



# R Package Anatomy

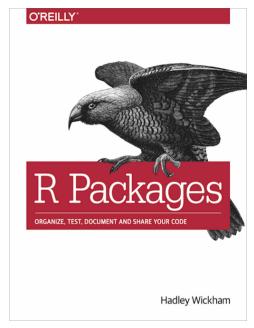
R Foundations for Life Scientists

Sebastian DiLorenzo

## Overview



- What is an R package?
- Possible package states
- Package structure:
  - ∘ Code | r/
  - Metadata | DESCRIPTION
  - Documentation | man/
  - Vignettes
  - Import & Export | NAMESPACE
  - o Data | data/
  - Compiled code | src/
- git, Github, Rstudio and you!
- CRAN and R CMD check



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

# What is an R package?



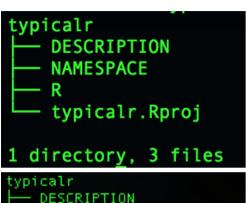
```
typicalr
DESCRIPTION
NAMESPACE
R
typicalr.Rproj

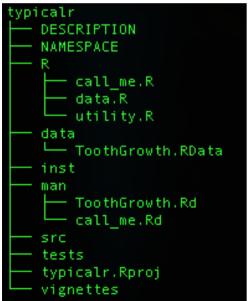
1 directory, 3 files
```

A strict and connected folder and file structure

## What is an R package?







A strict and connected folder and file structure

## What is an R package?



```
typicalr
DESCRIPTION
NAMESPACE
R
typicalr.Rproj

1 directory, 3 files
```

```
ypicalr
  DESCRIPTION
  NAMESPACE
       call me.R
       data.R
      utility.R
  data
   └─ ToothGrowth.RData
  inst
   man

    ToothGrowth.Rd

      call me.Rd
   src
   tests
   typicalr.Rproj
  vignettes
```

- A strict and connected folder and file structure
- Sharing code
- Improved quality and rigor
  - Documentation
  - Tests
  - Examples
- Efficiency
- Improvability
- Reproducibility

# Package naming



- A name that describes your packages function
  - Letters, numbers and periods
  - Must start with letter
  - Cannot end with period
- Make it googleable
- Check that it doesn't already exist!
  - CRAN
  - o github
  - Bioconductor

# Possible package states



There are five states a package can exist in:

- Source
- Bundled
- Binary
- Installed
- In-memory

### Source



The development version of your package. The collection of files on your computer.

### Bundled

- A compressed, tar.gz, source package with vignettes built
- .Rbuildignore files are excluded
  - Useful for data for example

# **Binary**

- A bundle that is built for a certain architecture
- Parsed format, skipping the development tools needed to take the package between source and being interpretable by R

## Installed



- A binary package decompressed into a package library for R
  - The package library is the directory or directories where <a href="library">library</a>(packagename) searches
    - .libPaths()

## **In-memory**

When you use a package, it is in memory. When developing, a package does not have to be installed to be in memory.

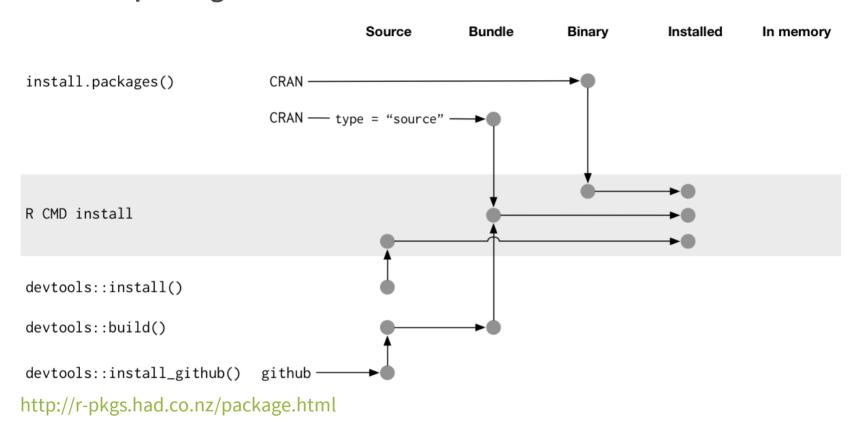
- packagename::function()
  - loads packagename

### Note:

- library(packagename)
  - loads and attaches packagename

# Possible package states





R/



```
typicalr
   DESCRIPTION
   NAMESPACE
       call_me.R
       data.R
       utility.R
   data
   └─ ToothGrowth.RData
  inst
   man
   ToothGrowth.Rd
     — call_me.Rd
   src
   tests
   typicalr.Rproj
   vignettes
```

R/



```
typicalr

DESCRIPTION

NAMESPACE

R

Call_me.R

data.R

utility.R

data

foothGrowth.RData

inst

man

call_me.Rd

call_me.Rd

src

tests

typicalr.Rproj

vignettes
```

```
typicalr/R
— call_me.R
— data.R
— utility.R
```

### Code

- Large functions in their own R files
- Utility functions, that your package uses, in one R file
- Bad code

```
o library(), require(), source()
o options(), par()
```

### **DESCRIPTION**



- Title
  - 65 characters, no punctuation
- Version
  - The version of the package
- Description
  - One paragraph

- Authors@R
  - Roles
    - **cre\***: Creator or maintainer.
    - aut\*: Author or authors, that have made significant contributions.
    - **ctb**: Contributors, have made smaller contributions.
    - **cph**: Copyright holder. Used if copyright is held by someone other than author, typically company.

<sup>\*:</sup>required

### **DESCRIPTION**



- Depends & Imports
  - Packages and versions that your package needs
  - Versions are optional
  - Depends: Attaches!
  - Imports: Loads!
- Suggests
  - Added functionality

- LazyData
  - Datasets occupy no memory until loaded
- License
  - Can be a file; LICENSE
  - Influences permissions of who can distribute and modify in what way
  - o Most common; MIT, GPL-3, CC0.
  - https://tldrlegal.com/
  - CRAN requires a license

# Versioning



# 0.0.0.9000

## major.minor.patch.dev

- Major
  - Large changes, not always backwards compatible
  - o Usually 1 upon first release out of dev
- Minor
  - Bug fixes & new features. Most common
- Patch
  - Small bugfixes, no new features.
- Dev
  - Only used while under development
  - Always starts at 9000

## man/



```
typicalr
DESCRIPTION
NAMESPACE
R
Call_me.R
data.R
data
ToothGrowth.RData
inst
man
Call_me.Rd
src
tests
typicalr.Rproj
vignettes
```

- Roxygen2
  - ?function
  - Comment block, #', preceding a function
  - Tags, @tags , map values
  - No tag for introduction
    - title\*
    - description
    - details
  - Special characters @\%, escape with \

#### call\_me.R

```
#' Output "Call me " followed by input.
#'
#' @param x A character or characters.
#' @return The string "Call me " and \code{x}. I'll write this
#' to display how to section with tags.
#' @examples
#' call_me("Maeby")
call_me <- function(x) {
   paste("Call me ", x, sep="")
}</pre>
```

## man/



### call\_me.R

```
#' Output "Call me " followed by input.
#'
#' @param x A character or characters.
#' @return The string "Call me " and \code{x}. I'll w.
#' to display how to section with tags.
#' @examples
#' call_me("Maeby")
call_me <- function(x) {
   paste("Call me ", x, sep="")
}</pre>
```

### call\_me.Rd

```
% Generated by roxygen2: do not edit by hand
% Please edit documentation in R/call_me.R
\name{call_me}
\alias{call_me}
\title{Output "Call me " followed by input.}
\usage{
call_me(x)
}
\arguments{
\item{x}{A character or characters.}
}
\value{
The string "Call me " and \code{x}. I'll write this
to display how to section with tags.
```

>?call\_me
call\_me {typicalr}

R Documentation

### Output "Call me " followed by input.

#### Description

Output "Call me " followed by input.

#### Usage

call me(x)

#### **Arguments**

x A character or characters.

#### **Value**

The string "Call me" and x. I'll write this to display how to section with tags.

#### Examples

call\_me("Maeby")

## man/ for datasets



```
typicalr
DESCRIPTION
NAMESPACE
R
call_me.R
data.R
utility.R
data
ToothGrowth.RData
inst
man
Call_me.Rd
call_me.Rd
src
tests
typicalr.Rproj
vignettes
```

```
head(ToothGrowth)

## len supp dose
## 1 4.2 VC 0.5
## 2 11.5 VC 0.5
## 3 7.3 VC 0.5
## 4 5.8 VC 0.5
## 5 6.4 VC 0.5
## 6 10.0 VC 0.5
```

### data.R:

```
#' The Effect of Vitamin C on Tooth Growth in Guinea Pigs
#'

#' The response is the length of odontoblasts (cells responsible for tooth growth)
#' in 60 guinea pigs. Each animal received one of three dose levels of vitamin C
#' (0.5, 1, and 2 mg/day) by one of two delivery methods, orange juice or ascorbic
#' acid (a form of vitamin C and coded as VC).
#'

#' @usage ToothGrowth
#'

#' @format A data frame with 60 observations on 3 variables.
#' \describe{
#' \item{len}{Tooth length}
#' \item{supp}{Supplement type (VC or OJ).}
#' \item{dose}{Dose in milligrams/day}
```

## vignettes/



- A more complete guide to your package
  - o For user/you
  - Examples and use cases
- knitr & rmarkdown
  - knitr: add r code to markdown
- vignettes/package-vignette.Rmd

```
usethis::use_vignette("typicalr-vignette")
```

typicalr-vignette.Rmd

```
title: "Vignette Title"
author: "Vignette Author"
date: "2021-09-24"
output: rmarkdown::html_vignette
vignette: >
   %\VignetteIndexEntry{Vignette Title}
   %\VignetteEngine{knitr::rmarkdown}
   %\VignetteEncoding{UTF-8}
```

## vignettes/



- A more complete guide to your package
  - o For user/you
  - o Examples and use cases
- knitr & rmarkdown
  - knitr: add r code to markdown
- vignettes/package-vignette.Rmd

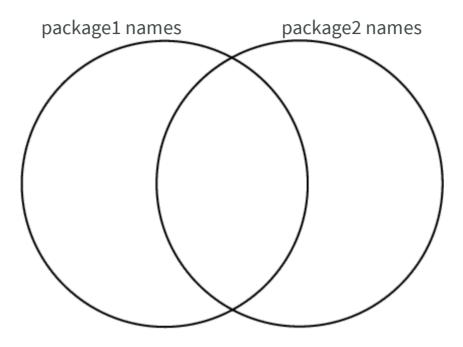
```
usethis::use_vignette("typicalr-vignette")
```

typicalr-vignette.Rmd

```
title: "typicalr"
author: "Sebastian DiLorenzo"
date: "2021-09-24"
output: rmarkdown::html_vignette
vignette: >
   %\VignetteIndexEntry{typicalr}
   %\VignetteEngine{knitr::rmarkdown}
   %\VignetteEncoding{UTF-8}
```

## **NAMESPACE**





- @imports and @importsFrom
  - Defines how/where a function in one package finds a function in another
  - @imports pkg
  - @importsFrom pkg function
- @export
  - Defines which functions are available to user
  - Do not export data

### **NAMESPACE**



call\_me.R:

```
#' Output "Call me " followed by input.
#'
#' @param x A character or characters.
#' @return The string "Call me " and \code{x}. I'll write this
#' to display how to section with tags.
#' @examples
#' call_me("Maeby")
#' @export
call_me <- function(x) {
   paste("Call me ",x,sep="")
}</pre>
```

utility.R:

```
#' @import knitr
NULL
```

NAMESPACE:

```
# Generated by roxygen2: do not edit by hand
export(call_me)
import(knitr)
```

### **NAMESPACE**



## Import in **DESCRIPTION** and in **NAMESPACE**!?

- DESCRIPTION Imports:
  - "My package needs this package to work"
- NAMESPACE @import
  - "When my package uses this function, use the one from the package in the NAMESPACE"
- Additional effects:
  - NAMESPACE removes need for ::
    - package::function() or function()

## data/



### Package types:

- Functional
  - Performs a or several functions
  - Contains no or small datasets, <1 MB</li>
- Dataset
  - Contains an interesting dataset
  - Easy to import
  - Few or no functions

```
#Create data in package automatically
usethis::use_data(object, package)

#Manually
save(object, file="path/to/package/data/objec")

#Access raw data
system.file("extdata","filename.csv", package
```

### Data types:

- Binary data, .Rdata or .rda
  - o data/ folder
  - A single object with the same name as the data file
- Function data
  - o R/sysdata.rda
  - Data that your functions need
- Raw data, .xlsx , .csv etc
  - inst/extdata folder

## src/



- Compiled code
  - o Rcpp
  - o rJava
- Scripts
  - o inst/
  - Dependencies

- 1. usethis::use\_rcpp()
  - Edit DESCRIPTION
  - 0 #' @useDynLib packagename
  - #' @importFrom Rcpp sourceCpp
- 2. .cpp file in src/

src/filename.cpp:

```
#include <Rcpp.h>
using namespace Rcpp;

// This is a simple example of exporting a C++ function to R. You can
// source this function into an R session using the Rcpp::sourceCpp
// function (or via the Source button on the editor toolbar). Learn
// more about Rcpp at:
//
// http://www.rcpp.org/
// http://adv-r.had.co.nz/Rcpp.html
// http://gallery.rcpp.org/
//
// [[Rcpp::export]]
NumericVector timesTwo(NumericVector x) {
    return x * 2;
}
```

## src/



- Compiled code
  - RcpprJava
- Scripts
  - o inst/
  - Dependencies

### R/RcppExports.R:

```
# Generated by using Rcpp::compileAttributes(
# Generator token: 10BE3573-1514-4C36-9D1C-5A

timesTwo <- function(x) {
    .Call('_typicalr_timesTwo', PACKAGE = 'ty)
}</pre>
```

```
1. usethis::use_rcpp()
```

- Edit DESCRIPTION
- 0 #' @useDynLib packagename
- #' @importFrom Rcpp sourceCpp
- 2. .cpp file in src/
- 3. pkgbuild::compile\_dll()
- 4. devtools::document()
- 5. Build & Reload
- 6. Add documentation to .cpp

# git, Github, Rstudio and you!



- git
  - Version control
  - Working in groups
  - Rstudio integration
- Github
  - Unoffical repository
  - o devtools::install\_github()
  - R Package development environment
  - Issues

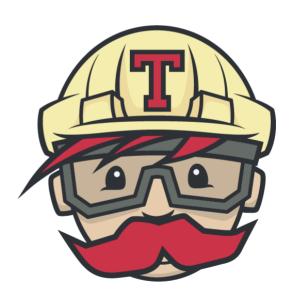


## CRAN and R CMD check

**NB**§S

- Comprehensive R Archive Network
  - R package repository
  - Sign of quality
- R CMD check
  - More than 50 individual checks
  - Three messages:
    - **ERROR**: Always fix.
    - WARNING: Should probably fix. Definitely for CRAN submit.
    - NOTE: Try to solve to CRAN submit, else do not bother.
  - o devtools::check()
- Travis-CI
  - Integrated with your github repository
  - Automates R CMD check
  - Multiple operating systems

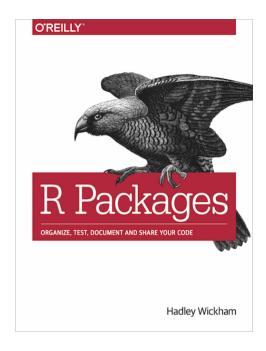




## Summary



- What is an R package?
- Possible package states
- Package structure:
  - ∘ Code | r/
  - Metadata | DESCRIPTION
  - Documentation | man/
  - Vignettes
  - Import & Export | NAMESPACE
  - Data | data/
  - Compiled code | src/
- git, Github, Rstudio and you!
- CRAN and R CMD check



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

# Thank you. Questions?



