Building R packages and publishing them on GitHub

Dr. Nicole Fortuna: n.fortuna@uq.edu.au

ResBazQld 2022

Purpose of the workshop

- An introduction to building an R package, using Git, and publishing to GitHub
- This is not a workshop on how to build a function, it is assumed that you will have some familiarity with how to do this already
  - I have provided some toy functions for you to work with. You will not be expected to build a function in this class
- Will demonstrate the utility of Rstudio for all aspects of package development and publication

Why bother with R packages?

- There are many benefits for building R packages
  - Forces you to document your code
  - Easily source all your functions
  - Easy to share
  - And importantly… it is now easy to do.

Taking advantage of the philosophy of R

- The R programming language is extremely successful
  - Has a large community
  - Free and open source
  - Easily extendable language

Tools for building packages

- RStudio
- Supporting packages
  - devtools
  - roxygen2
  - usethis

Hadley Wickham: the mastermind

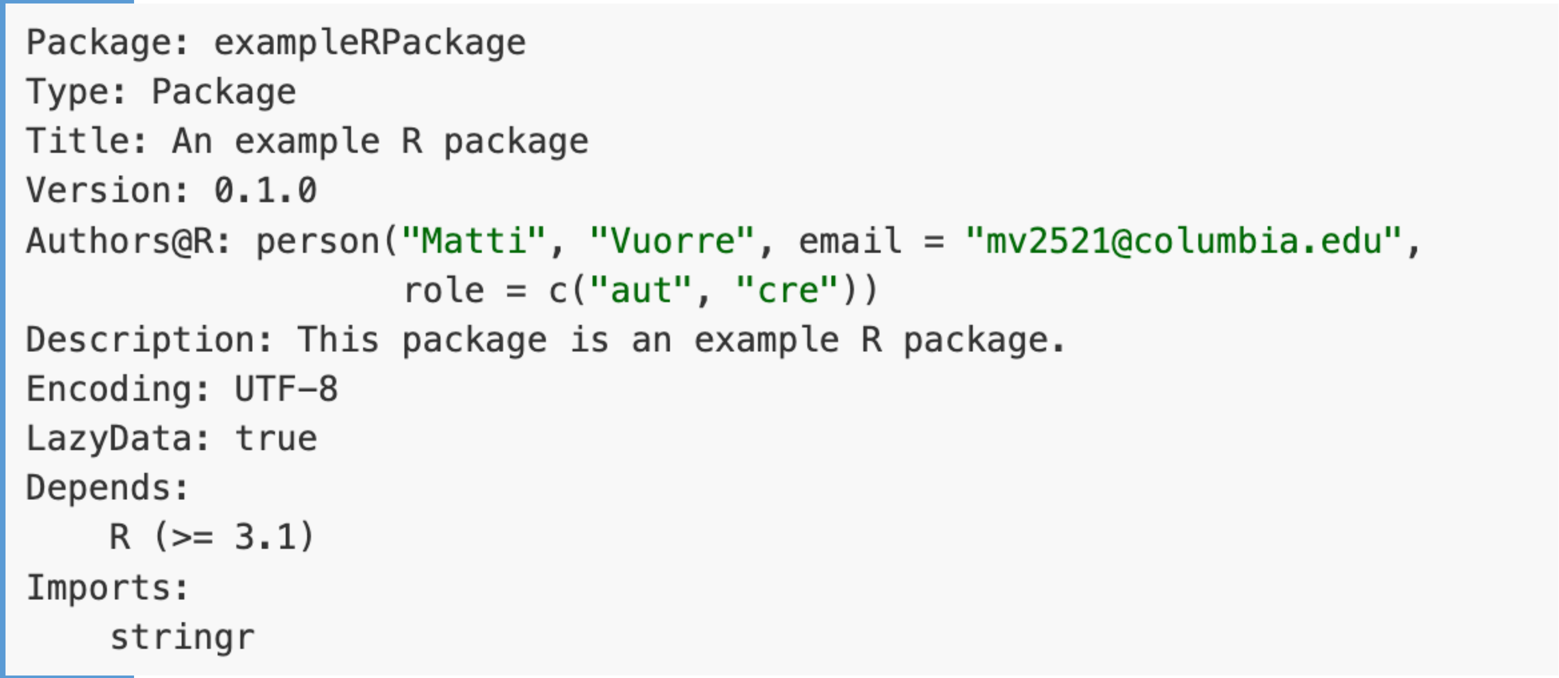
What is an R package?

- A library
- An extension of the existing language
- A layer of abstraction

The components of an R package

- R folder: code
- man folder: manuals
- DESCRIPTION: metadata
- NAMESPACE: exported functions

DESCRIPTION file



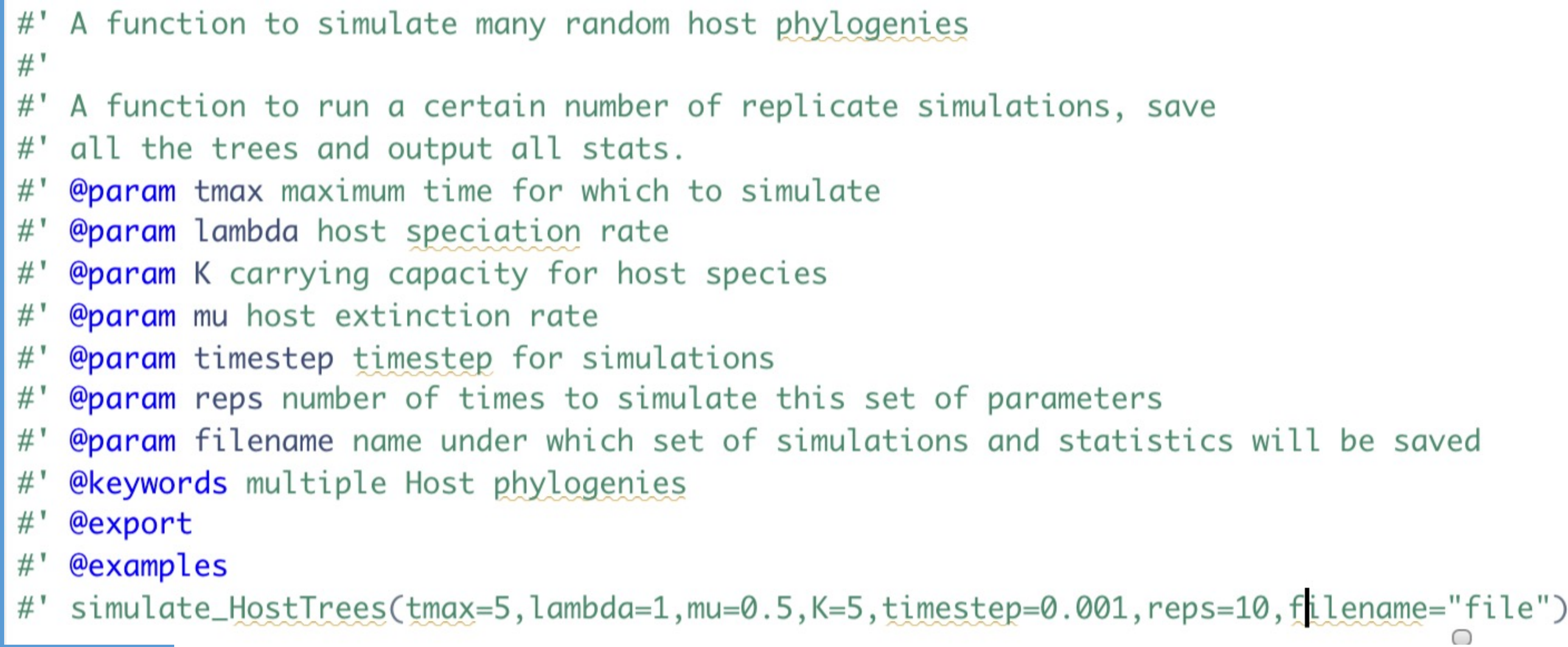
```
Package: exampleRPackage
Type: Package
Title: An example R package
Version: 0.1.0
Authors@R: person("Matti", "Vuorre", email = "mv2521@columbia.edu",
                  role = c("aut", "cre"))
Description: This package is an example R package.
Encoding: UTF-8
LazyData: true
Depends:
    R (>= 3.1)
Imports:
    stringr
```

[Crediting authors]

Documenting your Package

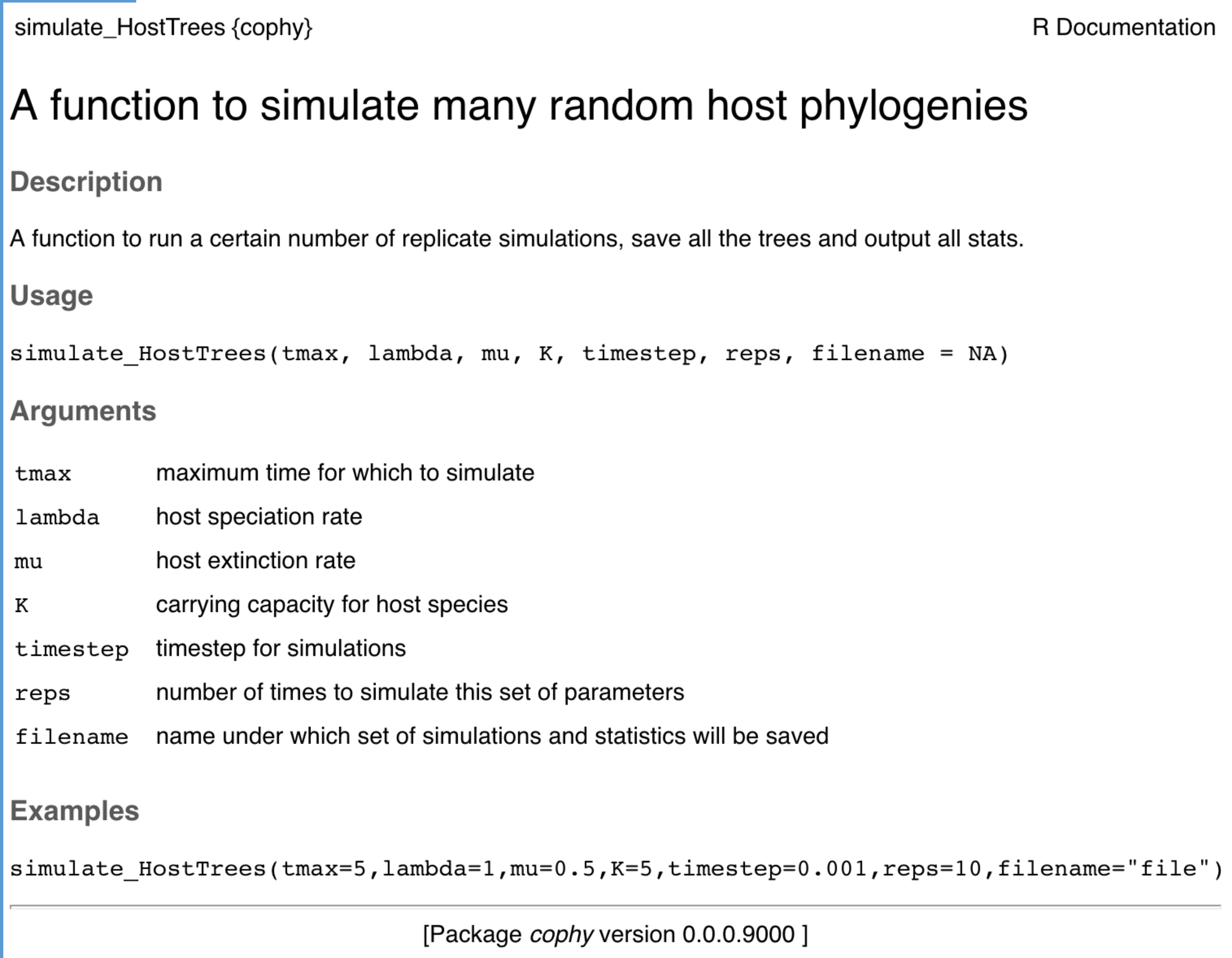
- Remembers what you did so that you don’t have to
- Makes it easy to share your code with other people

Documenting your Package: help files



```
#' A function to simulate many random host phylogenies
#'
#' A function to run a certain number of replicate simulations, save
#' all the trees and output all stats.
#' @param tmax maximum time for which to simulate
#' @param lambda host speciation rate
#' @param K carrying capacity for host species
#' @param mu host extinction rate
#' @param timestep timestep for simulations
#' @param reps number of times to simulate this set of parameters
#' @param filename name under which set of simulations and statistics will be saved
#' @keywords multiple Host phylogenies
#' @export
#' @examples
#' simulate_HostTrees(tmax=5,lambda=1,mu=0.5,K=5,timestep=0.001,reps=10,filename="file")
```

Documenting your Package: help files



simulate_HostTrees {cophy} R Documentation

A function to simulate many random host phylogenies

Description

A function to run a certain number of replicate simulations, save all the trees and output all stats.

Usage

```
simulate_HostTrees(tmax, lambda, mu, K, timestep, reps, filename = NA)
```

Arguments

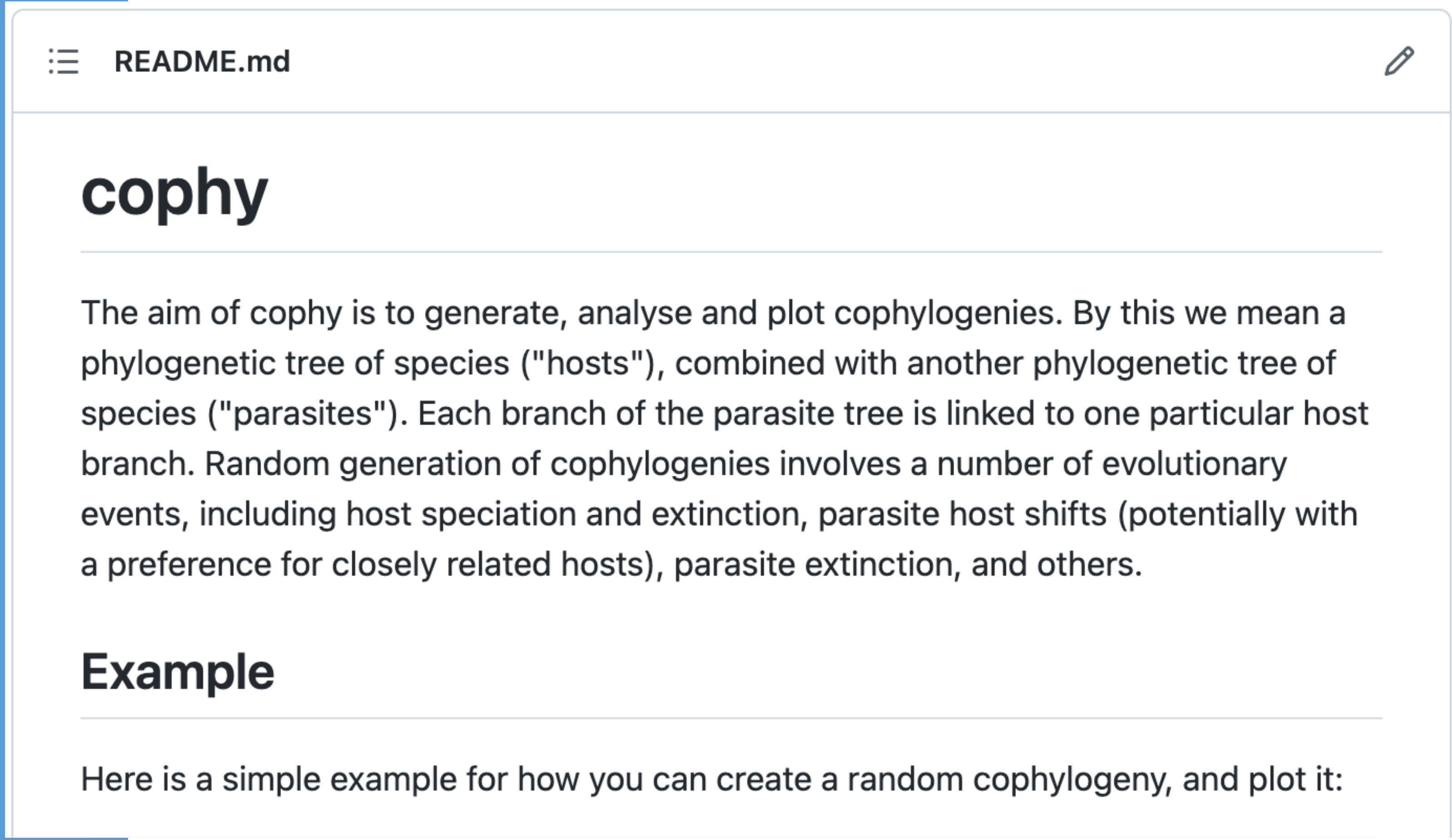
`tmax`	maximum time for which to simulate
`lambda`	host speciation rate
`mu`	host extinction rate
`K`	carrying capacity for host species
`timestep`	timestep for simulations
`reps`	number of times to simulate this set of parameters
`filename`	name under which set of simulations and statistics will be saved

Examples

```
simulate_HostTrees(tmax=5,lambda=1,mu=0.5,K=5,timestep=0.001,reps=10,filename="file")
```

[Package *cophy* version 0.0.0.9000]

Documenting your Package: README



cophy

The aim of cophy is to generate, analyse and plot cophylogenies. By this we mean a phylogenetic tree of species ("hosts"), combined with another phylogenetic tree of species ("parasites"). Each branch of the parasite tree is linked to one particular host branch. Random generation of cophylogenies involves a number of evolutionary events, including host speciation and extinction, parasite host shifts (potentially with a preference for closely related hosts), parasite extinction, and others.

Example

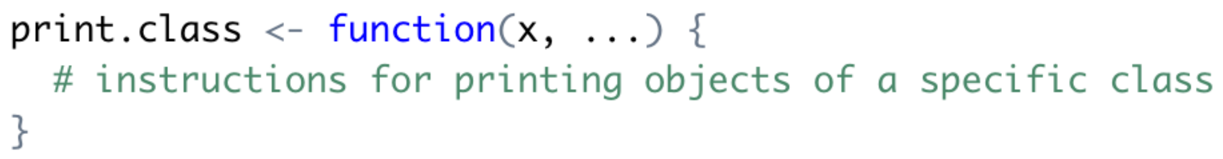
Here is a simple example for how you can create a random cophylogeny, and plot it:

Documenting your Package: vignettes

- Also designed to teach users how to engage with the package
- More specialised than a README

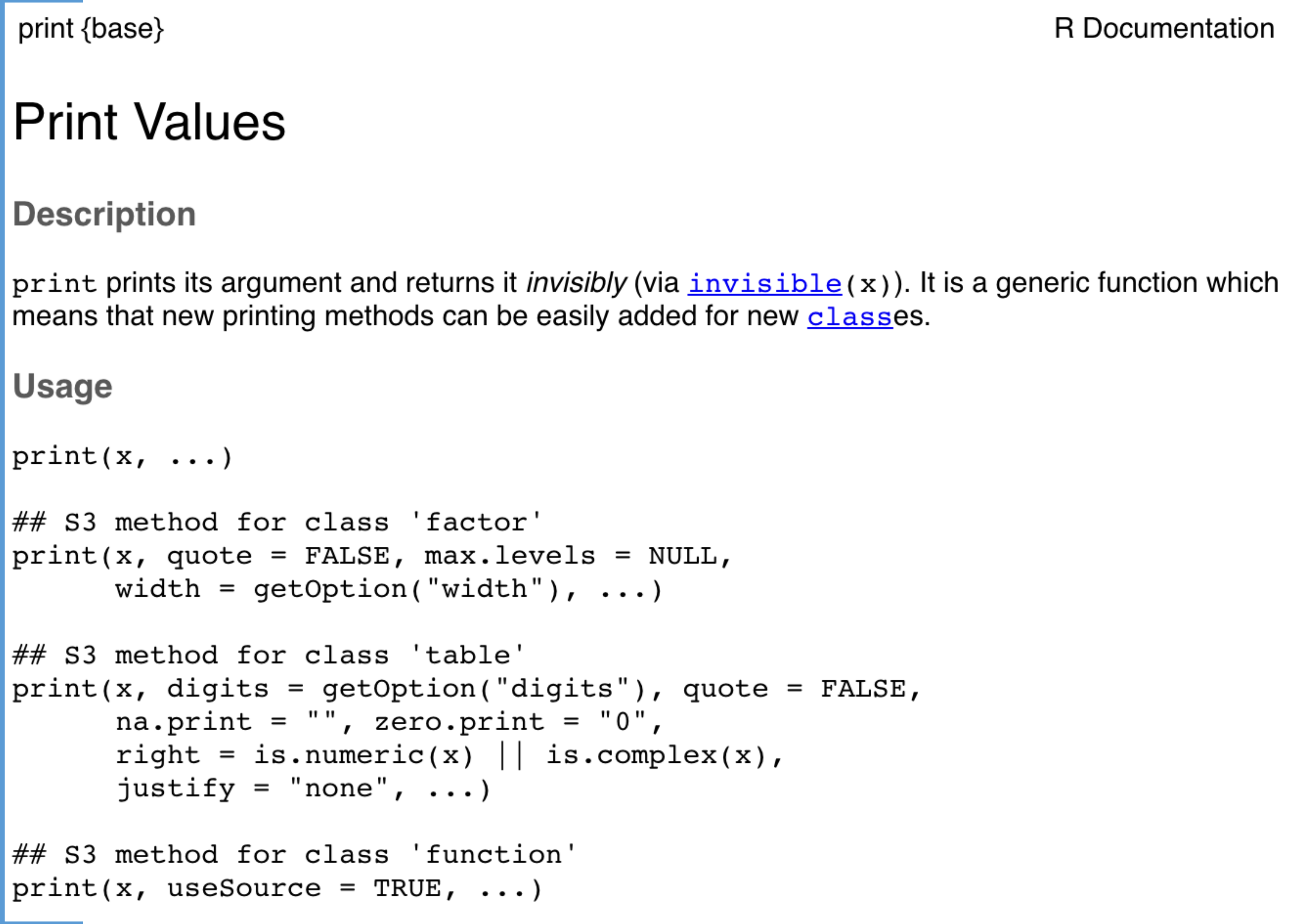
Taking advantage of the features of R

- S3 Classes and S3 methods
- These methods allow us to have one generic function, e.g. print(), that displays the object differently depending on its type: printing a linear model is very different to printing a data frame.
- To create a new printing method, assign a function to print.class



```r
print.class <- function(x, ...) {
  # instructions for printing objects of a specific class
}
```

Taking advantage of the features of R



print {base} R Documentation

Print Values

Description

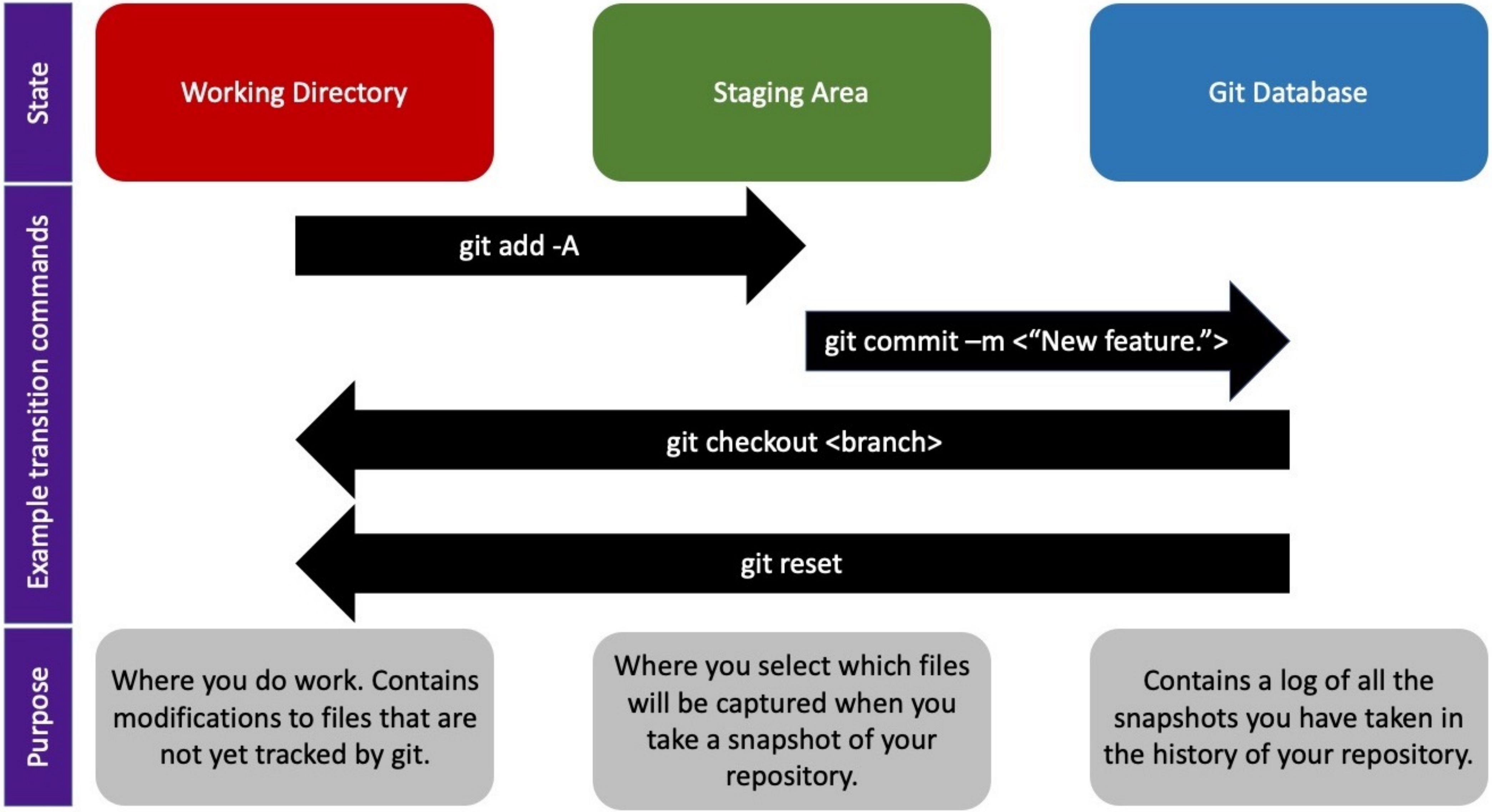
`print` prints its argument and returns it *invisibly* (via `invisible(x)`). It is a generic function which means that new printing methods can be easily added for new `class`es.

Usage

```
print(x, ...)

## S3 method for class 'factor'
print(x, quote = FALSE, max.levels = NULL,
      width = getOption("width"), ...)

## S3 method for class 'table'
print(x, digits = getOption("digits"), quote = FALSE,
      na.print = "", zero.print = "0",
      right = is.numeric(x) || is.complex(x),
      justify = "none", ...)

## S3 method for class 'function'
print(x, useSource = TRUE, ...)
```

Keeping track of your package

- As your package expands over time, it will become necessary to track how your package develops
- Git is one of the most popular version control systems available



State

Working Directory

Staging Area

Git Database

Example transition commands

git add -A

git commit –m <“New feature.”>

git checkout <branch>

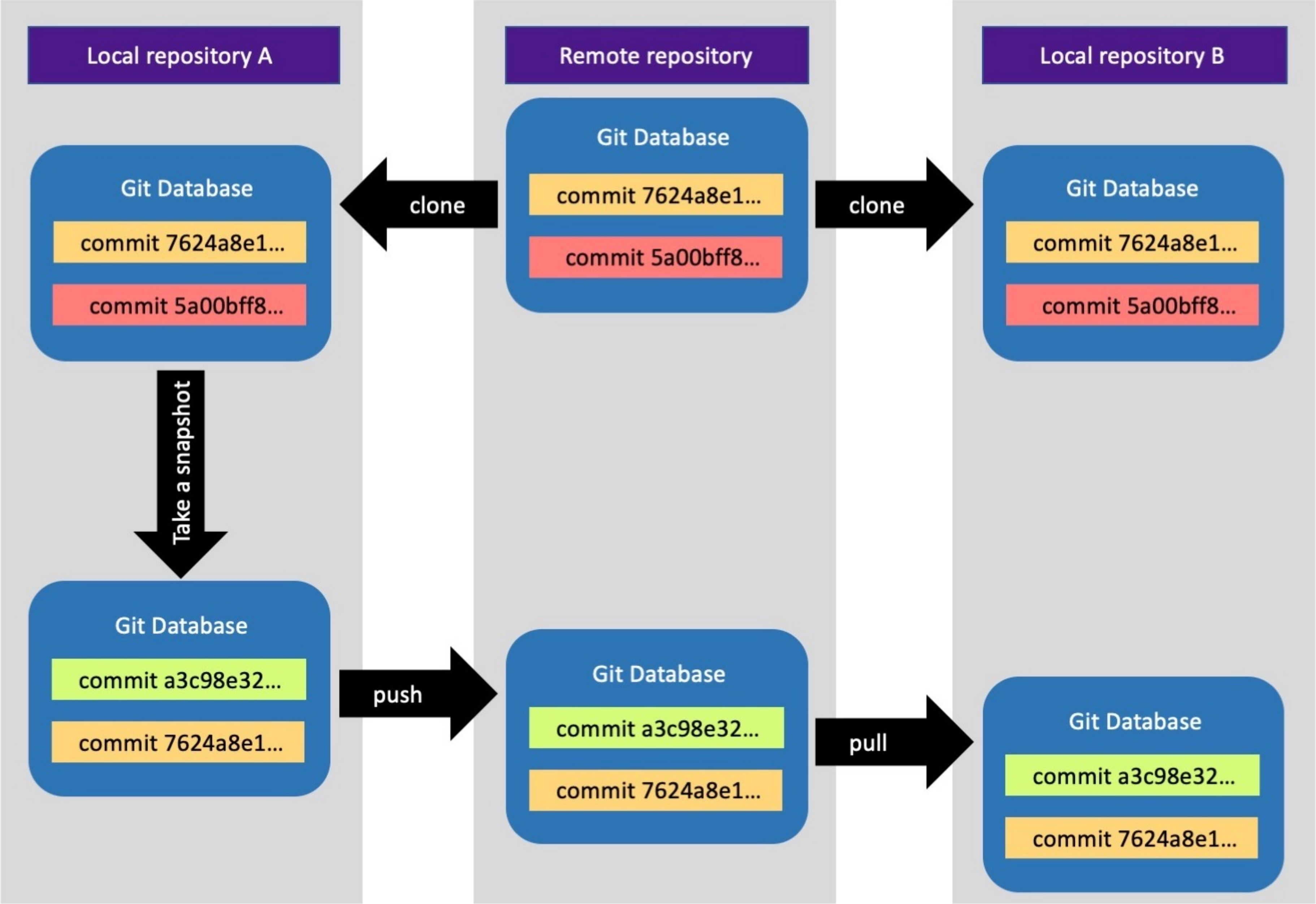
git reset

Purpose

Where you do work. Contains modifications to files that are not yet tracked by git.

Where you select which files will be captured when you take a snapshot of your repository.

Contains a log of all the snapshots you have taken in the history of your repository.


Local repository A
Remote repository
Local repository B
Git Database
commit 7624a8e1…
commit 5a00bff8…
clone
clone
Git Database
commit 7624a8e1…
commit 5a00bff8…
Git Database
commit 7624a8e1…
commit 5a00bff8…
Take a snapshot
Git Database
commit a3c98e32…
commit 7624a8e1…
push
Git Database
commit a3c98e32…
commit 7624a8e1…
pull
Git Database
commit a3c98e32…
commit 7624a8e1…

The benefits of Git

- Record of package over time
  - User can take ‘snapshots’ of the package at key times
- Ability to branch your project
- Share your code with others easily
- Useful for debugging

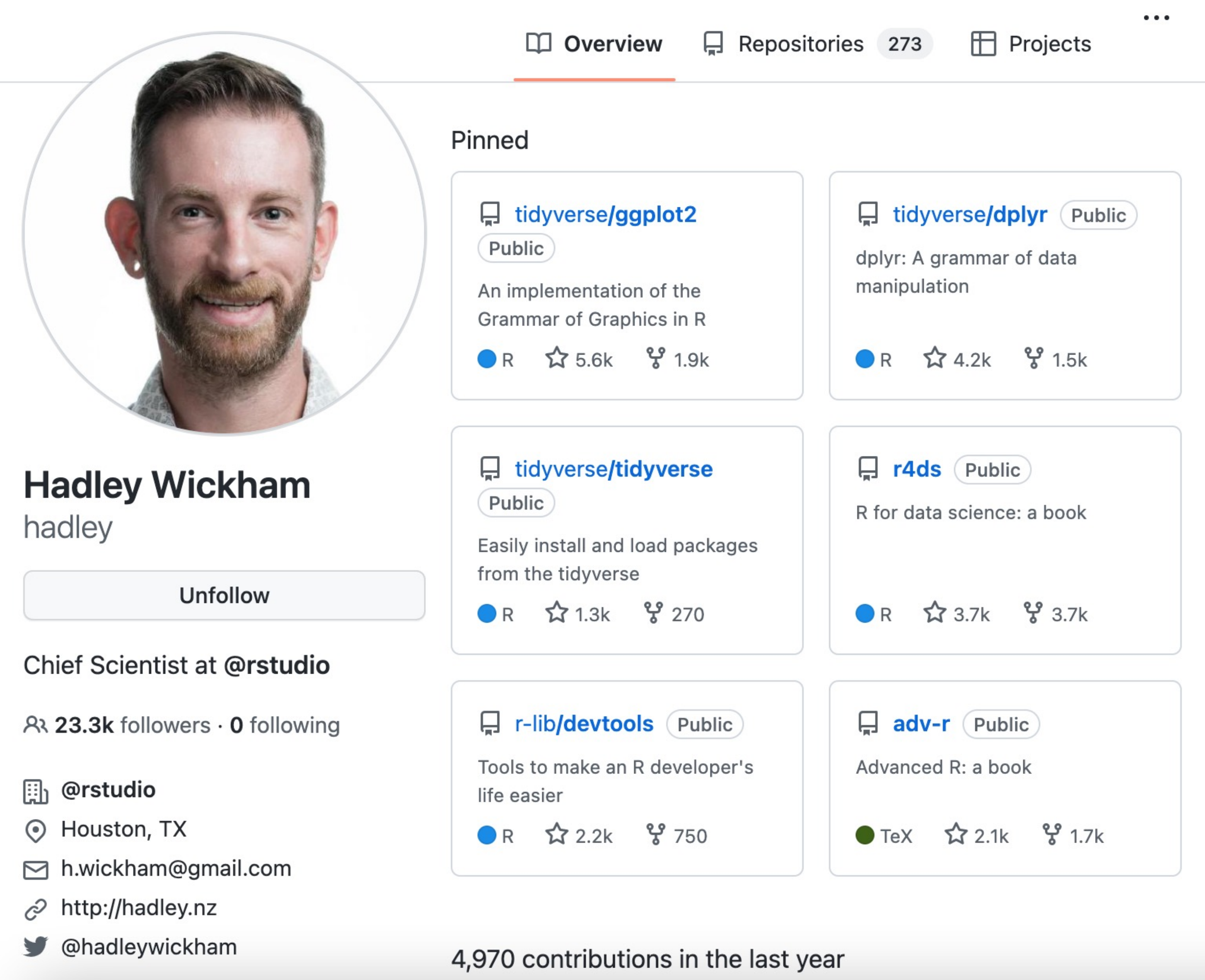
Where git truly shines

- The greatest benefits of Git can be recognised in community projects
- Facilitates integration of code by multiple authors (merging)
    - Automatically merges non-conflicting code
    - Indicates to the user any code that is conflicting for manual resolution

GitHub

- A code hosting platform integrated with the Git version control system
- Enables collaboration, and encourages open-source development

Publishing your package



Overview | Repositories 273 | Projects

Hadley Wickham
hadley

Unfollow

Chief Scientist at @rstudio

23.3k followers · 0 following

@rstudio
Houston, TX
h.wickham@gmail.com
http://hadley.nz
@hadleywickham

Pinned

tidyverse/ggplot2 Public
An implementation of the Grammar of Graphics in R
● R ☆ 5.6k 1.9k

tidyverse/dplyr Public
dplyr: A grammar of data manipulation
● R ☆ 4.2k 1.5k

tidyverse/tidyverse Public
Easily install and load packages from the tidyverse
● R ☆ 1.3k 270

r4ds Public
R for data science: a book
● R ☆ 3.7k 3.7k

r-lib/devtools Public
Tools to make an R developer's life easier
● R ☆ 2.2k 750

adv-r Public
Advanced R: a book
● TeX ☆ 2.1k 1.7k

4,970 contributions in the last year

Activity

https://github.com/NicoleZFortuna/Rpackages