

Reproducible Research:

Writing an R Package

Prof James P Curley

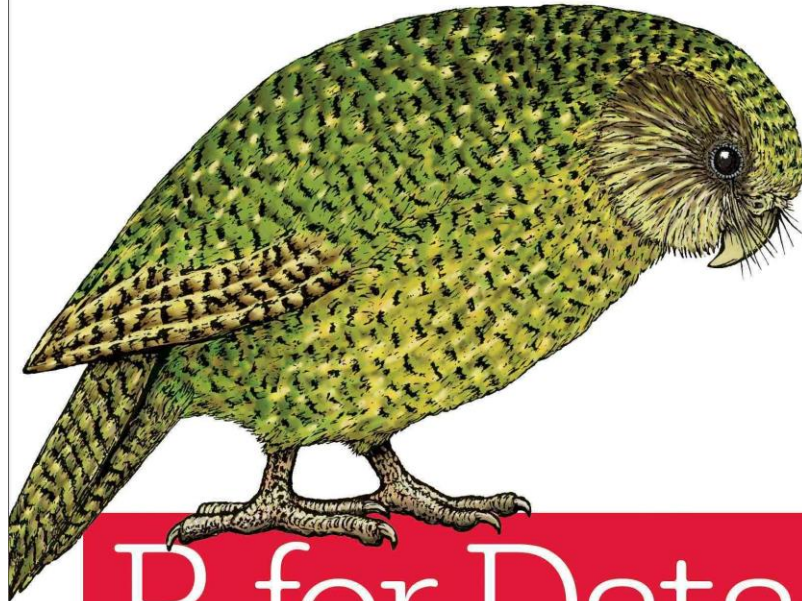
jc3181@columbia.edu

@jalapic

Materials will be available here:

- twitter.com/jalapic
- GitHub.com/jalapic/RPackage

O'REILLY®

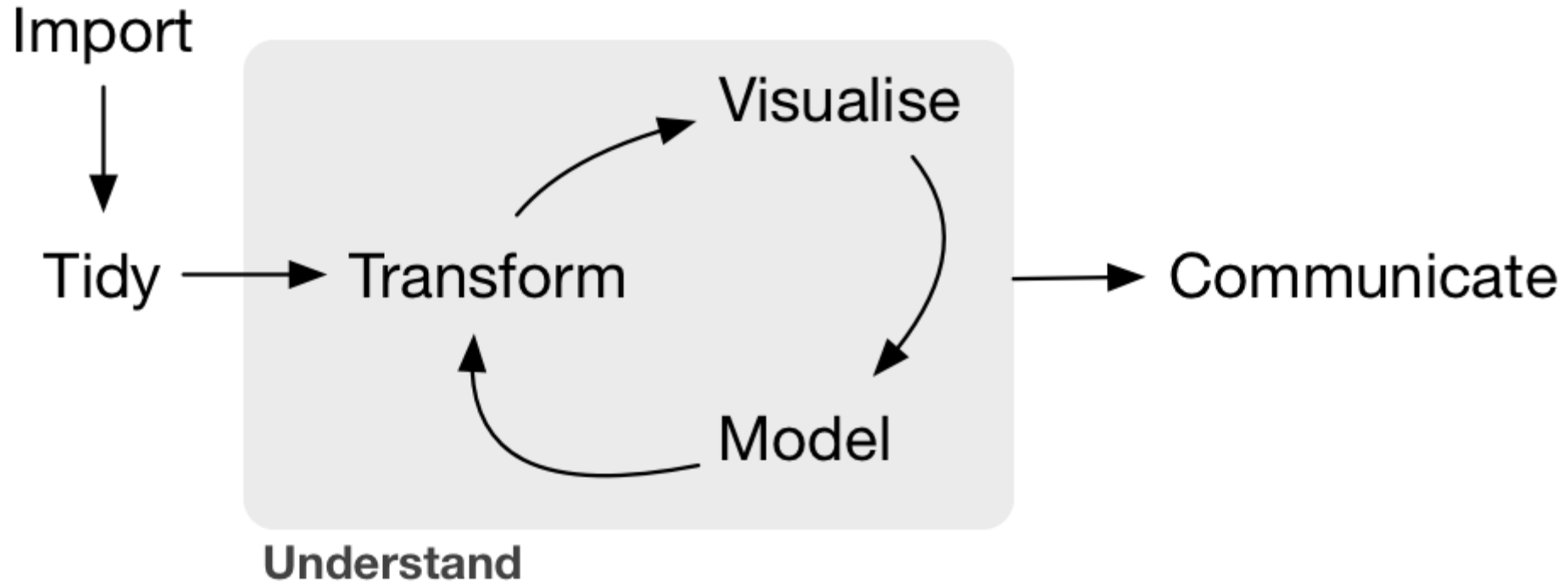


R for Data Science

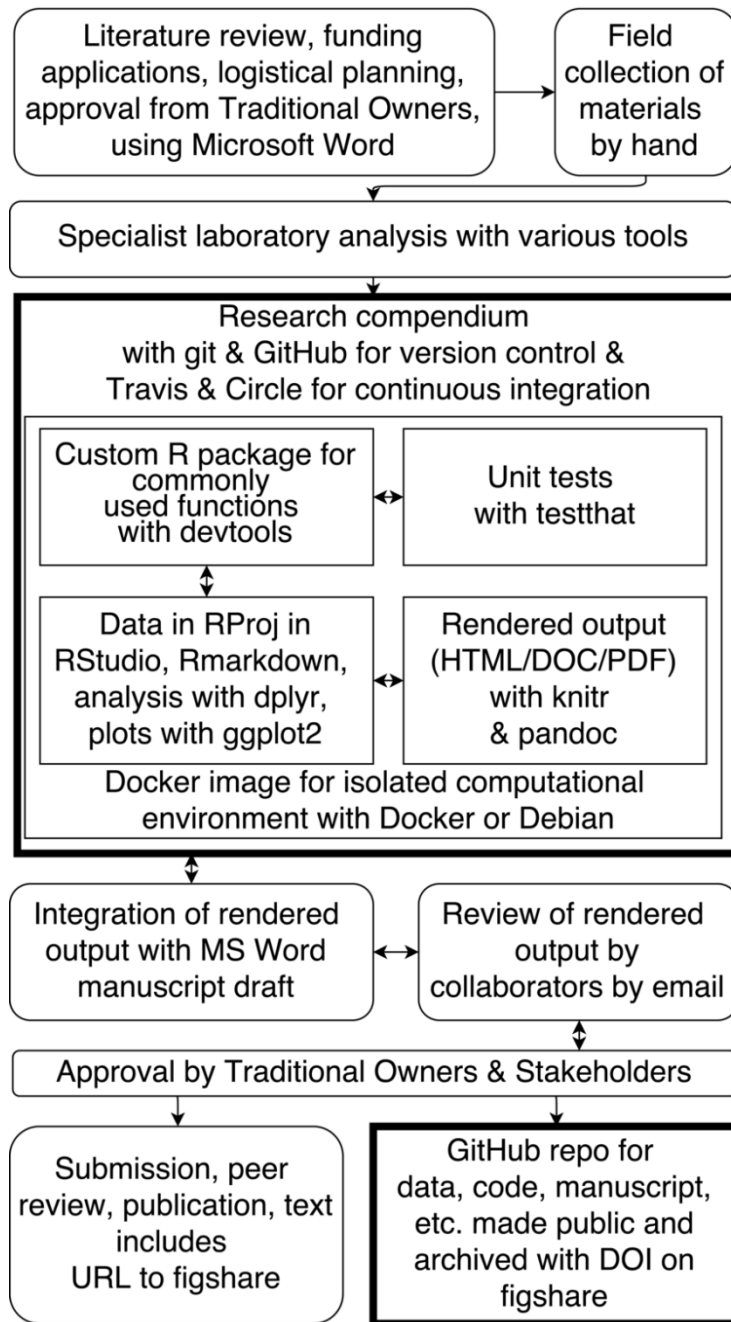
IMPORT, TIDY, TRANSFORM, VISUALIZE, AND MODEL DATA

Hadley Wickham &
Garrett Grolemund

Research Workflow – ideally automated and reproducible



Using R & Rstudio for Reproducible Research



Stage I:
Data Input

Stage II:
Data Processing

Stage III:
Data Analysis

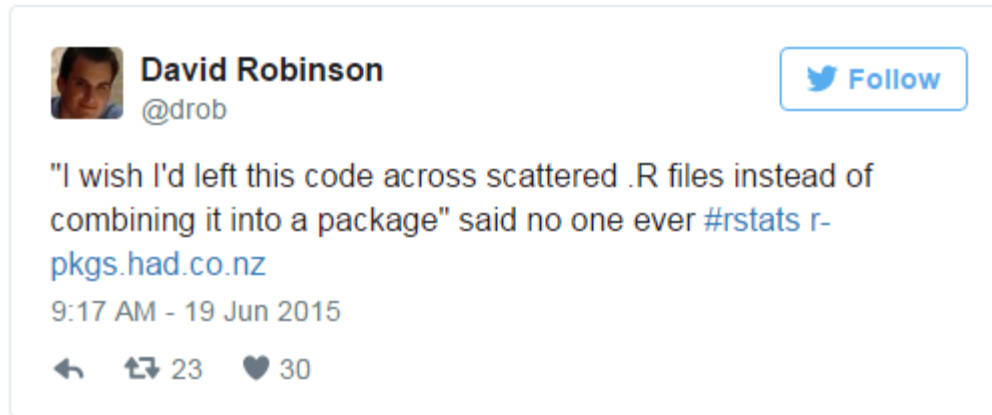
It can be overwhelming !

Let's start with one method –

Writing an R package

<https://www.practicereproducibleresearch.org/case-studies/benmarwick.html>

Inspiration for writing a R package



Organization: Keep all .R files and data related to a project in the same place

Documentation: Helpful documentation enables you to remember how to use functions you've written – as well as your colleagues

Collaboration: Others will be able to follow your work and reproduce it – they may even collaborate to help improve the work

Credit: Producing an R package is added productivity and can be evidence of contribution to research. Especially if the package is available on CRAN it shows that you have worked up to a high standard in making the package.

References

Very helpful tutorials/blogs:

<https://www.analyticsvidhya.com/blog/2017/03/create-packages-r-cran-github/>

http://web.mit.edu/insong/www/pdf/rpackage_instructions.pdf

<https://hilaryparker.com/2014/04/29/writing-an-r-package-from-scratch/>

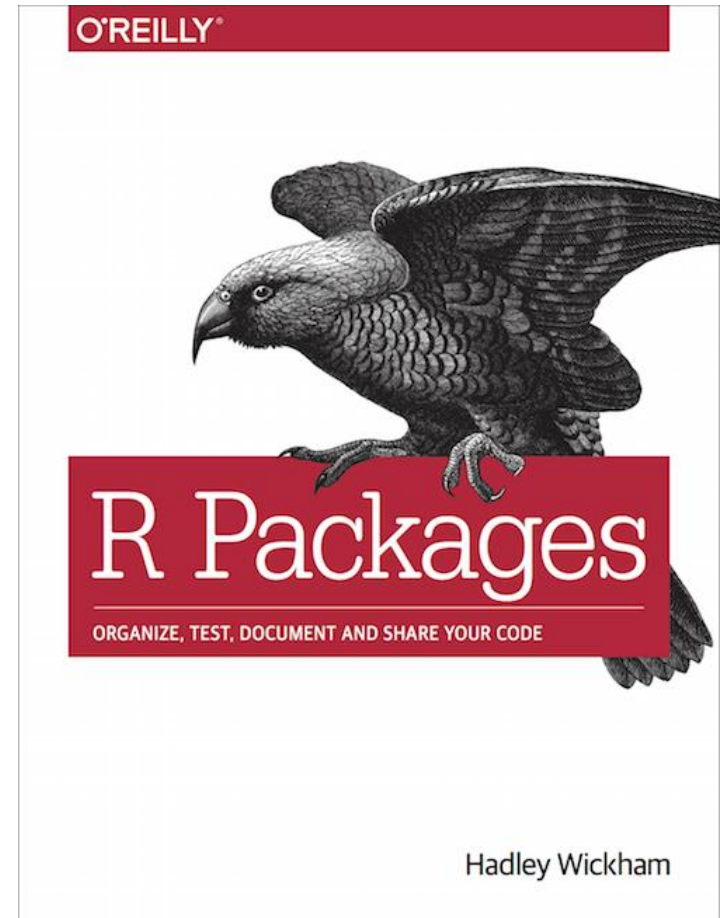
http://kbroman.org/pkg_primer/

http://stat545.com/packages00_index.html

<http://tinyheero.github.io/jekyll/update/2015/07/26/making-your-first-R-package.html>

If you're bold:

<https://cran.r-project.org/doc/manuals/r-release/R-exts.html>



The BEST resource

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

Writing our first basic R package

We will use Rstudio to build our package utilizing two packages:

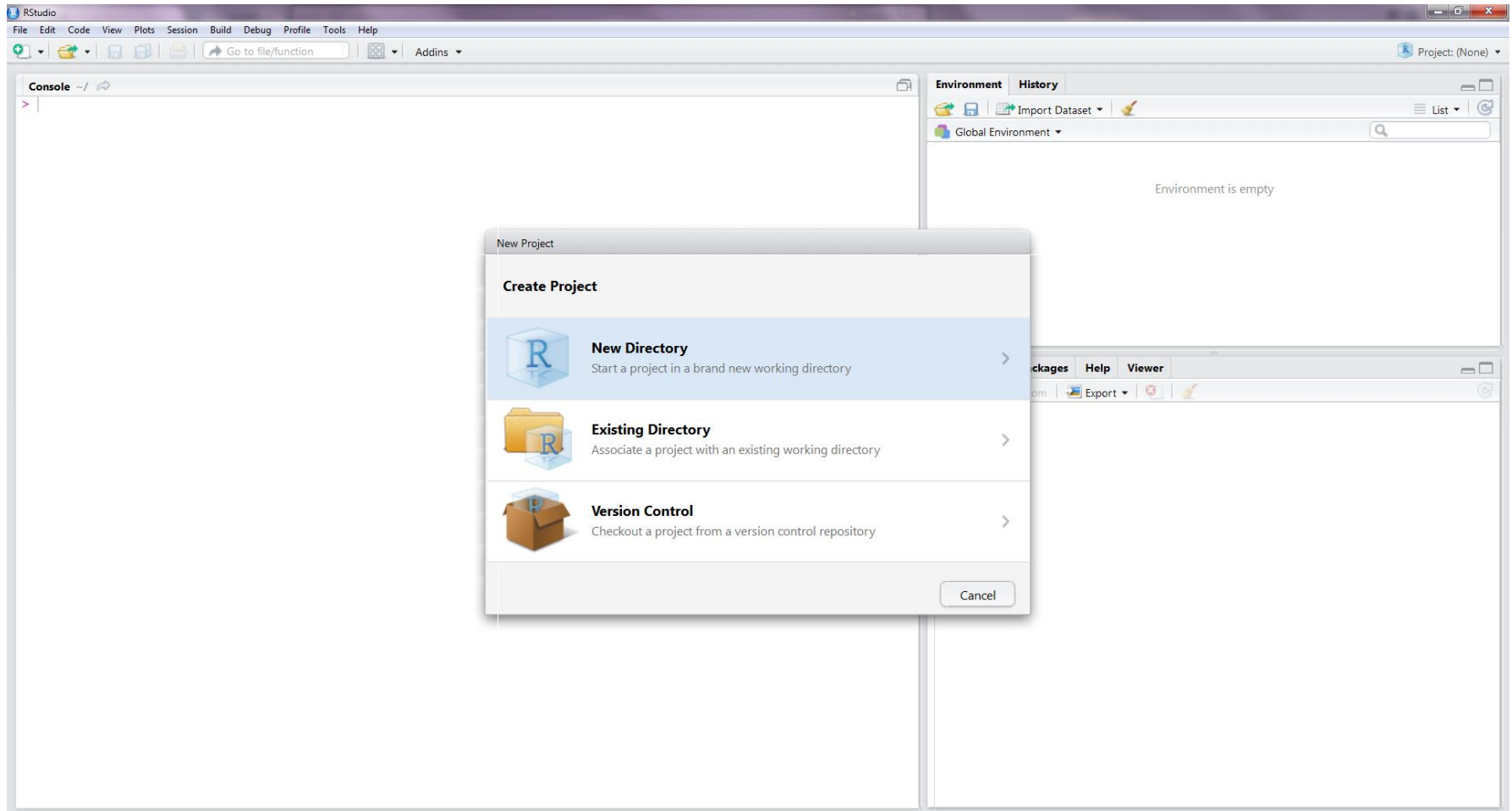
devtools

roxygen2

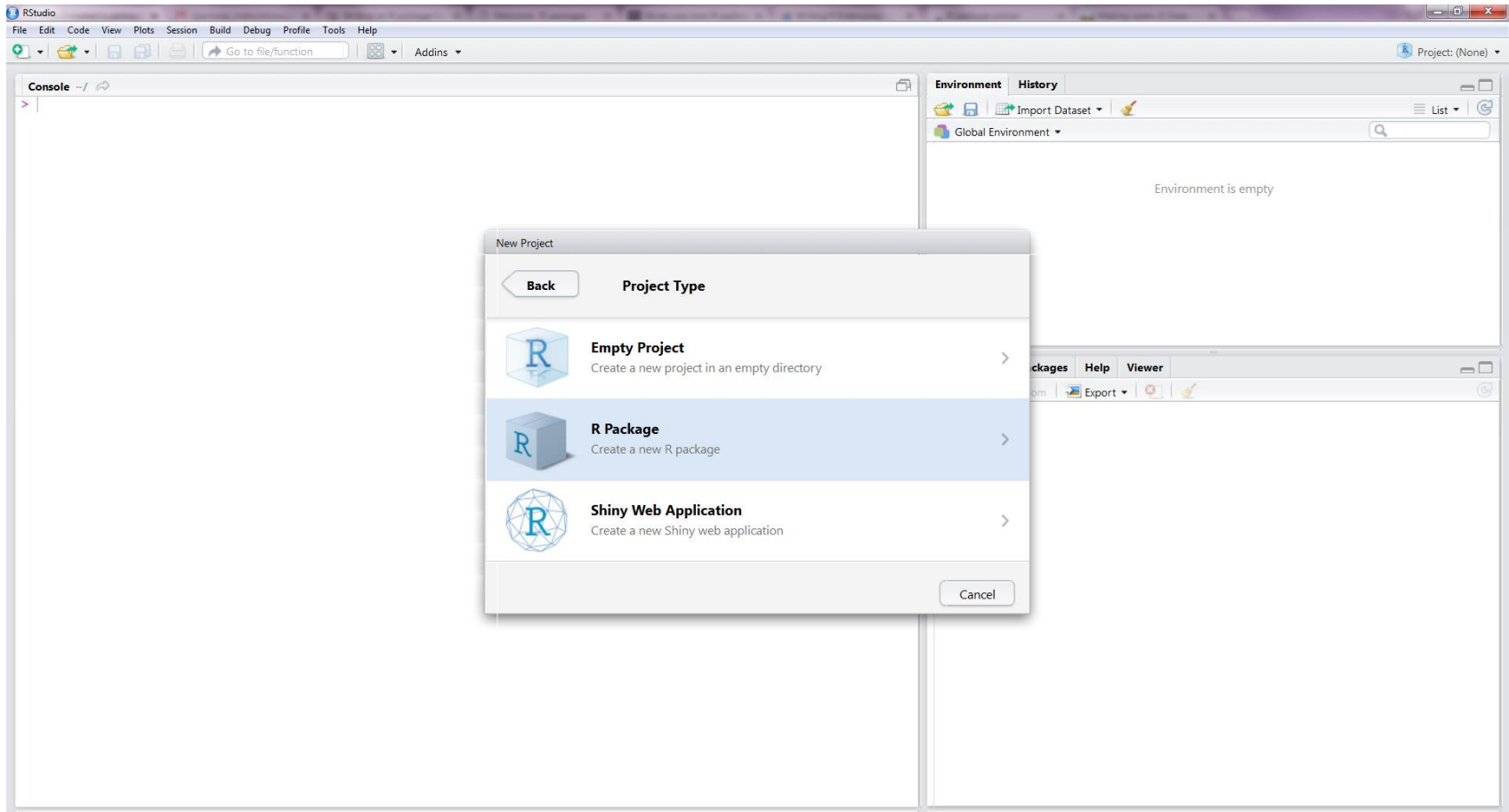
Project > New Project

or

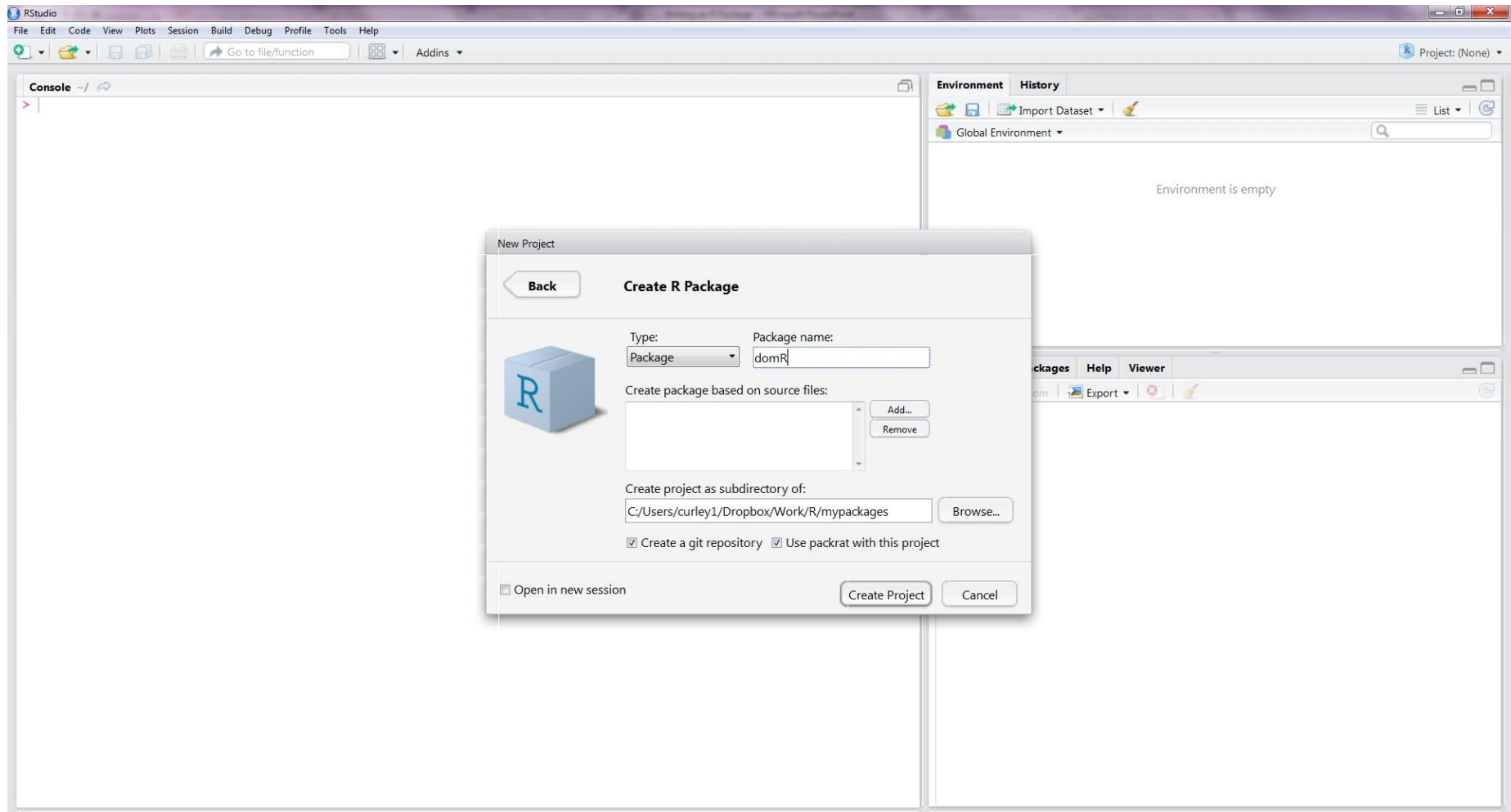
File > New Project



New Directory > R Package



Choose a package name and a folder location



Why “packrat” ?

You should see this happen....

The screenshot shows the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The toolbar below the menu has icons for file operations and a search bar. The status bar at the top right indicates 'Project: (None)'.

The console window on the left displays the following output:

```
R version 3.3.0 (2016-05-03) -- "Supposedly Educational"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

Initializing packrat project in directory:
- "C:/Users/curley1/Dropbox/Work/R/mypackages/domR"

Adding these packages to packrat:

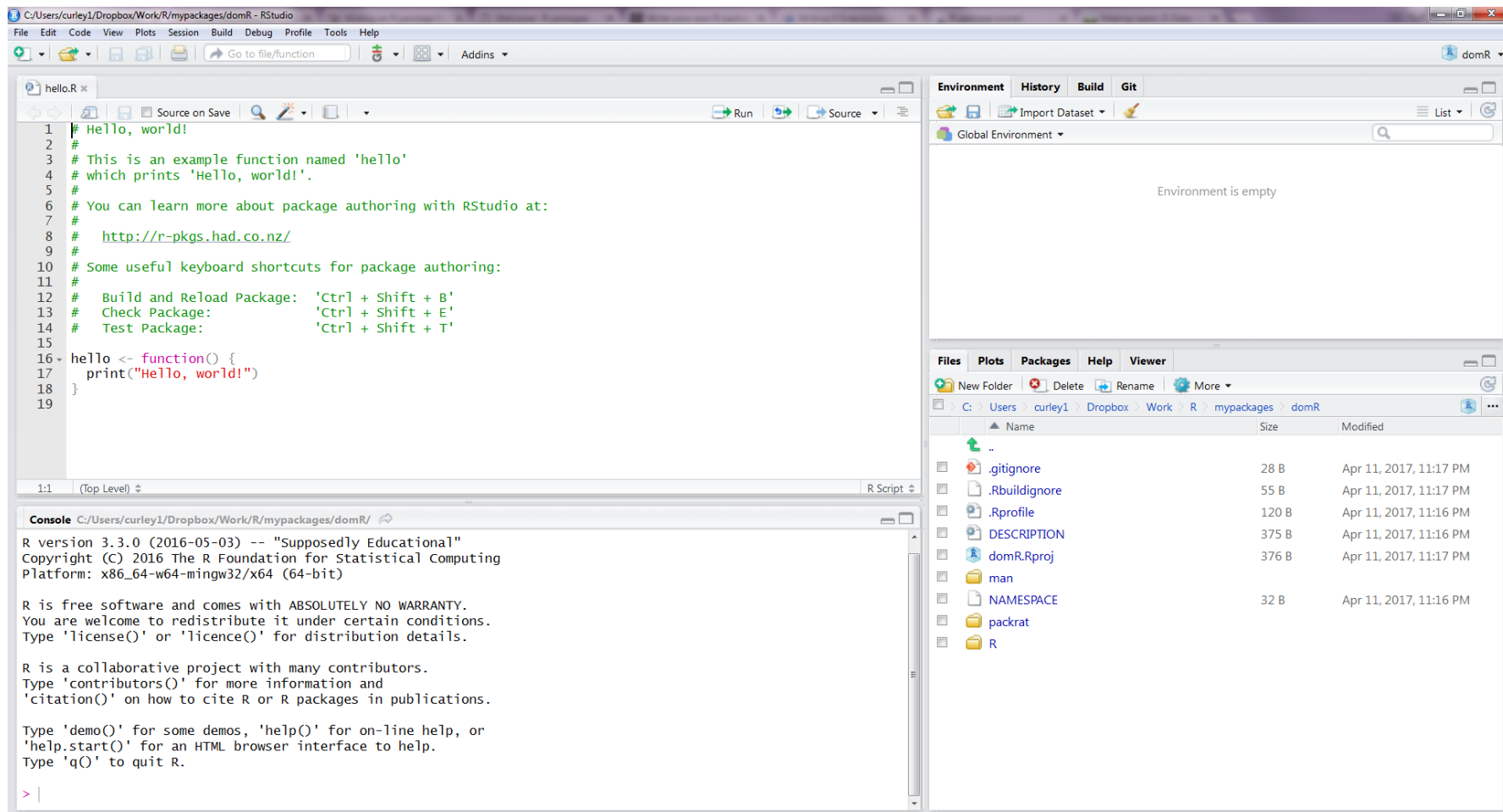
  packrat 0.4.7-1

Fetching sources for packrat (0.4.7-1) ... OK (CRAN archived)
Snapshot written to "C:/Users/curley1/Dropbox/Work/R/mypackages/domR/packrat/packrat.lock"
Installing packrat (0.4.7-1) ...
  OK (built source)
Initialization complete!
> |
```

The Environment pane on the right shows 'Global Environment' with the message 'Environment is empty'. Below it, the Packages pane lists installed packages:

Name	Description	Version
reshape	Flexibly reshape data.	0.8.5
reshape2	Flexibly Reshape Data: A Reboot of the Reshape Package	1.4.2
rio	A Swiss-Army Knife for Data I/O	0.4.16
rJava	Low-Level R to Java Interface	0.9-8
RJSONIO	Serialize R objects to JSON, JavaScript Object Notation	1.3-0
rlist	A Toolbox for Non-Tabular Data Manipulation	0.4.6.1

Several Files will be auto-generated in your folder like this: (I'll explain what these are)



Populate the DESCRIPTION file:

<http://r-pkgs.had.co.nz/description.html>

From this

```
1 Package: domR
2 Type: Package
3 Title: What the Package Does (Title Case)
4 Version: 0.1.0
5 Author: Who wrote it
6 Maintainer: The package maintainer <yourself@somewhere.net>
7 Description: More about what it does (maybe more than one line)
8     Use four spaces when indenting paragraphs within the Description.
9 License: What license is it under?
10 Encoding: UTF-8
11 LazyData: true
```

To something like this

```
1 Package: compete
2 Title: Analyzing Social Hierarchies
3 Type: Package
4 Version: 0.1
5 Author: c(person("James P.", "Curley", role = c("aut", "cre"),
6     email = "jc3181@columbia.edu")
7 Maintainer: James P. Curley <jc3181@columbia.edu>
8 Description: Organizing and Analyzing Social Dominance
9     Hierarchy Data.
10 Depends:
11     R (>= 3.1.0)
12 License: GPL-3
13 LazyData: true
14 Imports:
15     graphics,
16     igraph,
17     sna,
18     stats,
19     utils
20 URL: https://github.com/jalapic/compete
21 BugReports: https://github.com/jalapic/compete
22 RoxygenNote: 5.0.0
23
```

If package uses functions from other packages, include them in the “Imports” section.

It’s better in most cases to put them in “Imports” rather than “Depends”

Add .R functions to R folder

R code

The first principle of using a package is that all R code goes in `R/`. In this chapter, you'll learn about the `R/` directory, my recommendations for organising your functions into files, and some general tips on good style. You'll also learn about some important differences between functions in scripts and functions in packages.

R code workflow

The first practical advantage to using a package is that it's easy to re-load your code. You can either run `devtools::load_all()`, or in RStudio press **Ctrl/Cmd + Shift + L**, which also saves all open files, saving you a keystroke.

This keyboard shortcut leads to a fluid development workflow:

1. Edit an R file.
2. Press Ctrl/Cmd + Shift + L.
3. Explore the code in the console.
4. Rinse and repeat.

Congratulations! You've learned your first package development workflow. Even if you learn nothing else from this book, you'll have gained a useful workflow for editing and reloading R code.

See <http://r-pkgs.had.co.nz/r.html> for advice on code style and best practice

Add roxygen comments to top of .R functions script

The documentation workflow

In this section, we'll first go over a rough outline of the complete documentation workflow. Then, we'll dive into each step individually. There are four basic steps:

1. Add roxygen comments to your .R files.
2. Run `devtools::document()` (or press Ctrl/Cmd + Shift + D in RStudio) to convert roxygen comments to .Rd files. (`devtools::document()` calls `roxygen2::roxygenise()` to do the hard work.)
3. Preview documentation with `?`.
4. Rinse and repeat until the documentation looks the way you want.

The process starts when you add roxygen comments to your source file: roxygen comments start with `#'` to distinguish them from regular comments. Here's documentation for a simple function:

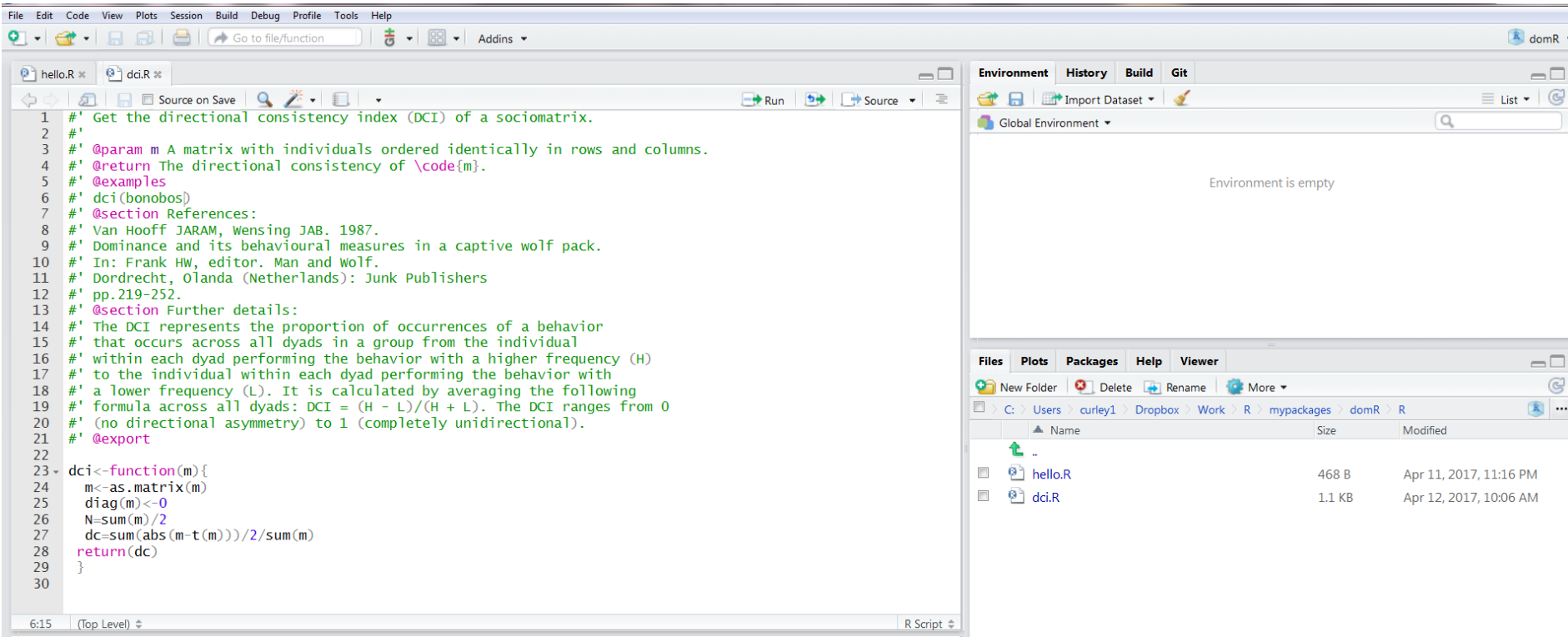
```
#' Add together two numbers.
#'
#' @param x A number.
#' @param y A number.
#' @return The sum of \code{x} and \code{y}.
#' @examples
#' add(1, 1)
#' add(10, 1)
add <- function(x, y) {
  x + y
}
```

See <http://r-pkgs.had.co.nz/man.html> for style guide and best practice

roxygen comments

title	= title of the function.
description	= detailed explanation of what the package does.
param	= parameters/arguments in function. You can use multiple 'param' return = what object the function returns
importFrom	= functions you need from other packages
examples	= sample code (keep it small)
section	= anything else relevant that you might want to add
export	= function to export

Add .R functions to R folder and add roxygen comments



The screenshot displays the RStudio interface. The main editor window shows an R script named `dcf.R` with the following content:

```
1 #' Get the directional consistency index (DCI) of a sociomatrix.
2 #'
3 #' @param m A matrix with individuals ordered identically in rows and columns.
4 #' @return The directional consistency of \code{m}.
5 #' @examples
6 #' dci(bonobos)
7 #' @section References:
8 #' Van Hooff JARAM, Wensing JAB. 1987.
9 #' Dominance and its behavioural measures in a captive wolf pack.
10 #' In: Frank HW, editor. Man and Wolf.
11 #' Dordrecht, Olanda (Netherlands): Junk Publishers
12 #' pp.219-252.
13 #' @section Further details:
14 #' The DCI represents the proportion of occurrences of a behavior
15 #' that occurs across all dyads in a group from the individual
16 #' within each dyad performing the behavior with a higher frequency (H)
17 #' to the individual within each dyad performing the behavior with
18 #' a lower frequency (L). It is calculated by averaging the following
19 #' formula across all dyads:  $DCI = (H - L) / (H + L)$ . The DCI ranges from 0
20 #' (no directional asymmetry) to 1 (completely unidirectional).
21 #' @export
22
23 dci<-function(m){
24   m<-as.matrix(m)
25   diag(m)<-0
26   N=sum(m)/2
27   dc=sum(abs(m-t(m)))/2/sum(m)
28   return(dc)
29 }
30
```

The right-hand pane shows the 'Environment' tab, which is currently empty. Below it, the 'Files' tab displays the directory structure: `C:\Users\curley1\Dropbox\Work\R\mypackages\domR`. The files listed are:

Name	Size	Modified
..		
hello.R	468 B	Apr 11, 2017, 11:16 PM
dcf.R	1.1 KB	Apr 12, 2017, 10:06 AM

Save functions with name of function

Do for next function ...

The screenshot displays the RStudio IDE interface. The main editor window shows an R script with the following content:

```
1 #' Transforms a frequency interaction sociomatrix (valued data) into a dichotomized 1/0 matrix
2 #'
3 #' @param m A matrix with individuals ordered identically in rows and columns.
4 #' @param type Determines the type of dichotomized matrix to be returned.
5 #' \strong{\code{type}="w1"} is the default which returns a win-loss matrix
6 #' with a '1' representing a consistent winner and a '0' representing a
7 #' consistent loser for each dyad of the matrix. A consistent winner is
8 #' defined as being the individual in each dyad that has absolutely more
9 #' wins than defeats. In the default condition if competitors have the
10 #' same number of wins each, they both receive a 0.
11 #' If \strong{\code{type}="wlties"} the default dichotomized win-loss
12 #' matrix will be returned but it will also return 0.5 into cells for tied
13 #' relationships.
14 #' @return A dichotomized win/loss or presence/absence matrix.
15 #' @examples
16 #' get_di_matrix(bonobos)
17 #' @section References:
18 #' Appleby, M. C. 1983. The probability of linearity in hierarchies.
19 #' Animal Behaviour, 31, 600-608.
20 #' @export
21
22 get_di_matrix <- function(m, type="w1"){
23
24
25
26 if(type=="w1") {
27   m <- as.matrix(m)
28   m <- (m > t(m)) + 0
29   return(m)
30 }
31
32 if(type=="wlties") {
33   m <- as.matrix(m)
34   m <- ((m > t(m)) + 0) + ((m == t(m)) + 0)/2
35   return(m)
36 }
37
38
39 }
40
```

The right-hand pane shows the Environment tab, which is currently empty, displaying the message "Environment is empty". Below the Environment tab is the Files tab, showing the file explorer for the project directory. The file list includes:

Name	Size	Modified
hello.R	468 B	Apr 11, 2017, 11:16 PM
dci.R	1.1 KB	Apr 12, 2017, 10:06 AM
get_di_matrix.R	1.3 KB	Apr 12, 2017, 10:08 AM

Keep going...

The screenshot displays the RStudio IDE interface. The main editor window contains an R script with the following code:

```
1 #' Get Triangle Transitivity of Sociomatrix
2 #'
3 #' @param m A frequency binary win-loss matrix
4 #' @return Pt and t.tri
5 #' @examples
6 #' ttri(caribou)
7 #' @importFrom igraph "graph.adjacency"
8 #' @importFrom igraph "triad.census"
9 #' @section Further details:
10 #' Algorithm described in D. Shizuka and D. B. McDonald, 2012,
11 #' A social network perspective on transitivity and linearity
12 #' in dominance hierarchies. Animal Behaviour.
13 #' DOI:10.1016/j.anbehav.2012.01.011
14 #' Code adapted from original code by Dai Shikua - see
15 #' http://www.shizukalab.com/toolkits/sna/triangle-transitivity
16 #' @export
17
18 ttri=function(m){
19   mat <- get_di_matrix(as.matrix(m))
20   diag(mat)=0
21   g = igraph::graph.adjacency(mat, mode="directed", diag=FALSE)
22   tri=igraph::triad.census(g)
23   Ntri=sum(tri*as.vector(c(0,0,0,0,0,0,0,0,1,1,0,1,1,1,1,1)))
24   sumtri = sum(tri*as.vector(c(0,0,0,0,0,0,0,0,1,0,0,1,1,0,5,0.75,0.75)) )
25   Pt=sumtri/Ntri
26   ttri.val = 4*(Pt-0.75)
27   return(list("Pt"=Pt, "ttri"=ttri.val))
28 }
29
```

The Environment pane on the right shows the Global Environment, which is currently empty. The Files pane at the bottom right displays the project structure, showing the following files:

Name	Size	Modified
hello.R	468 B	Apr 11, 2017, 11:16 PM
dci.R	1.1 KB	Apr 12, 2017, 10:06 AM
get_di_matrix.R	1.3 KB	Apr 12, 2017, 10:10 AM
ttri.R	990 B	Apr 12, 2017, 10:10 AM

Load .R files ...

```
> devtools::load_all()
Loading domR
Error in (function (dep_name, dep_ver = NA, dep_compare = NA) :
  Dependency package igraph not available.
> |
```

Fix error !

```
> install.packages("igraph")
Installing package into 'C:/Users/curley1/Dropbox/Work/R/mypackages/domR/packrat/lib/x86_64-w64-mingw32/3.3.0'
(as 'lib' is unspecified)
also installing the dependencies 'dichromat', 'munsell', 'labeling', 'xtable', 'iterators', 'gtable', 'plyr', 'scales', 'tibble', 'lazyeval', 'pkgmaker', 'registry', 'rngtools', 'gridBase', 'colorspace', 'RColorBrewer', 'foreach', 'doParallel', 'ggplot2', 'reshape2', 'NMF', 'irlba'

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.3/dichromat_2.0-0.zip'
Content type 'application/zip' length 147728 bytes (144 KB)
|
```

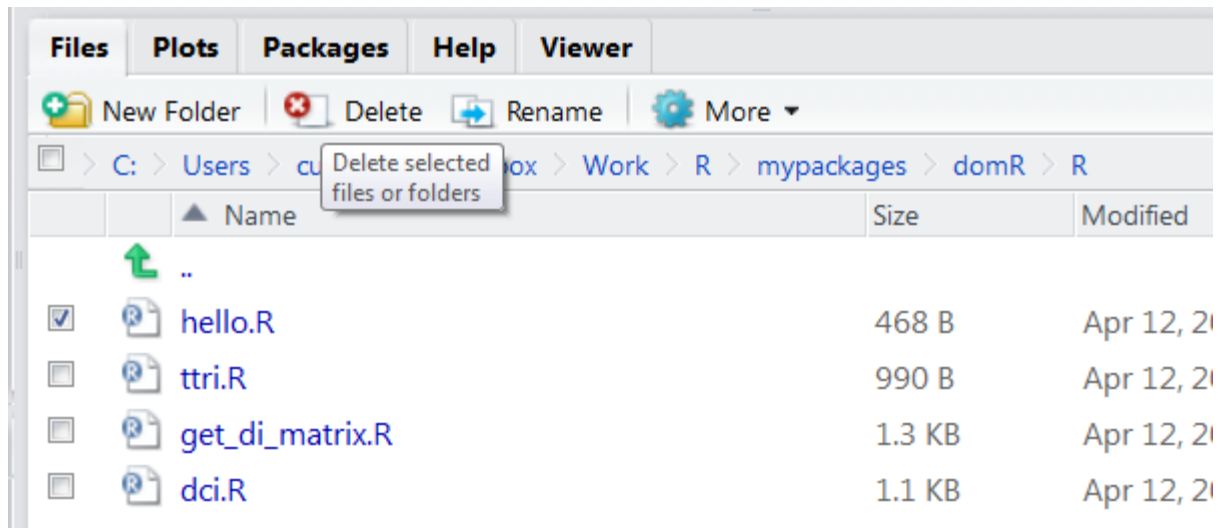
The downloaded binary packages are in
C:\Users\curley1\AppData\Local\Temp\RtmpqUptze\downloaded_packages

```
> devtools::load_all()
Loading domR
> |
```

We're good

Deleting files

If we don't want a file – remove it's .R file and its .Rd file (if it exists).
Easiest way is with the Delete icon in Files window



Add documentation Man

devtools::document()

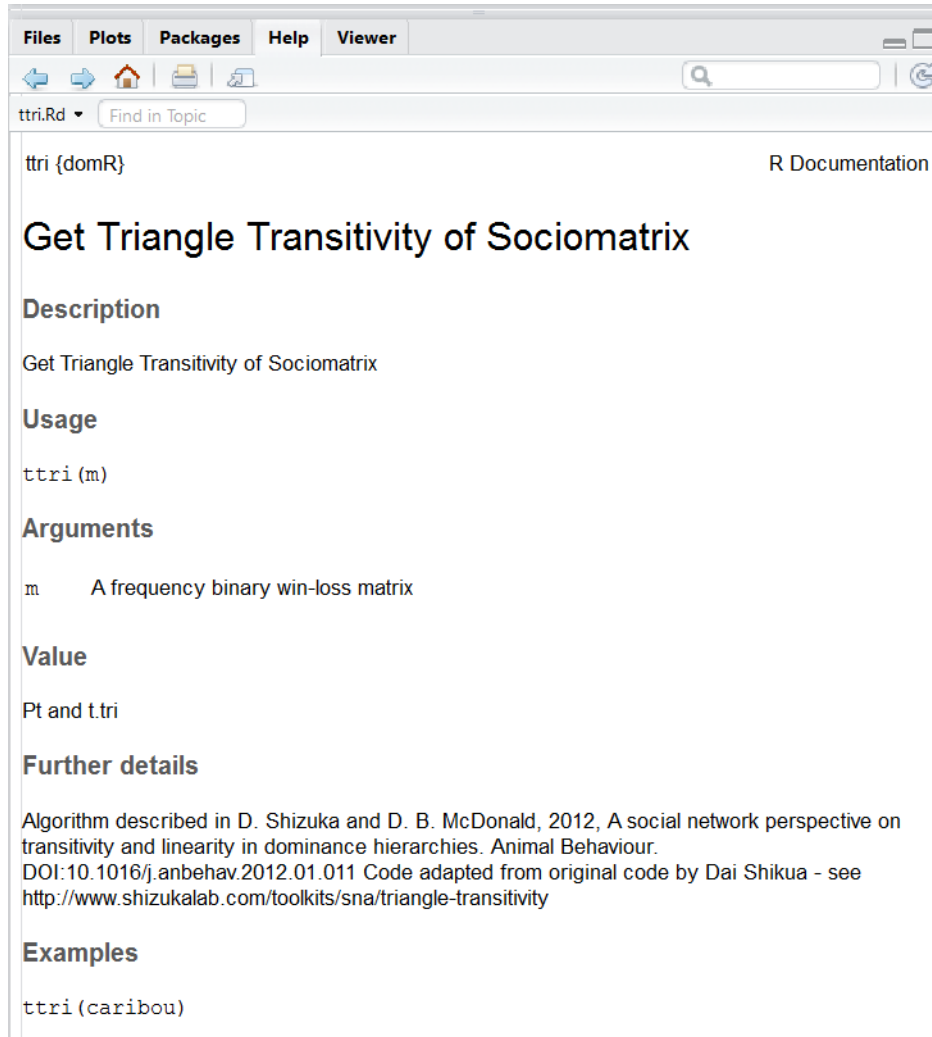
```
> devtools::document()  
Updating domR documentation  
Loading domR  
First time using roxygen2. Upgrading automatically...  
Warning: The existing 'NAMESPACE' file was not generated by roxygen2, and will not be overwritten.  
Writing dci.Rd  
Writing get_di_matrix.Rd  
Writing ttri.Rd  
>
```

	Name	Size	Modified
↑	..		
📄	dci.Rd	1.2 KB	Apr 12, 2017, 11:41 AM
📄	get_di_matrix.Rd	1.4 KB	Apr 12, 2017, 11:41 AM
📄	ttri.Rd	715 B	Apr 12, 2017, 11:41 AM

We now have docs !

Check them out with “?”

```
> ?ttri  
Using development documentation for ttri  
> |
```



The screenshot shows an RStudio window with the 'Viewer' tab active. The title bar indicates the file is 'ttri.Rd'. The main content area displays the R documentation for the 'ttri' function from the 'domR' package. The documentation includes sections for Description, Usage, Arguments, Value, and Further details. The 'Usage' section shows the function signature 'ttri(m)'. The 'Arguments' section describes 'm' as a frequency binary win-loss matrix. The 'Value' section indicates the function returns 'Pt and t.tri'. The 'Further details' section provides a reference to a paper by Shizuka and McDonald (2012) and a URL to the Shizuka Lab toolkit. The 'Examples' section shows the function being applied to a 'caribou' dataset.

ttri {domR} R Documentation

Get Triangle Transitivity of Sociomatrix

Description

Get Triangle Transitivity of Sociomatrix

Usage

```
ttri(m)
```

Arguments

m A frequency binary win-loss matrix

Value

Pt and t.tri

Further details

Algorithm described in D. Shizuka and D. B. McDonald, 2012, A social network perspective on transitivity and linearity in dominance hierarchies. Animal Behaviour. DOI:10.1016/j.anbehav.2012.01.011 Code adapted from original code by Dai Shikua - see <http://www.shizukalab.com/toolkits/sna/triangle-transitivity>

Examples

```
ttri(caribou)
```

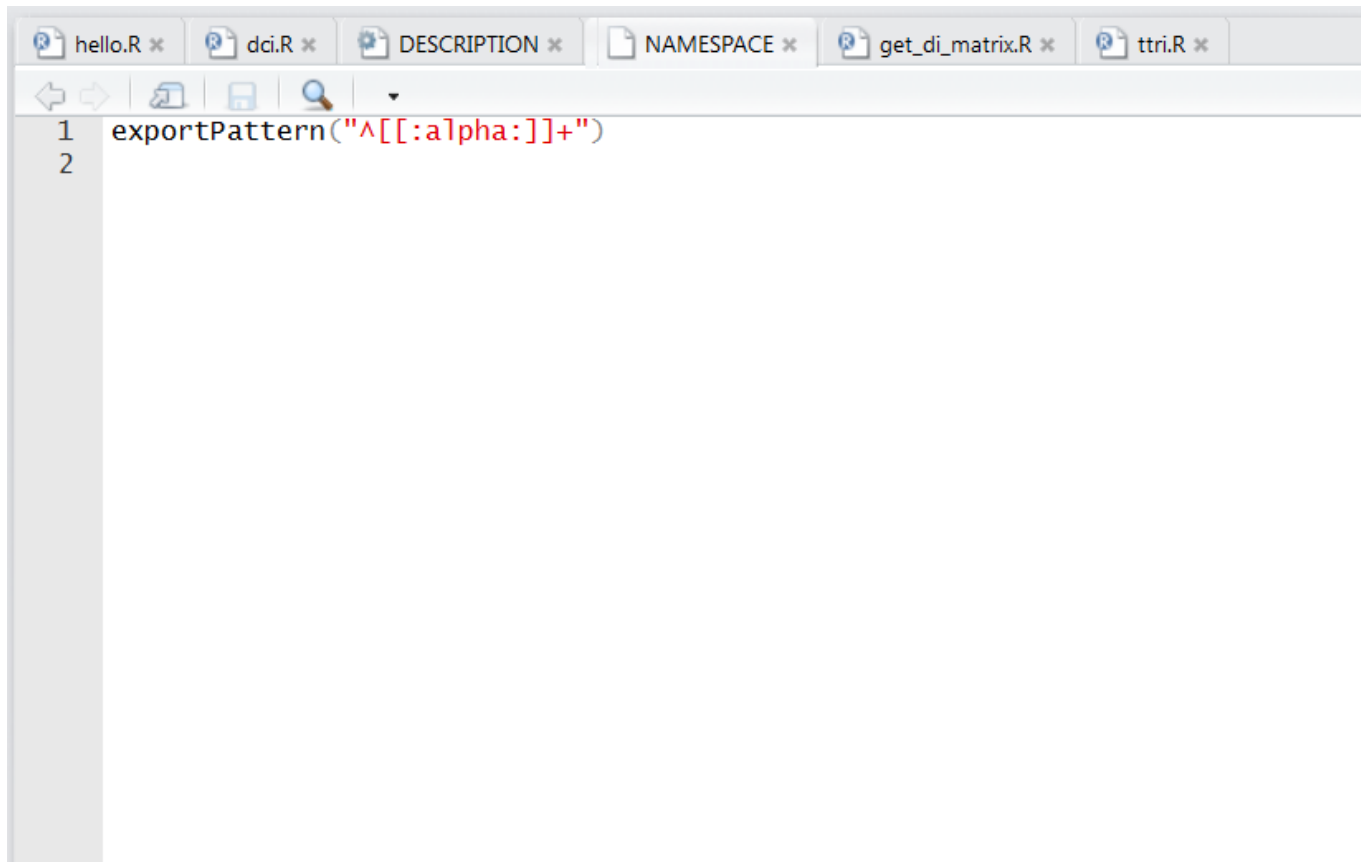



Warning: The existing 'NAMESPACE' file was not generated by roxygen2, and will not be overwritten.

What does this mean ???

NAMESPACE

This is what the NAMESPACE looked like on initial package set up:



The screenshot shows an RStudio editor window with the 'NAMESPACE' file open. The title bar at the top lists several files: 'hello.R', 'dci.R', 'DESCRIPTION', 'NAMESPACE', 'get_di_matrix.R', and 'ttri.R'. Below the title bar is a toolbar with icons for navigation and editing. The main editor area displays the following R code:

```
1 exportPattern("^[:alpha:]+")  
2
```

NAMESPACE

Google it:

Warning: The existing 'NAMESPACE' file was not generated by roxygen2, and will not be overwritten.

[Yihui Xie-2](#)



378 posts

Feb 06, 2017; 11:13am **Re: roxygen2 v6.0.0**

If your package source is version controlled (meaning you are free to regret any time), I'd recommend you to delete the three files NAMESPACE, chr.Rd, and essai-package.Rd. Then try to roxygenize again. Basically the warnings you saw indicates that roxygen2 failed to find the line

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

in your NAMESPACE and .Rd files, so it thinks these files were probably not previously generated by roxygen2. I think the cause is `package.skeleton()`, which generated the Rd files. Seriously, friends don't let friends use `package.skeleton()`... (it is 2017 now)

Regards,
Yihui

--

Yihui Xie <[\[hidden email\]](#)>

Web: <http://yihui.name>

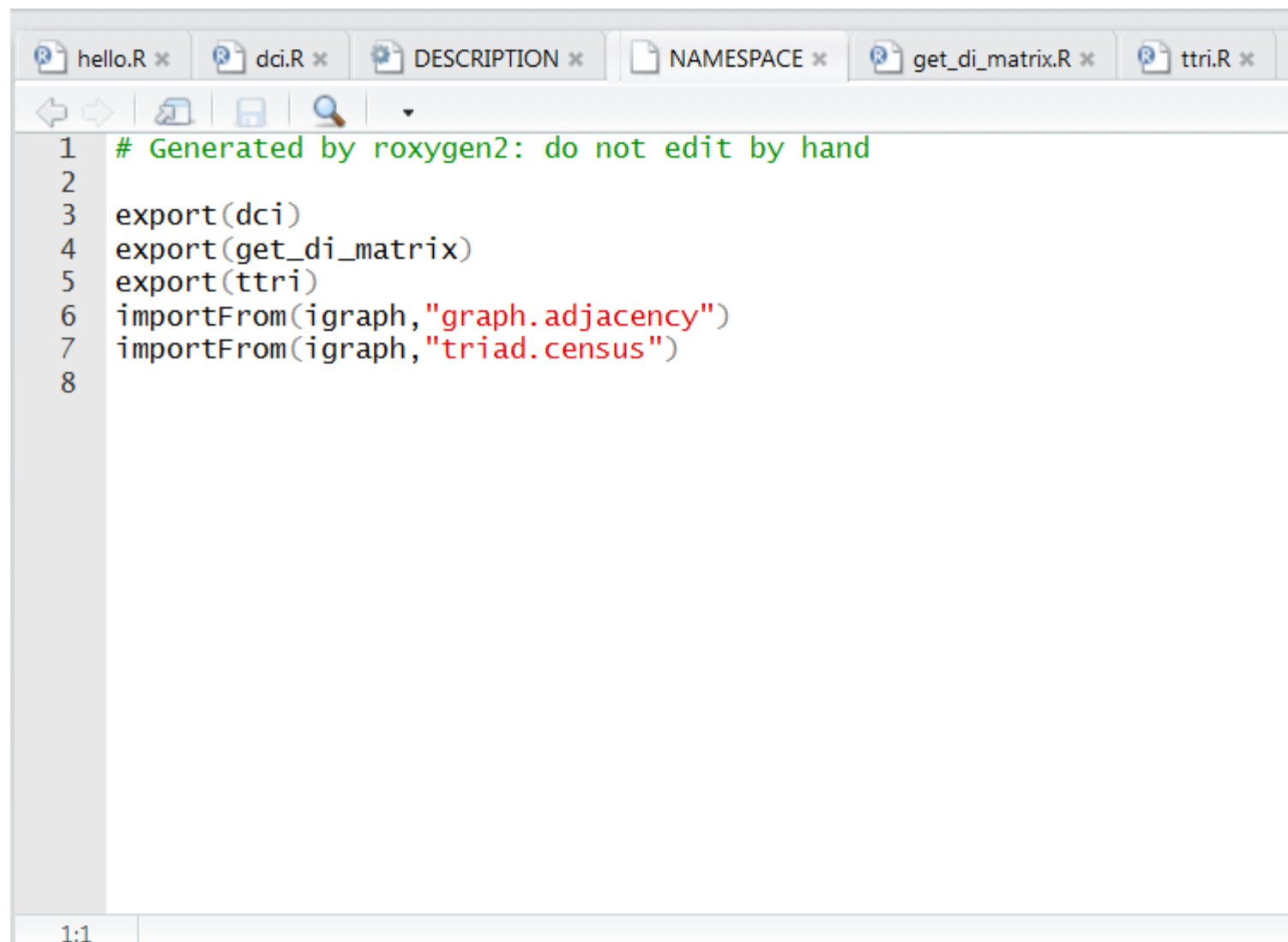
Do as Yihui says ...

Delete .Rd files we just made, Delete the NAMESPACE, problem fixed !!!

```
> devtools::document()  
Updating domR documentation  
Loading domR  
First time using roxygen2. Upgrading automatically...  
Writing NAMESPACE  
Writing dci.Rd  
Writing get_di_matrix.Rd  
Writing ttri.Rd  
> |
```



That's better – it should have the “do not edit by hand”



```
1 # Generated by roxygen2: do not edit by hand
2
3 export(dci)
4 export(get_di_matrix)
5 export(ttri)
6 importFrom(igraph, "graph.adjacency")
7 importFrom(igraph, "triad.census")
8
```

1:1

For every new .R file you will need to run:

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

To generate your documentation.

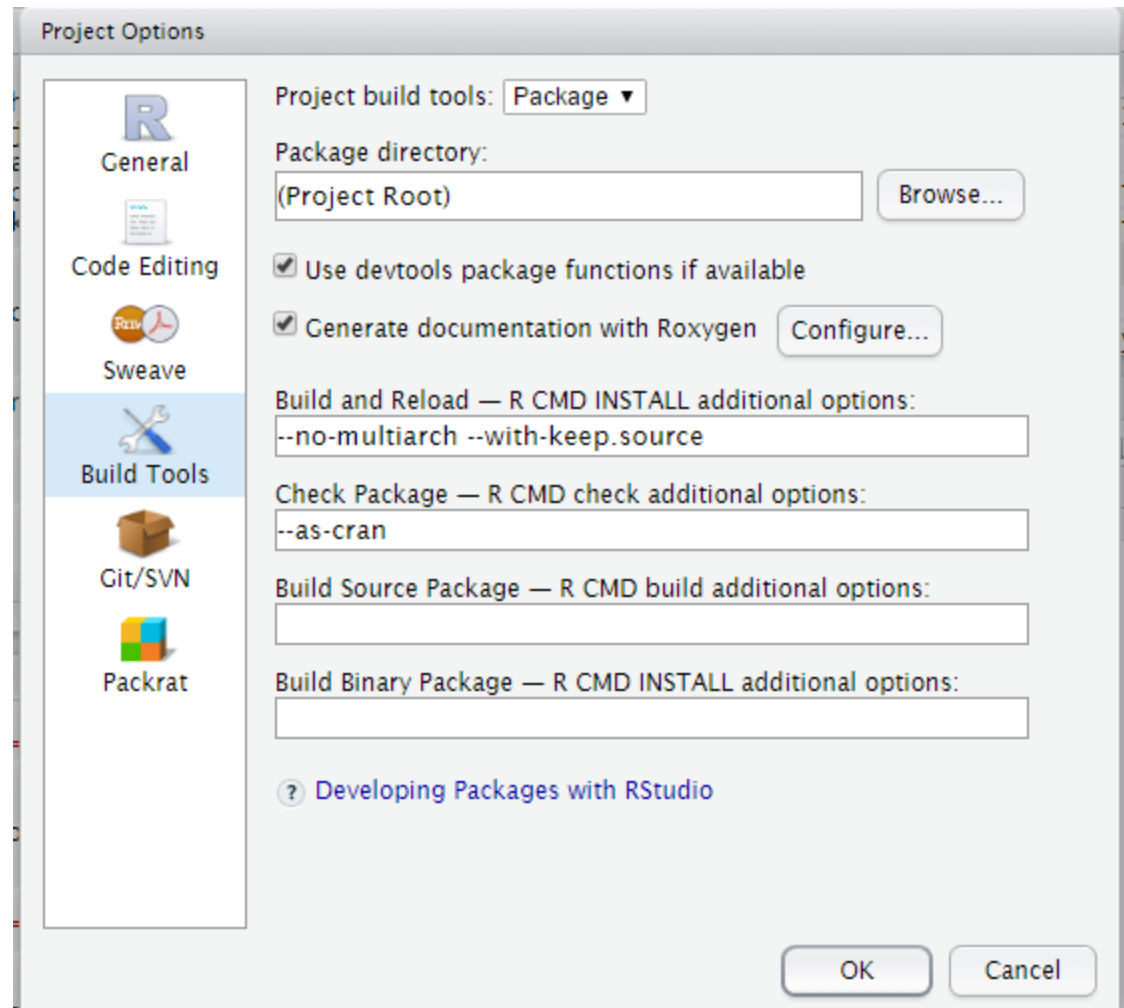
You will also need to run it if you change the information in any .R file

- This often happens after you run CHECKS and realize you made some errors in the text.

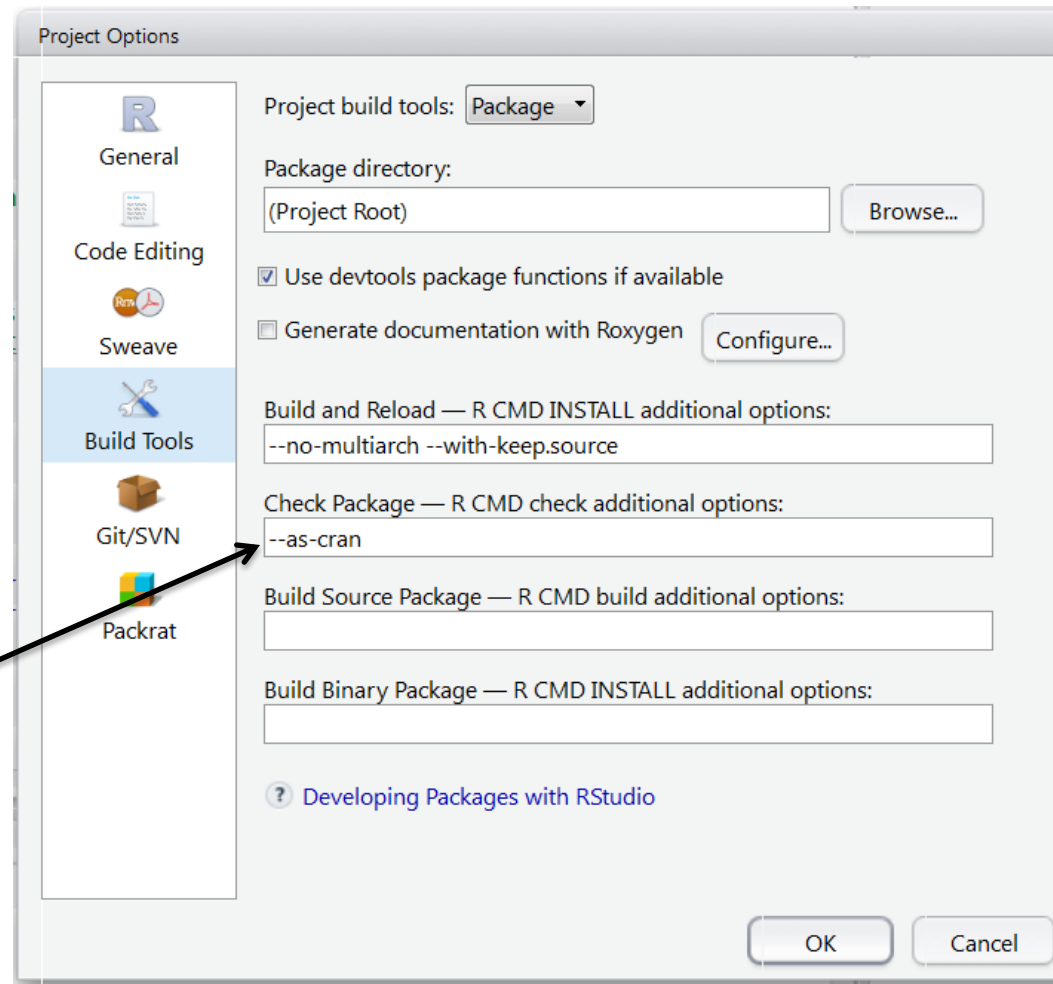
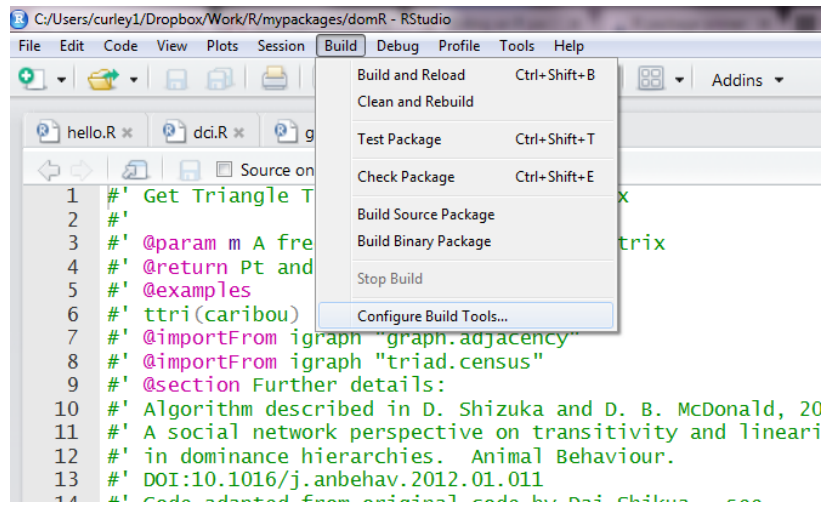


Build > Configure Build Tools.

Now check the “Generate documentation with Roxygen” option and put “--as-cran” under Check Package space to simulate the CRAN package checking and testing.



Configure Build Tools ... (this is for CRAN checks so not entirely necessary for all)



Add: --as-cran

Build & Reload

The screenshot shows the RStudio interface with the 'Build & Reload' button highlighted in the top right. The main editor displays the source code for the 'domR' package, which includes comments about the Directional Consistency Index (DCI) and a function definition for 'dci'. The console shows the output of the 'library(domR)' command, indicating that the package is being installed and reloaded. The file explorer on the right shows the directory structure of the package, including files like 'hello.R', 'ttri.R', 'get_di_matrix.R', and 'dci.R'.

```
1 #' Get the directional consistency index (DCI) of a sociomatrix.
2 #'
3 #' @param m A matrix with individuals ordered identically in rows and columns.
4 #' @return The directional consistency of \code{m}.
5 #' @examples
6 #' dci(bonobos)
7 #' @section References:
8 #' Van Hooff JARAM, Wensing JAB. 1987.
9 #' Dominance and its behavioural measures in a captive wolf pack.
10 #' In: Frank HW, editor. Man and Wolf.
11 #' Dordrecht, Olanda (Netherlands): Junk Publishers
12 #' pp.219-252.
13 #' @section Further details:
14 #' The DCI represents the proportion of occurrences of a behavior
15 #' that occurs across all dyads in a group from the individual
16 #' within each dyad performing the behavior with a higher frequency (H)
17 #' to the individual within each dyad performing the behavior with
18 #' a lower frequency (L). It is calculated by averaging the following
19 #' formula across all dyads:  $DCI = (H - L) / (H + L)$ . The DCI ranges from 0
20 #' (no directional asymmetry) to 1 (completely unidirectional).
21 #' @export
22
23 dci<-function(m){
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Console output:

```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

Restarting R session...

> library(domR)
>
```

Build & Reload output:

```
==> Rcmd.exe INSTALL --no-multiarch --with-keep.source domR
* installing to library 'C:/Users/curley1/Dropbox/work/R/mypackages/domR/packrat/lib/x86_64-w64-mingw32/3.3.0'
* installing *source* package 'domR' ...
** R
** preparing package for lazy loading
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded
* DONE (domR)
```

Name	Size	Modified
..		
hello.R	468 B	Apr 12, 2017, 10:33 AM
ttri.R	990 B	Apr 12, 2017, 10:34 AM
get_di_matrix.R	1.3 KB	Apr 12, 2017, 10:35 AM
dci.R	1.1 KB	Apr 12, 2017, 10:35 AM

You should be able to start using your functions now like a regular package:

Restarting R session...

```
> library(domR)
> m <- matrix(c(NA,2,30,6,19,122,0,NA,18,
+               0,19,85,0,1,NA,3,8,84,0,0,0,NA,267,50,0,
+               0,0,5,NA,10,1,0,4,4,1,NA), ncol=6)
>
> m
      [,1] [,2] [,3] [,4] [,5] [,6]
[1,]   NA    0    0    0    0    1
[2,]    2   NA    1    0    0    0
[3,]   30   18   NA    0    0    4
[4,]    6    0    3   NA    5    4
[5,]   19   19    8  267   NA    1
[6,]  122   85   84   50   10   NA
> ttri(m)
$Pt
[1] 1

$ttri
[1] 1

> |
```



Often you will want to add data to the package:

- for examples
- for users
- for functions

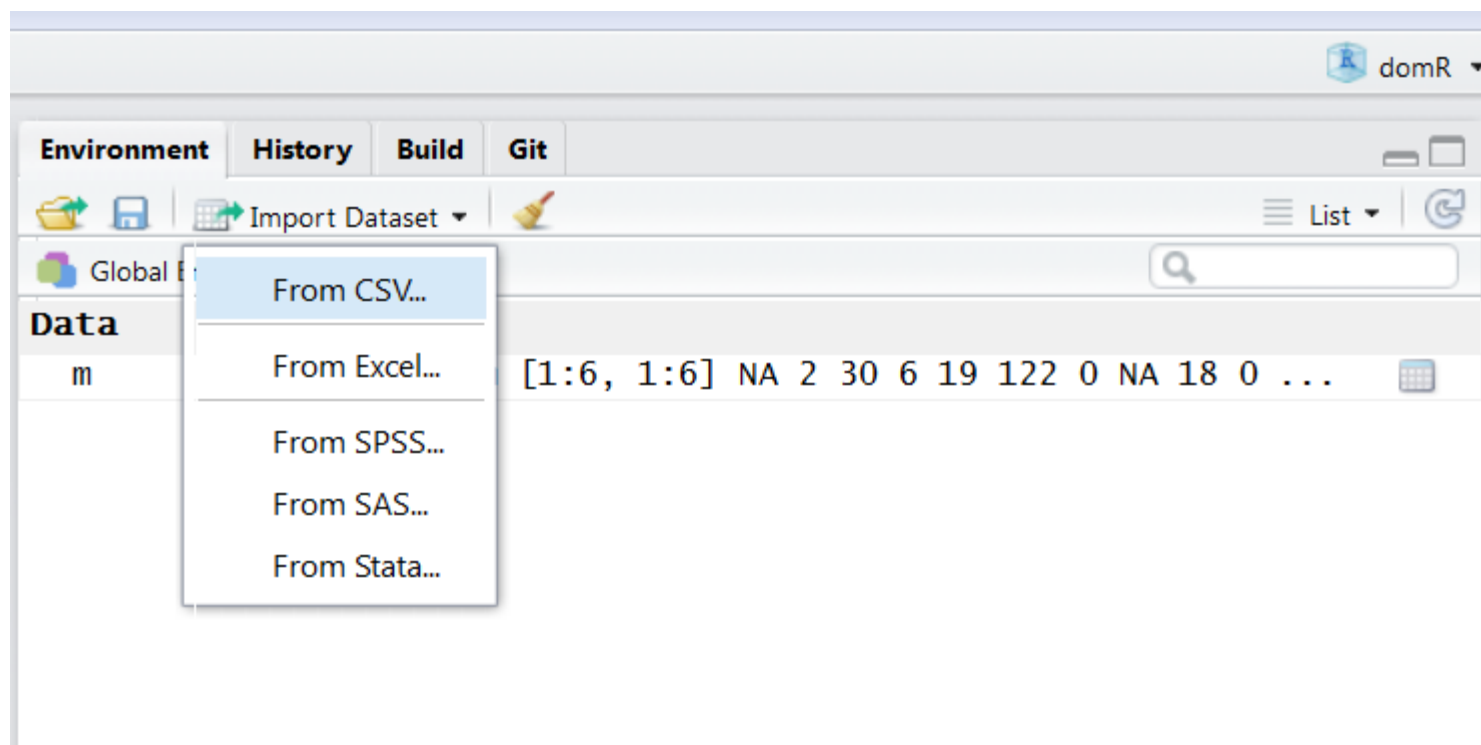


It's worth checking that the working directory is your project at this point:

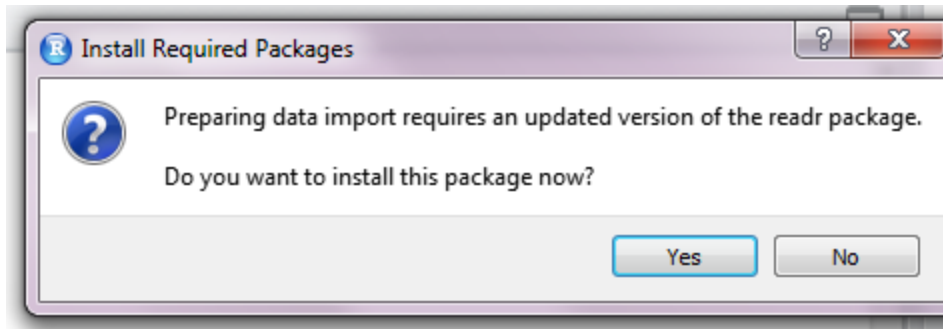
```
> setwd("C:/Users/curley1/Dropbox/Work/R/mypackages/domR")  
> getwd()  
[1] "C:/Users/curley1/Dropbox/Work/R/mypackages/domR"  
>
```

Here's how to make a data file available to users of your package

Example: Load “caribou.csv” into Global Environment



If you're following this tutorial and using "packrat", and you import using the Rstudio dropdown menu you will get this prompt: Hit "YES"



You could also import your data file in any other way to your working directory that you are comfortable with.

Importing ...

Import Text Data

File/Url:
C:/Users/curley1/Dropbox/Work/DataVisualizationTalks/Rpackage/caribou.csv
Browse...

Data Preview:

a (integer)	b (integer)	c (integer)	d (integer)	e (integer)	f (integer)	g (integer)	h (integer)	i (integer)	j (integer)	k (integer)	l (integer)	m (integer)	n (integer)	o (integer)	p (integer)	q (integer)	r (integer)	s (integer)
0	11	3	8	4	2	4	5	3	10	3	5	4	7	5	2	4	0	
0	0	4	11	6	2	5	6	1	5	4	4	4	4	3	7	4	2	
0	1	0	0	0	16	10	7	8	7	3	5	6	7	5	6	4	3	
2	0	8	0	6	9	4	5	3	6	15	9	7	5	9	8	5	2	
4	1	3	2	0	5	2	2	5	13	22	2	4	6	2	2	5	2	
0	0	0	0	1	0	5	1	6	3	8	9	4	2	3	5	8	8	
0	0	0	0	0	0	0	1	3	5	2	9	3	7	4	5	2	2	
0	0	0	0	3	1	2	0	3	2	4	1	1	2	5	2	1	1	
0	0	0	0	0	0	0	0	0	17	0	0	4	0	0	1	0	4	
0	0	0	0	0	0	0	0	0	0	16	6	5	4	0	3	2	0	
0	1	0	0	2	0	0	0	6	1	0	8	4	3	0	4	4	5	

Previewing first 50 entries.

Import Options:

Name: caribou

Skip: 0

☒ First Row as Names
☒ Trim Spaces
☒ Open Data Viewer

Delimiter: Comma
Quotes: Default
Locale: Configure...

Escape: None
Comment: Default
NA: Default

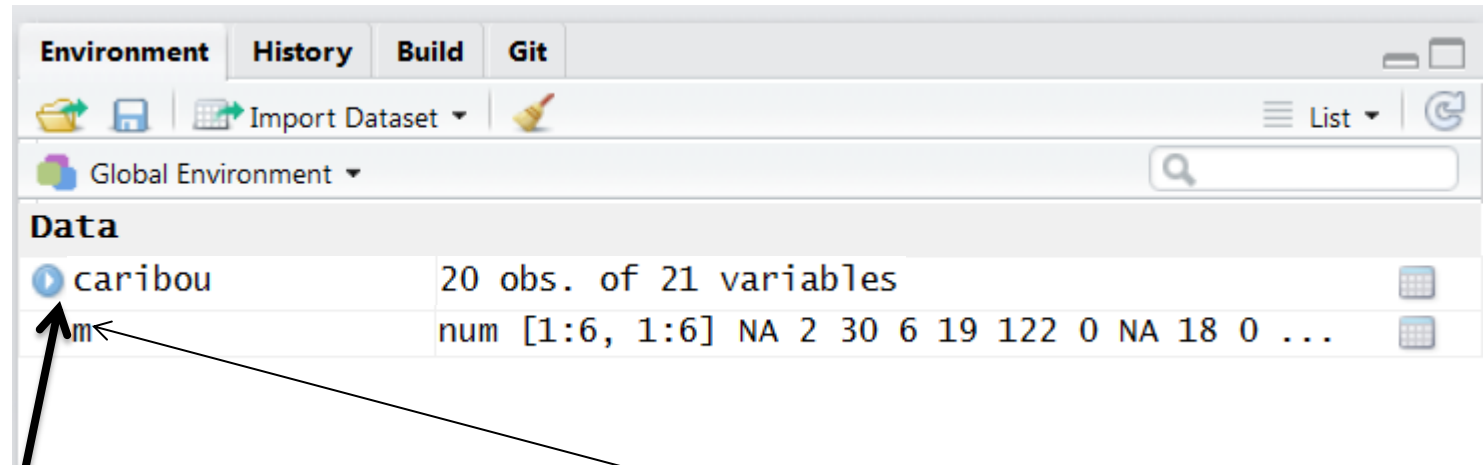
Code Preview:

```
library(readr)  
caribou <- read_csv("C:/Users/curley1/Dropbox/Work/DataVisualizationTalks/Rpackage/caribou.csv")  
View(caribou)
```

Import

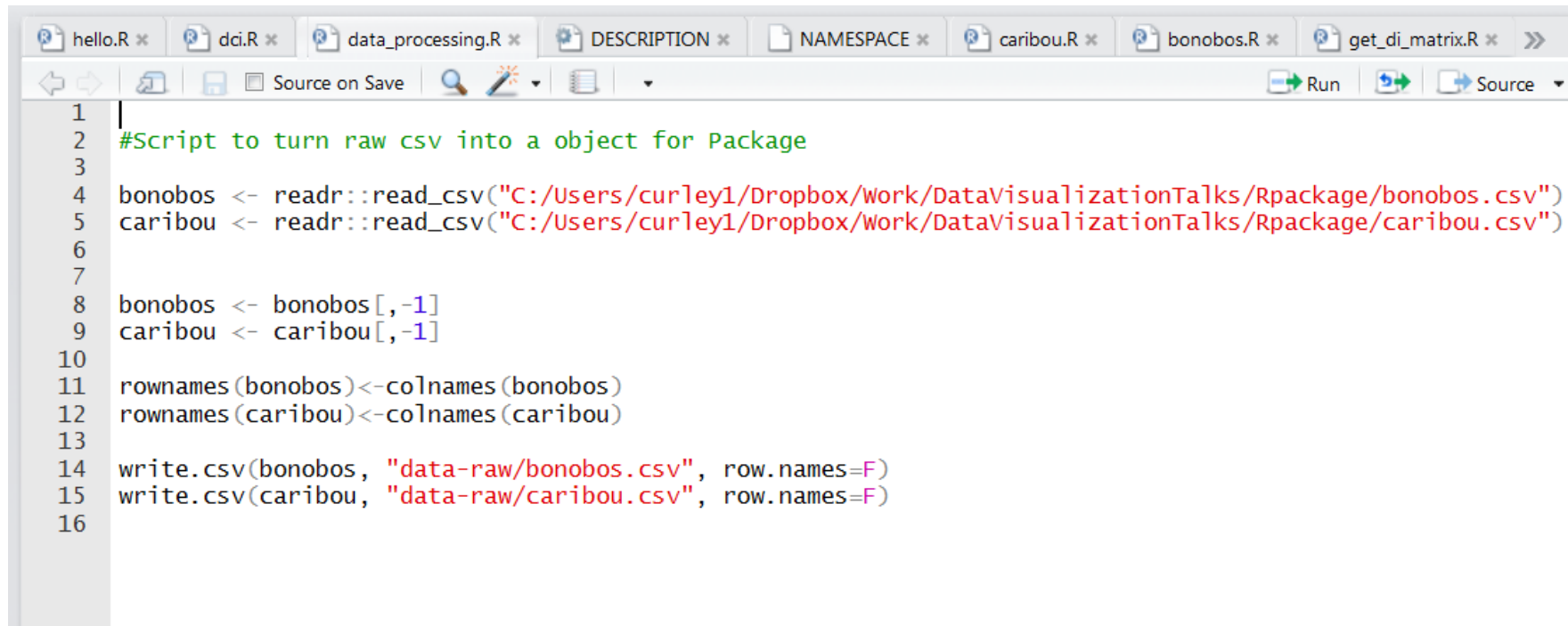
Cancel

It should be in your Global Environment ...



The "m" is the object we used to test our function earlier

The data object should have the name that you want it to be called in your package.



The image shows a screenshot of the RStudio IDE. The top pane displays several open files: 'hello.R', 'dci.R', 'data_processing.R', 'DESCRIPTION', 'NAMESPACE', 'caribou.R', 'bonobos.R', and 'get_di_matrix.R'. The bottom pane shows the source code for 'data_processing.R'. The code is as follows:

```
1 |
2 #Script to turn raw csv into a object for Package
3
4 bonobos <- readr::read_csv("C:/Users/curley1/Dropbox/Work/DataVisualizationTalks/Rpackage/bonobos.csv")
5 caribou <- readr::read_csv("C:/Users/curley1/Dropbox/Work/DataVisualizationTalks/Rpackage/caribou.csv")
6
7
8 bonobos <- bonobos[, -1]
9 caribou <- caribou[, -1]
10
11 rownames(bonobos) <- colnames(bonobos)
12 rownames(caribou) <- colnames(caribou)
13
14 write.csv(bonobos, "data-raw/bonobos.csv", row.names=F)
15 write.csv(caribou, "data-raw/caribou.csv", row.names=F)
16
```

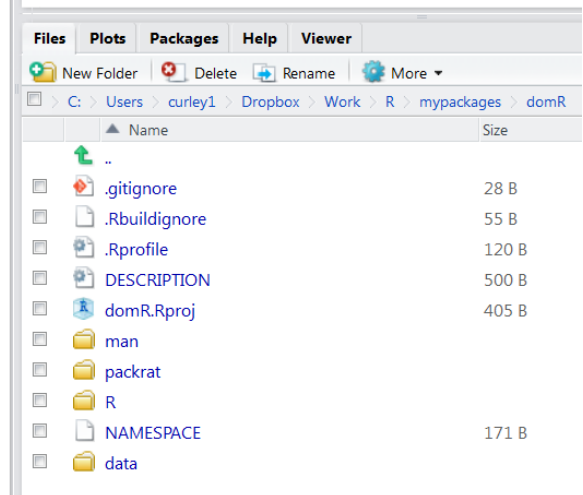
Creating our first data object

devtools::use_data()

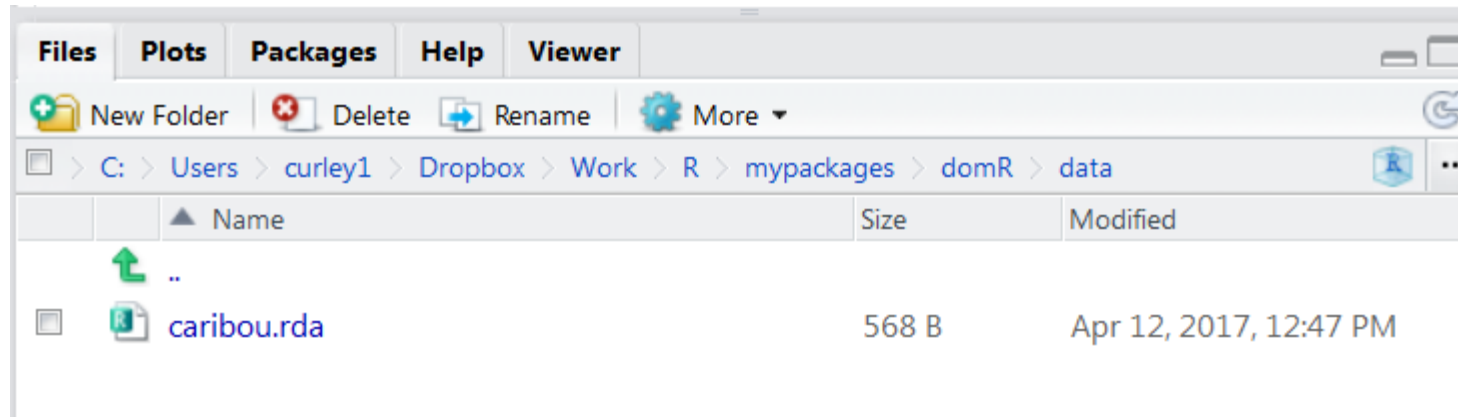
1. Will create a “DATA” folder
2. Will save the object as a “.rda” file in the DATA folder
3. Use “overwrite=T” to replace an existing file of same name (or just add file if no file of that name)

devtools::use_data(caribou,overwrite=T)

```
Console C:/Users/curley1/Dropbox/Work/R/mypackages/domR/
> devtools::use_data(caribou,overwrite=T)
Saving caribou as caribou.rda to C:\Users\curley1\Dropbox\Work\R\mypackages\domR\data
>
> |
```



The file should now be in the “DATA” folder:

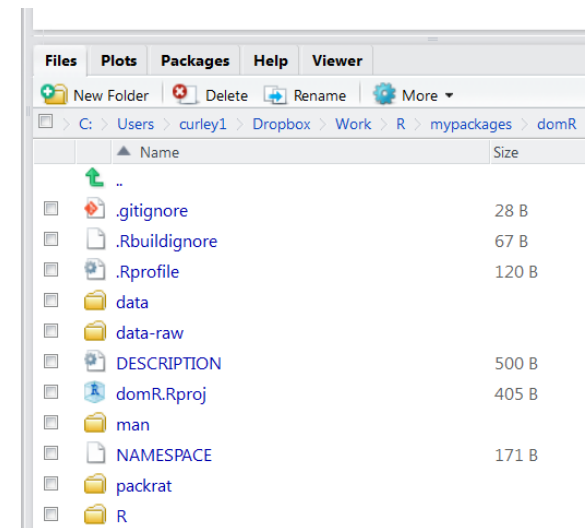


What about raw data ?

devtools::use_data-raw()

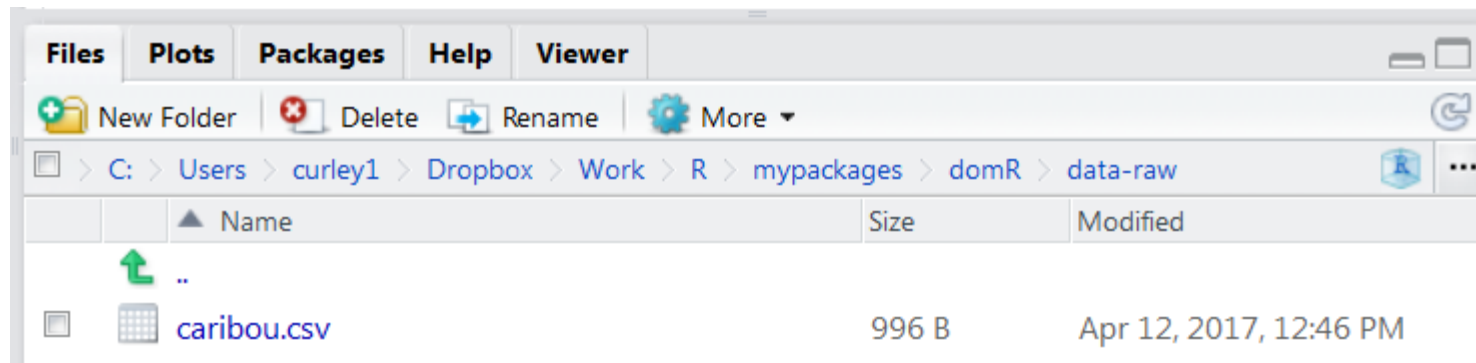
- best to keep this and any .R files used in processing data
- to enable anybody to go from raw data to final outcome, should be able to determine how data was processed

```
> devtools::use_data(caribou,overwrite=T)
Saving caribou as caribou.rda to C:\Users\curley1\Dropbox\Work\R\mypackages\domR\data
>
> devtools::use_data_raw()
* Creating `data-raw`.
* Adding `data-raw` to `.Rbuildignore`.
Next:
* Add data creation scripts in data-raw
* Use devtools::use_data() to add data to package
>
> |
```



Files that you want to save into “data-raw” can be saved in any format:

```
>  
> write.csv(caribou, "data-raw/caribou.csv", row.names=F)  
> |
```



Can do for multiple datasets

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code to load two CSV files and save them as RDS files.

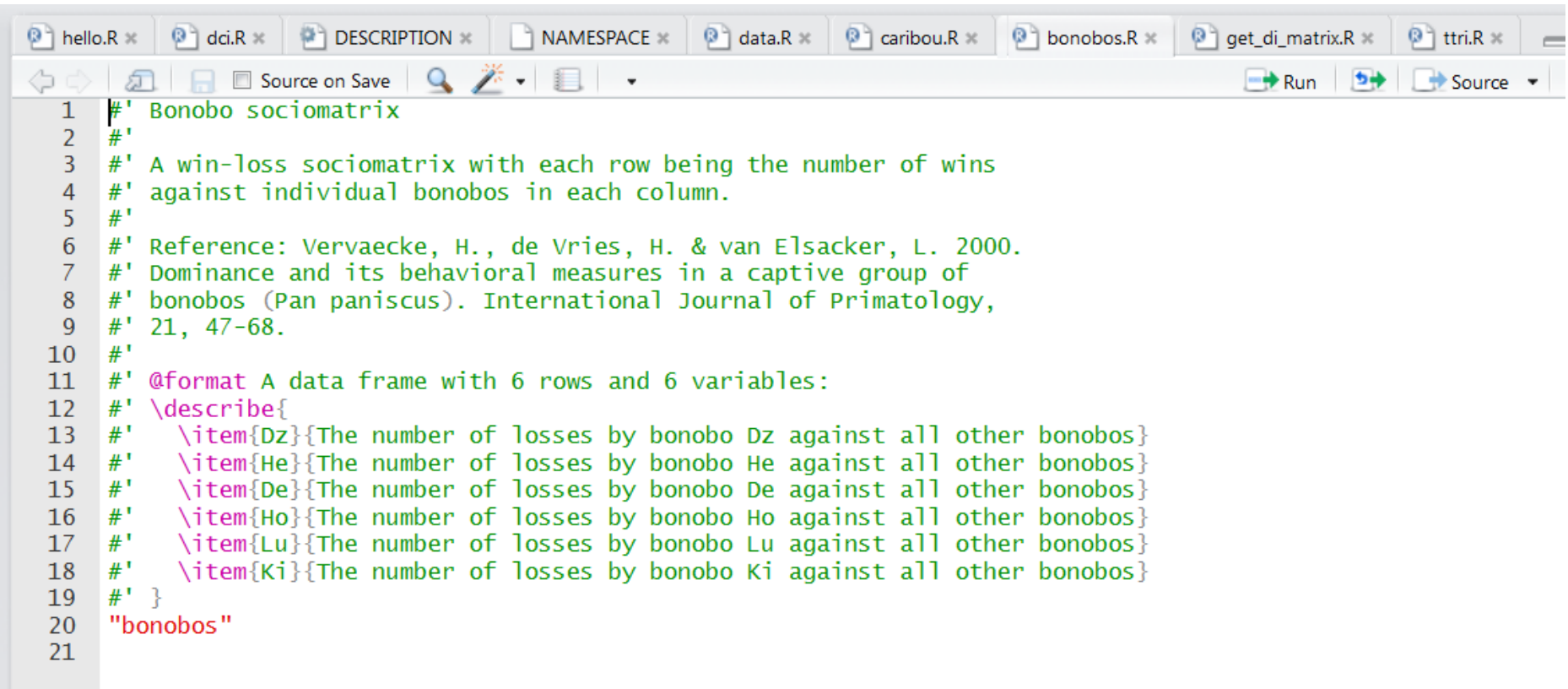
```
1 bonobos <- read.csv("data-row/bonobos.csv",  
2 stringsAsFactors = FALSE)  
3 caribou <- read.csv("data-row/caribou.csv",  
4 stringsAsFactors = FALSE)  
5 devtools::use_data(bonobos, caribou, overwrite = TRUE)  
6
```
- Console:** Shows the execution of the code, including the message "Saving bonobos, caribou as bonobos.rda, caribou.rda to C:\Users\curley1\Dropbox\Work\R\mypackages\domR\data".

```
> bonobos <- read.csv("data-row/bonobos.csv",  
+ stringsAsFactors = FALSE)  
+ caribou <- read.csv("data-row/caribou.csv",  
+ stringsAsFactors = FALSE)  
+ devtools::use_data(bonobos, caribou, overwrite = TRUE)  
Saving bonobos, caribou as bonobos.rda, caribou.rda to C:\Users\curley1\Dropbox\Work\R\mypackages\domR\data  
> |
```
- Environment:** Lists the loaded datasets: bonobos (6 obs. of 7 variables) and caribou (20 obs. of 21 variables).
- Files Panel:** Shows the file explorer for the project, displaying the newly created `caribou.rda` (416 B) and `bonobos.rda` (232 B) files in the `data` folder.

Here, using a file called “data.R” that I will save in “data-row” that contains the code to load files and save as “.rda” files in “data” folder.

I also added “data_processing.R” into “data-row” folder

Add .R files to R folder for datasets

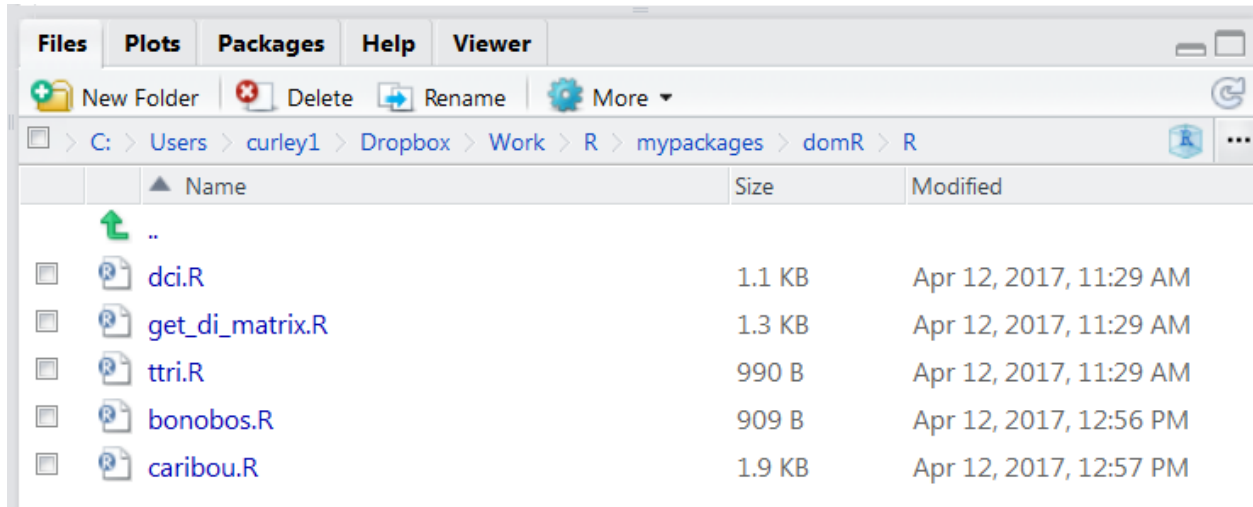


```
1 #' Bonobo sociomatrix
2 #'
3 #' A win-loss sociomatrix with each row being the number of wins
4 #' against individual bonobos in each column.
5 #'
6 #' Reference: Vervaecke, H., de Vries, H. & van Elsacker, L. 2000.
7 #' Dominance and its behavioral measures in a captive group of
8 #' bonobos (Pan paniscus). International Journal of Primatology,
9 #' 21, 47-68.
10 #'
11 #' @format A data frame with 6 rows and 6 variables:
12 #' \describe{
13 #'   \item{Dz}{The number of losses by bonobo Dz against all other bonobos}
14 #'   \item{He}{The number of losses by bonobo He against all other bonobos}
15 #'   \item{De}{The number of losses by bonobo De against all other bonobos}
16 #'   \item{Ho}{The number of losses by bonobo Ho against all other bonobos}
17 #'   \item{Lu}{The number of losses by bonobo Lu against all other bonobos}
18 #'   \item{Ki}{The number of losses by bonobo Ki against all other bonobos}
19 #' }
20 "bonobos"
21
```

Remember:

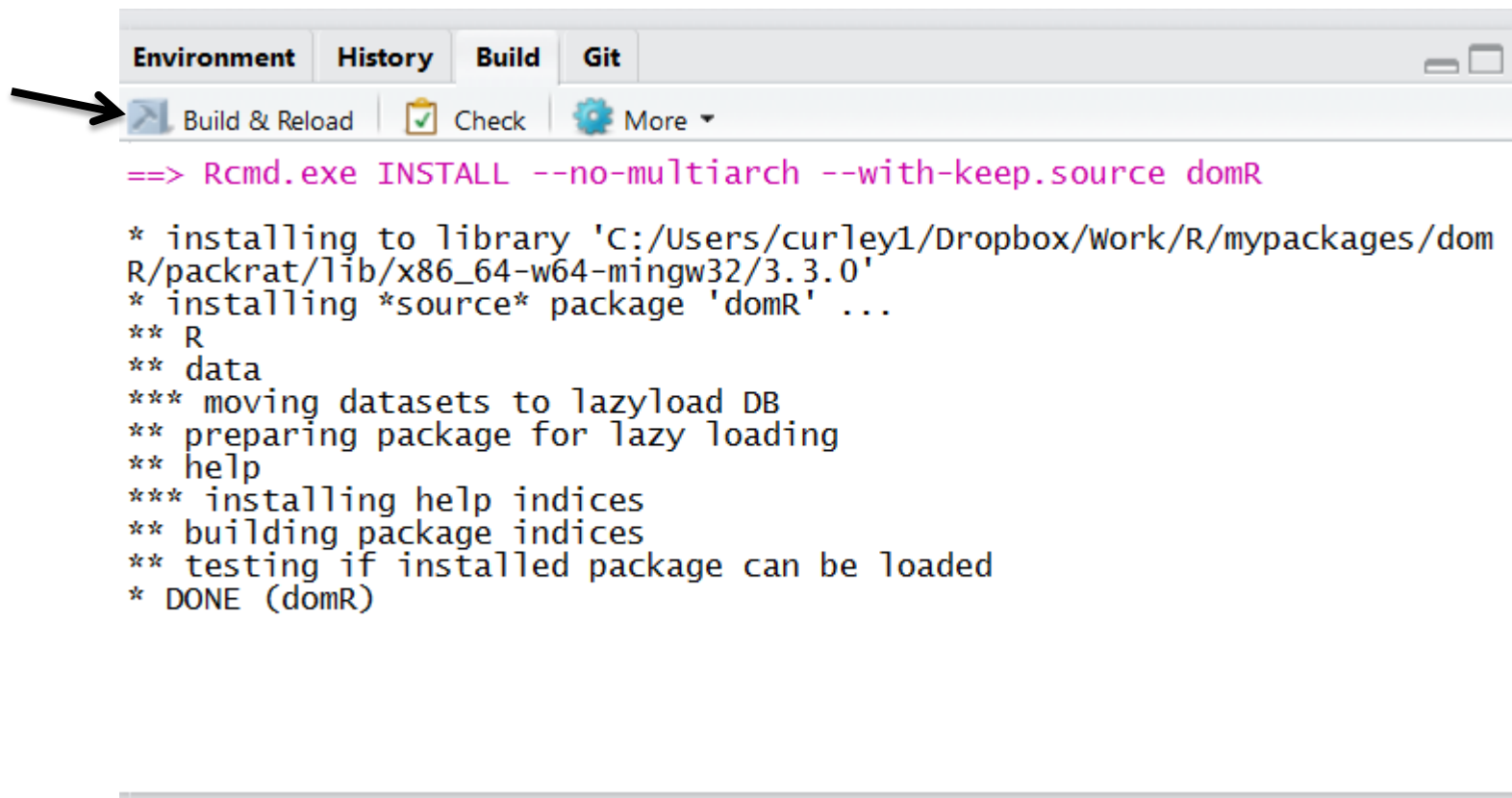
```
devtools::document()
```

Add .R files to R folder for datasets



These .R files are needed when you run Checks.

It's always ok to rebuild



The screenshot shows the RStudio interface with the 'Build' tab selected. A black arrow points to the 'Build & Reload' button, which is represented by a blue icon of a hammer and a wrench. The 'Check' button, represented by a green checkmark icon, is also visible. The 'More' button, represented by a blue gear icon, is partially visible. The console output shows the command `Rcmd.exe INSTALL --no-multiarch --with-keep.source domR` and the subsequent installation steps for the 'domR' package.

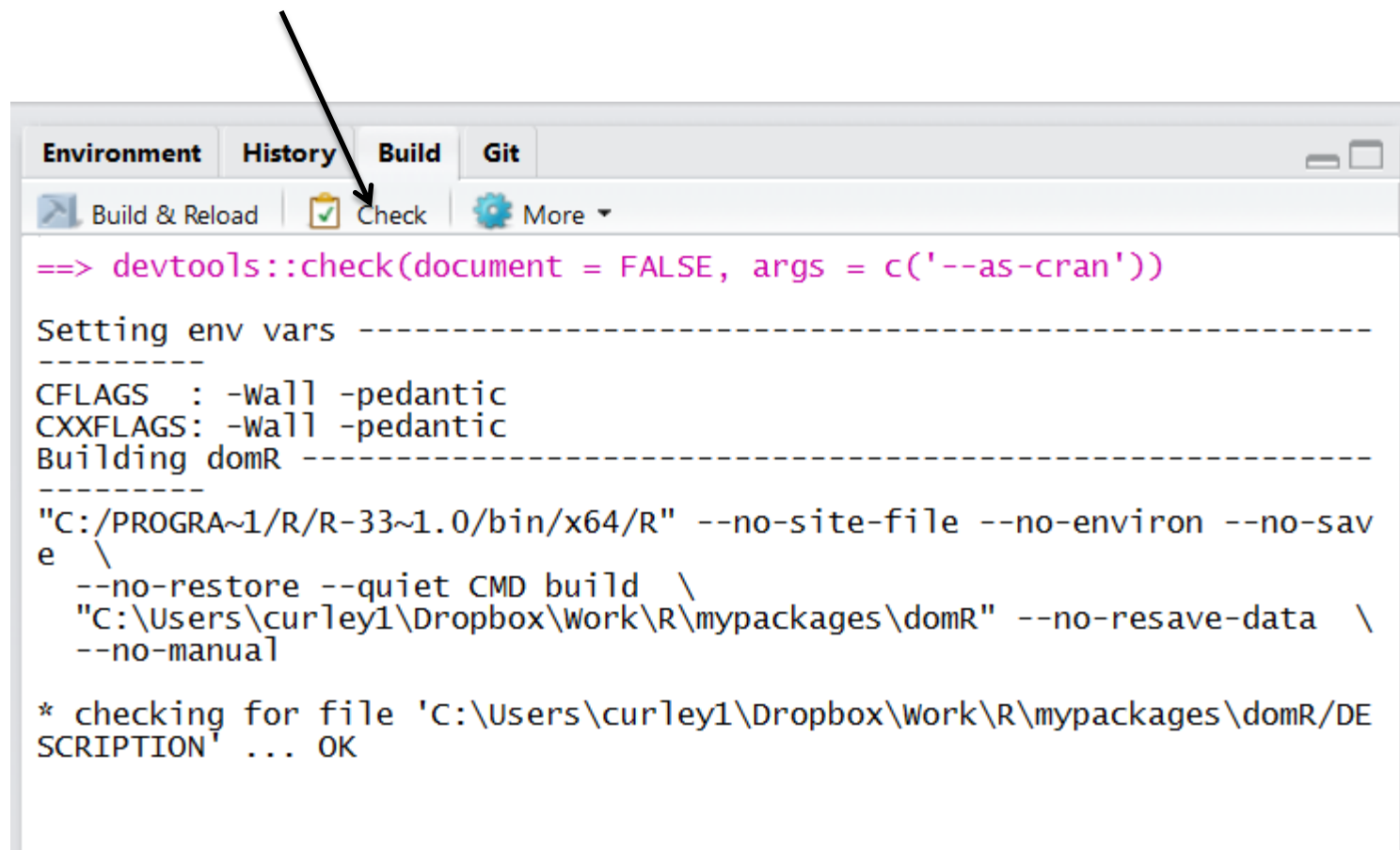
```
Environment History Build Git
Build & Reload Check More

==> Rcmd.exe INSTALL --no-multiarch --with-keep.source domR

* installing to library 'C:/Users/curley1/Dropbox/Work/R/mypackages/dom
R/packrat/lib/x86_64-w64-mingw32/3.3.0'
* installing *source* package 'domR' ...
** R
** data
*** moving datasets to lazyload DB
** preparing package for lazy loading
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded
* DONE (domR)
```

Running a CRAN check

It starts by hitting this button



Running a CRAN check

It then goes like this:



Running a CRAN check

The results will have at the bottom:

- ERRORS - you have to fix these immediately !
- WARNINGS - you really should fix these immediately !
- NOTES - you could not bother – but you should.

The point of this check is to ensure that your package will work consistently.

It can be hard to work out what the ERRORS/WARNINGS/NOTES are referring to, but it's usually some error in the code or documentation that you've written. Persevere with it – Google it !!!

It can help to run checks after every new thing you add to the package – that way you know what went wrong immediately.

* DONE
Status: OK

R CMD check results
0 errors | 0 warnings | 0 notes

R CMD check succeeded



Adding vignettes

devtools::use_vignette()

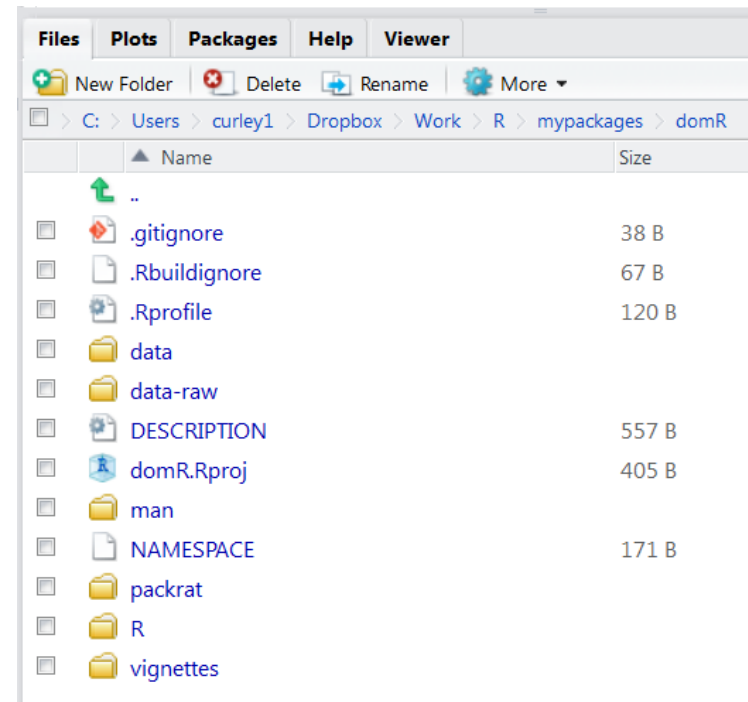
For other colleagues, for peer-reviewers, for other users, for yourself !!!

```
Console C:/Users/curley1/Dropbox/Work/R/mypackages/domR/
> devtools::use_vignette("introduction")
'rmarkdown' must be installed for this functionality.
Would you like to install it?

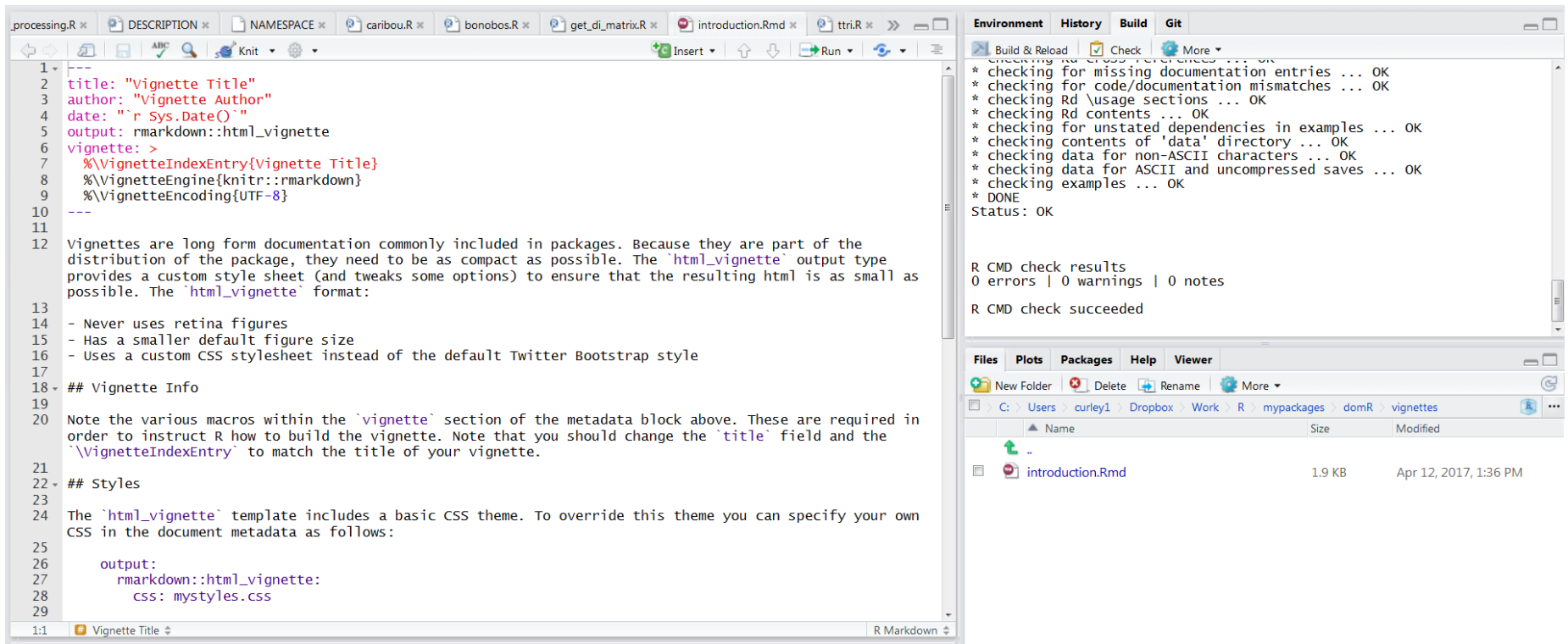
1: Yes
2: No

Selection:
Enter an item from the menu, or 0 to exit
Selection: 1
```

```
The downloaded binary packages are in
  C:\Users\curley1\AppData\Local\Temp\RtmpUByFX\downloaded_packages
* Creating `vignettes`.
* Adding `inst/doc` to ../.gitignore
> |
```



You can now edit the vignette....



The screenshot displays the RStudio interface with an R Markdown file named 'introduction.Rmd' open in the editor. The file contains a metadata block for a vignette and a section on styles. The right-hand pane shows the output of the R CMD check, indicating that all checks passed successfully. The bottom pane shows the file explorer with the 'introduction.Rmd' file highlighted.

```
1 ---  
2 title: "Vignette Title"  
3 author: "Vignette Author"  
4 date: "`r Sys.Date()`"  
5 output: rmarkdown::html_vignette  
6 vignette: >  
7   %\VignetteIndexEntry{Vignette Title}  
8   %\VignetteEngine{knitr::rmarkdown}  
9   %\VignetteEncoding{UTF-8}  
10 ---  
11  
12 Vignettes are long form documentation commonly included in packages. Because they are part of the  
13 distribution of the package, they need to be as compact as possible. The 'html_vignette' output type  
14 provides a custom style sheet (and tweaks some options) to ensure that the resulting html is as small as  
15 possible. The 'html_vignette' format:  
16  
17 - Never uses retina figures  
18 - Has a smaller default figure size  
19 - Uses a custom CSS stylesheet instead of the default Twitter Bootstrap style  
20  
21 ## Vignette Info  
22  
23 Note the various macros within the 'vignette' section of the metadata block above. These are required in  
24 order to instruct R how to build the vignette. Note that you should change the 'title' field and the  
25 '\VignetteIndexEntry' to match the title of your vignette.  
26  
27 ## Styles  
28  
29 The 'html_vignette' template includes a basic CSS theme. To override this theme you can specify your own  
30 CSS in the document metadata as follows:  
31  
32 output:  
33   rmarkdown::html_vignette:  
34     css: mystyles.css
```

Environment History Build Git
Build & Reload Check More
* checking for missing documentation entries ... OK
* checking for code/documentation mismatches ... OK
* checking Rd usage sections ... OK
* checking Rd contents ... OK
* checking for unstated dependencies in examples ... OK
* checking contents of 'data' directory ... OK
* checking data for non-ASCII characters ... OK
* checking data for ASCII and uncompressed saves ... OK
* checking examples ... OK
* DONE
Status: OK

R CMD check results
0 errors | 0 warnings | 0 notes

R CMD check succeeded

Files Plots Packages Help Viewer
New Folder Delete Rename More
C > Users > curley1 > Dropbox > Work > R > mypackages > domR > vignettes
Name Size Modified
..
introduction.Rmd 1.9 KB Apr 12, 2017, 1:36 PM

1:1 Vignette Title R Markdown

Install your package into your R system library

```
devtools::install()
```

Your package should now be available on your machine whenever you need it.

Access it like you would any other package:

```
library("domR")
```

Publishing your package - GitHub

1. Create your repository on GitHub
2. Add the contents of your entire folder
3. Install/share from GitHub directly using the following:

```
devtools::install_github("jalapic/domR")
```

Publishing your package - CRAN

1. `devtools::release()`
2. Follow ALL of the instructions as they appear in the console
3. Add a file called “cran-comments.md” to the main folder.
4. This file should contain information based on the CRAN checks (see example).
5. Add this file to `.Rbuildignore`

See here: <http://r-pkgs.had.co.nz/release.html>

Things to cover in version 2 of this tutorial:

- Writing tests
- Writing a package website using “pkgdown”
- Further ‘how-to’ on CRAN submission
- Adding other data types e.g. internal only data into a package.