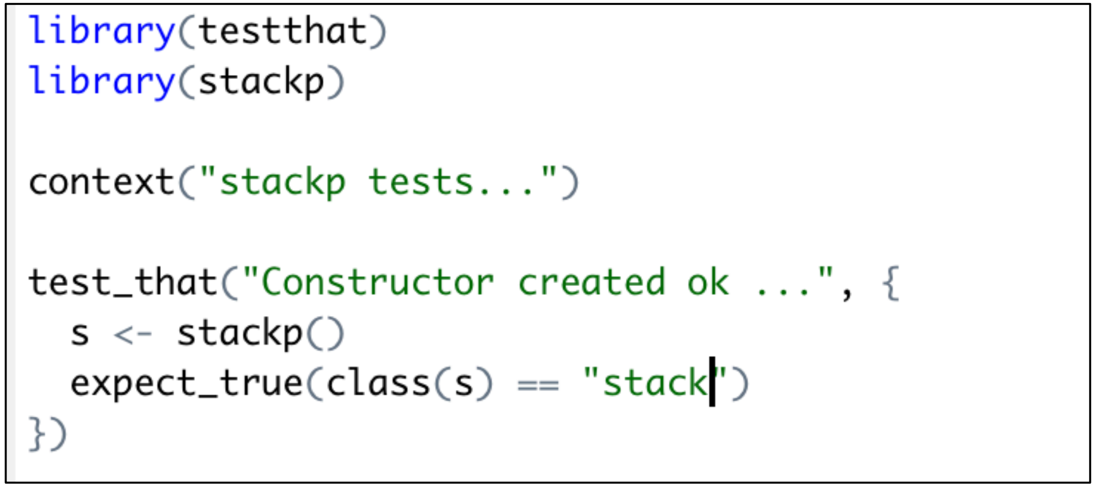
- Compare with known outputs.
- Compare with results calculated another way.
- Whenever you find a bug, first figure out the right answer and write a test.
- Test areas that are likely to fail (i.e. complicated bits).
- Focus on improving tests over time, not being perfect when you first start.

testthat.R & test_prereqs.R



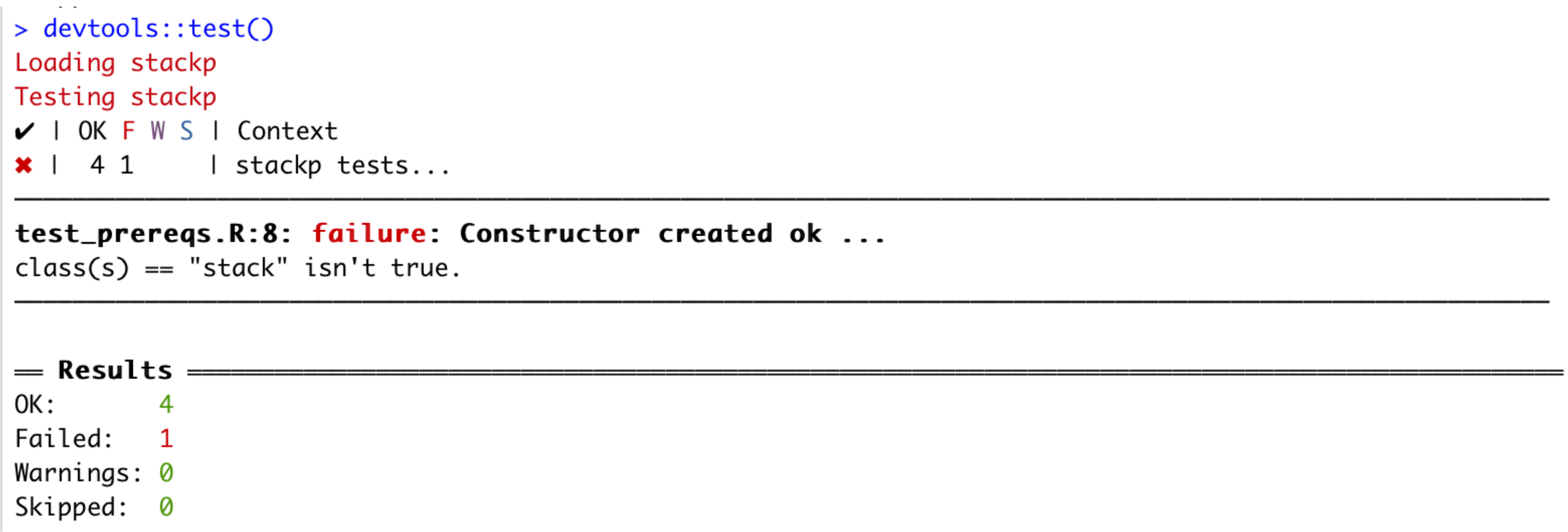


```
library(testthat)

test_check("stackp")
```



```
library(testthat)
library(stackp)

context("stackp tests...")

test_that("Constructor created ok ...", {
  s <- stackp()
  expect_true(class(s) == "stack")
})
```

Running tests…



```
> devtools::test()
Loading stackp
Testing stackp
✔ | OK F W S | Context
✖ |  4 1     | stackp tests...
──────────────────────────────────────────────────────────────────────────────────
test_prereqs.R:8: failure: Constructor created ok ...
class(s) == "stack" isn't true.
──────────────────────────────────────────────────────────────────────────────────

══ Results ═══════════════════════════════════════════════════════════════════════
OK:       4
Failed:   1
Warnings: 0
Skipped:  0
```

testthat functions

Abbreviation	**Test**
expect_equal()	Uses all.equal(), ignores floating point differences
expect_identical()	Uses identical() for stricter numerical testing.
expect_equivalent()	Like expect_equal(), but also ignores differences in attributes.
expect_is()	Check that inherits from a given class.
expect_true() / expect_false()	Catch all expectations for anything not otherwise covered

testthat functions

Abbreviation	Test
expect_matches()	Does any value match the supplied regular expression?
expect_output()	Does printed output match the supplied regular expression?
expect_message()	Does displayed messages match the supplied regular expression?
expect_warning()	Do any warnings match supplied regular expression?
expect_error()	Do any errors match supplied regular expression?

(8) Checking, Build and Install

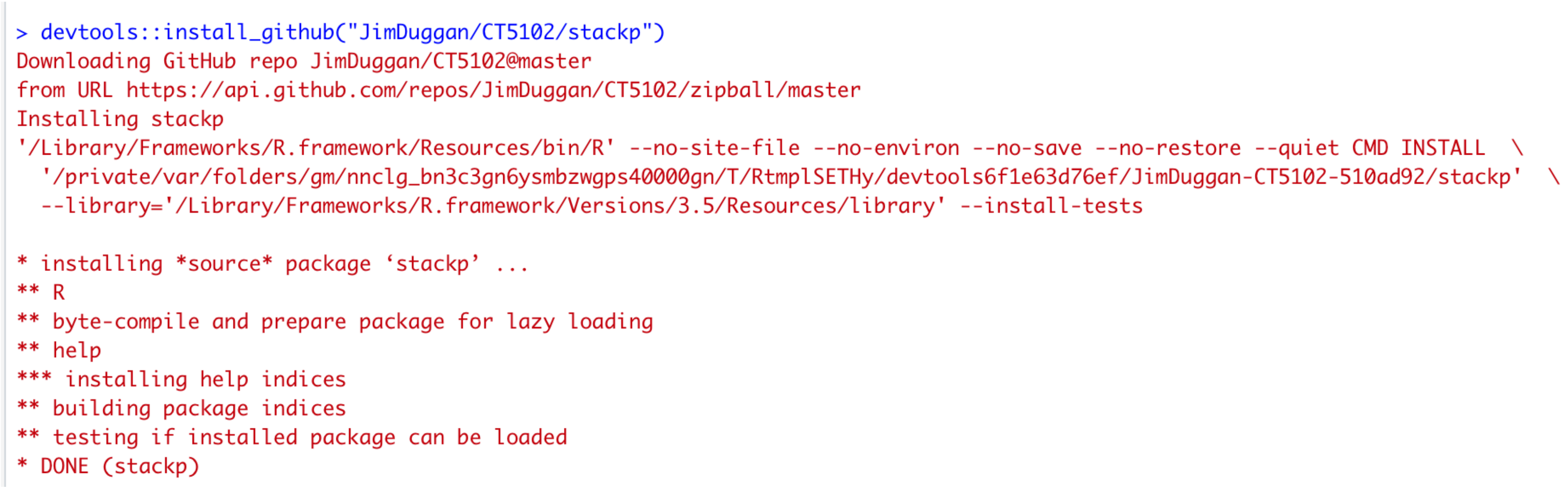
- **devtools::check()**
  - Runs automated checks for common problems in R packages
- **devtools::build()**
  - Builds a tar.gz file for sharing
- **install_github(“JimDuggan/CT5102/stackp”)**

Types of Problem

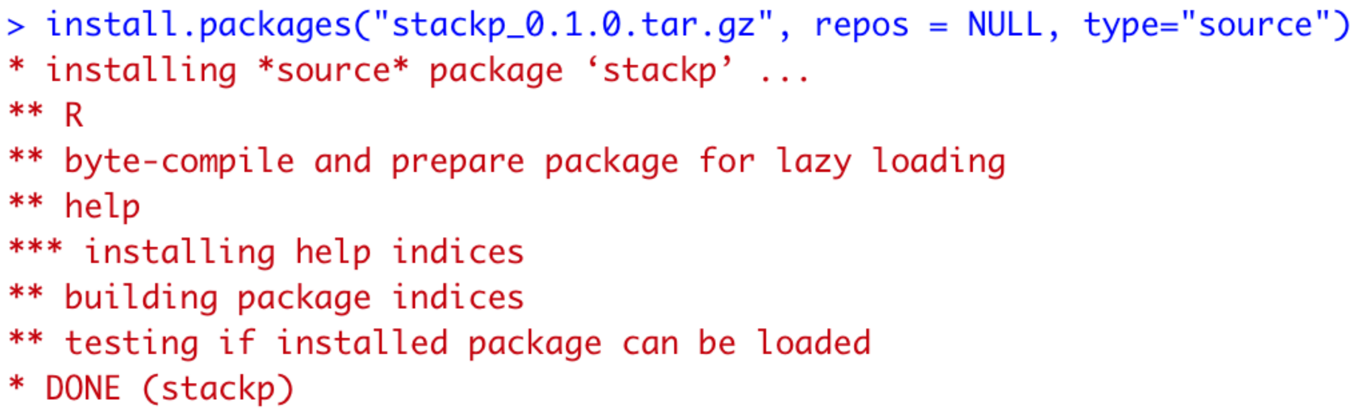
- Error
  - Must fix!
- Warning
  - Fix if submitting to CRAN
- NOTE
  - Fix if submitting to CRAN

	Local	**CRAN**
ERROR	Fix	Fix
WARNING		Fix
NOTE		Fix

Install from github & local file

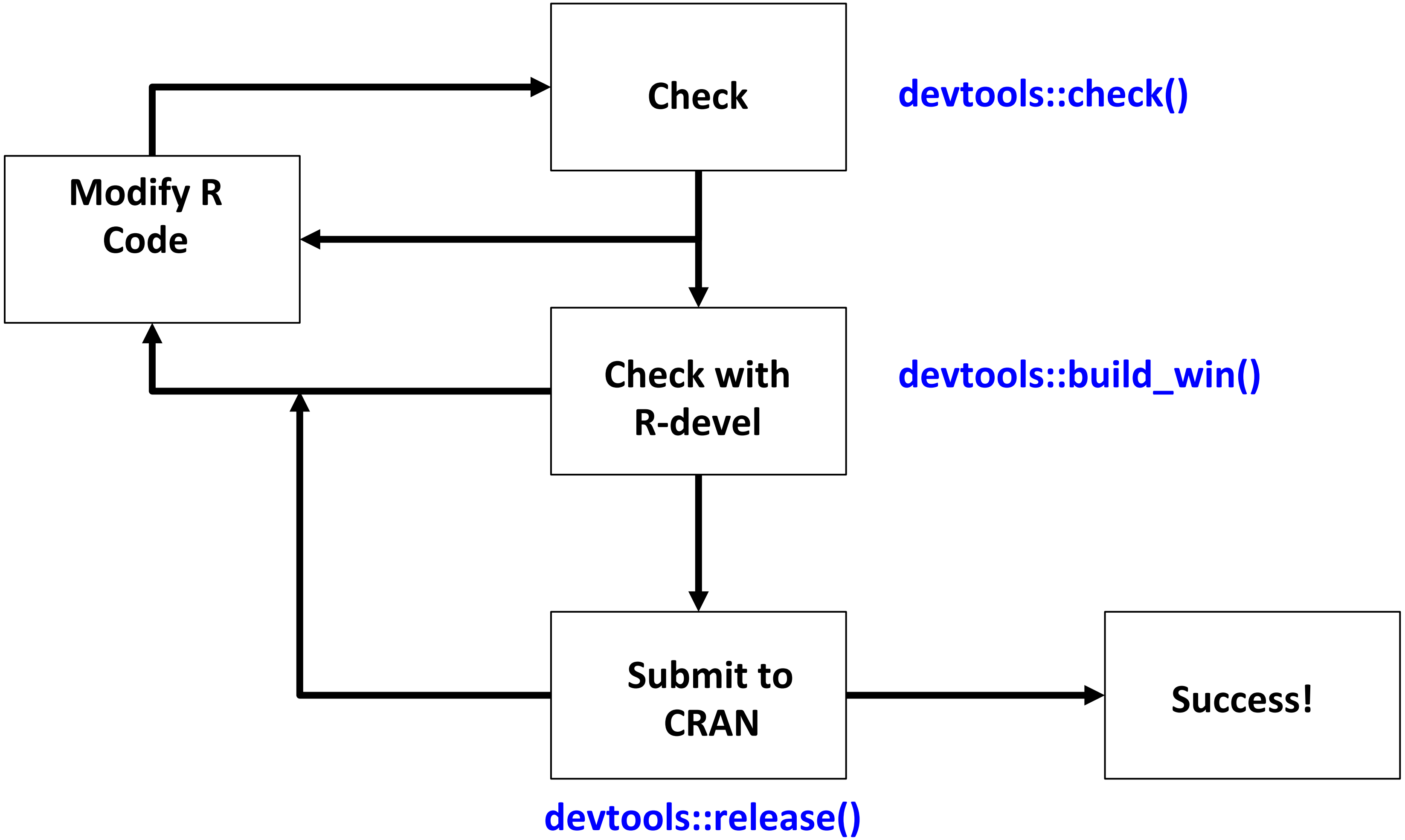


```
> devtools::install_github("JimDuggan/CT5102/stackp")
Downloading GitHub repo JimDuggan/CT5102@master
from URL https://api.github.com/repos/JimDuggan/CT5102/zipball/master
Installing stackp
'/Library/Frameworks/R.framework/Resources/bin/R' --no-site-file --no-environ --no-save --no-restore --quiet CMD INSTALL  \
  '/private/var/folders/gm/nnclg_bn3c3gn6ysmbzwgps40000gn/T/RtmplSETHy/devtools6f1e63d76ef/JimDuggan-CT5102-510ad92/stackp'  \
  --library='/Library/Frameworks/R.framework/Versions/3.5/Resources/library' --install-tests

* installing *source* package ‘stackp’ ...
** R
** byte-compile and prepare package for lazy loading
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded
* DONE (stackp)
```

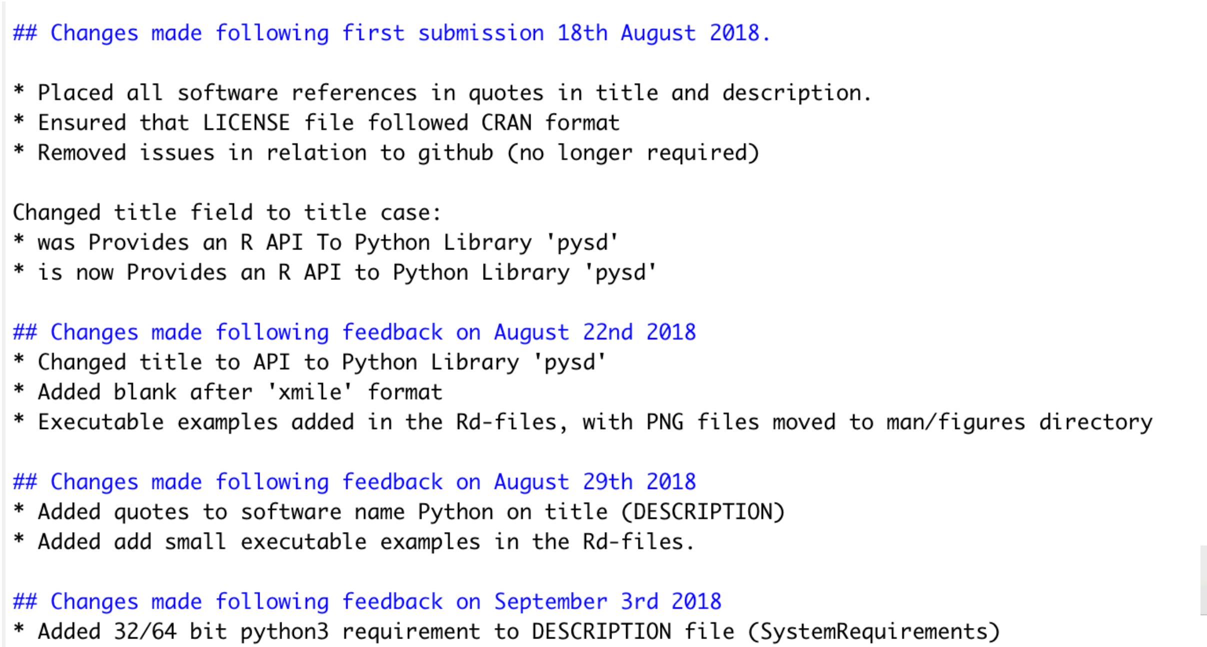


```
> install.packages("stackp_0.1.0.tar.gz", repos = NULL, type="source")
* installing *source* package ‘stackp’ ...
** R
** byte-compile and prepare package for lazy loading
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded
* DONE (stackp)
```

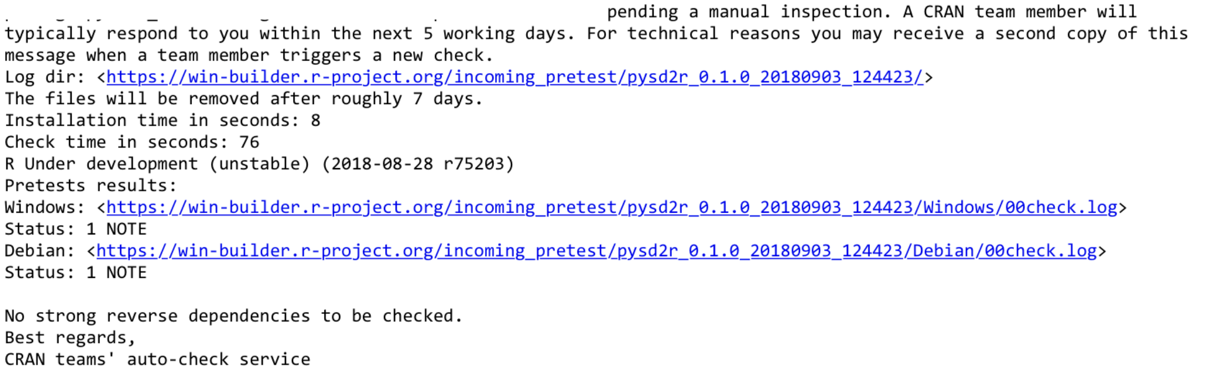
CRAN Process (Wickham)



Final Stages of CRAN Process
cran-comments.md



```
## Changes made following first submission 18th August 2018.

* Placed all software references in quotes in title and description.
* Ensured that LICENSE file followed CRAN format
* Removed issues in relation to github (no longer required)

Changed title field to title case:
* was Provides an R API To Python Library 'pysd'
* is now Provides an R API to Python Library 'pysd'

## Changes made following feedback on August 22nd 2018
* Changed title to API to Python Library 'pysd'
* Added blank after 'xmile' format
* Executable examples added in the Rd-files, with PNG files moved to man/figures directory

## Changes made following feedback on August 29th 2018
* Added quotes to software name Python on title (DESCRIPTION)
* Added add small executable examples in the Rd-files.

## Changes made following feedback on September 3rd 2018
* Added 32/64 bit python3 requirement to DESCRIPTION file (SystemRequirements)
```



```
                                                            pending a manual inspection. A CRAN team member will
typically respond to you within the next 5 working days. For technical reasons you may receive a second copy of this
message when a team member triggers a new check.
Log dir: <https://win-builder.r-project.org/incoming_pretest/pysd2r_0.1.0_20180903_124423/>
The files will be removed after roughly 7 days.
Installation time in seconds: 8
Check time in seconds: 76
R Under development (unstable) (2018-08-28 r75203)
Pretests results:
Windows: <https://win-builder.r-project.org/incoming_pretest/pysd2r_0.1.0_20180903_124423/Windows/00check.log>
Status: 1 NOTE
Debian: <https://win-builder.r-project.org/incoming_pretest/pysd2r_0.1.0_20180903_124423/Debian/00check.log>
Status: 1 NOTE

No strong reverse dependencies to be checked.
Best regards,
CRAN teams' auto-check service
```

Summary

- Package structure facilitates excellent software design
- Sharing of code, design of useful modules
- Can be used:
  - "in-house" − private
  - github (shared)
  - CRAN − widely available