

R Programming For Natural Resource Professionals

Week 2: Data Types, Data Structures,
Finding Help, Coding Etiquette, and Workflows

Homework reminders

- Be sure to cite sources, including your colleagues.
- Remember to comment your code.
 - In many cases, line-by-line commenting is appropriate
- Follow instructions exactly.
- Homework will be returned via email. Grades will be on Canvas.
 - Read through the R markdown that I return. It includes notes from me (JH: ...)

Paper discussions

- Two papers for discussion on Tues, Feb 6
 - Low availability of code in ecology: A call for urgent action
 - Tidy data
- As you read, note ideas for:
 - Thoughts
 - Questions
 - Epiphanies
- Will set up assign discussion groups on day of discussion
 - Reporting out via Google Docs file

Topics for today

- Data types
- Data structures
- Finding help
- Coding etiquette
- Workflows

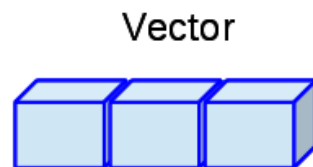
Data types

Data type	Example	Verify
Logical	TRUE, FALSE	<pre>a <- TRUE class(a)</pre>
Numeric	42.1, 4, 2215	<pre>b <- 23.4 class(b)</pre>
Integer	2L, 24L, 0L	<pre>d <- 2L class(d)</pre>
Character	"a", "good", "perch"	<pre>f <- "perch" class(f)</pre>
Complex	1+4i	<pre>g <- "1+4i" class(g)</pre>

Topics for today

- Data types
- Data structures
- Finding help
- Coding etiquette
- Workflows

Data structures



Vector: A sequence of items of the same type.

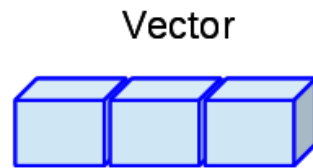
Most basic data structure in R

Items in a vector can be accessed using `[]`

Length of vector displayed using `length()`

```
> apple <- c("red", "green", "yellow")  
> apple  
> class(apple)  
> length(apple)  
> apple[2]
```

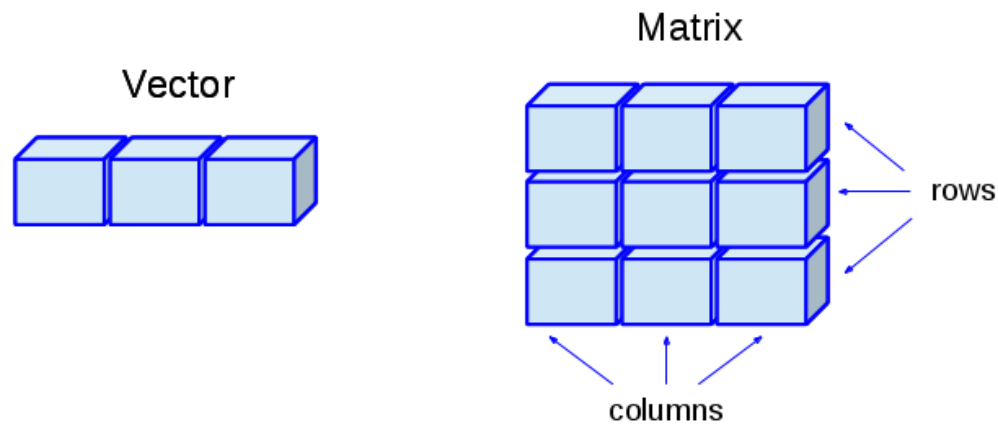
Data structures



Factor: A common type of vector used in plotting and modeling.
Forces values of the vector into categories

```
> apple <- c("red", "green", "yellow")  
> factor_apple <- as.factor(apple)  
> factor_apple  
> str(factor_apple)
```


Data structures

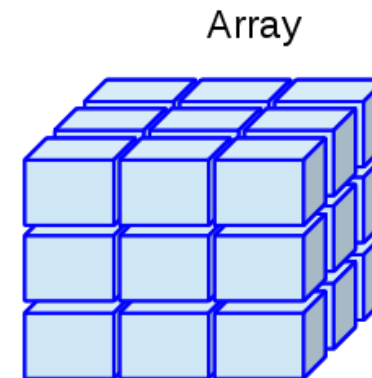
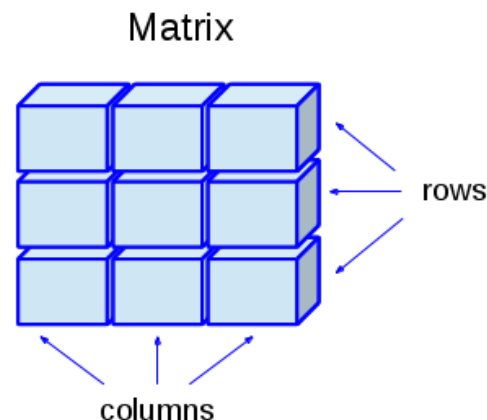
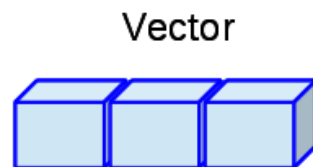


Matrix: A two-dimensional array

```
> matrix1 <- matrix(c("a", "a", "b", "c", "c", "a"),  
  nrow = 2, ncol = 3, byrow = TRUE)
```

```
> matrix1
```

Data structures



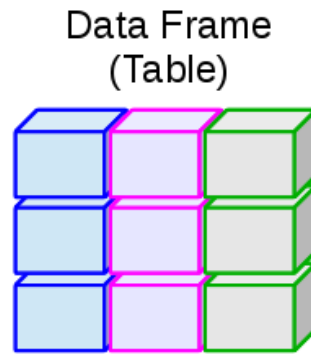
Array: A multidimensional matrix

Any number of dimensions.

The array function's dim attribute is used to specify dimensionality.

```
> array1 <- array(c("green", "yellow"), dim = c(3,3,2))  
> array1
```

Data structures



Data frame: A table, similar to a matrix, but each variable (column) can be a different data type

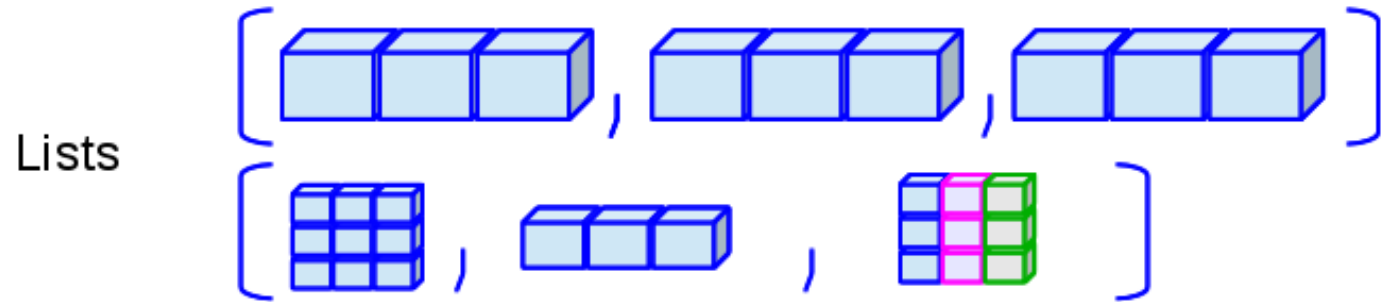
```
> df1 <- data.frame(gender = c("Male", "Female",  
"Female"), height = c(152, 171.5, 165), weight = c(81,  
93, 83), age = c(42, 38, 26))
```

```
> df1
```

```
> str(df1)
```

```
> summary(df1)
```

Data structures



Lists: Can contain many different types of elements, such as vectors, arrays, data frames, or even other lists.

```
> list1 <- list(c(2,5,3), 21.3, "tree")  
> str(list1)  
> list1[3]  
> list1[[3]]  
> list1[[1]][[2]]
```

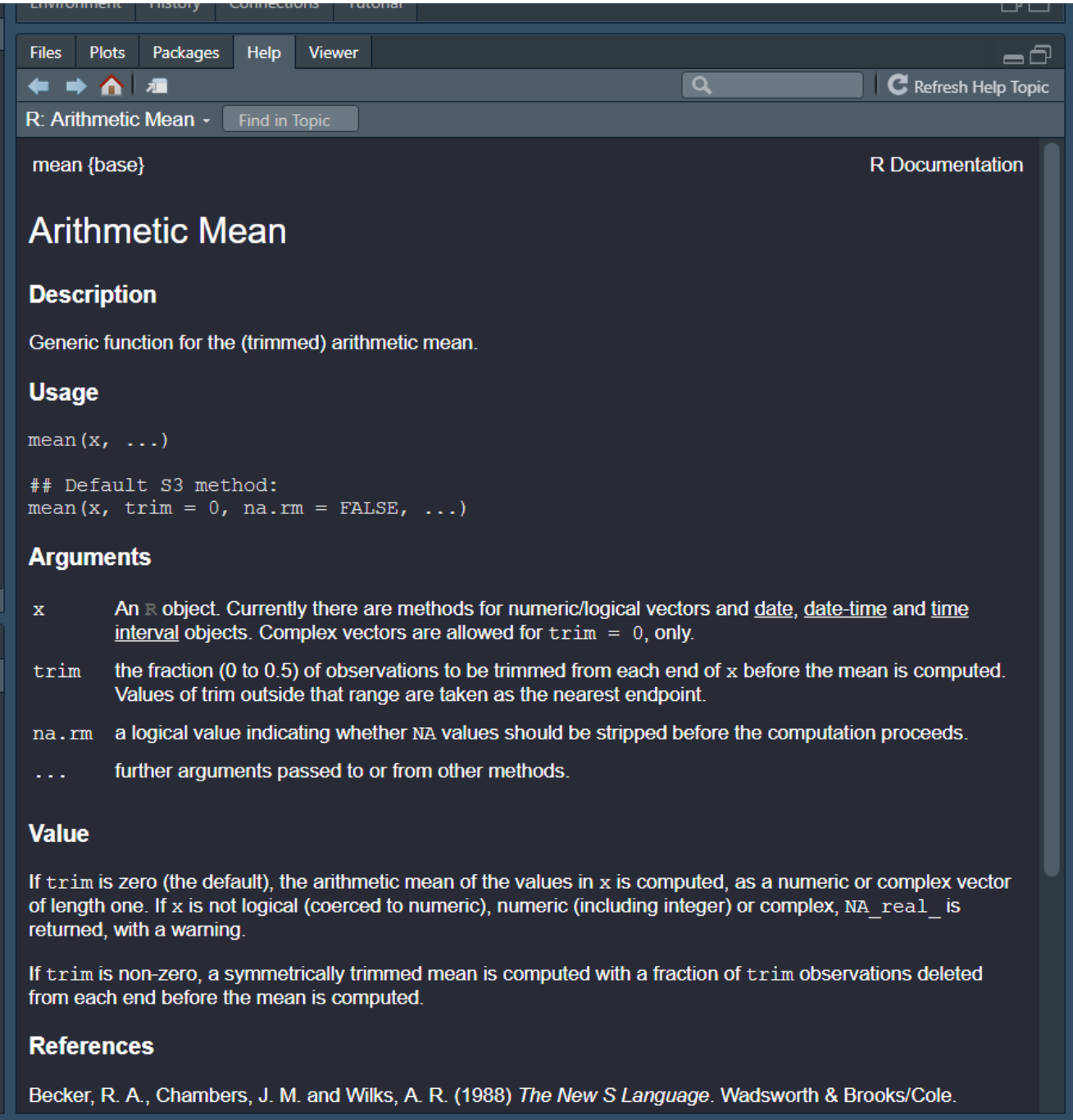
Topics for today

- Data types
- Data structures
- Finding help
- Coding etiquette
- Workflows

Finding help

R's built-in help pages

> ?mean



The screenshot shows the R help interface for the 'mean' function. The top navigation bar includes tabs for 'Files', 'Plots', 'Packages', 'Help', and 'Viewer'. Below this is a search bar and a 'Refresh Help Topic' button. The main content area is titled 'R: Arithmetic Mean' and includes a 'Find in Topic' button. The text 'mean {base}' and 'R Documentation' are visible in the top right of the content area. The page is structured with sections: 'Description' (Generic function for the (trimmed) arithmetic mean.), 'Usage' (mean(x, ...)), 'Arguments' (x: An R object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for trim = 0, only. trim: the fraction (0 to 0.5) of observations to be trimmed from each end of x before the mean is computed. Values of trim outside that range are taken as the nearest endpoint. na.rm: a logical value indicating whether NA values should be stripped before the computation proceeds. ...: further arguments passed to or from other methods.), 'Value' (If trim is zero (the default), the arithmetic mean of the values in x is computed, as a numeric or complex vector of length one. If x is not logical (coerced to numeric), numeric (including integer) or complex, NA_real_ is returned, with a warning. If trim is non-zero, a symmetrically trimmed mean is computed with a fraction of trim observations deleted from each end before the mean is computed.), and 'References' (Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.).

mean {base} R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)
```

Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)

Arguments

x	An R object. Currently there are methods for numeric/logical vectors and <u>date</u> , <u>date-time</u> and <u>time interval</u> objects. Complex vectors are allowed for trim = 0, only.
trim	the fraction (0 to 0.5) of observations to be trimmed from each end of x before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.
...	further arguments passed to or from other methods.

Value

If trim is zero (the default), the arithmetic mean of the values in x is computed, as a numeric or complex vector of length one. If x is not logical (coerced to numeric), numeric (including integer) or complex, NA_real_ is returned, with a warning.

If trim is non-zero, a symmetrically trimmed mean is computed with a fraction of trim observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

Finding help



Online resources

Stack Overflow

Package vignettes on CRAN

- <https://cran.r-project.org/web/packages/vegan/index.html>

R-bloggers

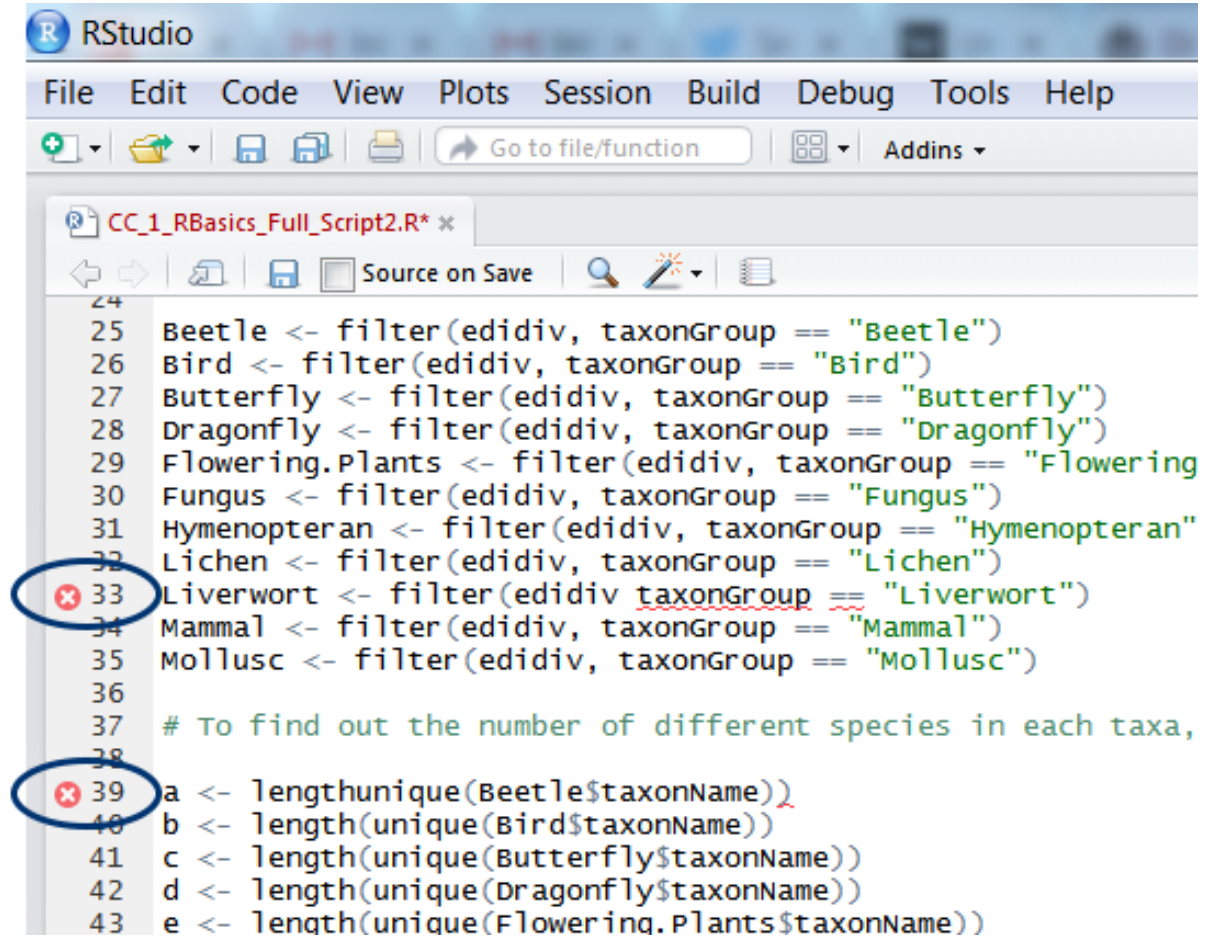
Package's GitHub sites

Finding help

Realtime Error Checking in Rstudio

Syntax errors are common and easy to make.

- Open bracket or parentheses
- Missing commas
- Extra character or other typo



The screenshot shows the RStudio interface with a script file named 'CC_1_RBasics_Full_Script2.R'. The script contains several lines of R code. Two lines are circled in blue, each with a red 'x' icon indicating a syntax error. The first error is on line 33, where the variable 'Liverwort' is assigned a filter function, but the closing parenthesis for the 'filter' function is missing. The second error is on line 39, where the variable 'a' is assigned the result of 'lengthunique(Beetle\$taxonName)', but the closing parenthesis for the 'lengthunique' function is missing. The rest of the script is as follows:

```
24
25 Beetle <- filter(edidiv, taxonGroup == "Beetle")
26 Bird <- filter(edidiv, taxonGroup == "Bird")
27 Butterfly <- filter(edidiv, taxonGroup == "Butterfly")
28 Dragonfly <- filter(edidiv, taxonGroup == "Dragonfly")
29 Flowering.Plants <- filter(edidiv, taxonGroup == "Flowering
30 Fungus <- filter(edidiv, taxonGroup == "Fungus")
31 Hymenopteran <- filter(edidiv, taxonGroup == "Hymenopteran")
32 Lichen <- filter(edidiv, taxonGroup == "Lichen")
33 Liverwort <- filter(edidiv, taxonGroup == "Liverwort")
34 Mammal <- filter(edidiv, taxonGroup == "Mammal")
35 Mollusc <- filter(edidiv, taxonGroup == "Mollusc")
36
37 # To find out the number of different species in each taxa,
38
39 a <- lengthunique(Beetle$taxonName))
40 b <- length(unique(Bird$taxonName))
41 c <- length(unique(Butterfly$taxonName))
42 d <- length(unique(Dragonfly$taxonName))
43 e <- length(unique(Flowering.Plants$taxonName))
```


Common errors

Error: Could not find function 'functionName'

Likely solution: Package containing function not loaded. `library(packageName)`

Error: There is no package called 'packageName'

Likely solution: Package isn't installed

Error: object 'objectName' not found

Likely solution: Check your environmental panel that object is loaded. Check for typos.

Error: unexpected symbol in 'lineOfCode'

Likely solution: A forgotten or extra comma, bracket, etc.

Topics for today

- Data types
- Data structures
- Finding help
- Coding etiquette
- Workflows

Coding Etiquette

File names should be meaningful

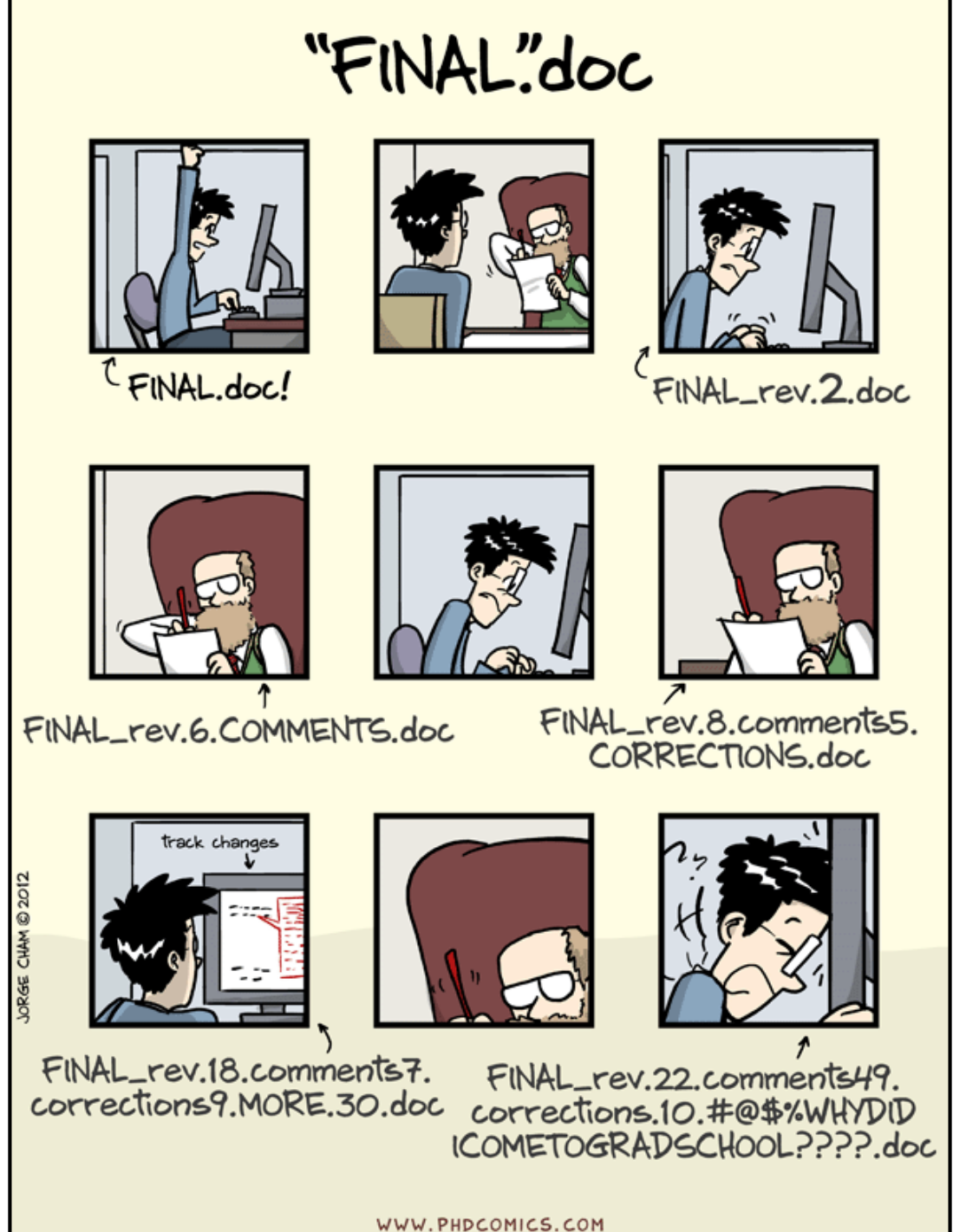
- dataWrangling.R
- repeatedMeasuresModel.R
- modelSelection.R

Coding Etiquette

If files must be run in sequence,
give them numerical prefixes:

- 1_dataWrangling.R
- 2_modelSelection.R
- 3_plotting.R

Word from the wise- no matter how
much you think it is appropriate, never
name anything "final."



Coding Etiquette

Function names should be **verbs**

Object names should be **nouns**

Good function names

parse_data

generate_boxplots

Good object names

green_bay_env

mccurdy_woodlot_dbh

Coding Etiquette

Do not assign names to existing functions

For example:

- C
- T
- F
- data

Coding Etiquette

Put **space** around all operators and after commas.
Just like English. Make your code readable.

Good: `average <- mean(feet / 12 + inches, na.remove = TRUE)`

Bad: `average<-mean(feet/12+inches,na.remove=TRUE)`

Coding Etiquette

Assignment should always be done using “<-”, never “=”

Good: `x <- 10`

Bad: `x = 10`

Coding Etiquette

Case: lots of case options. Just be consistent.

- lower_snake
- UPPER_SNAKE
- lowerCamelCase
- UpperCamelCase
- kebab-case

Coding Etiquette

Use commenting frequently

Titling scripts:

```
#####  
#### Length frequency plots ####  
#####
```

In R Studio: ##### generates a collapsible code block

Topics for today

- Data types
- Data structures
- Finding help
- Coding etiquette
- **Workflows**

Workflows

Goal: Portable code

- R scripts and data files recreate the environment

R Studio facilitates a project-oriented workflow to help with this

Workflows

Projects include:

- Input data
- R scripts
- A default home directory
 - Enables writing of relative paths (e.g., data/species_counts.csv)

One folder = one project

Workflows

Projects directory structures can vary, but a good starting point is:

1. data folder: all input data and metadata
2. doc folder: manuscript and other documents
3. output folder: intermediate outputs (e.g., wrangled data)
4. R folder: R scripts
5. Rmd: Rmarkdown scripts
6. reports folder: R Markdown files that generate reports

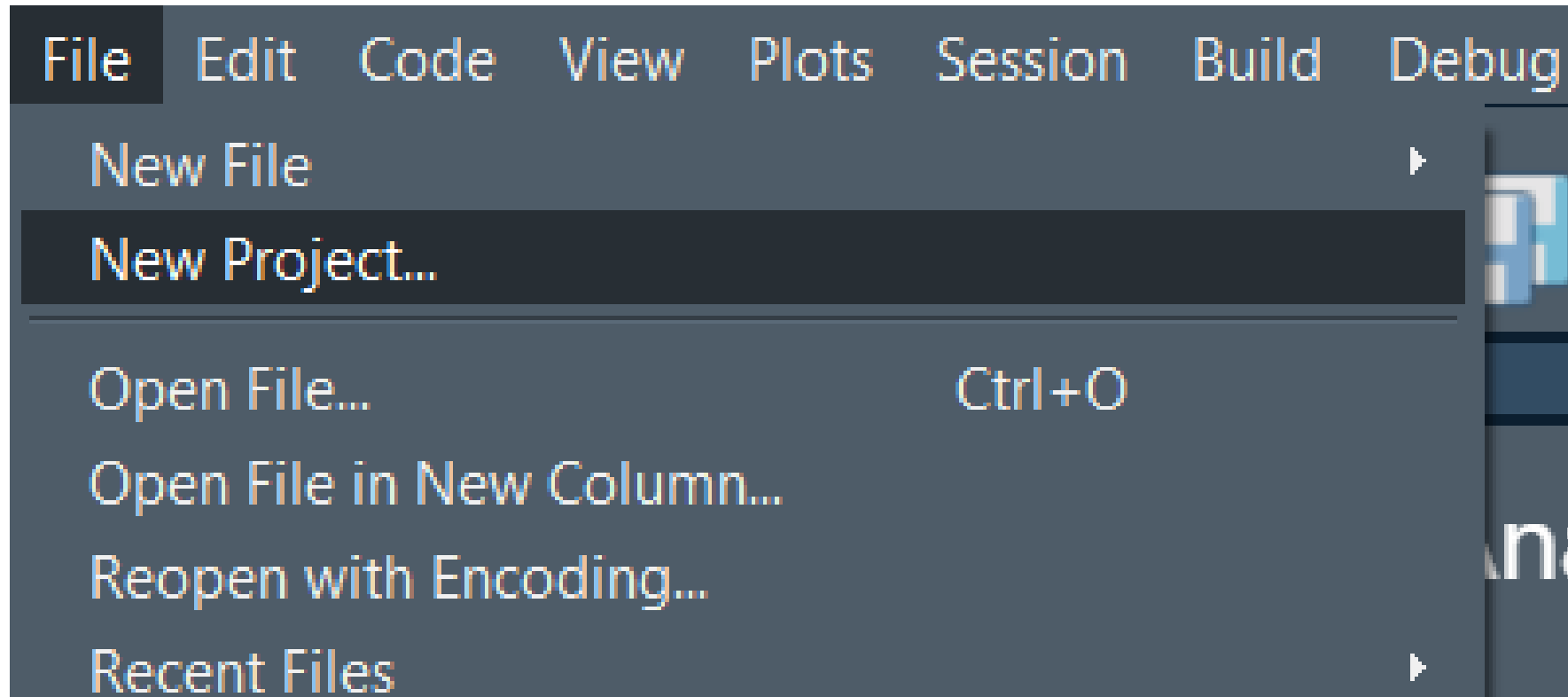
Workflows

Projects should contain scripts organized for a logical workflow:

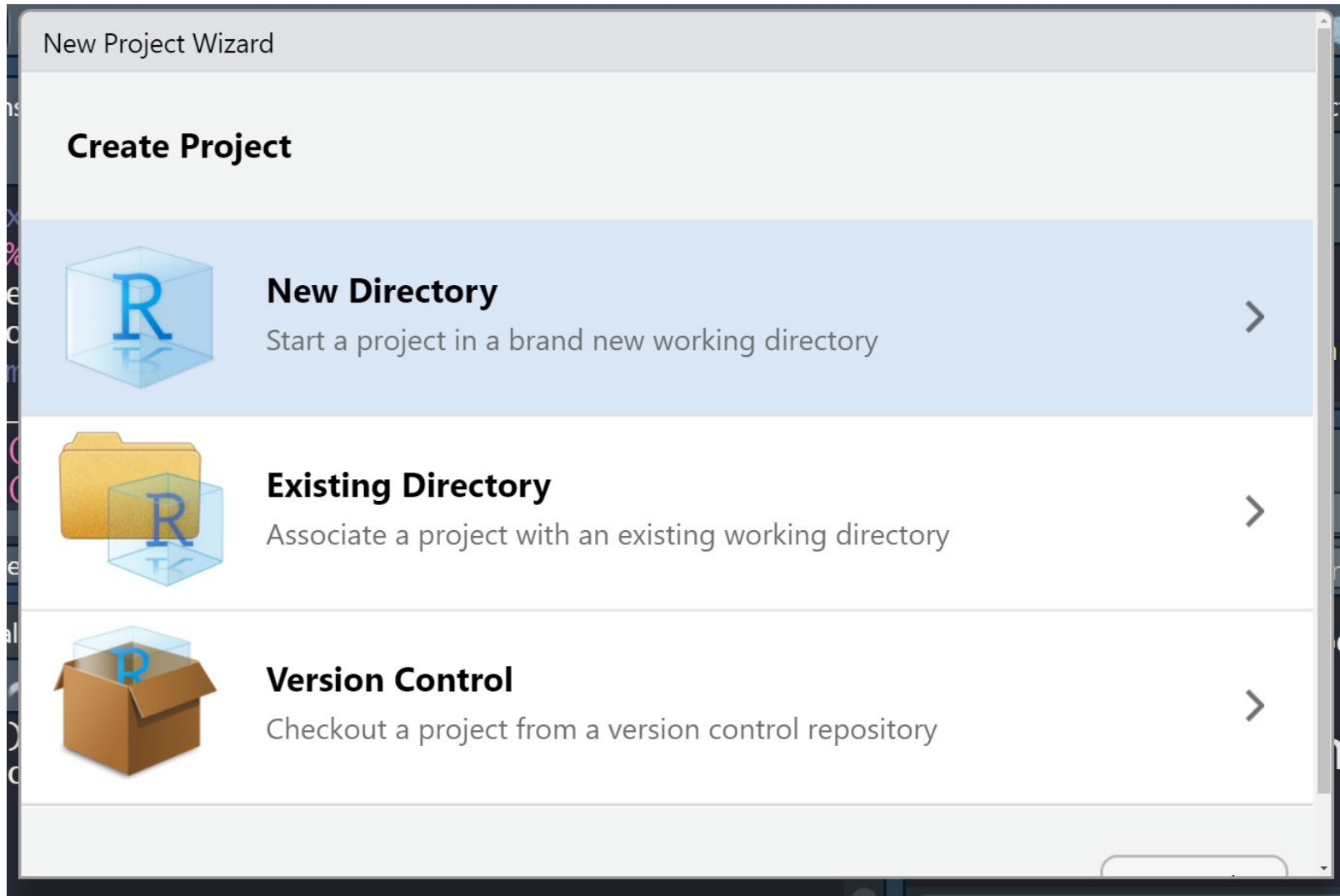
1. Load and merge data (Always work from your raw data!)
2. Data wrangling
3. Data analyses
4. Generate outputs such as tables and figures

Create project for RforNatRes

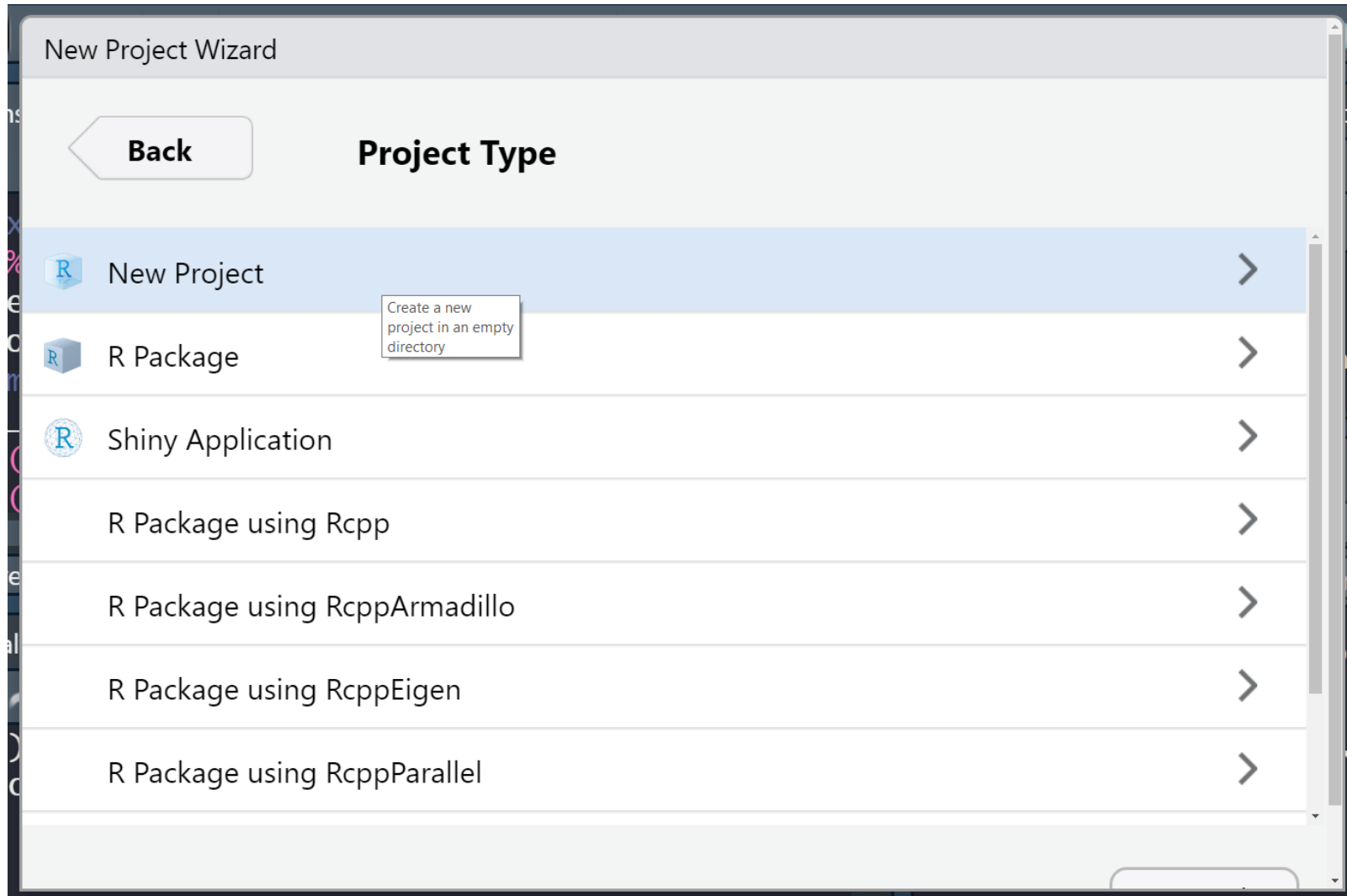
 RStudio



Create project for RforNatRes




Create project for RforNatRes



Create project for RforNatRes

New Project Wizard

[Back](#) **Create New Project**



Directory name:

Create project as subdirectory of:
 [Browse...](#)

☐ Create a git repository

☐ Use renv with this project

☐ Open in new session

