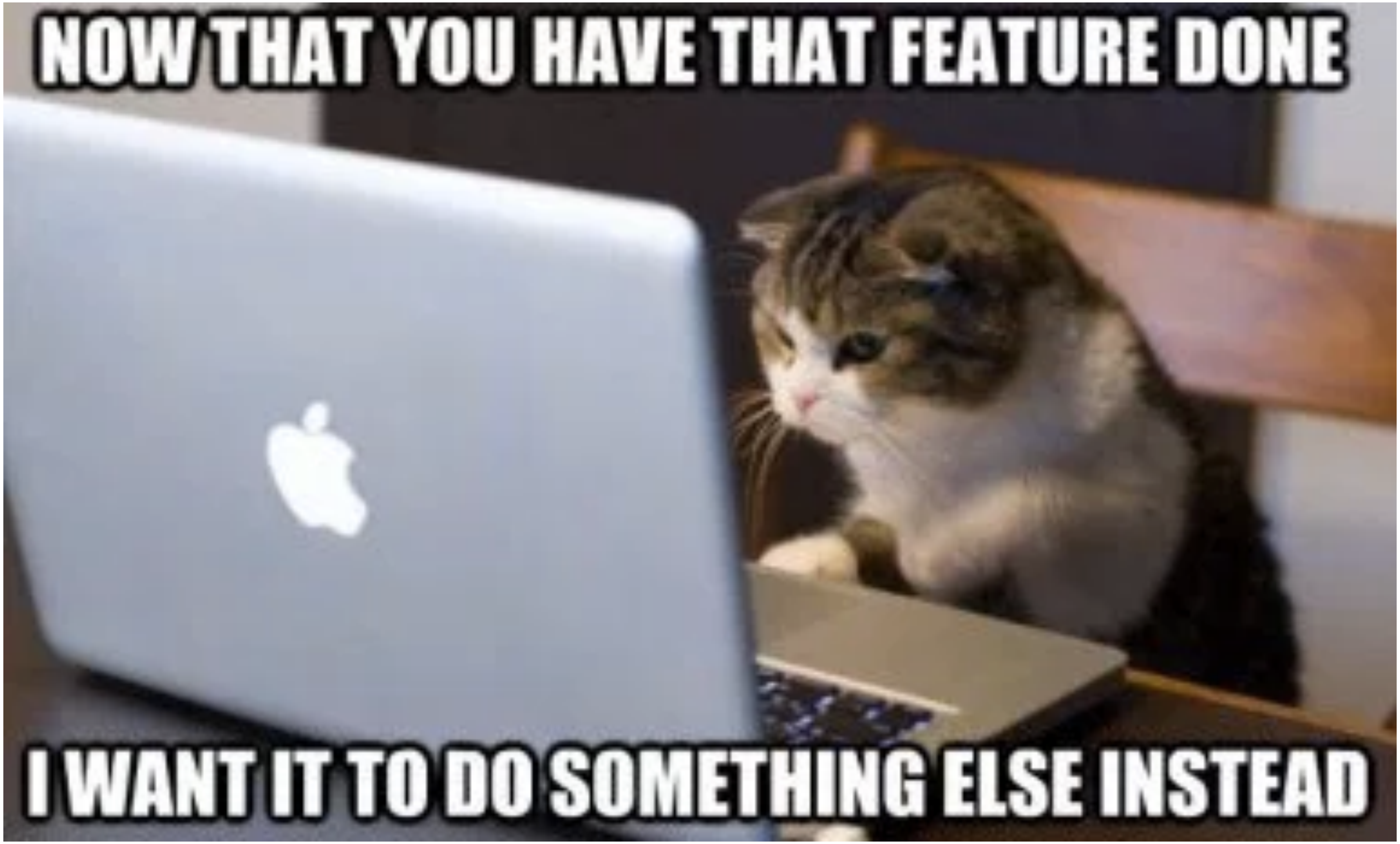
R Bootcamp (continued)

Dani Navarro
Amy Perfors


NOW THAT YOU HAVE THAT FEATURE DONE
I WANT IT TO DO SOMETHING ELSE INSTEAD

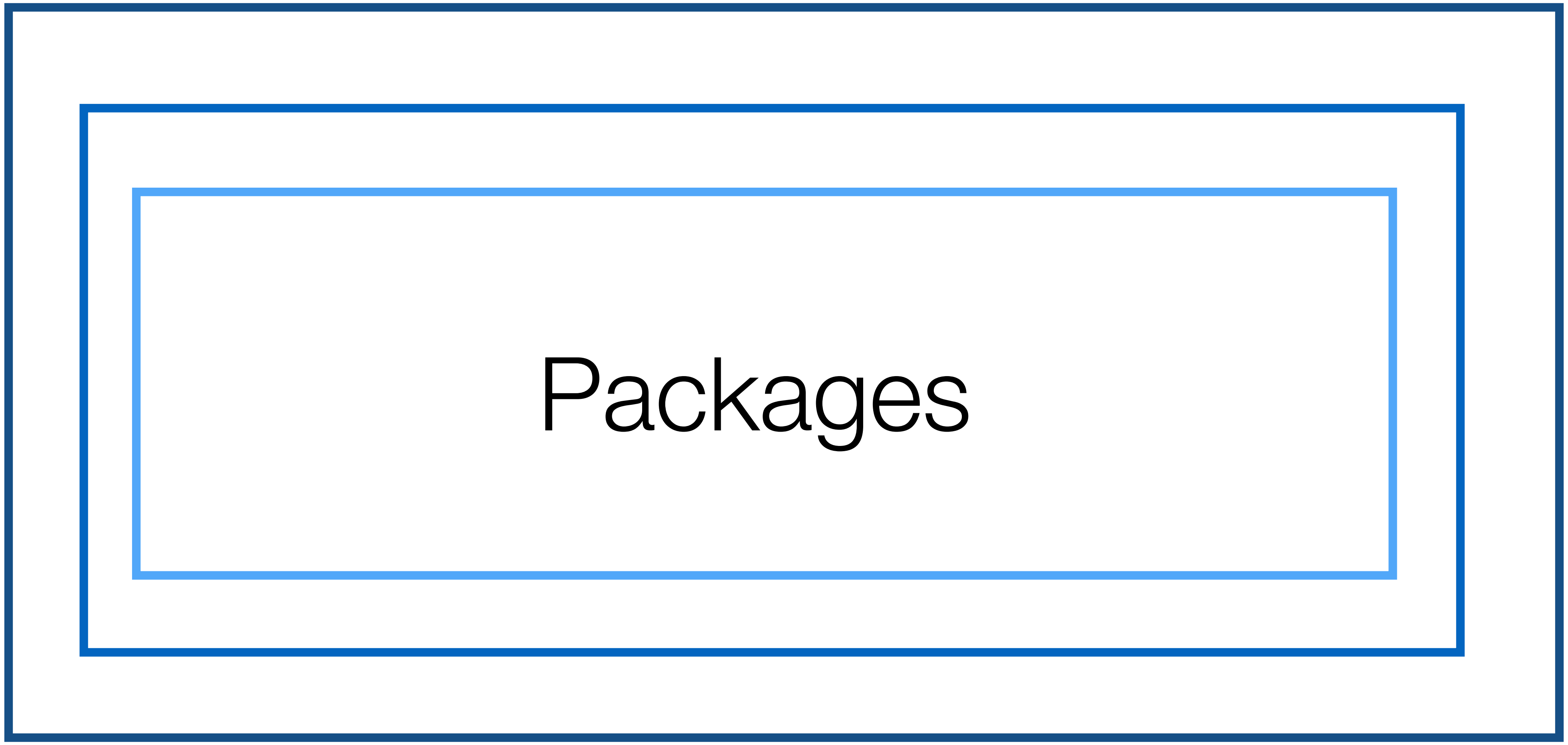
This time

More useful things: data, data, data

Before we begin: you should make sure you've opened RStudio on your laptop so you can follow along

You will need to download the following datasets:
`toydata.RData` and `toydata.csv`

They are on the resource page here:
http://chdsummerschool.com/resources.html



Packages

Packages

- What is a package?
    - A collection of R functions and data sets added to the R “ecosystem”
    - They extend the functionality of R: there’s 5000+ packages out there
    - You can download them from the internet (easiest way: via RStudio)
        - (It accesses the R archive network called CRAN but you really don’t need to care about this)

Terminology

- **Installed** means...
  - That the package files are stored on your computer
  - Your version of R is able to load the package

- **Loaded** means...
  - That R has opened the package, and “knows” what it contains
  - You can use the functions / data stored in the package

- As a result:
  - A package must be **installed** before you can **load it**
  - A package must be **loaded** before you can **use it**

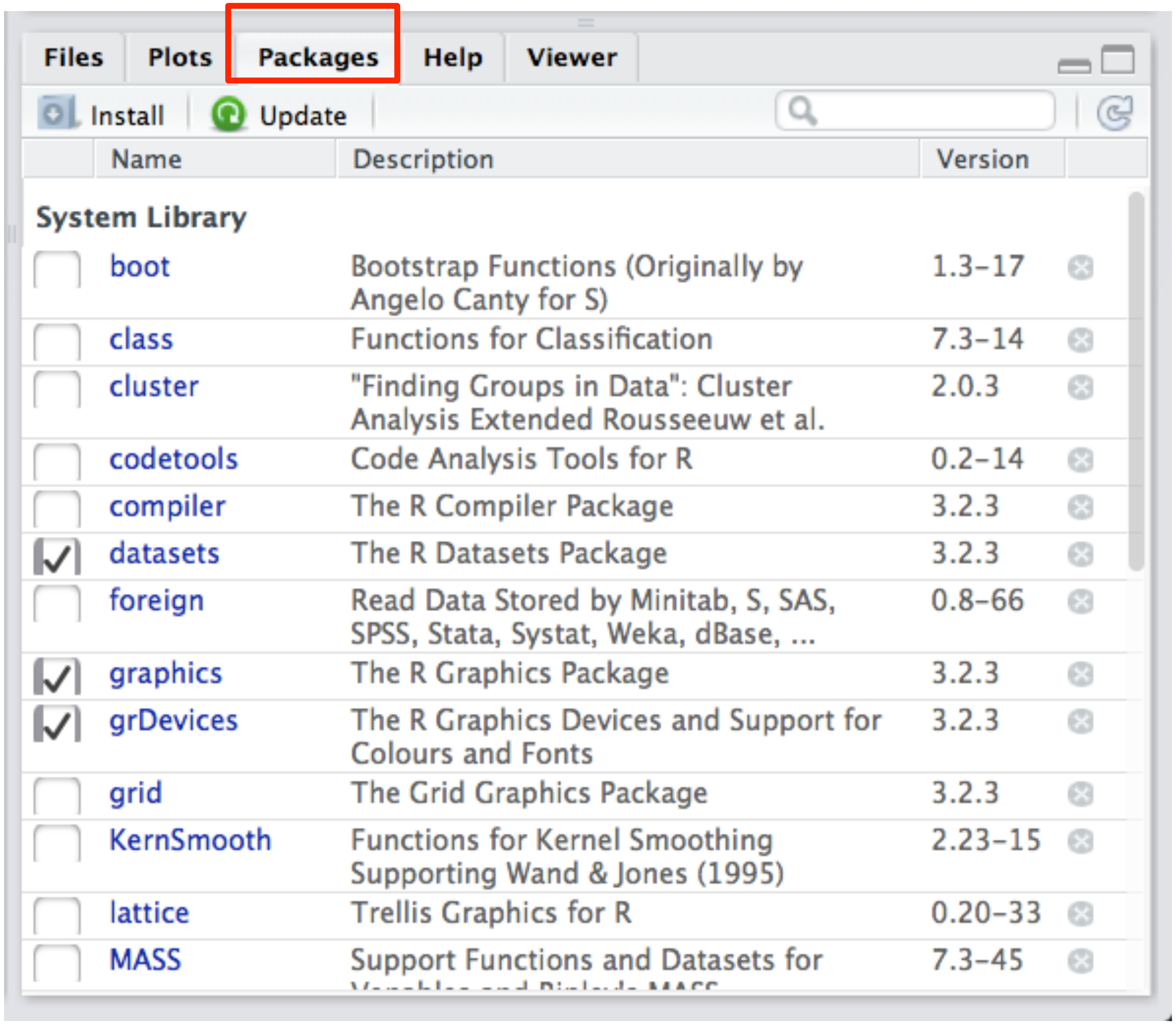
Why does it work like that???

- R is big
  - 5000+ packages means can cause confusion
  - Different authors will use the same name to refer to different functions!
  - e.g., there are multiple packages that define a logit() function.

- Separating install from load avoids inconsistency:
  - R only has to resolve the names of things in the loaded packages!
  - **Install** everything you might want to use sometime
  - **Load** only those things you need to use now!

The Rstudio “packages” panel

(lower right part
of RStudio)

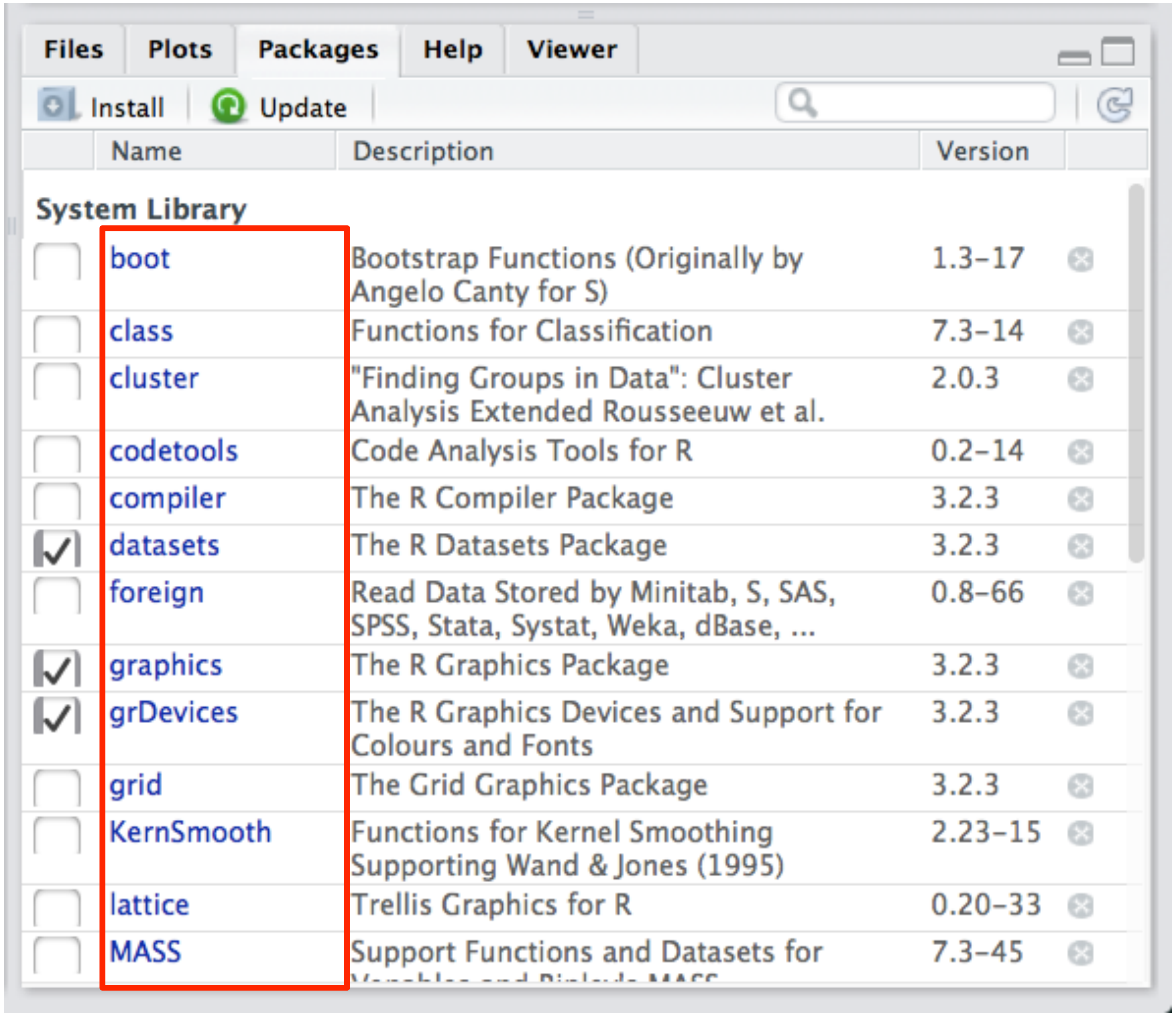


System Library

	Name	Description	Version
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☐	class	Functions for Classification	7.3-14
☐	cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
☐	codetools	Code Analysis Tools for R	0.2-14
☐	compiler	The R Compiler Package	3.2.3
☑	datasets	The R Datasets Package	3.2.3
☐	foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
☑	graphics	The R Graphics Package	3.2.3
☑	grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
☐	grid	The Grid Graphics Package	3.2.3
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
☐	lattice	Trellis Graphics for R	0.20-33
☐	MASS	Support Functions and Datasets for	7.3-45

The Rstudio “packages” panel

These are the names of the packages that are installed

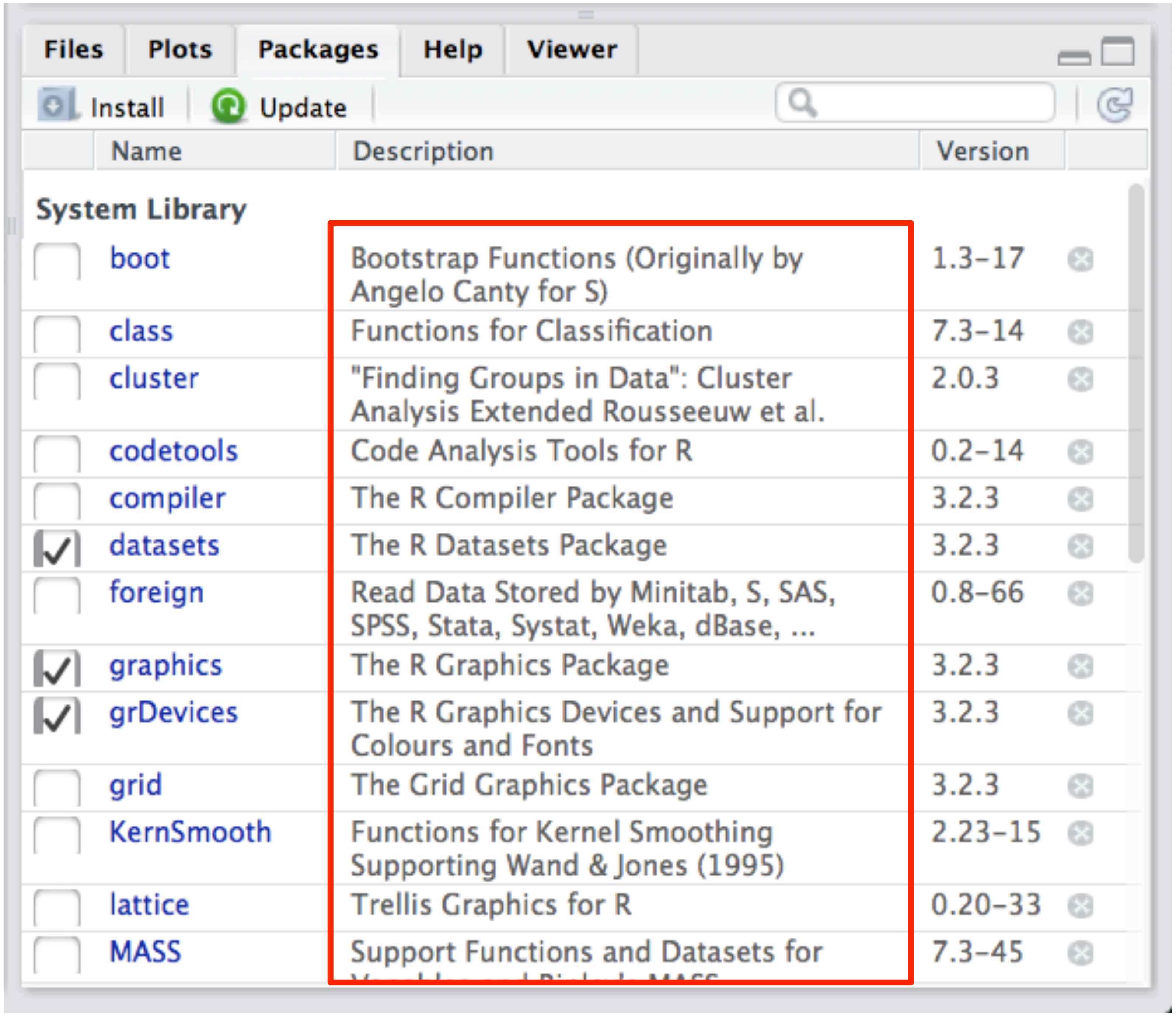


System Library

	Name	Description	Version
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☐	class	Functions for Classification	7.3-14
☐	cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
☐	codetools	Code Analysis Tools for R	0.2-14
☐	compiler	The R Compiler Package	3.2.3
☑	datasets	The R Datasets Package	3.2.3
☐	foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
☑	graphics	The R Graphics Package	3.2.3
☑	grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
☐	grid	The Grid Graphics Package	3.2.3
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
☐	lattice	Trellis Graphics for R	0.20-33
☐	MASS	Support Functions and Datasets for	7.3-45

The Rstudio “packages” panel

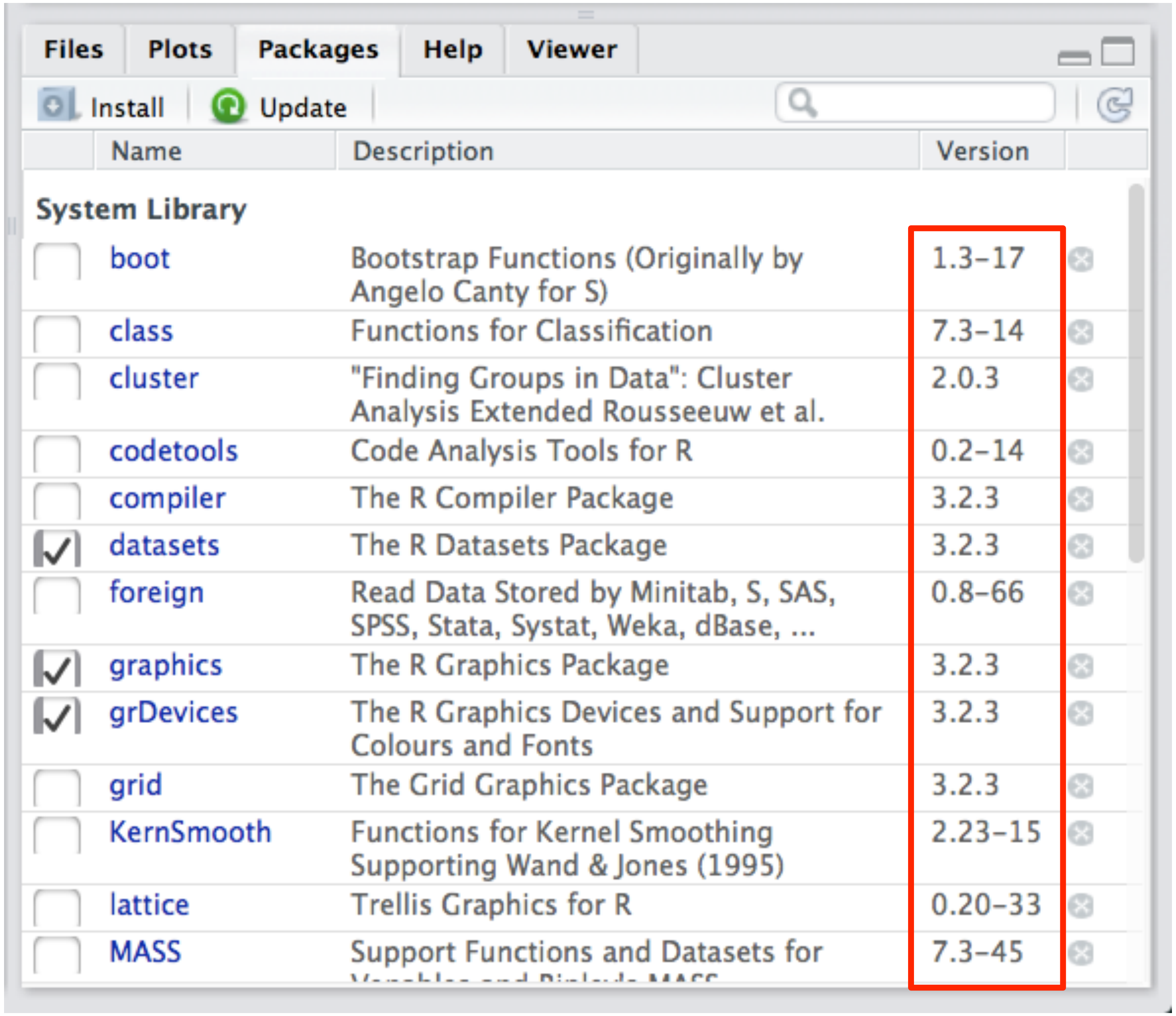
This describes what the package does



	Name	Description	Version
	System Library		
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☐	class	Functions for Classification	7.3-14
☐	cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
☐	codetools	Code Analysis Tools for R	0.2-14
☐	compiler	The R Compiler Package	3.2.3
☑	datasets	The R Datasets Package	3.2.3
☐	foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
☑	graphics	The R Graphics Package	3.2.3
☑	grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
☐	grid	The Grid Graphics Package	3.2.3
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
☐	lattice	Trellis Graphics for R	0.20-33
☐	MASS	Support Functions and Datasets for	7.3-45

The Rstudio “packages” panel

This tells you what version you have

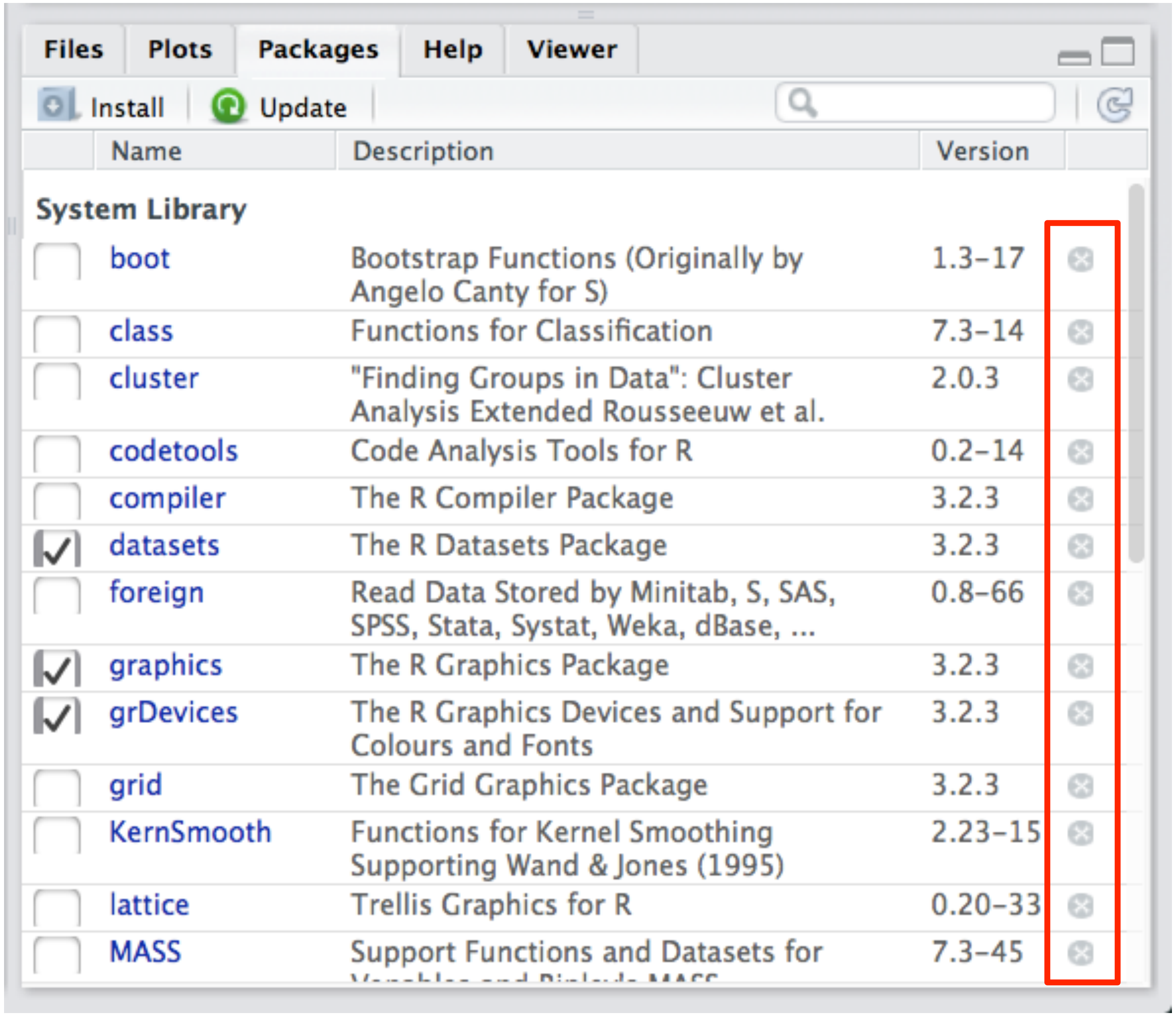


System Library

	Name	Description	Version
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☐	class	Functions for Classification	7.3-14
☐	cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
☐	codetools	Code Analysis Tools for R	0.2-14
☐	compiler	The R Compiler Package	3.2.3
☑	datasets	The R Datasets Package	3.2.3
☐	foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
☑	graphics	The R Graphics Package	3.2.3
☑	grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
☐	grid	The Grid Graphics Package	3.2.3
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
☐	lattice	Trellis Graphics for R	0.20-33
☐	MASS	Support Functions and Datasets for	7.3-45

The Rstudio “packages” panel

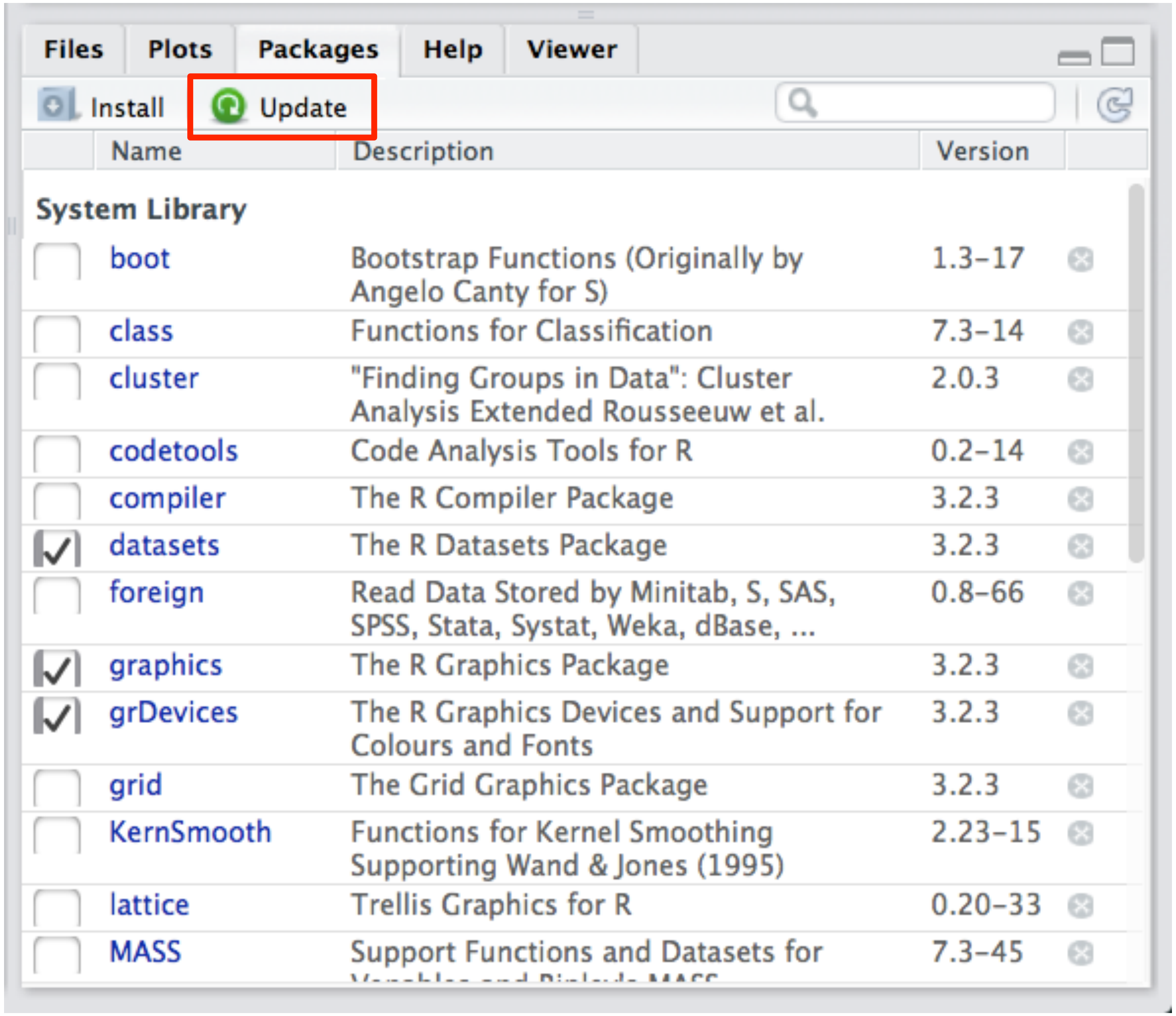
Clicking this will uninstall the package



	Name	Description	Version
	System Library		
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☐	class	Functions for Classification	7.3-14
☐	cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
☐	codetools	Code Analysis Tools for R	0.2-14
☐	compiler	The R Compiler Package	3.2.3
☑	datasets	The R Datasets Package	3.2.3
☐	foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
☑	graphics	The R Graphics Package	3.2.3
☑	grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
☐	grid	The Grid Graphics Package	3.2.3
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
☐	lattice	Trellis Graphics for R	0.20-33
☐	MASS	Support Functions and Datasets for	7.3-45

The Rstudio “packages” panel

This will check whether any new versions of the package are available

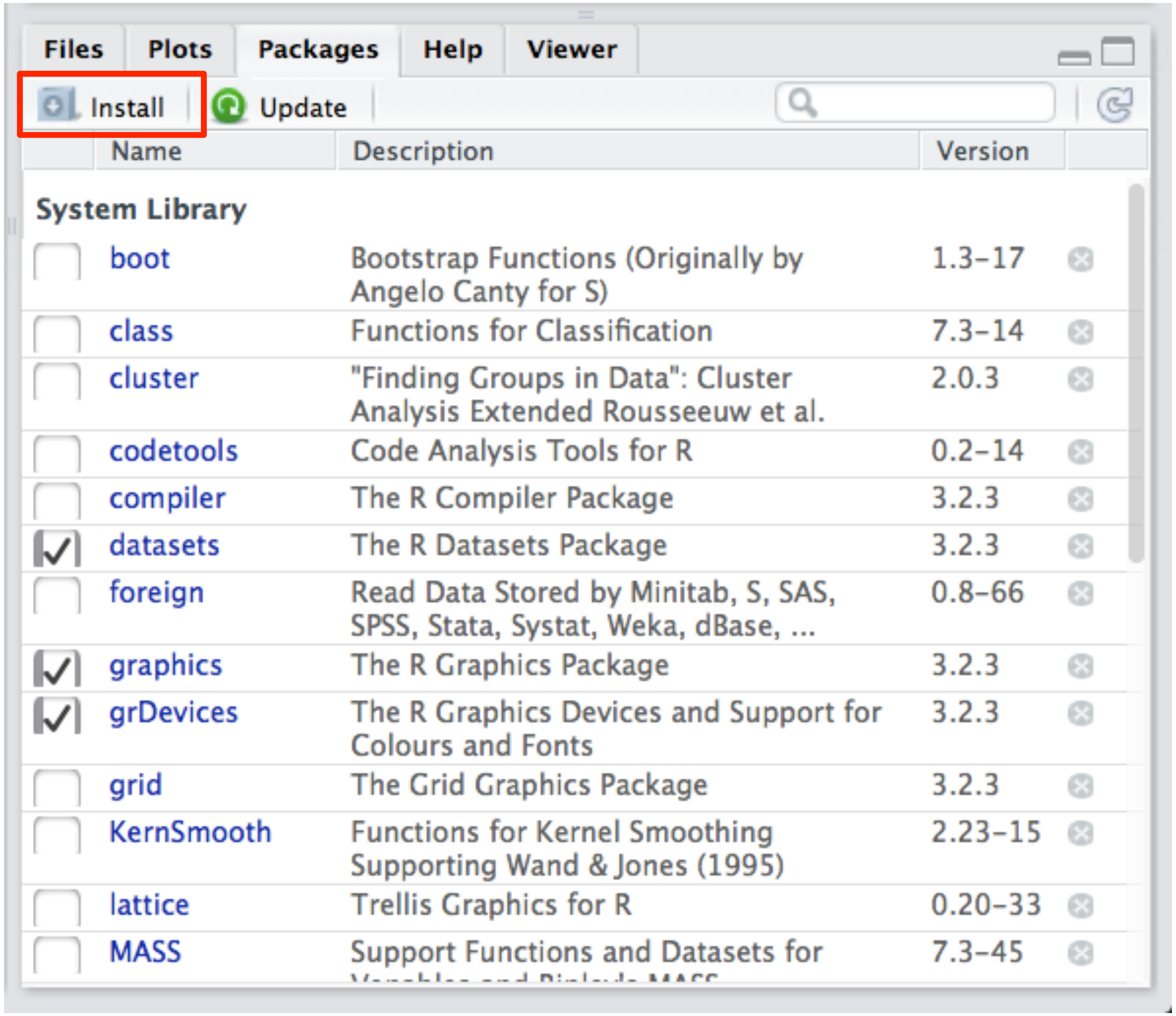


System Library

	Name	Description	Version
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☐	class	Functions for Classification	7.3-14
☐	cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
☐	codetools	Code Analysis Tools for R	0.2-14
☐	compiler	The R Compiler Package	3.2.3
☑	datasets	The R Datasets Package	3.2.3
☐	foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
☑	graphics	The R Graphics Package	3.2.3
☑	grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
☐	grid	The Grid Graphics Package	3.2.3
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
☐	lattice	Trellis Graphics for R	0.20-33
☐	MASS	Support Functions and Datasets for	7.3-45

The Rstudio "packages" panel

This is how you install new packages (we'll come back to this)



System Library

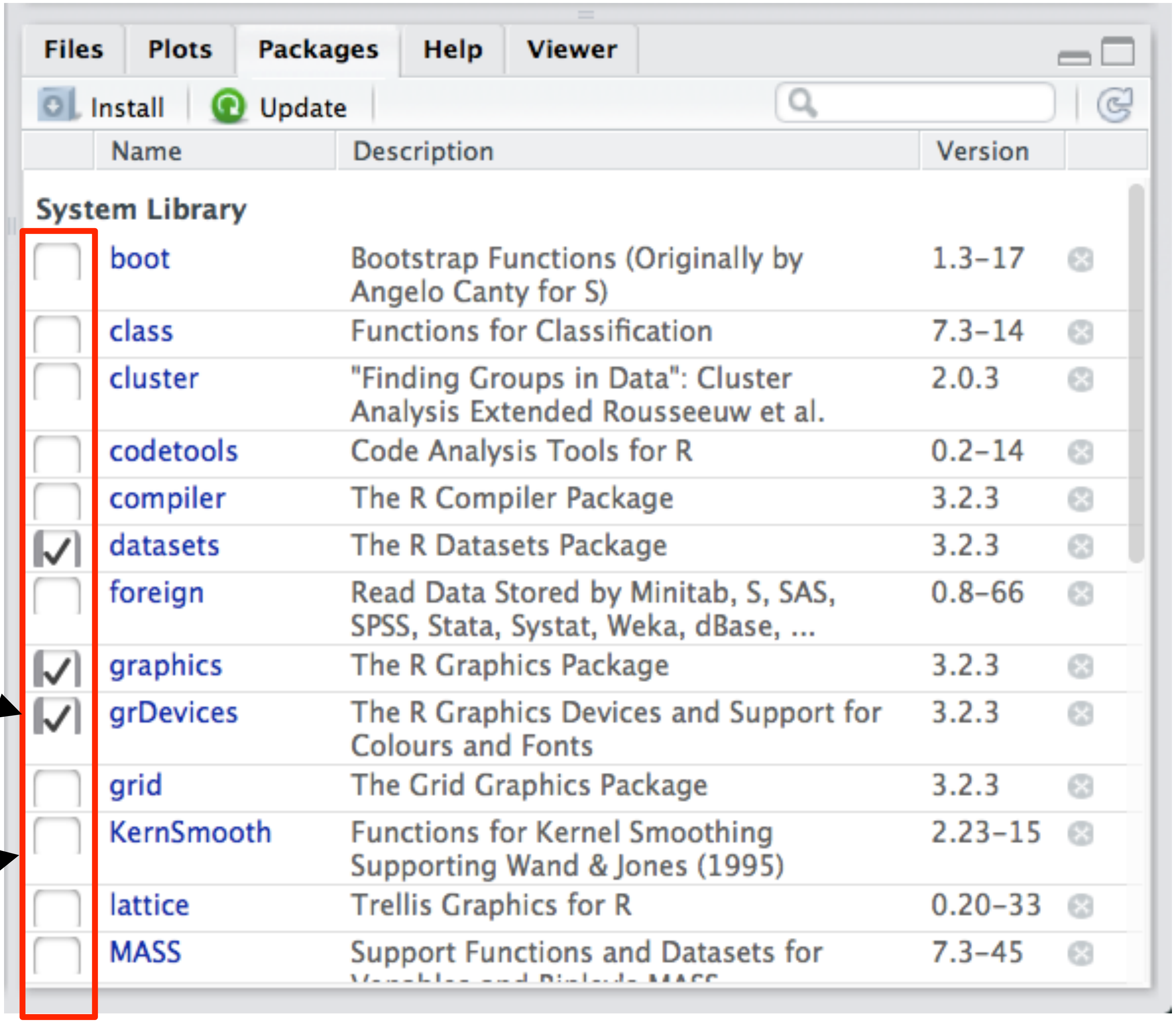
	Name	Description	Version
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☐	class	Functions for Classification	7.3-14
☐	cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
☐	codetools	Code Analysis Tools for R	0.2-14
☐	compiler	The R Compiler Package	3.2.3
☑	datasets	The R Datasets Package	3.2.3
☐	foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
☑	graphics	The R Graphics Package	3.2.3
☑	grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
☐	grid	The Grid Graphics Package	3.2.3
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
☐	lattice	Trellis Graphics for R	0.20-33
☐	MASS	Support Functions and Datasets for	7.3-45

The Rstudio “packages” panel

Click here to load or unload a package

loaded

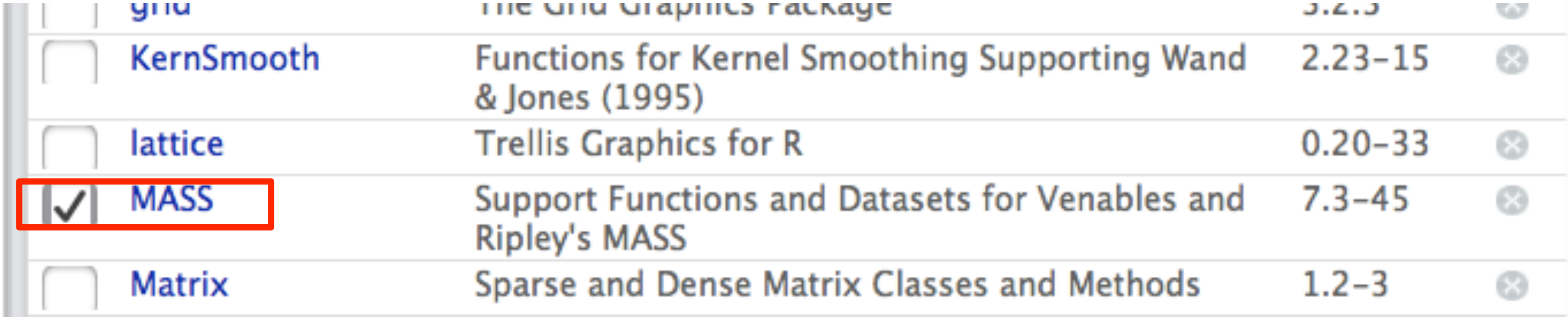
unloaded



	Name	Description	Version
System Library			
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☐	class	Functions for Classification	7.3-14
☐	cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
☐	codetools	Code Analysis Tools for R	0.2-14
☐	compiler	The R Compiler Package	3.2.3
☑	datasets	The R Datasets Package	3.2.3
☐	foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
☑	graphics	The R Graphics Package	3.2.3
☑	grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
☐	grid	The Grid Graphics Package	3.2.3
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
☐	lattice	Trellis Graphics for R	0.20-33
☐	MASS	Support Functions and Datasets for	7.3-45

Let’s load the MASS package

(just click on it)



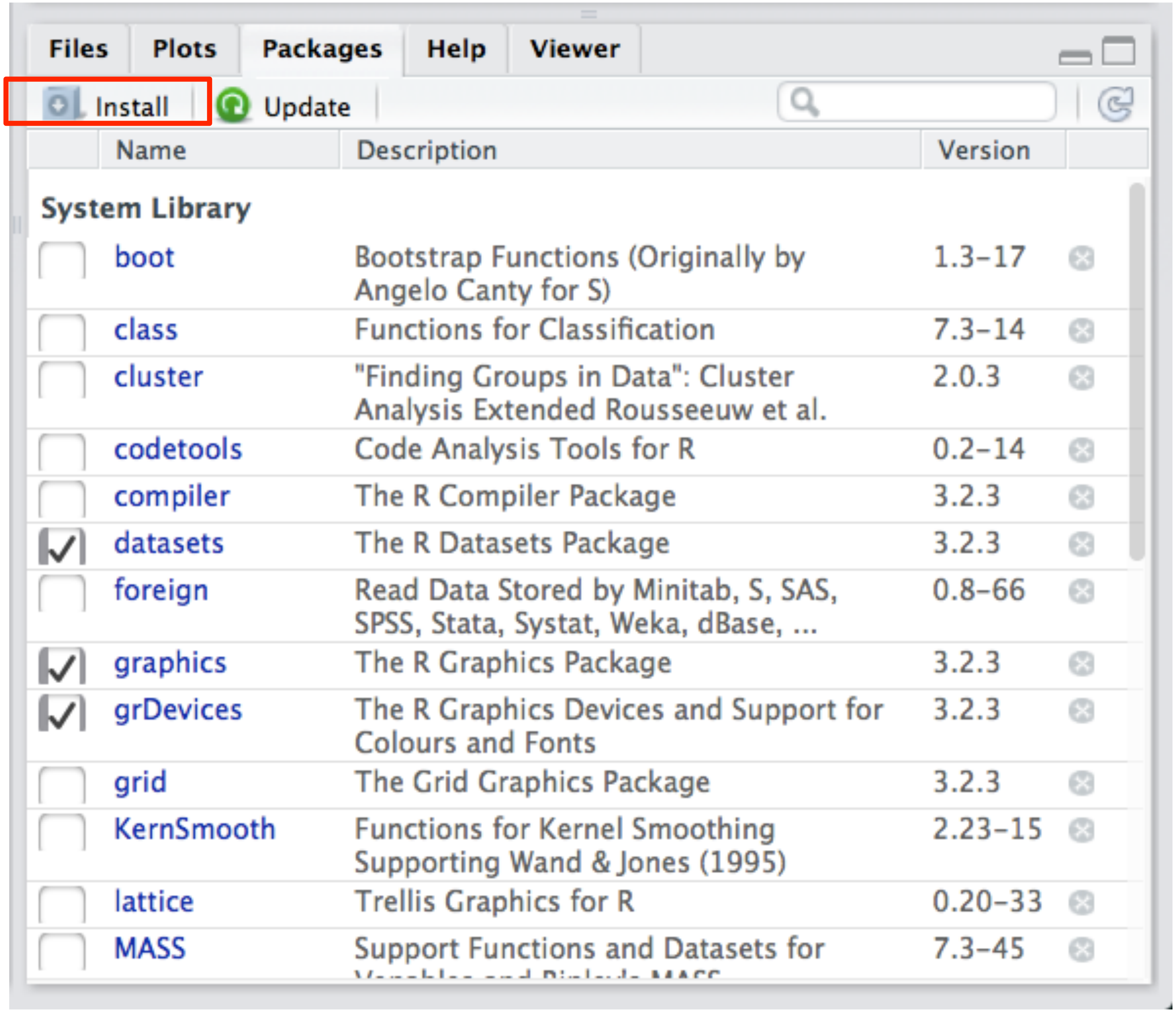
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15	⊗
☐	lattice	Trellis Graphics for R	0.20-33	⊗
☑	MASS	Support Functions and Datasets for Venables and Ripley's MASS	7.3-45	⊗
☐	Matrix	Sparse and Dense Matrix Classes and Methods	1.2-3	⊗

```
> library("MASS", lib.loc="/Library/Frameworks/R.framework/Versions
/3.2/Resources/library")
```

This command appears in the R console automatically: this is the “real” way that the package gets loaded. The Rstudio package panel is just a user-friendly way of producing this command. You could also load the package by typing it in the console, but that’s a lot harder.

Installing packages

Click here



System Library

	Name	Description	Version
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☐	class	Functions for Classification	7.3-14
☐	cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
☐	codetools	Code Analysis Tools for R	0.2-14
☐	compiler	The R Compiler Package	3.2.3
☑	datasets	The R Datasets Package	3.2.3
☐	foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
☑	graphics	The R Graphics Package	3.2.3
☑	grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
☐	grid	The Grid Graphics Package	3.2.3
☐	KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
☐	lattice	Trellis Graphics for R	0.20-33
☐	MASS	Support Functions and Datasets for	7.3-45

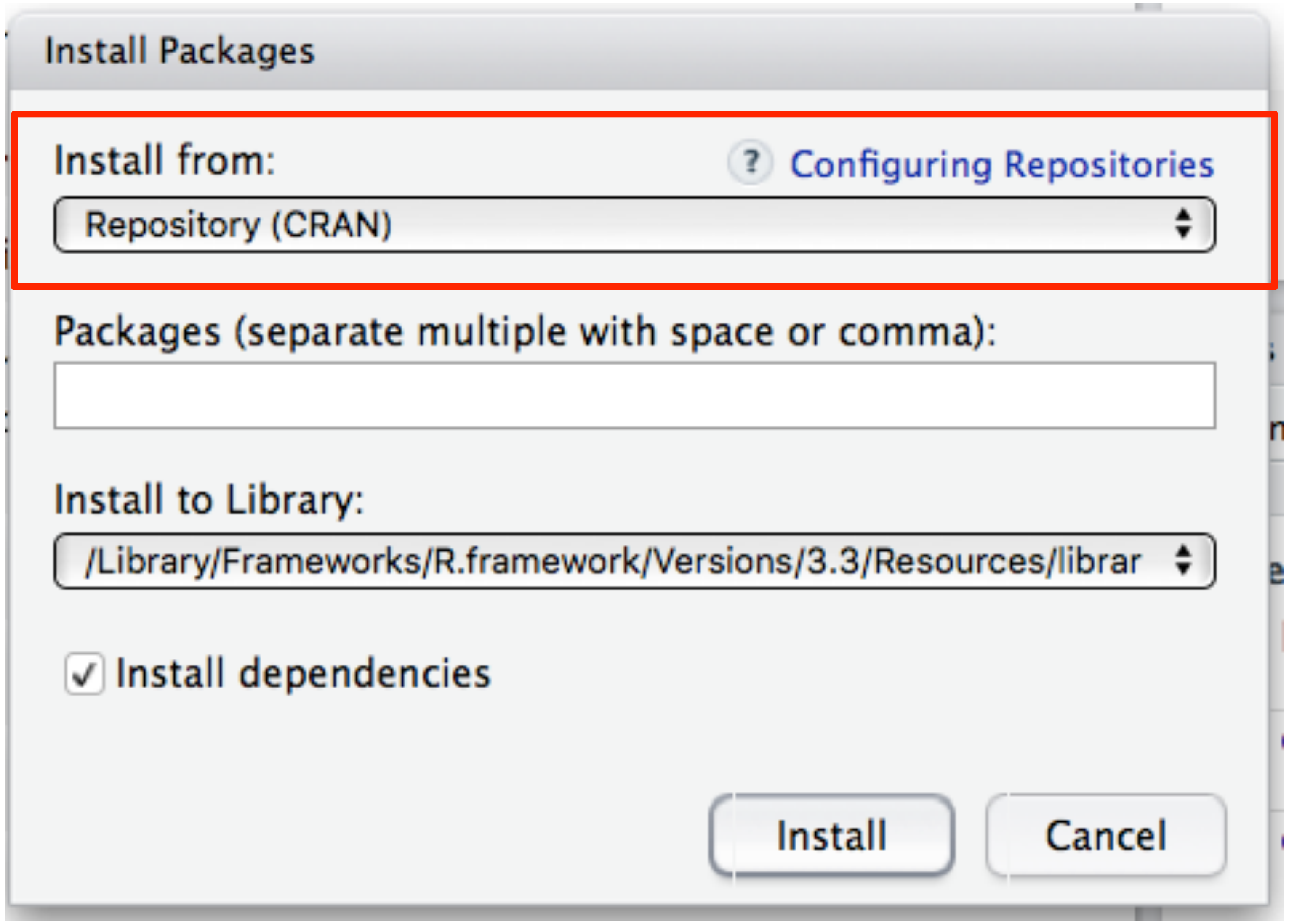
You'll note that this list doesn't have 5000 packages in it

What if you want one that isn't in it?

Installing packages

Where to install it from?

(ignore this: default is fine)



Install Packages

Install from: Repository (CRAN)

Packages (separate multiple with space or comma):

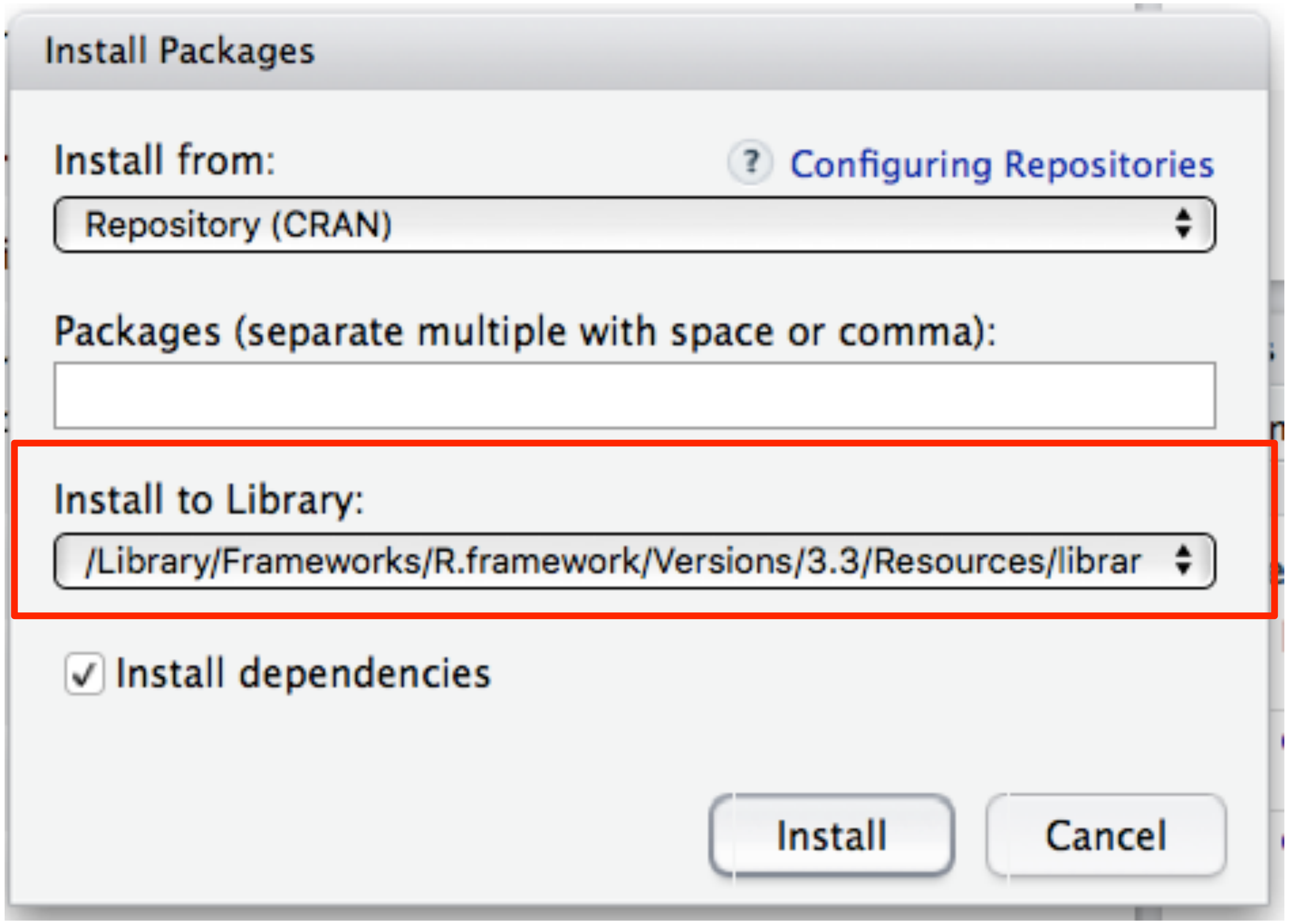
Install to Library: /Library/Frameworks/R.framework/Versions/3.3/Resources/librar

☑ Install dependencies

Installing packages

Where should packages be stored?

(default is also fine)



Install Packages

Install from: Repository (CRAN)

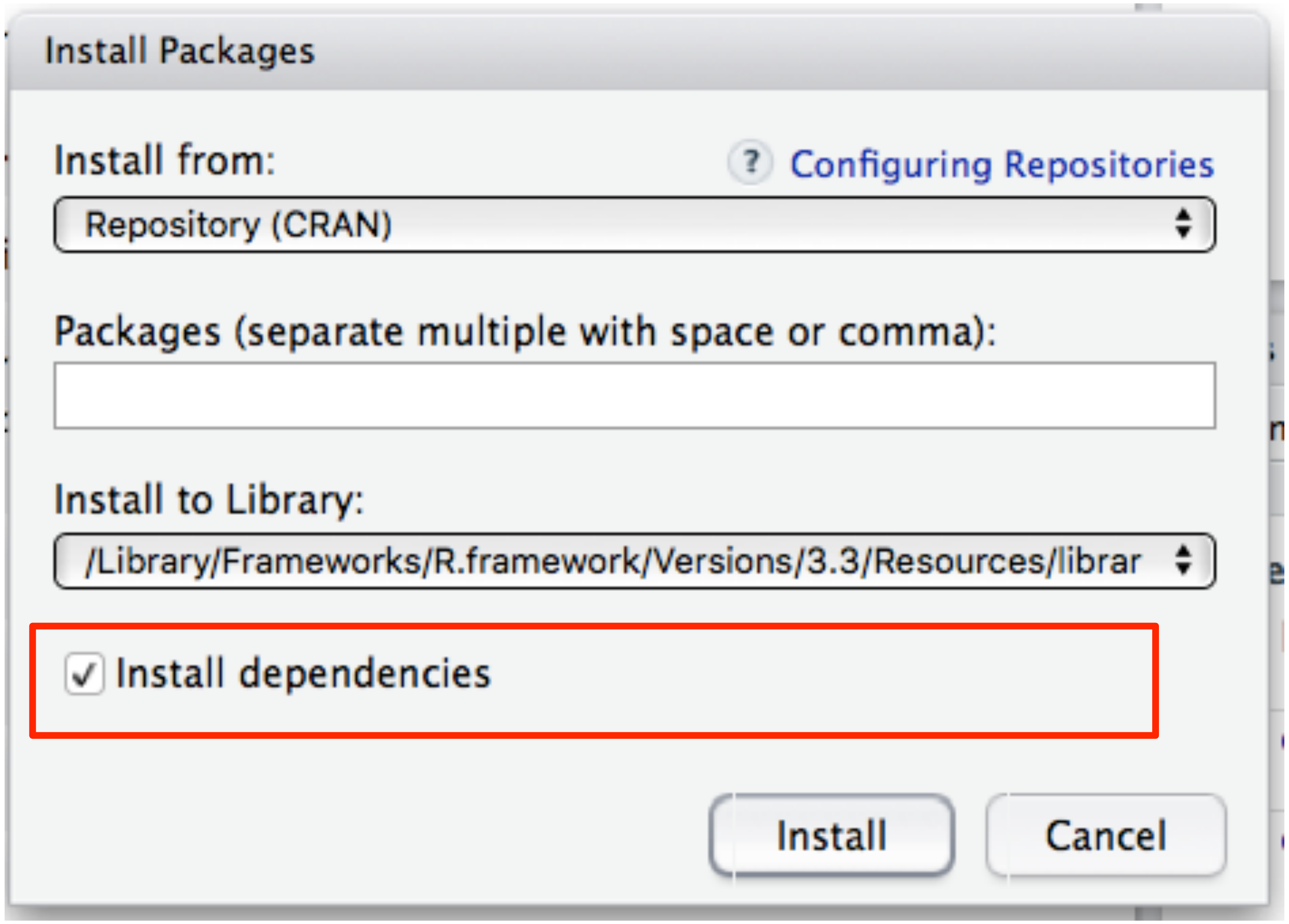
Packages (separate multiple with space or comma):

Install to Library: /Library/Frameworks/R.framework/Versions/3.3/Resources/librar

☑ Install dependencies

Installing packages

Should dependencies be installed? Leave this checked, because the answer is almost always “yes”



Repository (CRAN)

Packages (separate multiple with space or comma):

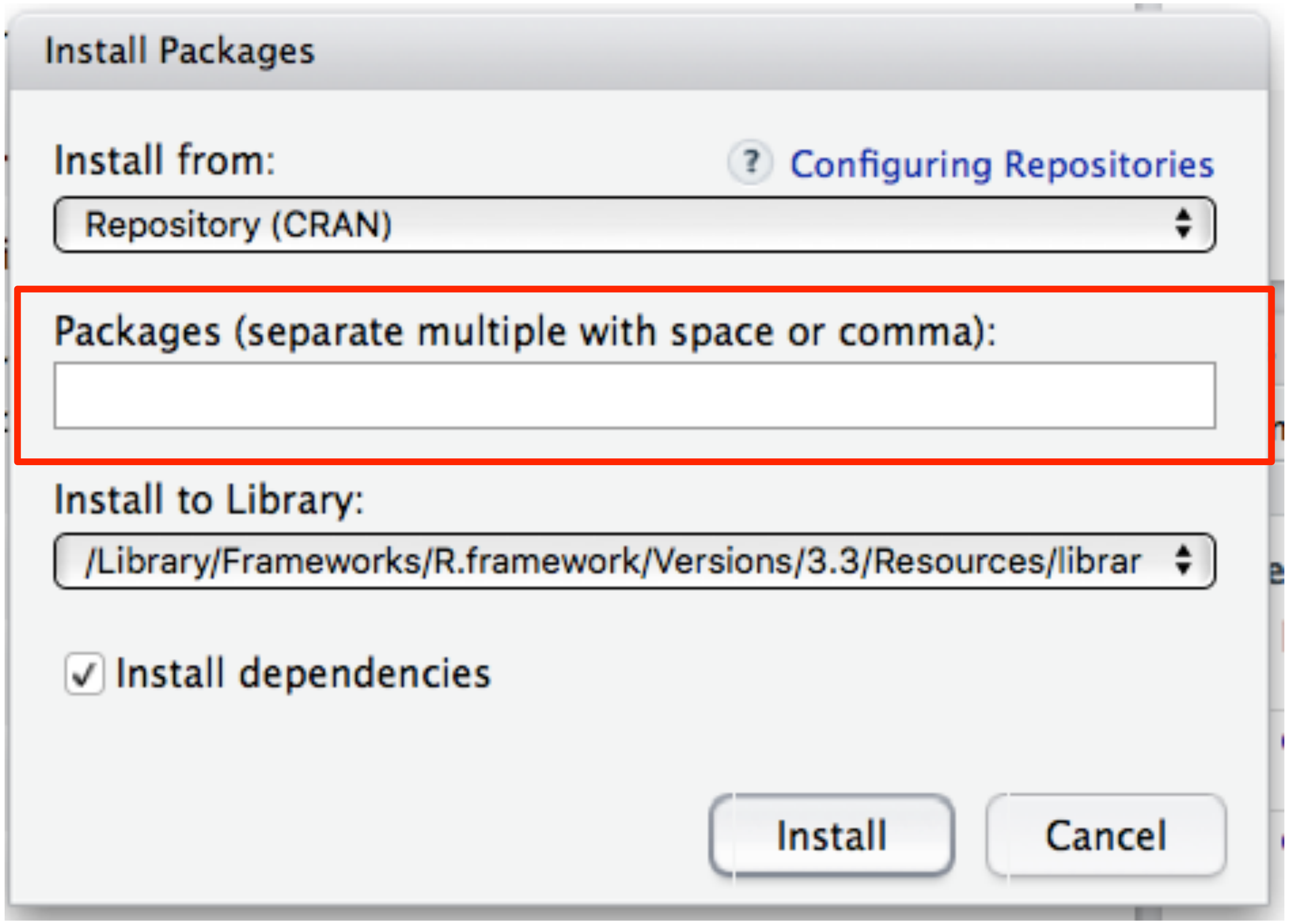
Install to Library:

/Library/Frameworks/R.framework/Versions/3.3/Resources/librar

☑ Install dependencies

Installing packages

Which packages to install? <u>This</u> is the important bit!



Install Packages

Install from: Repository (CRAN)

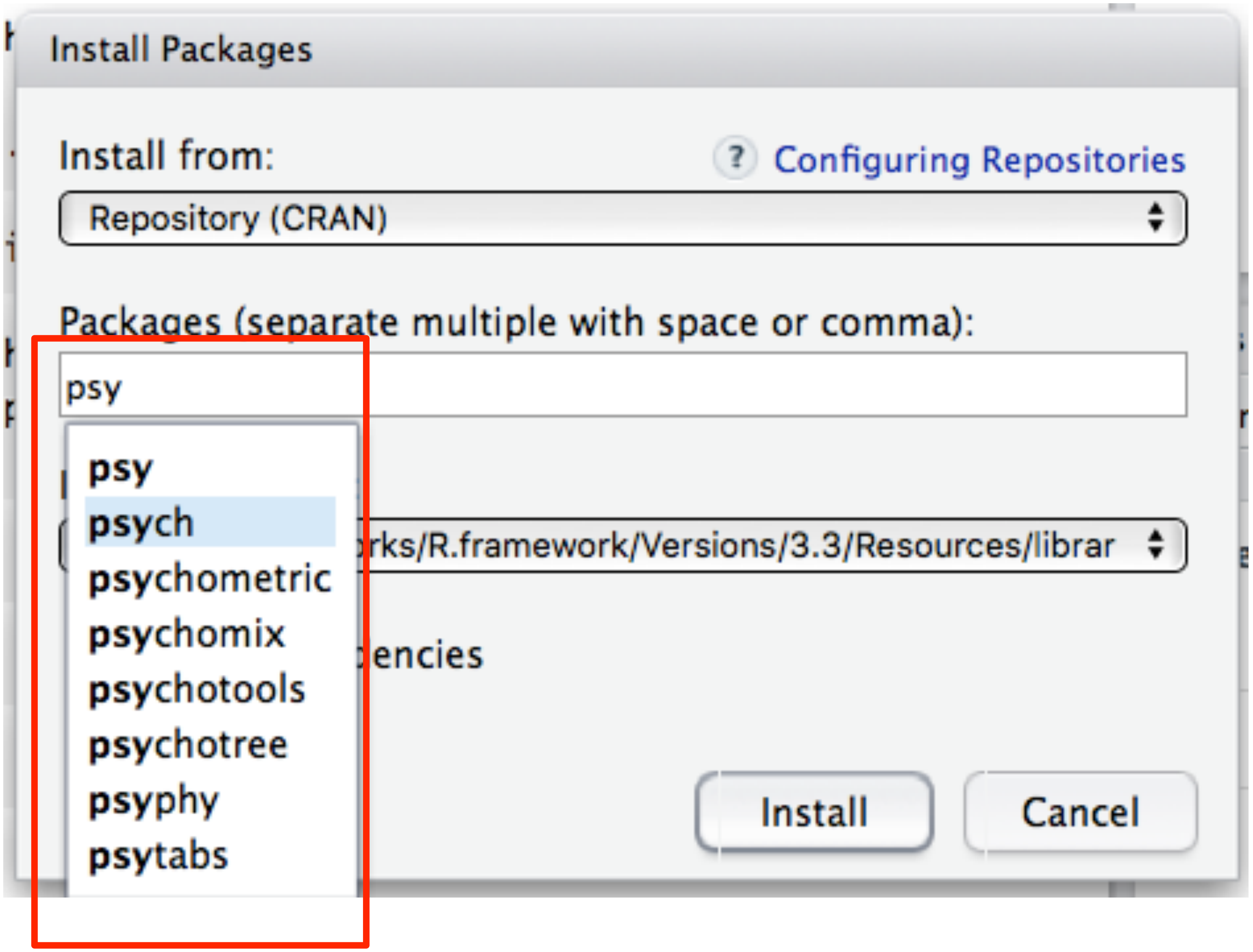
Packages (separate multiple with space or comma):

Install to Library: /Library/Frameworks/R.framework/Versions/3.3/Resources/librar

☑ Install dependencies

Installing packages

Start typing… and Rstudio gives you a list of possible packages



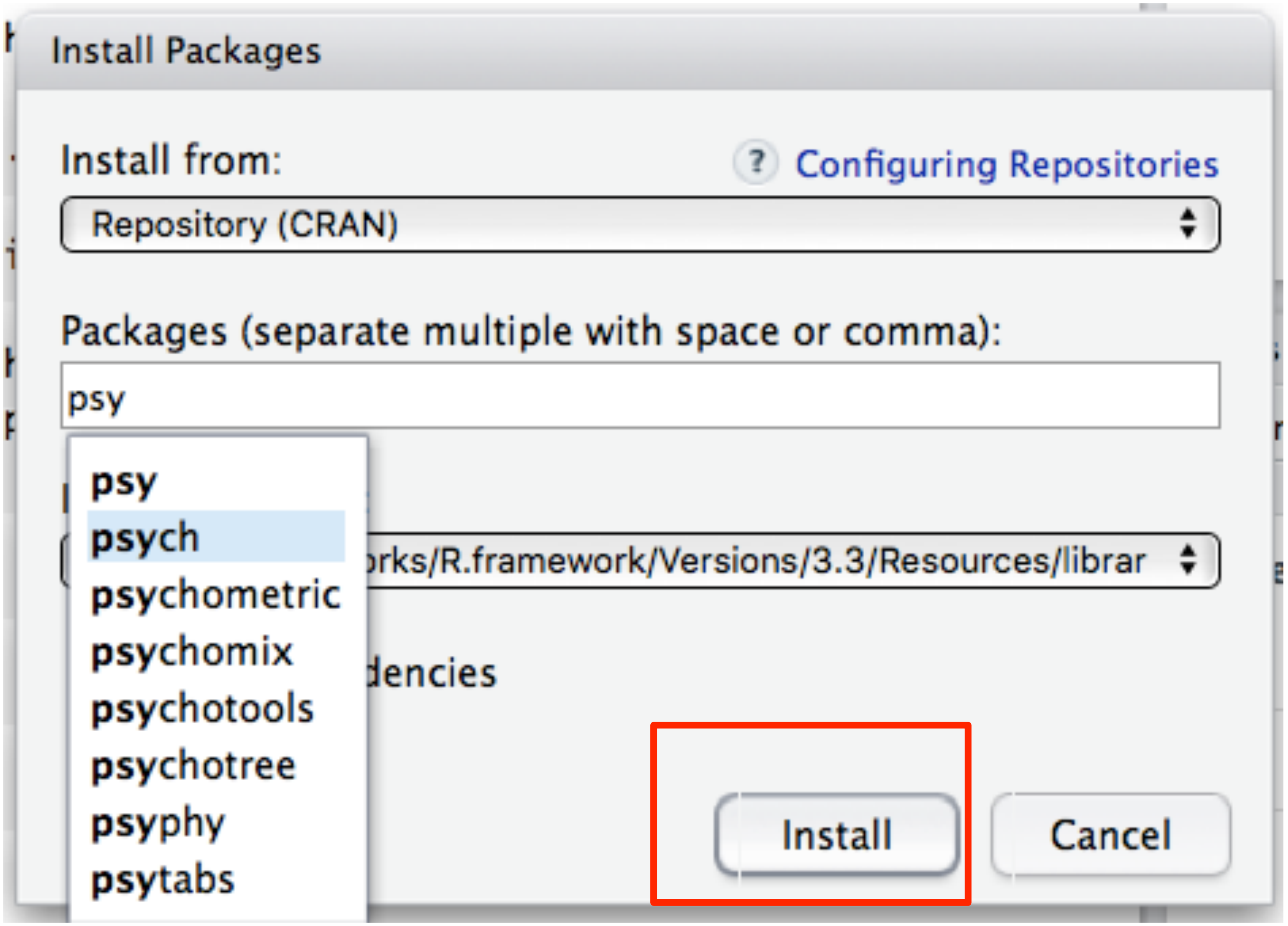
Install Packages

Install from: Repository (CRAN)

Packages (separate multiple with space or comma): psy

rks/R.framework/Versions/3.3/Resources/librar

Installing packages



Click “install” once
you’ve typed the name
of the package you want

Here’s what happens…

```
> install.packages("psych")
```

This is the command that
appears in the R console

Here’s what happens…

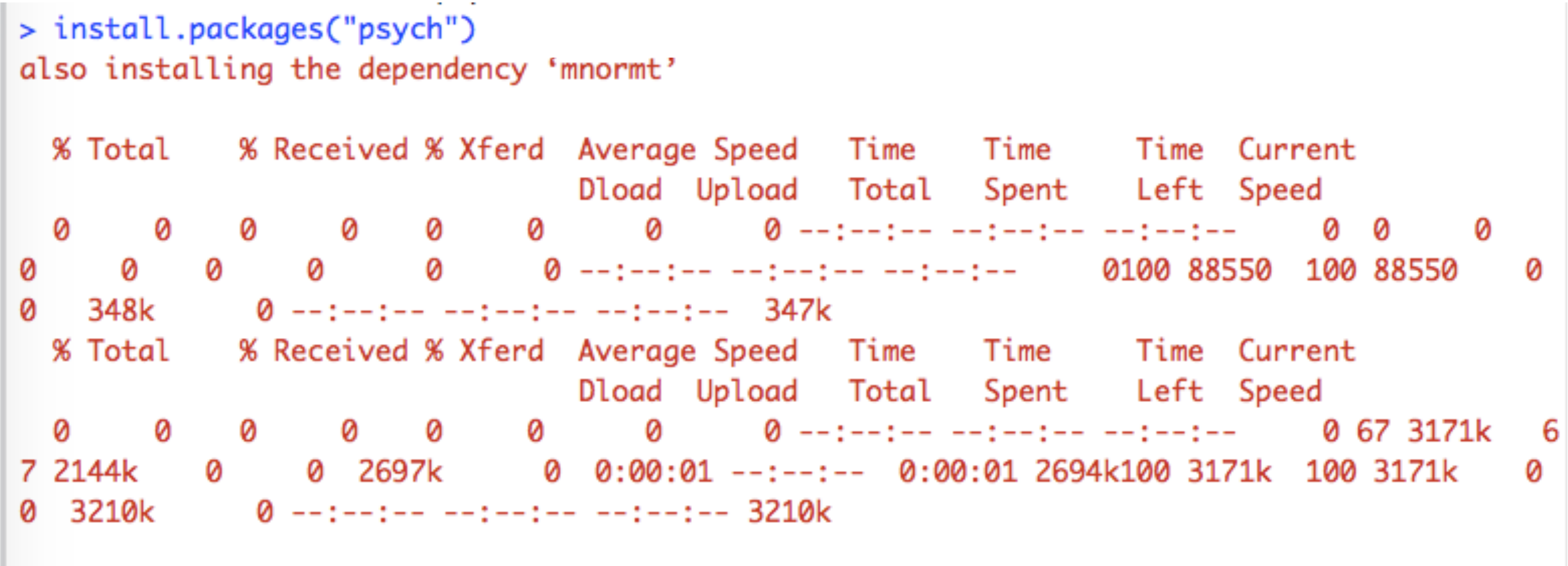
```
> install.packages("psych")
also installing the dependency ‘mnormt’
```

R keeps track of “dependencies”

Some packages rely on content of other packages. So if you try to load package A, but it requires content from package B (which you don’t have loaded), R will load package B too.

You generally don’t need to care about this.

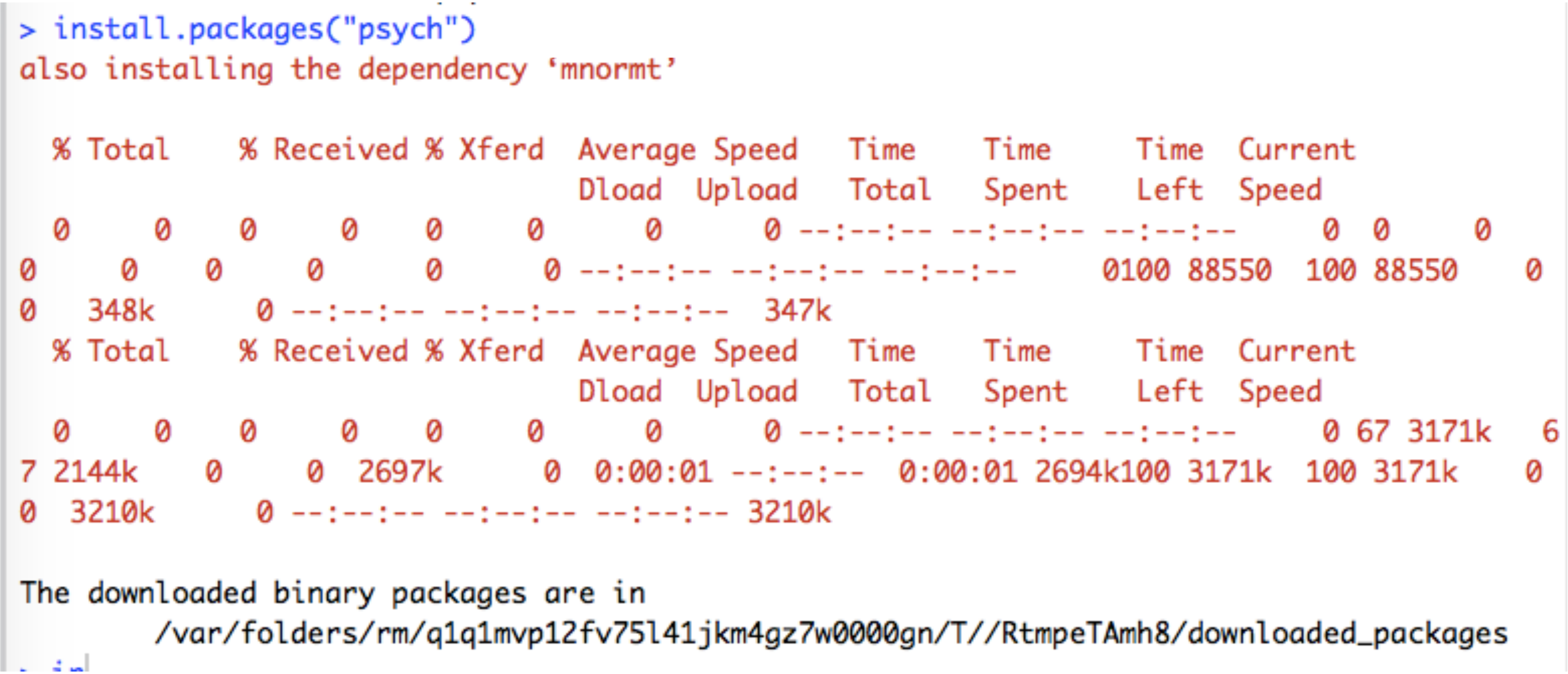
Here’s what happens…



```
> install.packages("psych")
also installing the dependency ‘mnormt’

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0  0     0
0     0    0     0      0     0 --:--:-- --:--:-- --:--:--     0100 88550  100 88550    0
0   348k      0 --:--:-- --:--:-- --:--:--  347k
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0 67 3171k    6
7 2144k    0     0  2697k      0  0:00:01 --:--:--  0:00:01 2694k100 3171k  100 3171k    0
0  3210k      0 --:--:-- --:--:-- --:--:-- 3210k
```

This blahdiblah means it is currently downloading successfully…

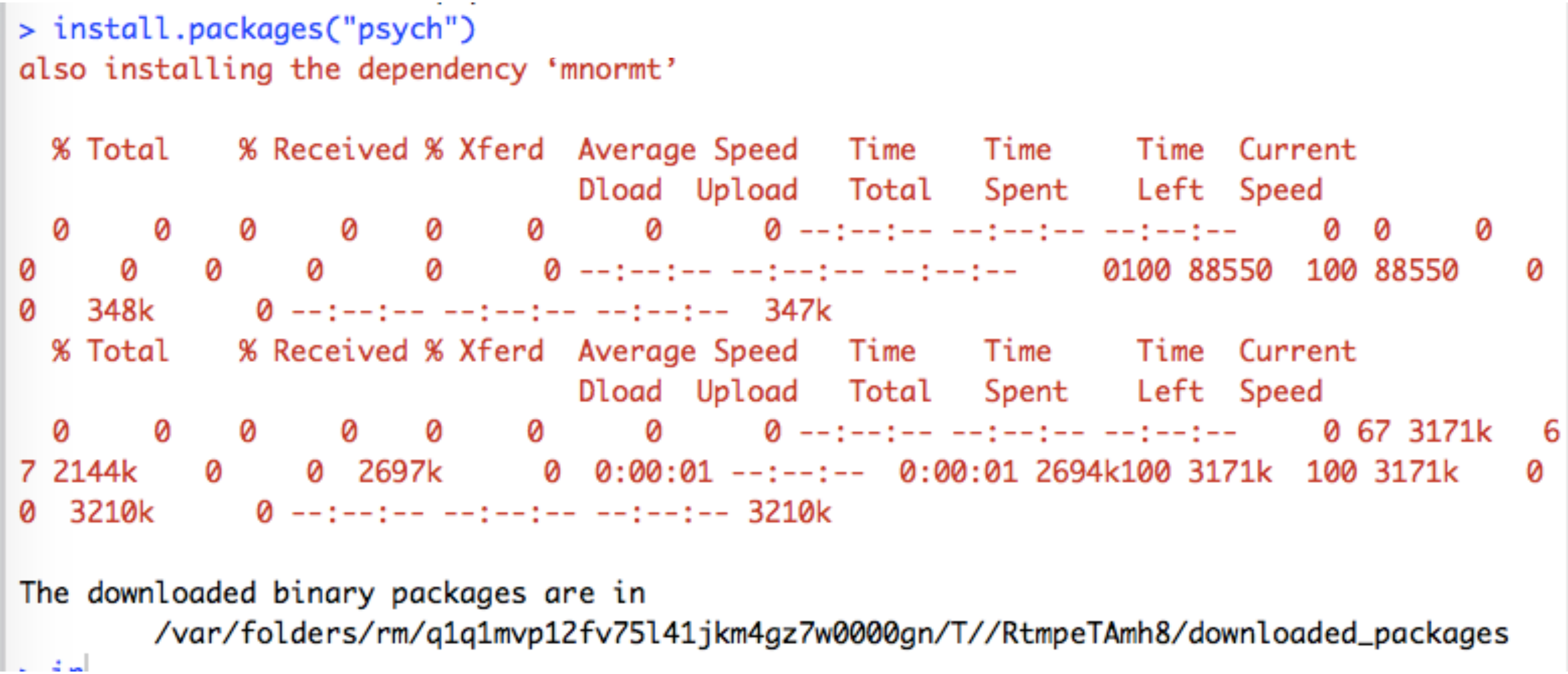
Here’s what happens…



```
> install.packages("psych")
also installing the dependency ‘mnormt’

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0  0      0
0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0100 88550  100 88550    0
0   348k      0 --:--:-- --:--:-- --:--:--  347k
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0 67 3171k    6
7 2144k    0     0  2697k      0  0:00:01 --:--:--  0:00:01 2694k100 3171k  100 3171k    0
0  3210k      0 --:--:-- --:--:-- --:--:-- 3210k

The downloaded binary packages are in
	/var/folders/rm/q1q1mvp12fv75l41jkm4gz7w0000gn/T//RtmpeTAmh8/downloaded_packages
```

This last bit tells you where it is being stored temporarily

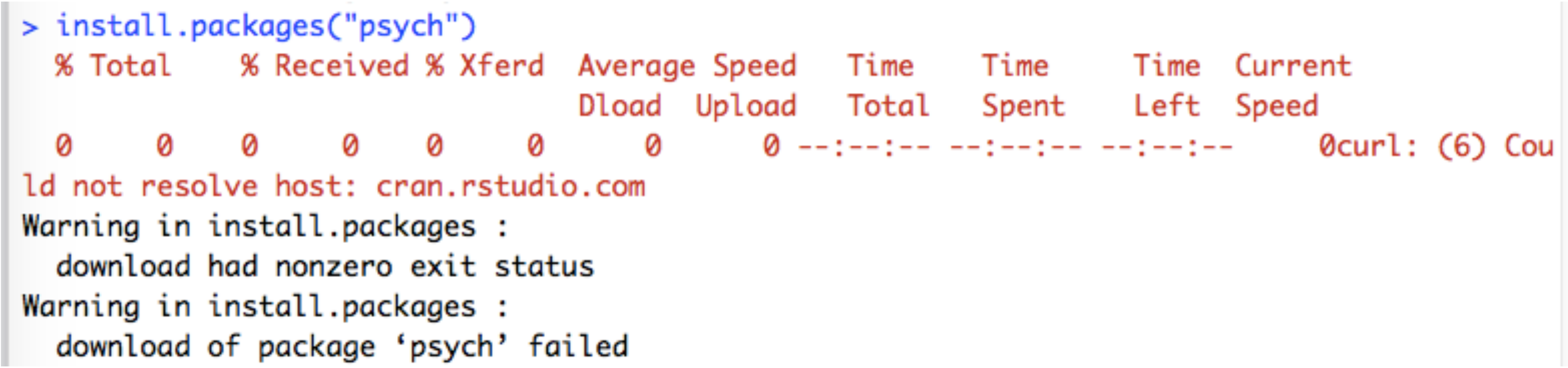
Here’s what happens…



```
> install.packages("psych")
also installing the dependency ‘mnormt’

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0  0     0
0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0100 88550  100 88550    0
0   348k         0 --:--:-- --:--:-- --:--:--  347k
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0 67 3171k    6
7 2144k    0     0  2697k      0  0:00:01 --:--:--  0:00:01 2694k100 3171k  100 3171k    0
0  3210k         0 --:--:-- --:--:-- --:--:-- 3210k

The downloaded binary packages are in
        /var/folders/rm/q1q1mvp12fv75l41jkm4gz7w0000gn/T//RtmpeTAmh8/downloaded_packages
```

The only thing you really need to care about is… do you see some output that looks like this? If yes, all is well. If you get something else, you might have a problem

A common problem…



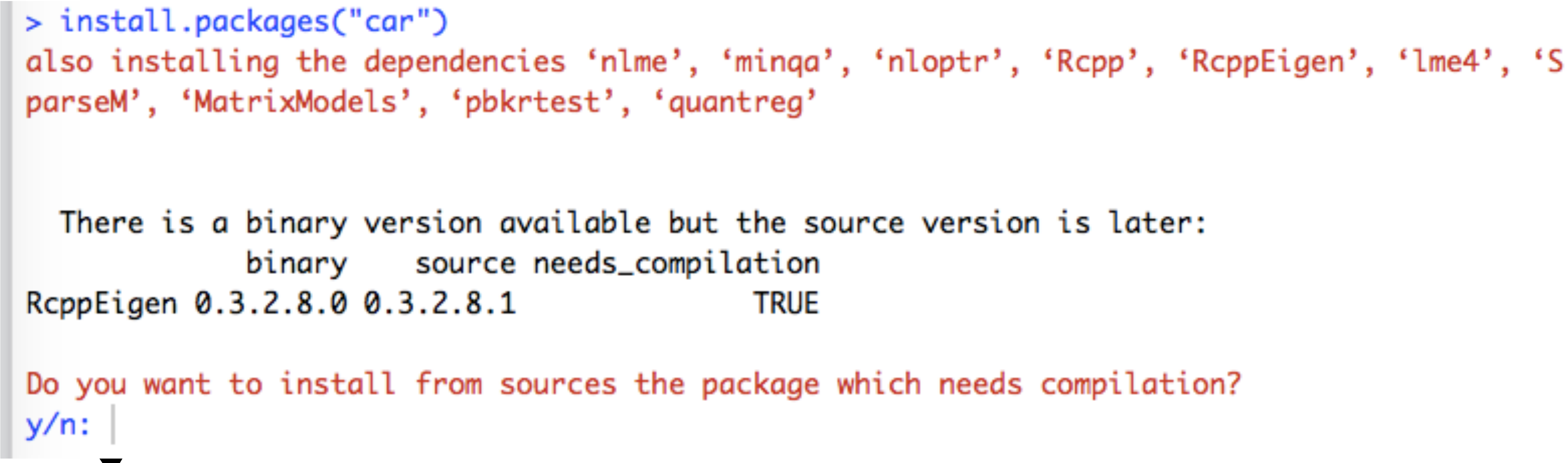
```
> install.packages("psych")
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0curl: (6) Cou
ld not resolve host: cran.rstudio.com
Warning in install.packages :
  download had nonzero exit status
Warning in install.packages :
  download of package ‘psych’ failed
```

This means that R can’t access the internet. The most common reasons are (a) your internet connection isn’t on! (b) your firewall or antivirus software is blocking R.

Packages you should install

- lsr - that's the package Dani wrote!
- psych - it has some useful tools for psychologists
- car - it's handy for lots of things

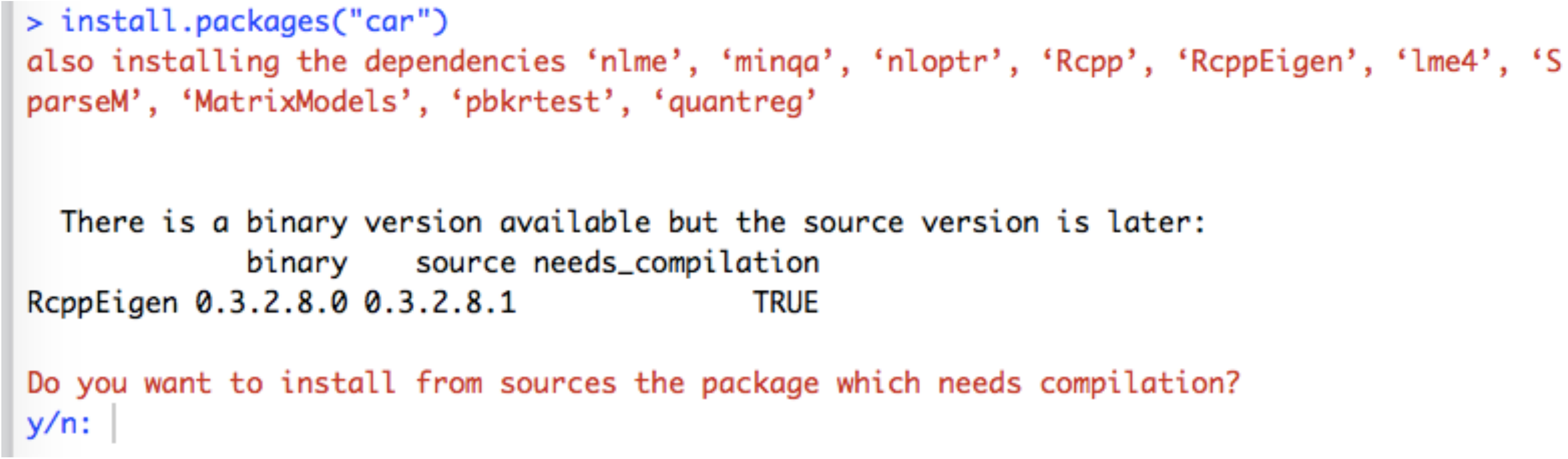
Note re: car() package



```
> install.packages("car")
also installing the dependencies ‘nlme’, ‘minqa’, ‘nloptr’, ‘Rcpp’, ‘RcppEigen’, ‘lme4’, ‘S
parseM’, ‘MatrixModels’, ‘pbkrtest’, ‘quantreg’


  There is a binary version available but the source version is later:
             binary    source needs_compilation
RcppEigen 0.3.2.8.0 0.3.2.8.1              TRUE

Do you want to install from sources the package which needs compilation?
y/n: 
```

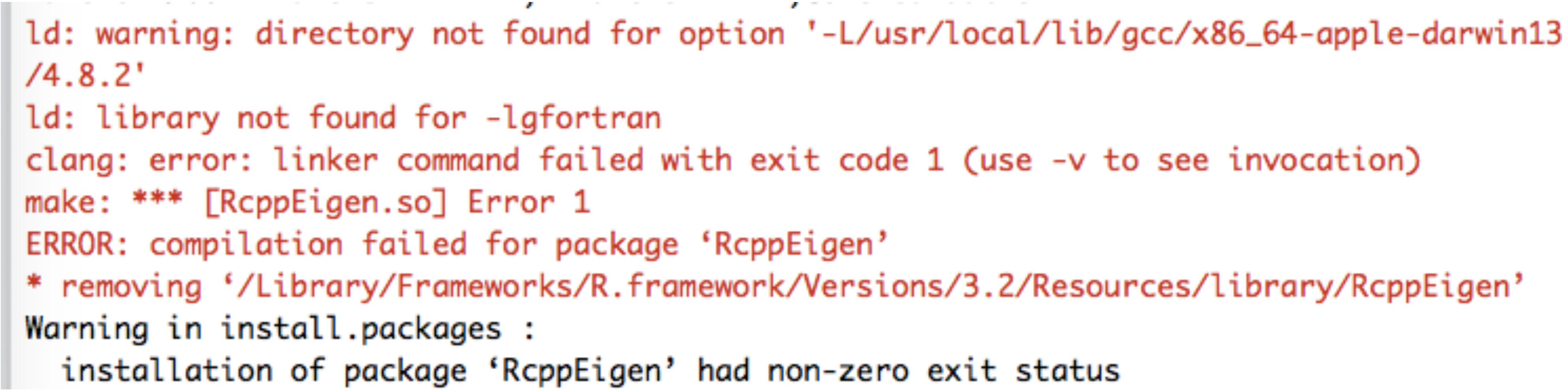
say “y”

Note re: car() package



```
> install.packages("car")
also installing the dependencies ‘nlme’, ‘minqa’, ‘nloptr’, ‘Rcpp’, ‘RcppEigen’, ‘lme4’, ‘S
parseM’, ‘MatrixModels’, ‘pbkrtest’, ‘quantreg’


  There is a binary version available but the source version is later:
           binary    source needs_compilation
RcppEigen 0.3.2.8.0 0.3.2.8.1              TRUE

Do you want to install from sources the package which needs compilation?
y/n: 
```

if you get this, don’t worry - it won’t affect anything we need for this class



```
ld: warning: directory not found for option '-L/usr/local/lib/gcc/x86_64-apple-darwin13
/4.8.2'
ld: library not found for -lgfortran
clang: error: linker command failed with exit code 1 (use -v to see invocation)
make: *** [RcppEigen.so] Error 1
ERROR: compilation failed for package ‘RcppEigen’
* removing ‘/Library/Frameworks/R.framework/Versions/3.2/Resources/library/RcppEigen’
Warning in install.packages :
  installation of package ‘RcppEigen’ had non-zero exit status
```

So far you've just **installed** the packages (they're on your computer but R is not currently using them)

Now you have to **load** them

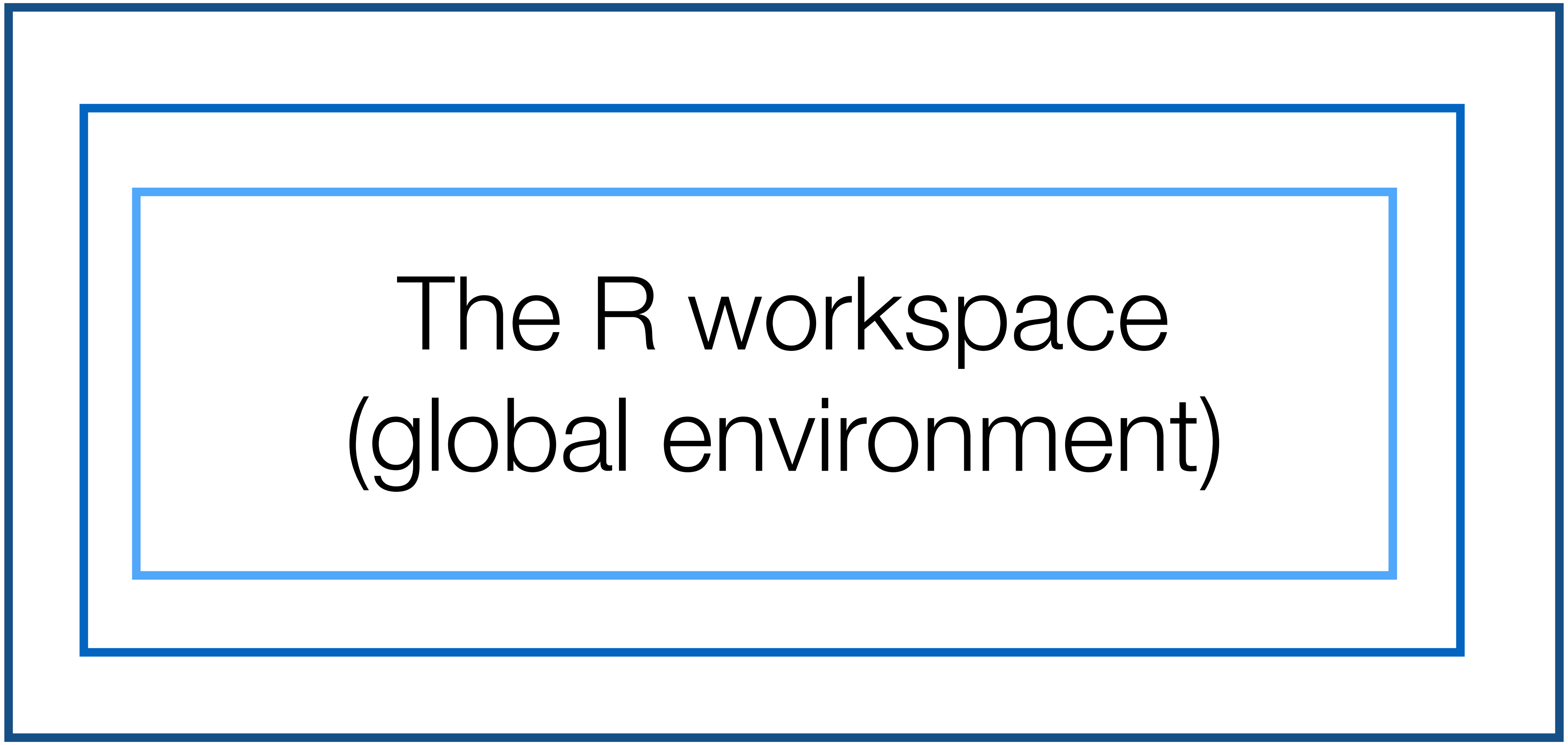
Conflicts between packages?

```
> library( psych )
> library( car )

Attaching package: ‘car’

The following object is masked from ‘package:psych’:

    logit
```

psych and car both contain a function called logit(). When I load both packages, the more recently loaded one (car) takes precedence...

This is the warning message that R prints out.

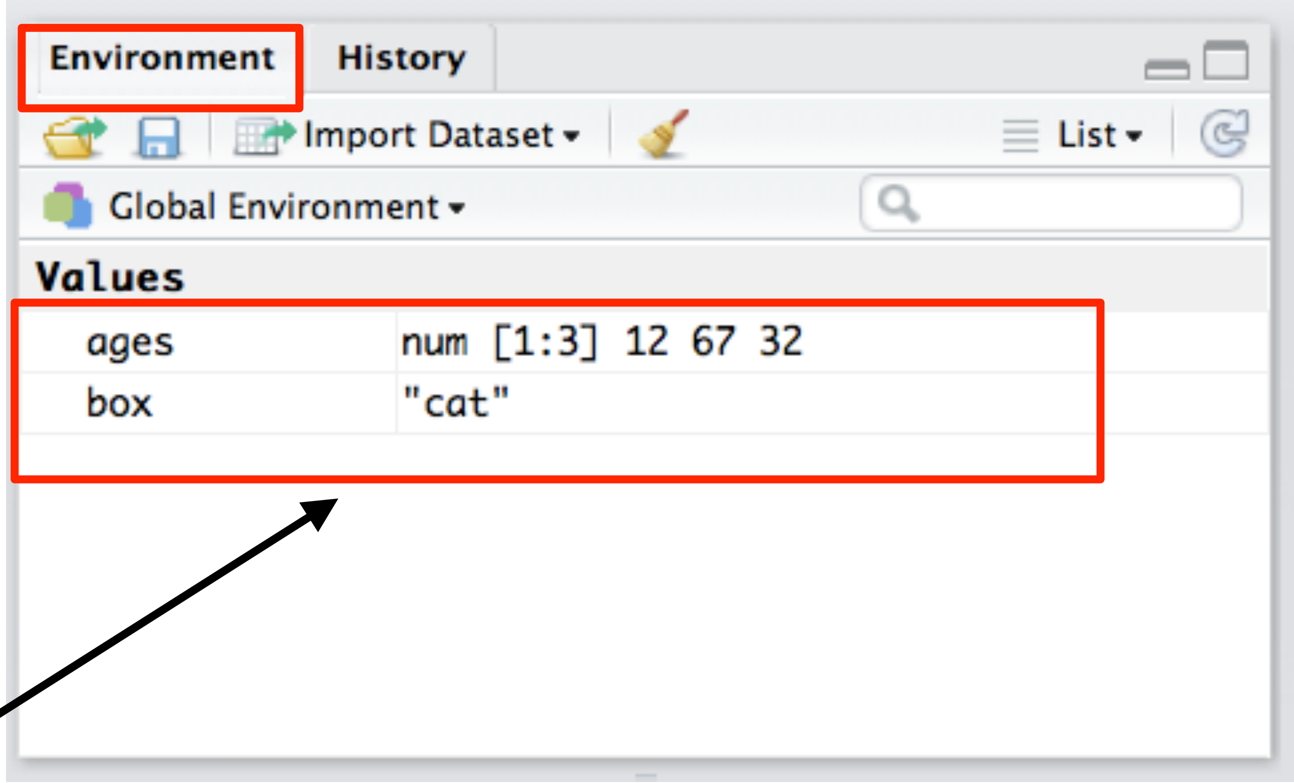
It says that “logit” exists in both packages... and that the version in “psych” is “masked” (i.e., you can’t access it)



The R workspace
(global environment)

The Rstudio “environment” panel

The Rstudio environment panel lists information about the variables that you’ve created (or loaded)

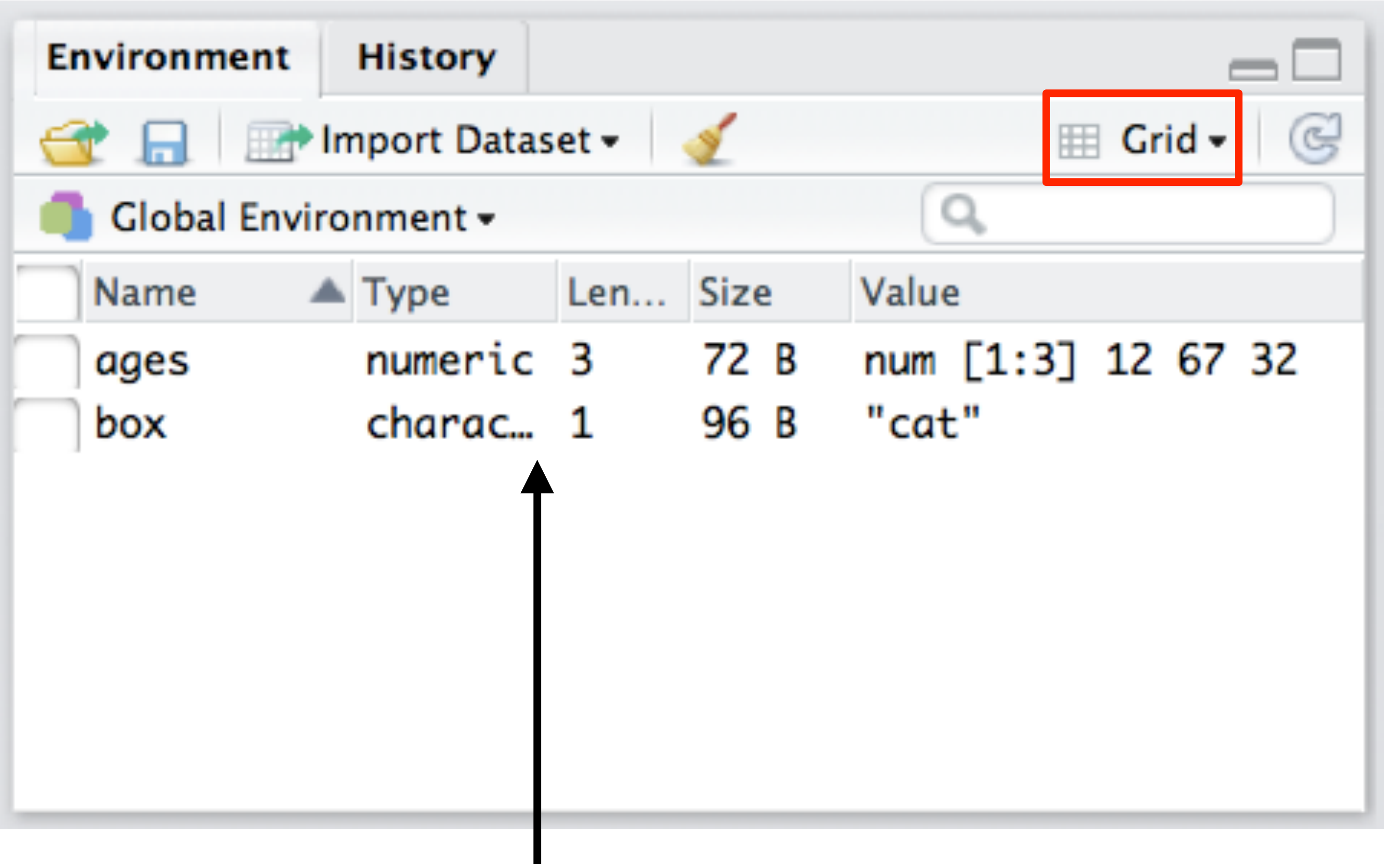


```
Values
   ages        num [1:3] 12 67 32
   box         "cat"
```

```
> box <- “cat”
> ages <- c(12,67,32)
```

When I create variables, they appear in the environment panel

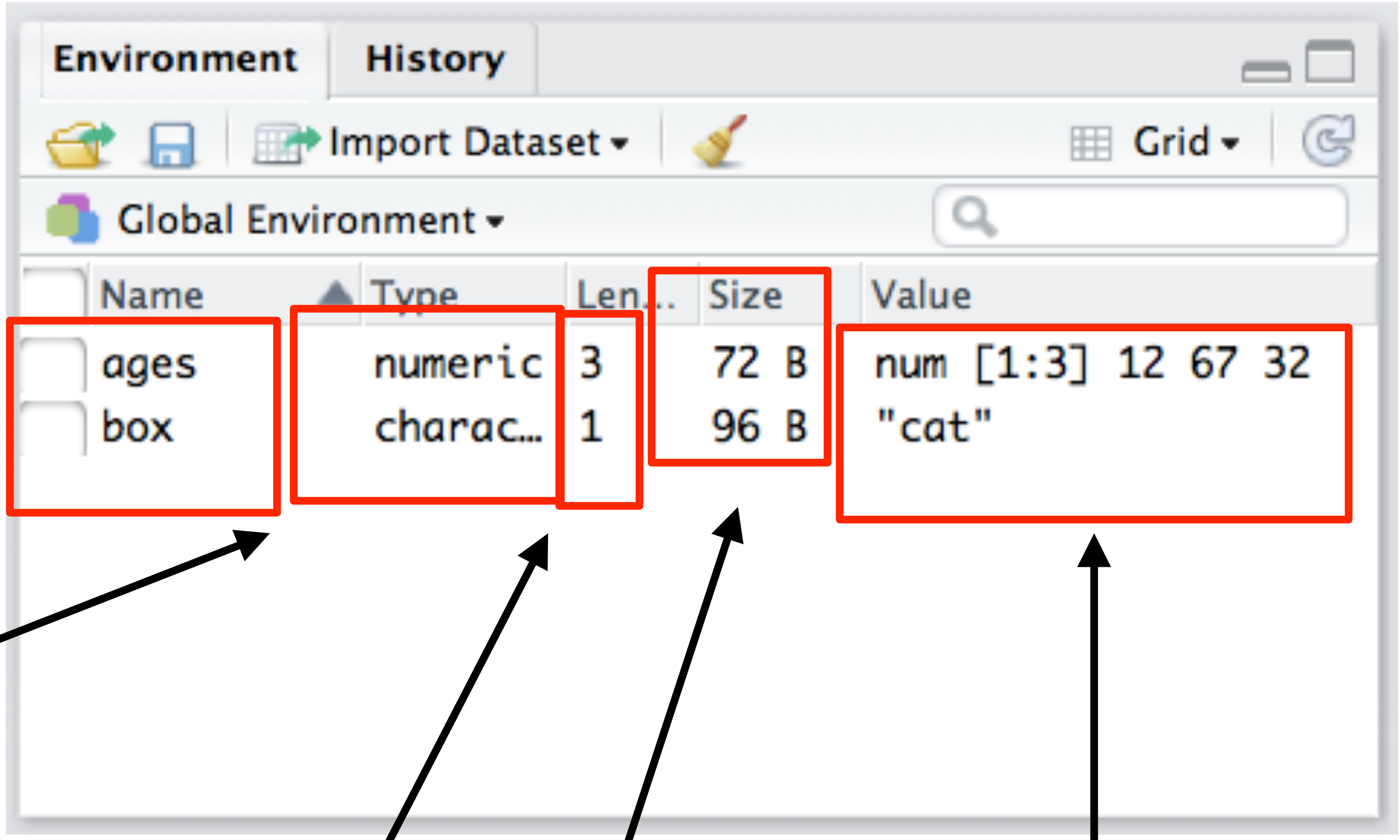
The Rstudio “environment” panel



Name	Type	Len...	Size	Value
ages	numeric	3	72 B	num [1:3] 12 67 32
box	charac...	1	96 B	"cat"

When I switch to “grid” view I see more information

The Rstudio “environment” panel



Name	Type	Len...	Size	Value
ages	numeric	3	72 B	num [1:3] 12 67 32
box	charac...	1	96 B	"cat"

Names of the variables

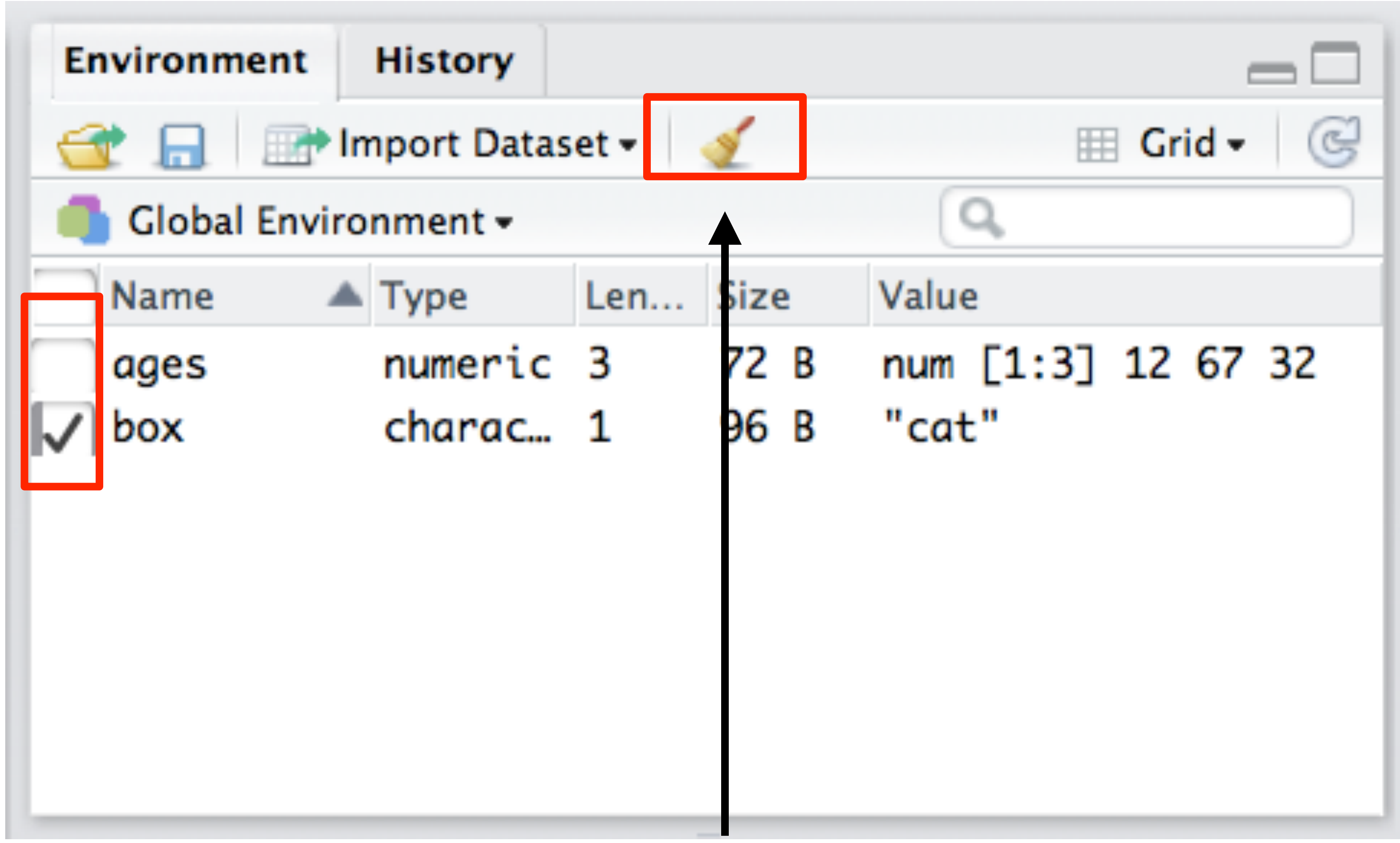
What kind of information is stored in this variable? (e.g., numeric)

How “long” is it? That is, how many elements does it have?

An attempt to summarise the information stored in the variable

How much of your computer’s memory does it take up?

Getting rid of variables?

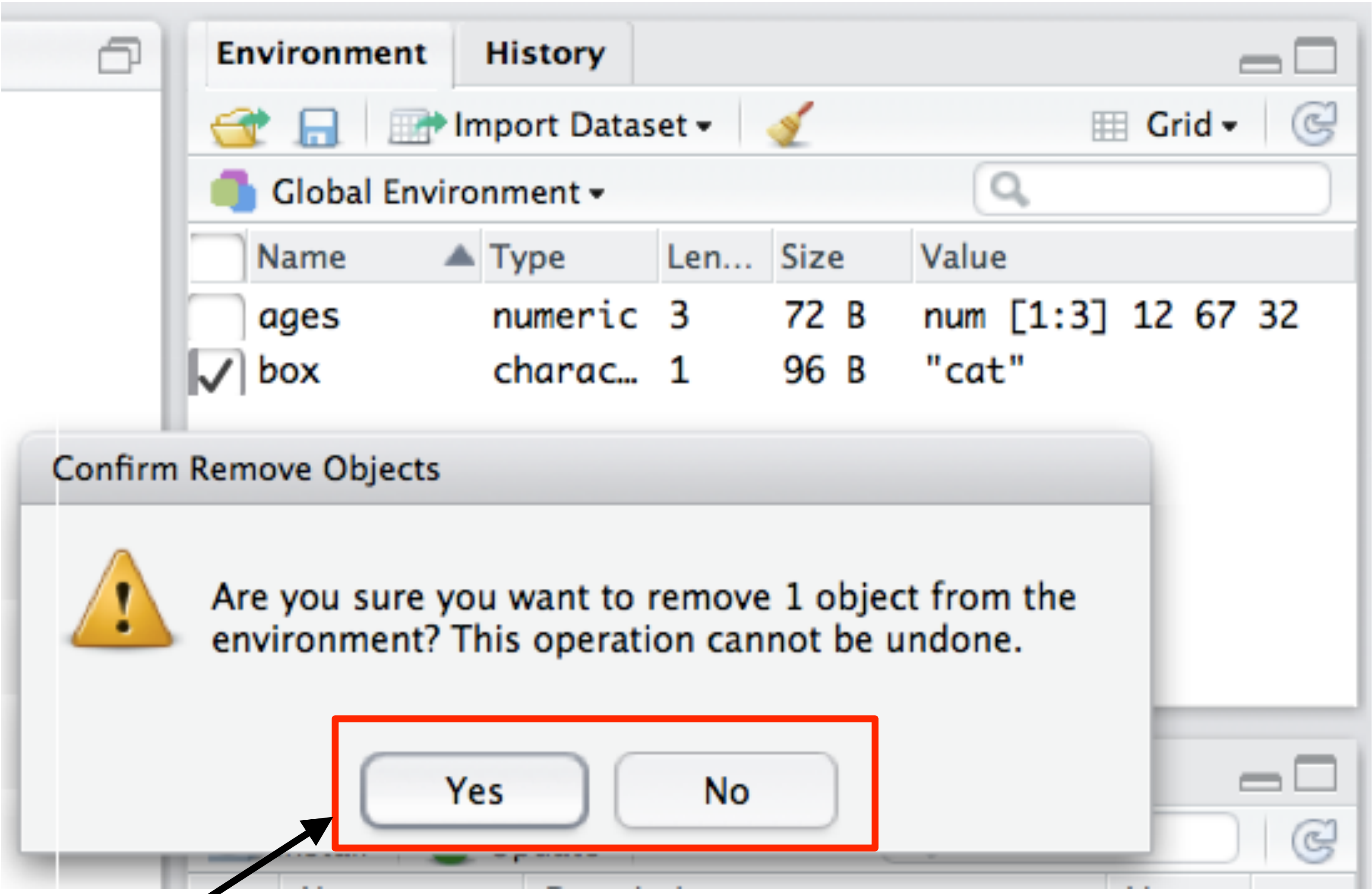


	Name	Type	Len...	Size	Value
☐	ages	numeric	3	72 B	num [1:3] 12 67 32
☑	box	charac...	1	96 B	"cat"

Click here to
select variables

Now click here to delete
the selected variable(s)

Getting rid of variables?



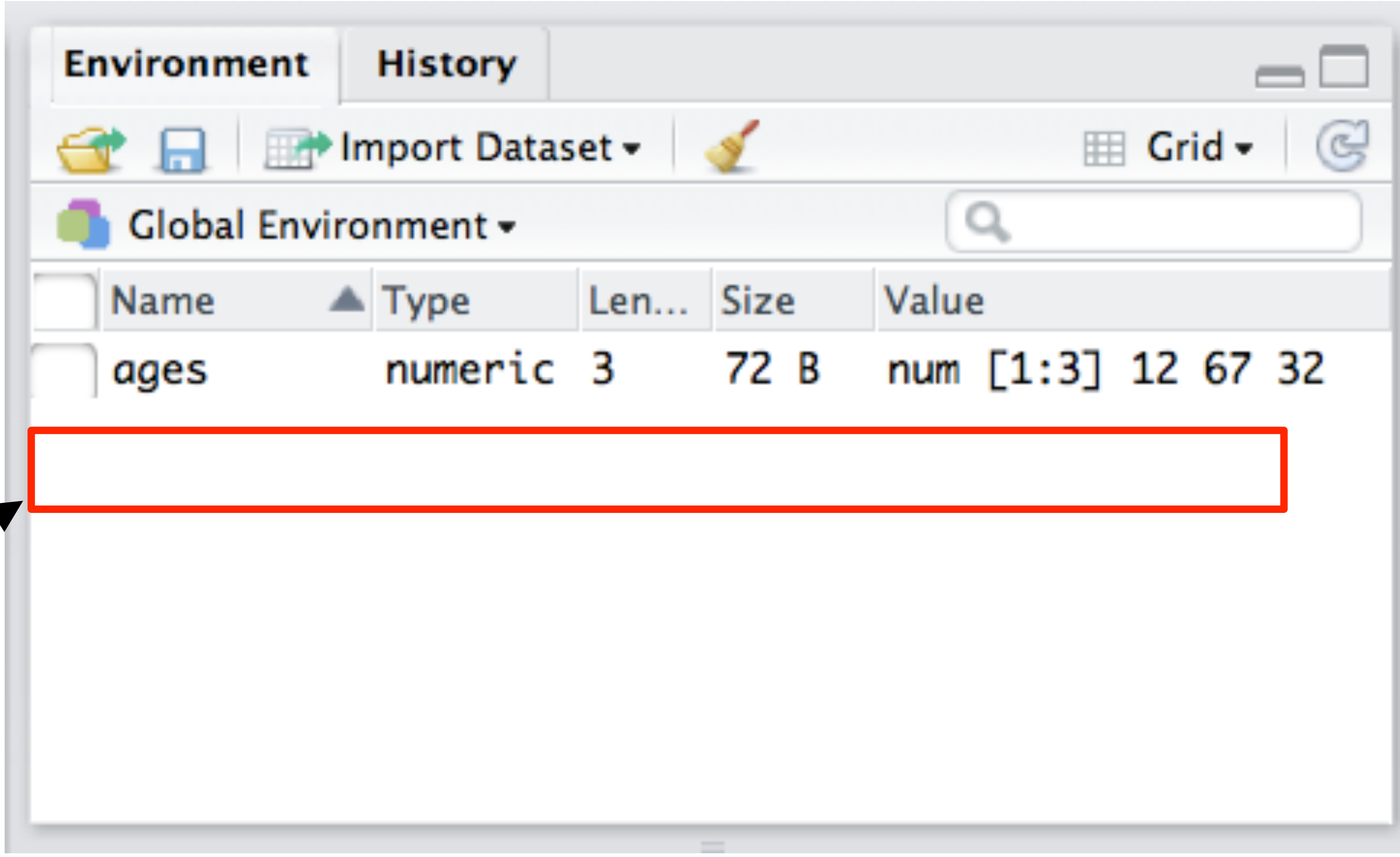
☐	Name	Type	Len...	Size	Value
☐	ages	numeric	3	72 B	num [1:3] 12 67 32
☑	box	charac...	1	96 B	"cat"

Confirm Remove Objects

Are you sure you want to remove 1 object from the environment? This operation cannot be undone.

Click yes to delete. Click no if you've made a mistake and you want to keep the variable!

Getting rid of variables?



Name	Type	Len...	Size	Value
ages	numeric	3	72 B	num [1:3] 12 67 32

The selected variable(s) are now gone. Unless you've got them saved somewhere, you can't get them back!

Doing it with R commands…

```
> box <- “cat”
> ages <- c(12,67,32)
```

Create the variables

```
> library(lsr)
> who()
   -- Name --    -- Class --    -- Size --
   ages          numeric        3
   box           character      1
```

Load the “lsr” package

The who( ) function in the lsr package lists the variables in a fairly readable way

```
> rm(ages)
> who()
   -- Name --    -- Class --    -- Size --
   box           character      1
```

The rm( ) function “removed” a variable

Use who( ) to confirm that it’s gone

Exercises

1. Make a variable called `myFavourite` with the name of your favourite food, and another called `ugh` with one of your least favourites. Use the command line to make sure they are in your workspace, and then to remove `ugh`.

2. Install and load the packages called `gplots` and `ggplot2`.

Loading a workspace file (i.e., an “Rdata” file)

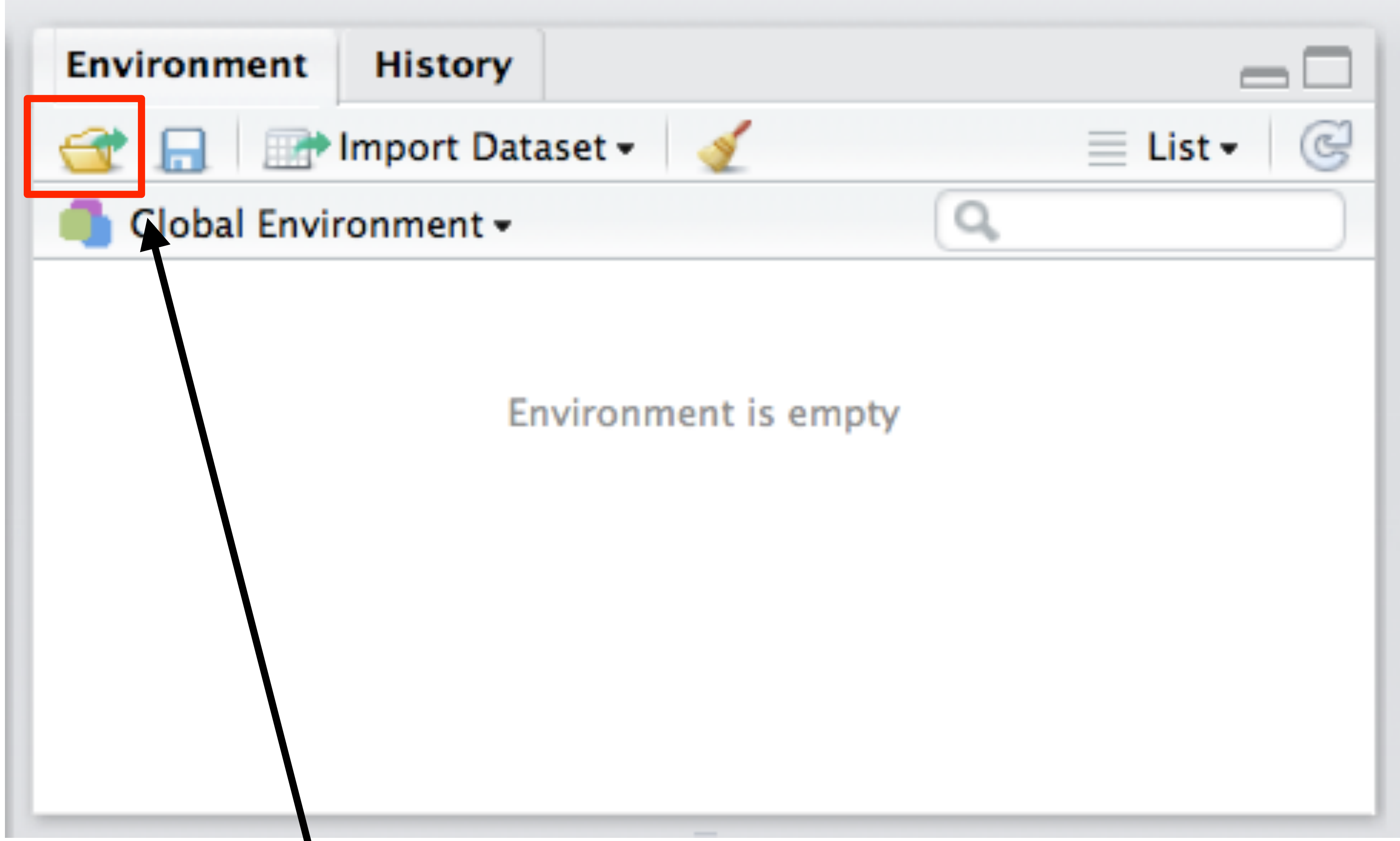
What does it <u>mean</u> to load data?

- Loading means:
  - You’ve copied the variables in a file into your R workspace
  - You can now use these variables for your analysis
- Changing the copy doesn’t change the original
  - The original stays in the file
  - Any changes/deletions you make only get saved if you choose to
- We’ll talk about saving shortly.
- But first, let’s load….

Workspace files

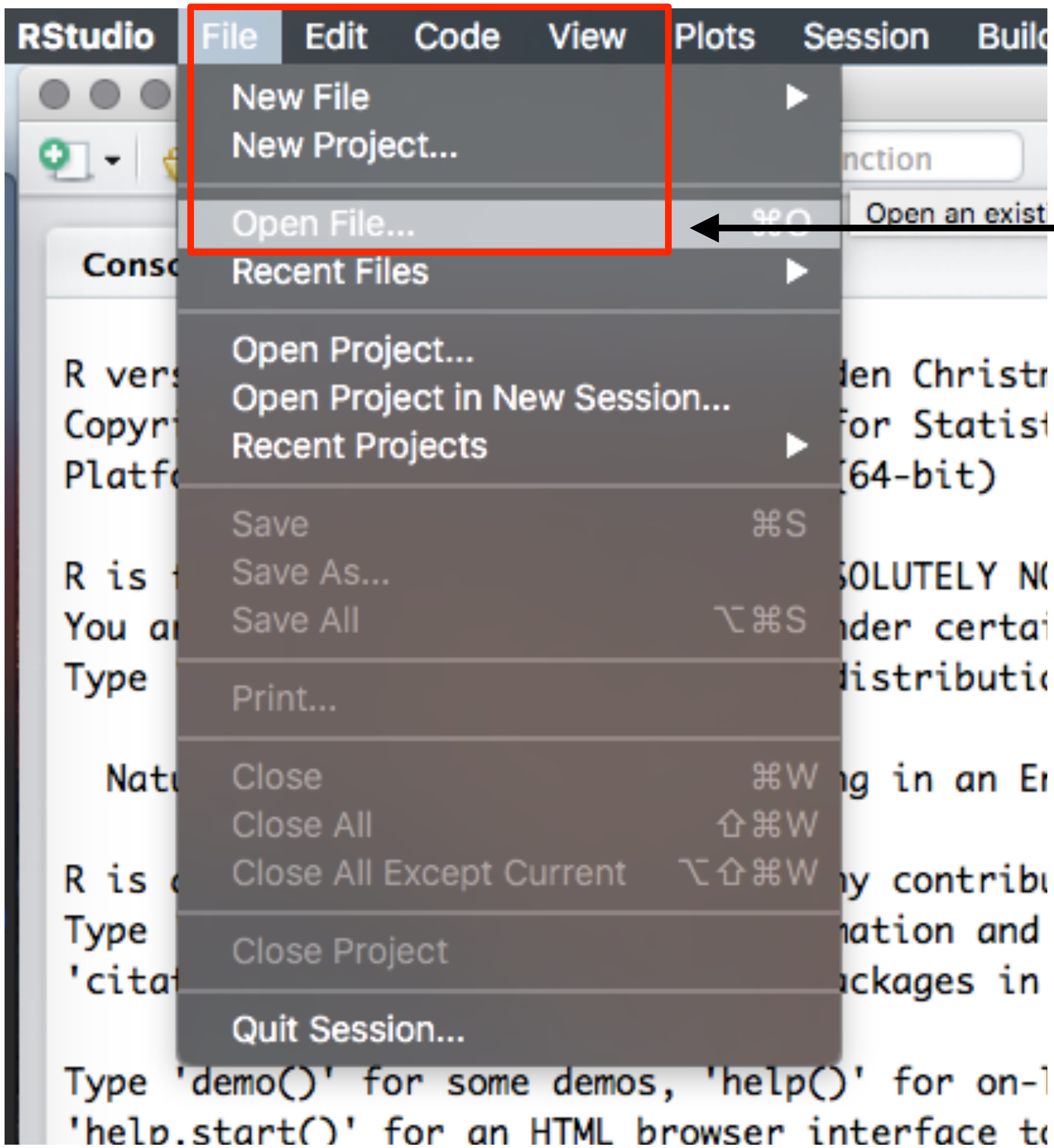
- The primary file format used by R is .Rdata (though it can also load Excel, csv, etc)
  - .Rdata files are saved workspaces
  - They contain whatever data sets, variables, functions etc that the workspace included when the file was created
- How to load an .Rdata file?
  - Hard(er) way: use the `load()` function manually
  - Easy way #1: double click on the .Rdata file in Finder/Explorer, and it should load automatically
  - **Easy way #2:** open using the Rstudio menus

Using Rstudio to load Rdata files



This is the “file open” button

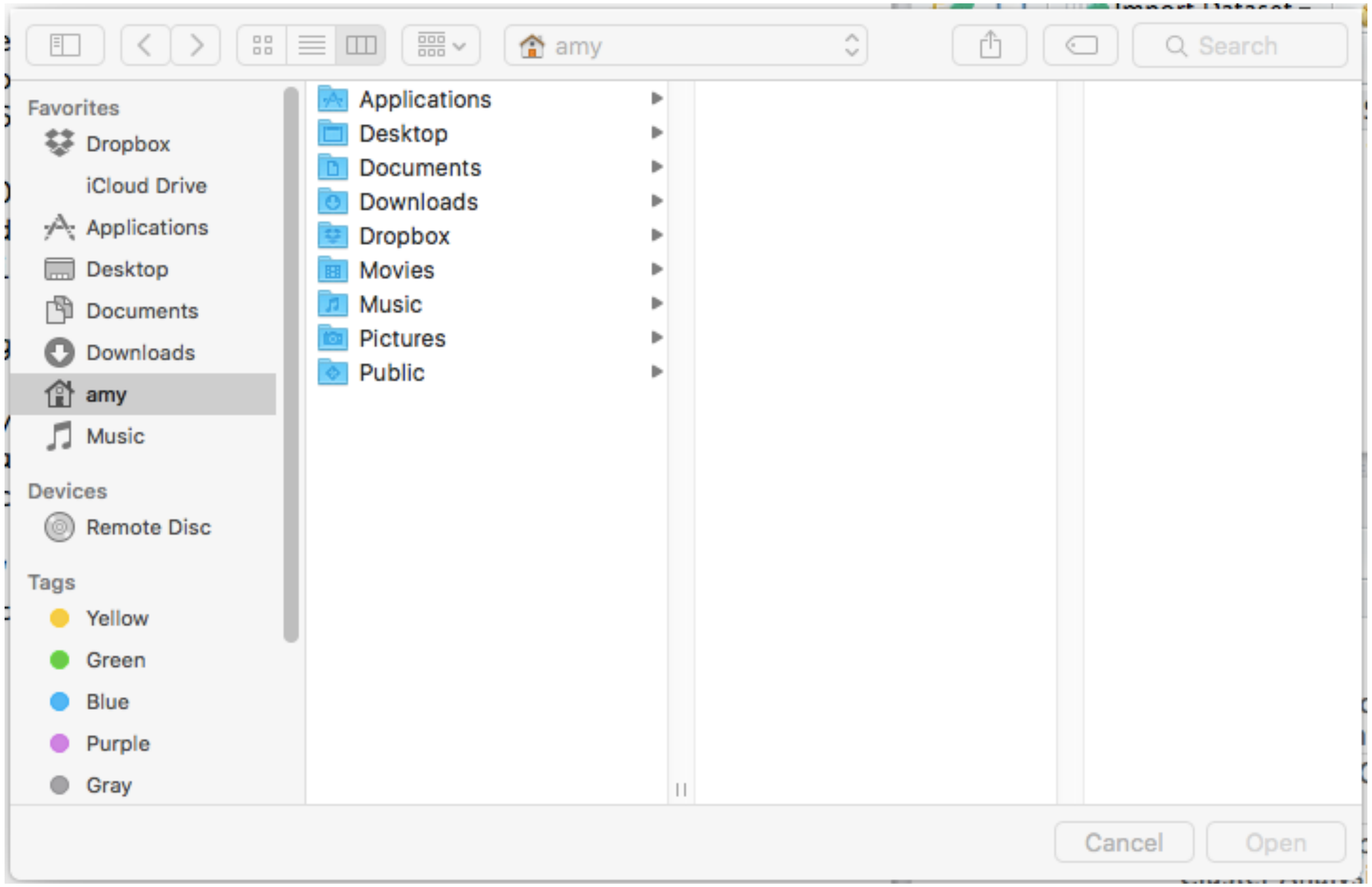
Using Rstudio to load Rdata files



```
R ver                              den Christr
Copyr                              for Statist
Platf                              (64-bit)

R is                               SOLUTELY NO
You a                              nder certai
Type                               distributio

  Natu                             ng in an En

R is                               ny contribu
Type                               mation and
'cita                              ckages in

Type 'demo()' for some demos, 'help()' for on-l
'help.start()' for an HTML browser interface to
```

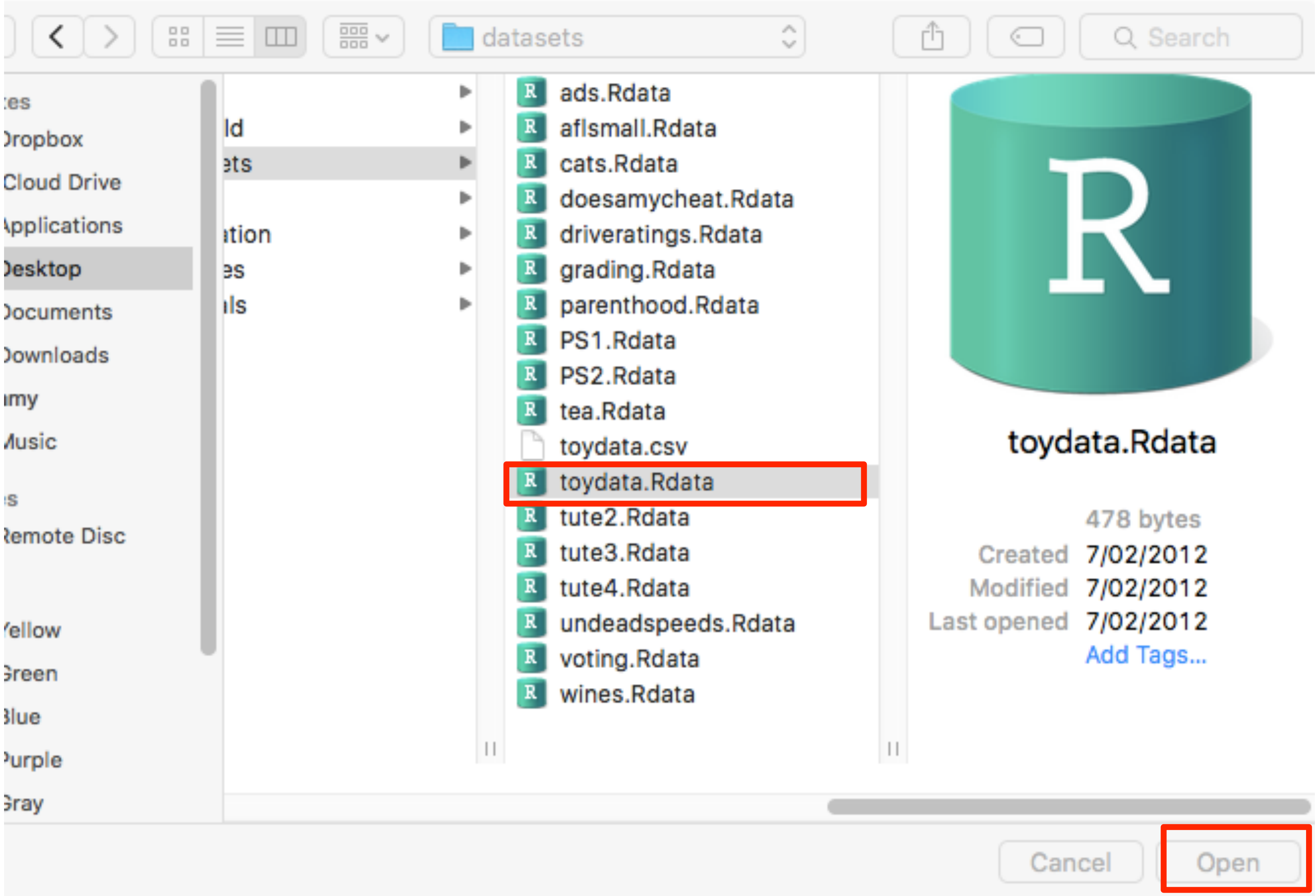
You can also use the File menu to do the same thing if you want to...

This opens a file open dialog box...



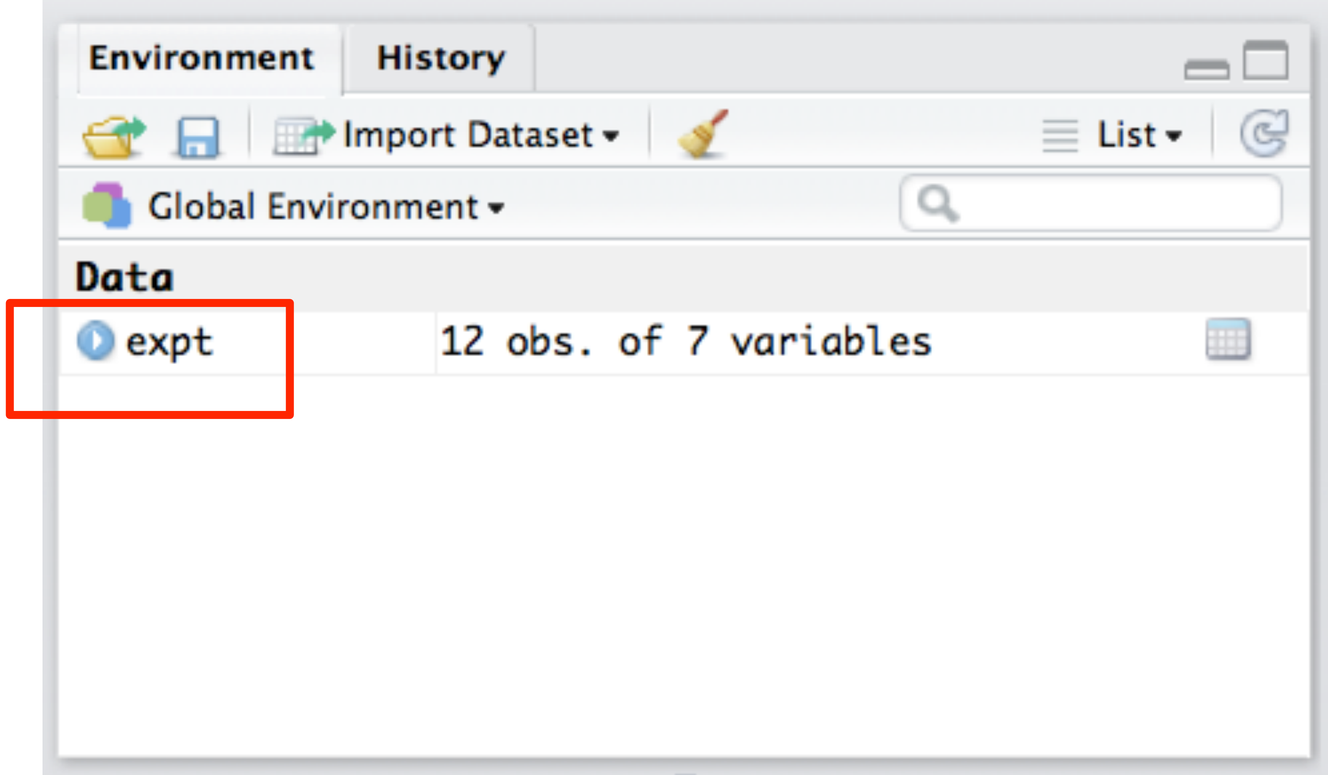
It will look different on different operating systems... it will look like a familiar Windows thing on a Windows computer, a standard Mac thing on a Mac computer etc etc...

Browse for the file you want, and open:



Clicking open will load the “toydata.Rdata” file you downloaded earlier from MyUni

A **copy** of the variable(s) saved in the file are now added to the workspace



expt 12 obs. of 7 variables

```
> load(“~/Documents/teaching/2018/summerschool/datasets/toydata.Rdata")
```

A command like this will appear in the R console (the command is what <u>actually</u> does the work)

Manipulating data

```
> expt

   id age gender treatment hormone happy  sad
1   1  25   male   control     6.7  2.00 6.12
2   2  24   male     drug1    38.5  3.36 3.53
3   3  25   male     drug2    25.0  3.40 4.82
4   4  28   male   control    98.4  5.69 0.34
5   5  23   male     drug1    42.4  4.56 4.48
6   6  28   male     drug2    20.3  2.89 4.57
7   7  25 female   control    18.5  3.18 4.82
8   8  29 female     drug1    65.2  4.78 2.24
9   9  21 female     drug2    56.4  4.51 2.64
10 10  26 female   control    55.7  3.90 2.71
11 11  19 female     drug1    41.9  2.83 2.94
12 12  30 female     drug2    54.1  3.45 1.87
```

The variable we just loaded is
a **“data frame”**

We’ve actually seen one already

```
> subject <- c( "STAT1", "STAT1", "STAT2", “STAT2" )
> person <- c( "ann", "bec", "ann", "bec" )
> grades <- c( 82, 71, 63, 80 )
```

```
> data.frame( person, subject, grades )

  person subject grades
1    ann   STAT1     82
2    bec   STAT1     71
3    ann   STAT2     63
4    bec   STAT2     80
```

Remember this bit?

A data frame is actually a bunch of vectors all bundled together…

Data frames

- Data frames are the typical way to store a data set in R

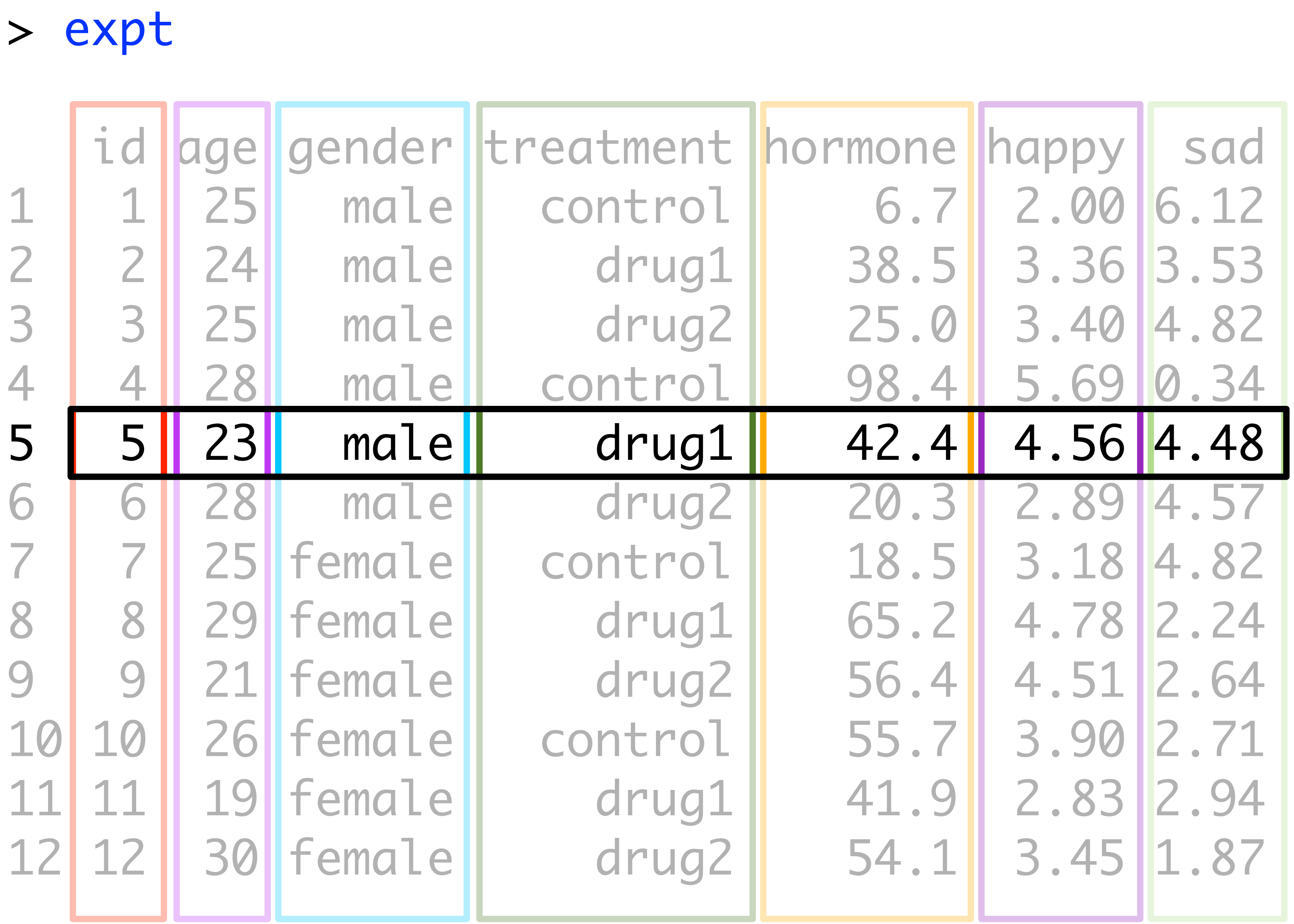
- What is a data frame?
  - It is a collection of variables “bundled” together
  - Organised into a “case by variable” matrix
  - Each row is a “case”
  - Each column is a named “variable”

- Let’s go through this idea more slowly…

Here are the 7 vectors

```
> expt
   id age gender treatment hormone happy  sad
1   1  25   male   control     6.7  2.00 6.12
2   2  24   male     drug1    38.5  3.36 3.53
3   3  25   male     drug2    25.0  3.40 4.82
4   4  28   male   control    98.4  5.69 0.34
5   5  23   male     drug1    42.4  4.56 4.48
6   6  28   male     drug2    20.3  2.89 4.57
7   7  25 female   control    18.5  3.18 4.82
8   8  29 female     drug1    65.2  4.78 2.24
9   9  21 female     drug2    56.4  4.51 2.64
10 10  26 female   control    55.7  3.90 2.71
11 11  19 female     drug1    41.9  2.83 2.94
12 12  30 female     drug2    54.1  3.45 1.87
```

They have a special relationship...



```
> expt
   id age gender treatment hormone happy  sad
1   1  25   male   control     6.7  2.00 6.12
2   2  24   male     drug1    38.5  3.36 3.53
3   3  25   male     drug2    25.0  3.40 4.82
4   4  28   male   control    98.4  5.69 0.34
5   5  23   male     drug1    42.4  4.56 4.48
6   6  28   male     drug2    20.3  2.89 4.57
7   7  25 female   control    18.5  3.18 4.82
8   8  29 female     drug1    65.2  4.78 2.24
9   9  21 female     drug2    56.4  4.51 2.64
10 10  26 female   control    55.7  3.90 2.71
11 11  19 female     drug1    41.9  2.83 2.94
12 12  30 female     drug2    54.1  3.45 1.87
```

The 5th element of each variable refers to the same person (the same "**case**")

```
> expt

   id age gender treatment hormone happy  sad
1   1  25   male   control     6.7  2.00 6.12
2   2  24   male     drug1    38.5  3.36 3.53
3   3  25   male     drug2    25.0  3.40 4.82
4   4  28   male   control    98.4  5.69 0.34
5   5  23   male     drug1    42.4  4.56 4.48
6   6  28   male     drug2    20.3  2.89 4.57
7   7  25 female   control    18.5  3.18 4.82
8   8  29 female     drug1    65.2  4.78 2.24
9   9  21 female     drug2    56.4  4.51 2.64
10 10  26 female   control    55.7  3.90 2.71
11 11  19 female     drug1    41.9  2.83 2.94
12 12  30 female     drug2    54.1  3.45 1.87
```

But! They are still ordinary variables…

```
> expt$age
 [1] 25 24 25 28 23 28 25 29 21 26 19 30
```

`expt$age` tells R to look for a vector called `age` stored in a data frame called `expt`.

```
> expt

   id age gender treatment hormone happy  sad
1   1  25   male   control     6.7  2.00 6.12
2   2  24   male     drug1    38.5  3.36 3.53
3   3  25   male     drug2    25.0  3.40 4.82
4   4  28   male   control    98.4  5.69 0.34
5   5  23   male     drug1    42.4  4.56 4.48
6   6  28   male     drug2    20.3  2.89 4.57
7   7  25 female   control    18.5  3.18 4.82
8   8  29 female     drug1    65.2  4.78 2.24
9   9  21 female     drug2    56.4  4.51 2.64
10 10  26 female   control    55.7  3.90 2.71
11 11  19 female     drug1    41.9  2.83 2.94
12 12  30 female     drug2    54.1  3.45 1.87
```

But! They are still ordinary variables…

```
> expt$gender
 [1]  male   male   male   male   male
 [6]  male   female female female female
 [11] female female
 Levels: male female
```

Hm. That's odd. We'll come back to that one in a moment

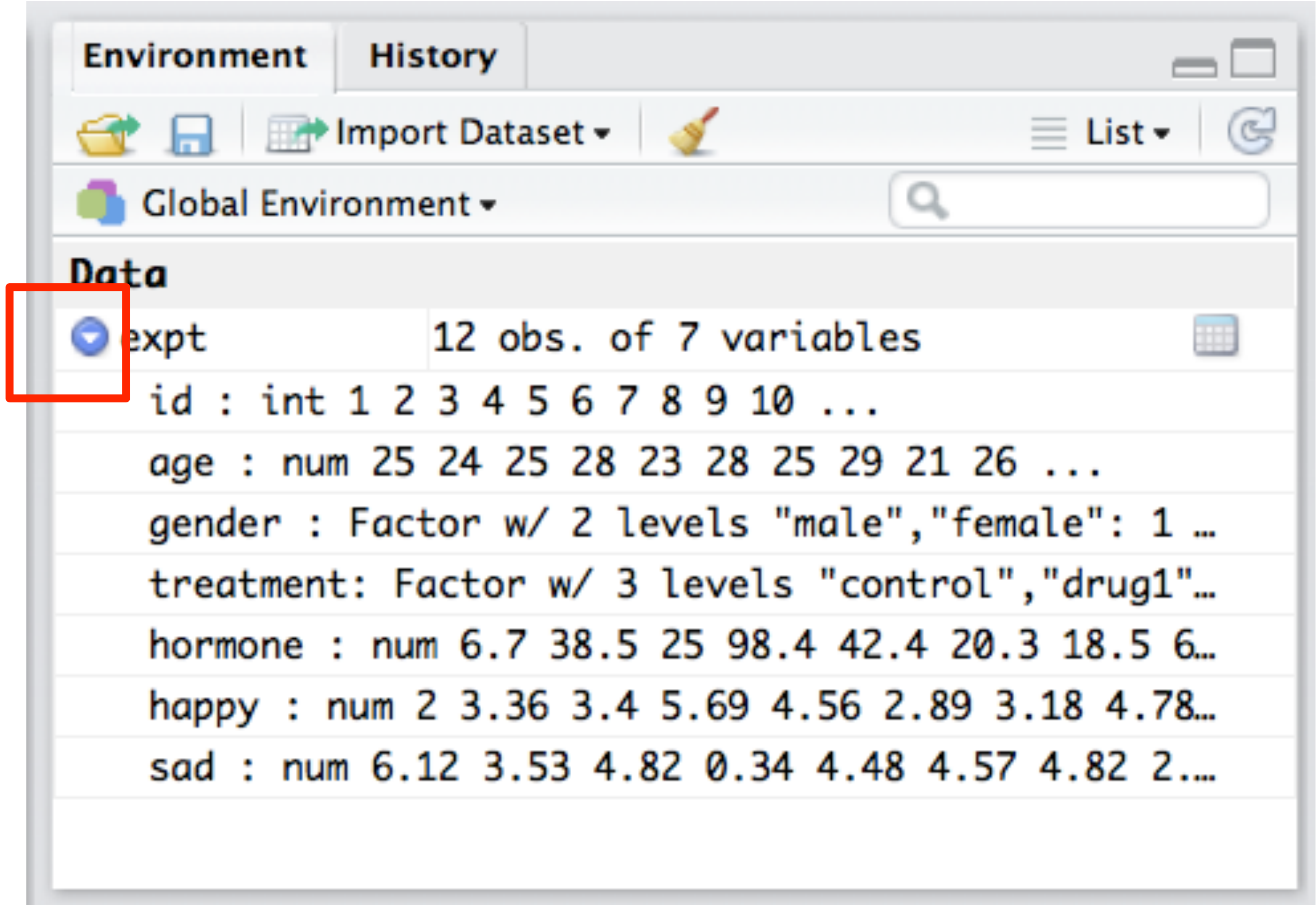
```
> expt

   id age gender treatment hormone happy  sad
1   1  25   male   control     6.7  2.00 6.12
2   2  24   male     drug1    38.5  3.36 3.53
3   3  25   male     drug2    25.0  3.40 4.82
4   4  28   male   control    98.4  5.69 0.34
5   5  23   male     drug1    42.4  4.56 4.48
6   6  28   male     drug2    20.3  2.89 4.57
7   7  25 female   control    18.5  3.18 4.82
8   8  29 female     drug1    65.2  4.78 2.24
9   9  21 female     drug2    56.4  4.51 2.64
10 10  26 female   control    55.7  3.90 2.71
11 11  19 female     drug1    41.9  2.83 2.94
12 12  30 female     drug2    54.1  3.45 1.87
```

But! They are still ordinary variables…

```
> expt$happy
 [1] 2.00 3.36 3.40 5.69 4.56 2.89 3.18
 [8] 4.78 4.51 3.90 2.83 3.45
```

Okay, clearly this $ trick works for all of them...

You can also view the dataset using RStudio

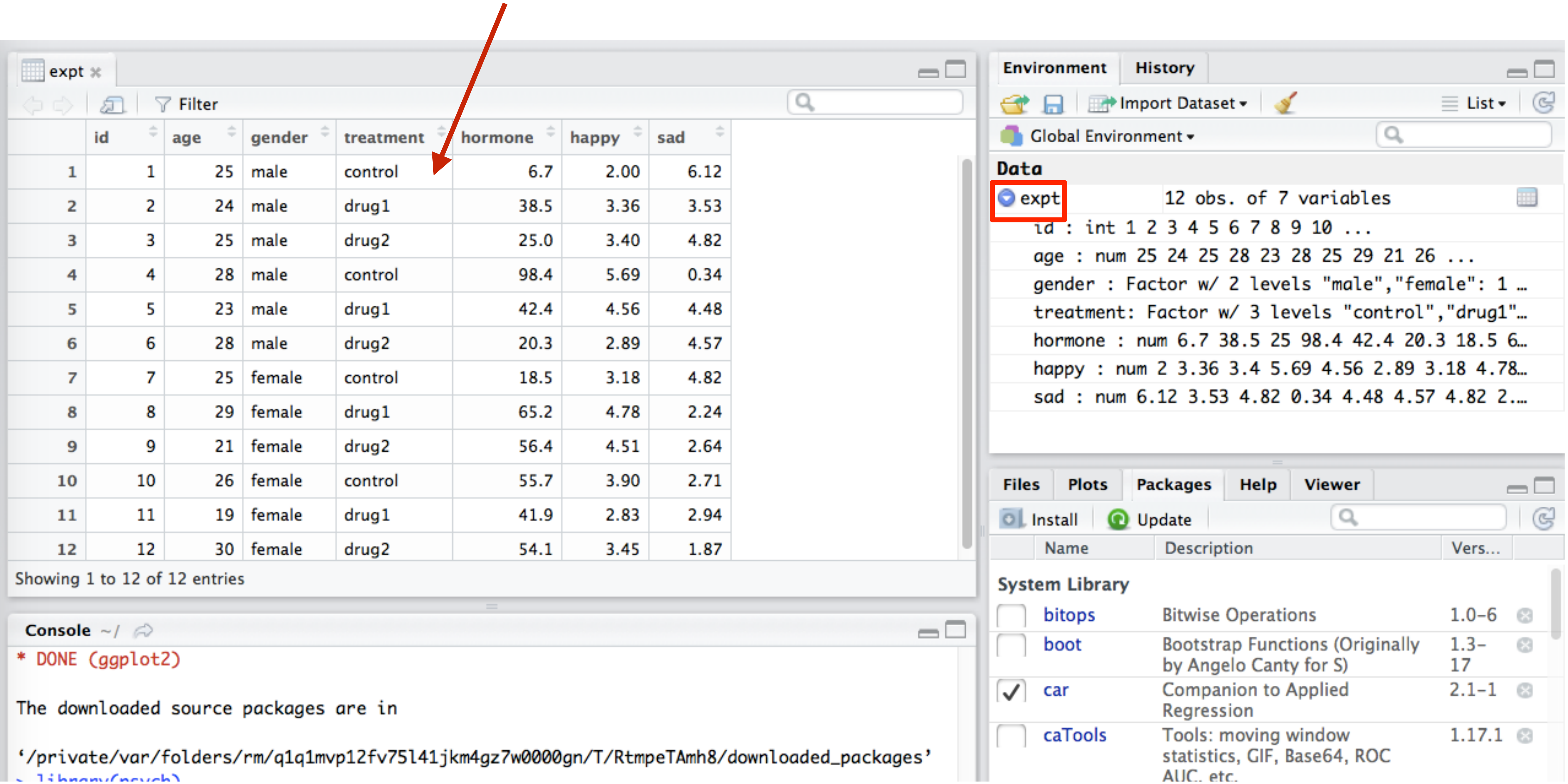


```
expt        12 obs. of 7 variables
  id : int 1 2 3 4 5 6 7 8 9 10 ...
  age : num 25 24 25 28 23 28 25 29 21 26 ...
  gender : Factor w/ 2 levels "male","female": 1 ...
  treatment: Factor w/ 3 levels "control","drug1"...
  hormone : num 6.7 38.5 25 98.4 42.4 20.3 18.5 6...
  happy : num 2 3.36 3.4 5.69 4.56 2.89 3.18 4.78...
  sad : num 6.12 3.53 4.82 0.34 4.48 4.57 4.82 2....
```

Clicking here lets you see the entire dataset

You can also view the dataset using RStudio

Clicking it again shows you the dataset in another panel.



	id	age	gender	treatment	hormone	happy	sad
1	1	25	male	control	6.7	2.00	6.12
2	2	24	male	drug1	38.5	3.36	3.53
3	3	25	male	drug2	25.0	3.40	4.82
4	4	28	male	control	98.4	5.69	0.34
5	5	23	male	drug1	42.4	4.56	4.48
6	6	28	male	drug2	20.3	2.89	4.57
7	7	25	female	control	18.5	3.18	4.82
8	8	29	female	drug1	65.2	4.78	2.24
9	9	21	female	drug2	56.4	4.51	2.64
10	10	26	female	control	55.7	3.90	2.71
11	11	19	female	drug1	41.9	2.83	2.94
12	12	30	female	drug2	54.1	3.45	1.87

Showing 1 to 12 of 12 entries

```
expt            12 obs. of 7 variables
   id : int 1 2 3 4 5 6 7 8 9 10 ...
   age : num 25 24 25 28 23 28 25 29 21 26 ...
   gender : Factor w/ 2 levels "male","female": 1 …
   treatment: Factor w/ 3 levels "control","drug1"…
   hormone : num 6.7 38.5 25 98.4 42.4 20.3 18.5 6…
   happy : num 2 3.36 3.4 5.69 4.56 2.89 3.18 4.78…
   sad : num 6.12 3.53 4.82 0.34 4.48 4.57 4.82 2.…
```

```
* DONE (ggplot2)

The downloaded source packages are in

‘/private/var/folders/rm/q1q1mvp12fv75l41jkm4gz7w0000gn/T/RtmpeTAmh8/downloaded_packages’
```

	Name	Description	Vers...
System Library			
☐	bitops	Bitwise Operations	1.0-6
☐	boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
☑	car	Companion to Applied Regression	2.1-1
☐	caTools	Tools: moving window statistics, GIF, Base64, ROC AUC, etc.	1.17.1

Variables inside data frames behave the same way as any other variable

```
> expt$age
 [1] 25 24 25 28 23 28 25 29 21 26 19 30


> expt$age + 100
 [1] 125 124 125 128 123 128 125 129 121 126 119 130


> expt$age[1]
[1] 25
```

You can change the values of variables in a data frame in the usual way...

```
> expt$age[1] <- 1000
> expt

   id  age gender treatment hormone happy  sad
1   1 1000   male   control     6.7  2.00 6.12
2   2   24   male     drug1    38.5  3.36 3.53
3   3   25   male     drug2    25.0  3.40 4.82
4   4   28   male   control    98.4  5.69 0.34
5   5   23   male     drug1    42.4  4.56 4.48
6   6   28   male     drug2    20.3  2.89 4.57

etc
```

```
> expt$age[1] <- 25  # change it back!
```

You can add variables to a data frame...

```
> expt$over25 <- expt$age > 25
> expt

   id age gender treatment hormone happy  sad over25
1   1  25   male   control     6.7  2.00 6.12  FALSE
2   2  24   male     drug1    38.5  3.36 3.53  FALSE
3   3  25   male     drug2    25.0  3.40 4.82  FALSE
4   4  28   male   control    98.4  5.69 0.34   TRUE
5   5  23   male     drug1    42.4  4.56 4.48  FALSE
6   6  28   male     drug2    20.3  2.89 4.57   TRUE
7   7  25 female   control    18.5  3.18 4.82  FALSE
8   8  29 female     drug1    65.2  4.78 2.24   TRUE
9   9  21 female     drug2    56.4  4.51 2.64  FALSE
10 10  26 female   control    55.7  3.90 2.71   TRUE
11 11  19 female     drug1    41.9  2.83 2.94  FALSE
12 12  30 female     drug2    54.1  3.45 1.87   TRUE
```

Removing them is even easier...

```
> expt$over25 <- NULL
> expt

   id age gender treatment hormone happy  sad
1   1  25   male   control     6.7  2.00 6.12
2   2  24   male     drug1    38.5  3.36 3.53
3   3  25   male     drug2    25.0  3.40 4.82
4   4  28   male   control    98.4  5.69 0.34
5   5  23   male     drug1    42.4  4.56 4.48
6   6  28   male     drug2    20.3  2.89 4.57
7   7  25 female   control    18.5  3.18 4.82
8   8  29 female     drug1    65.2  4.78 2.24
9   9  21 female     drug2    56.4  4.51 2.64
10 10  26 female   control    55.7  3.90 2.71
11 11  19 female     drug1    41.9  2.83 2.94
12 12  30 female     drug2    54.1  3.45 1.87
```

NULL is a special “value” in R that means “this variable does not exist” or “it has no value”. It is different to NA, which means “the variable exists (and in principle has a value), but the value is missing/unknown”

Selecting elements from a data frame

```
> expt$age[1]
[1] 25
```

expt$age is a vector, and we're requesting the 1st element of it

```
> expt[1, 2]
[1] 25
```

expt is a data frame, and we're requesting the value found in the 1st row, and the 2nd column

```
> expt[1, "age"]
[1] 25
```

expt is a data frame, and we're requesting the value found in the 1st row, and the column named “age”

Selecting a whole row

```
> expt[ 4, ]

  id age gender treatment hormone happy  sad
4  4  28   male   control    98.4  5.69 0.34
```

Selecting multiple rows

```
> expt[ c(1,4,7), ]

  id age gender treatment hormone happy  sad
1  1  25   male   control     6.7  2.00 6.12
4  4  28   male   control    98.4  5.69 0.34
7  7  25 female   control    18.5  3.18 4.82
```

Selecting rows and columns?

```
> expt[ c(1,4,7), c("age","gender") ]

  age gender
1  25   male
4  28   male
7  25 female
```

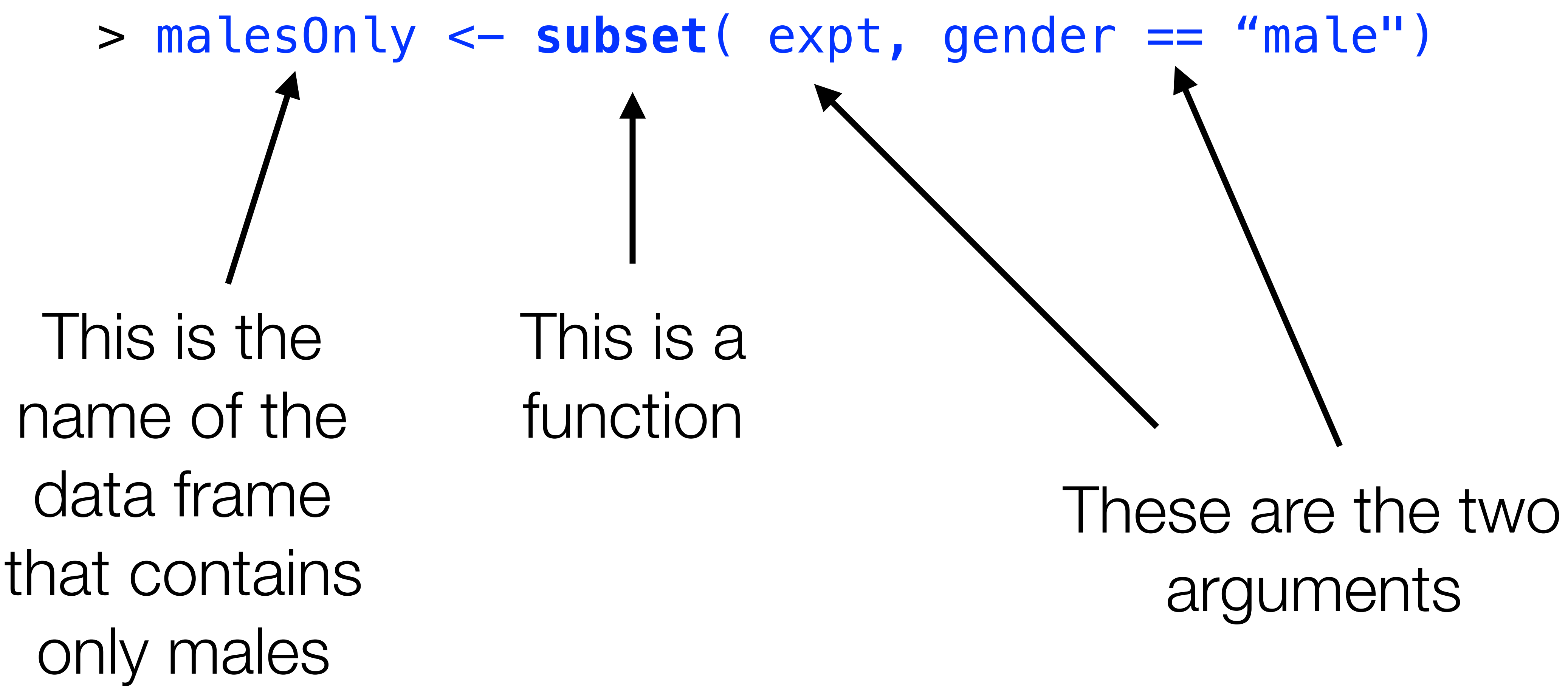
Selecting rows that match a criterion?

```
> theMales <- expt$gender == "male"
> expt[ theMales, ]

  id age gender treatment hormone happy  sad
1  1  25   male   control     6.7  2.00 6.12
2  2  24   male     drug1    38.5  3.36 3.53
3  3  25   male     drug2    25.0  3.40 4.82
4  4  28   male   control    98.4  5.69 0.34
5  5  23   male     drug1    42.4  4.56 4.48
6  6  28   male     drug2    20.3  2.89 4.57
```

Using subset() to do the same thing

```
> malesOnly <- subset( expt, gender == “male")
> malesOnly

  id age gender treatment hormone happy  sad
1  1  25   male   control     6.7  2.00 6.12
2  2  24   male     drug1    38.5  3.36 3.53
3  3  25   male     drug2    25.0  3.40 4.82
4  4  28   male   control    98.4  5.69 0.34
5  5  23   male     drug1    42.4  4.56 4.48
6  6  28   male     drug2    20.3  2.89 4.57
```

Using subset() to do the same thing



```
> malesOnly <- subset( expt, gender == “male")
```

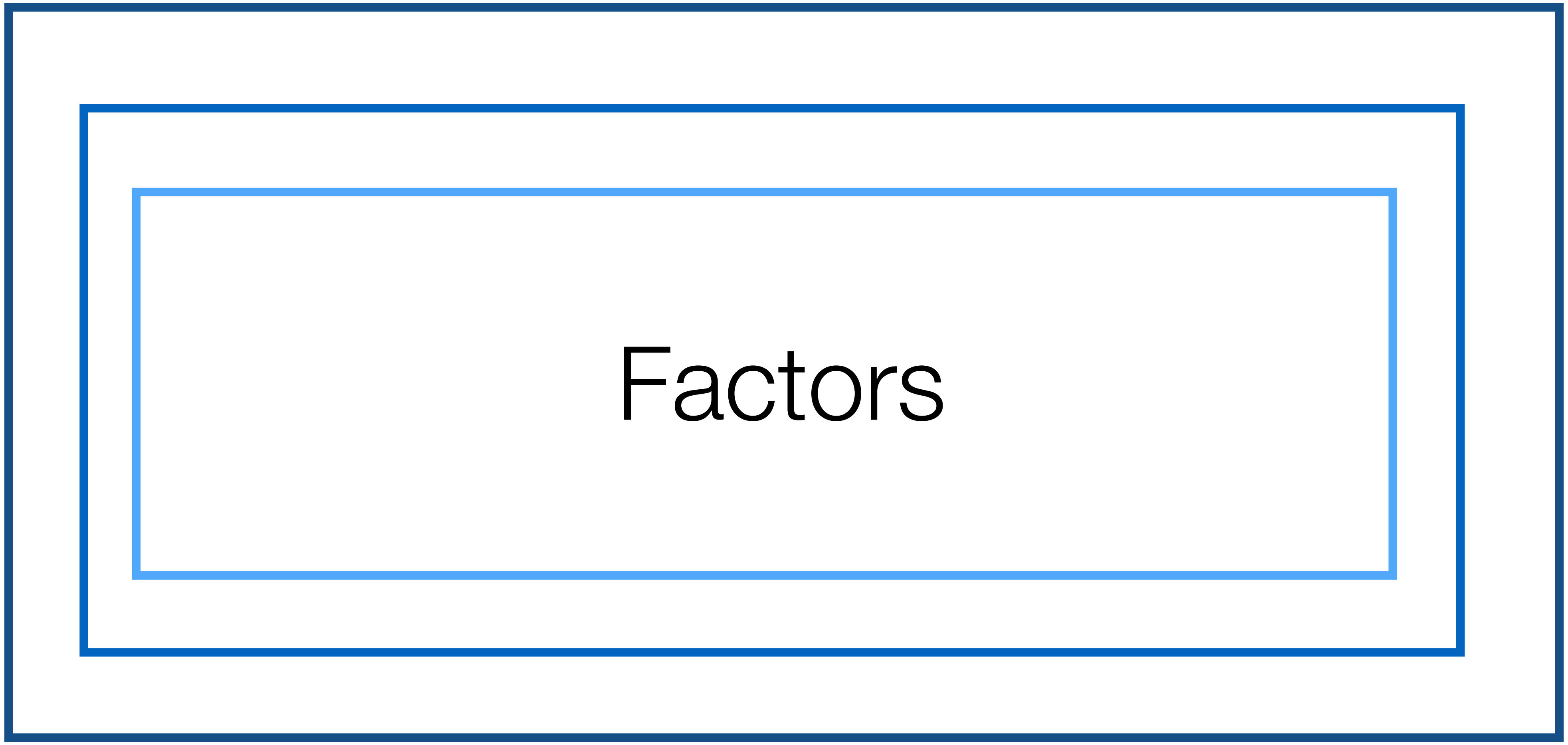
This is the name of the data frame that contains only males

This is a function

These are the two arguments

Exercises

1. Make a new dataframe called `d` which is just a copy of `expt`.
2. In `d`, add 1.5 to every entry for `hormone`.
3. Create a new variable in `d` called `depressed` which is `sad` minus `happy`.
4. Find out how many people are over 25 and took more than 20.0 of the hormone.
5. Create a new dataframe consisting of just the control condition.
6. Create another new dataframe consisting of both drug1 and drug2 conditions.



Factors

Okay, what's going on with "gender"?

```
> expt$gender

 [1] male   male   male   male   male   male   female
 [8] female female female female female
Levels: female male
```

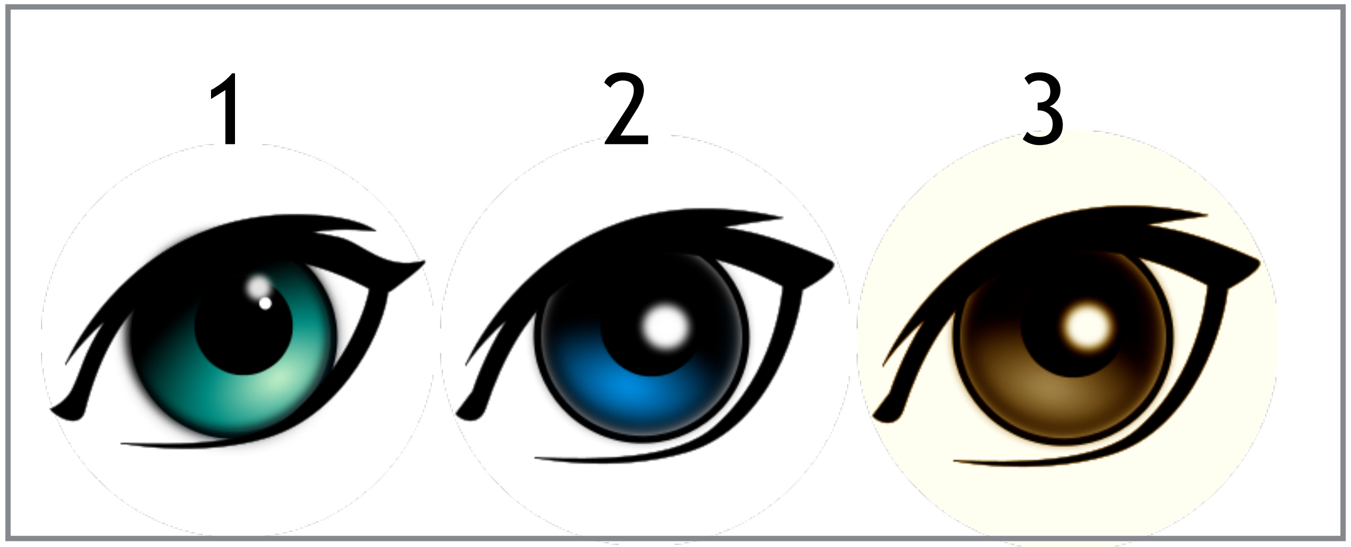
This is new!

expt$gender is actually a “factor”...

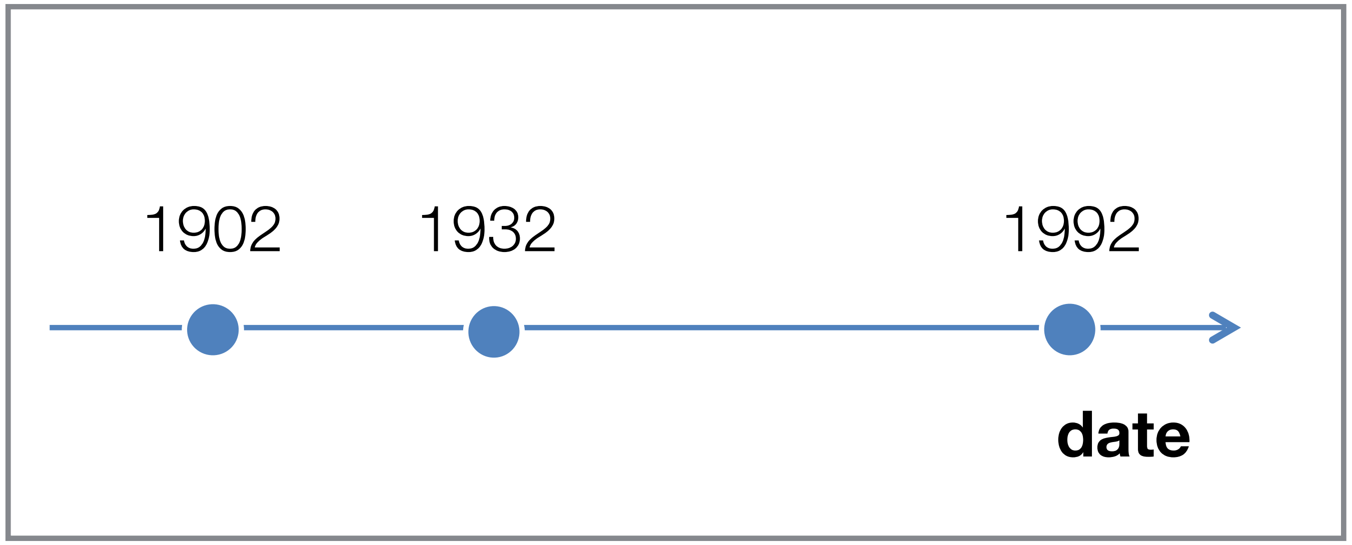
```
> expt$gender

 [1] male   male   male   male   male   male   female
 [8] female female female female female
Levels: female male
```

```
> class( expt$gender )
[1] "factor"
```

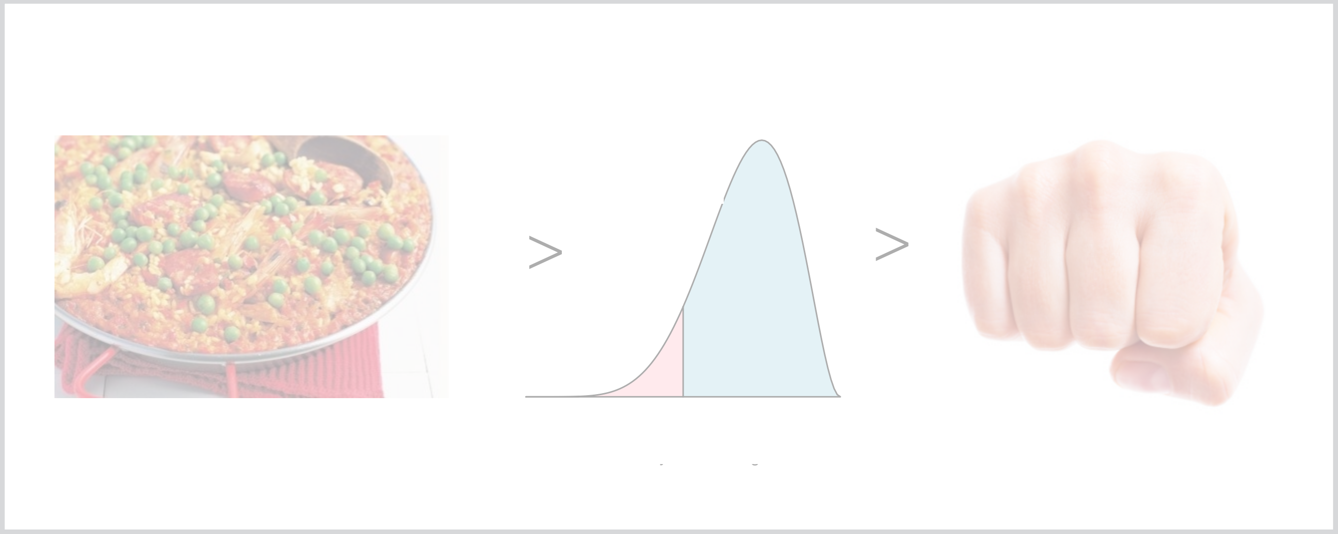
Factors “look” like character data,
but they’re a bit more subtle than that...



In R, nominal scale data are stored as **factors**



Interval and ratio scale data are stored as **numeric** variables



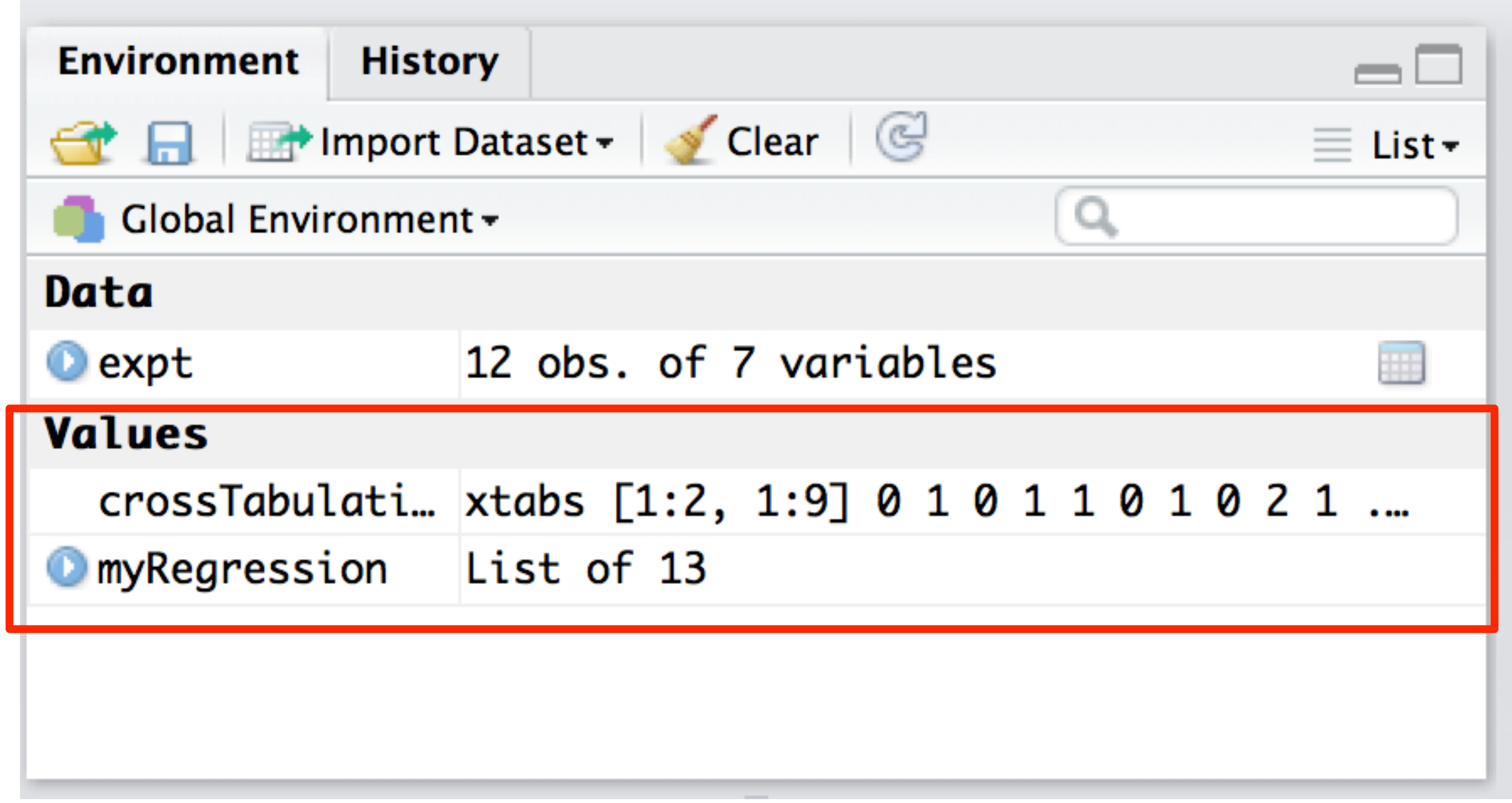
Ordinal scale data are stored as **ordered factors** (not covered in this class)

What’s this about?

- R needs to know if a variable is nominal scale
    - A “factor” is a nominal scale variable
    - Created using `factor()` and `as.factor()` [not in this class]
    - Some analyses we’ll learn later require factors!
    - The textbook has more details on working with factors
    - For this class, every time you need a factor it will already be properly converted to a factor!

Saving your current variables to a file

Suppose you've done some work and you want to save the workspace...



Data

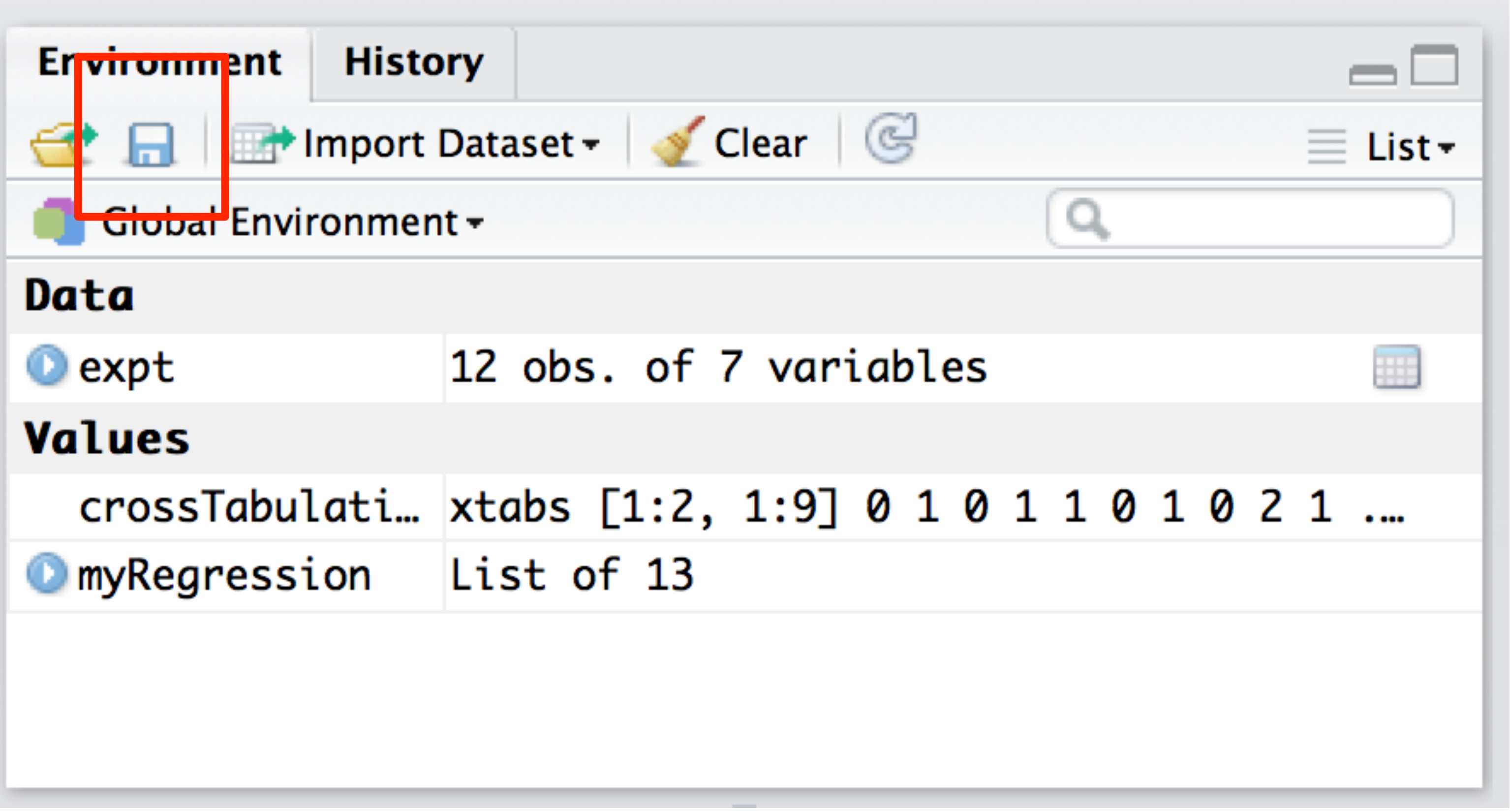
expt	12 obs. of 7 variables

Values

crossTabulati…	xtabs [1:2, 1:9] 0 1 0 1 1 0 1 0 2 1 ...
myRegression	List of 13

I must have done some work, there's all this new stuff in the workspace!

The save button is your friend



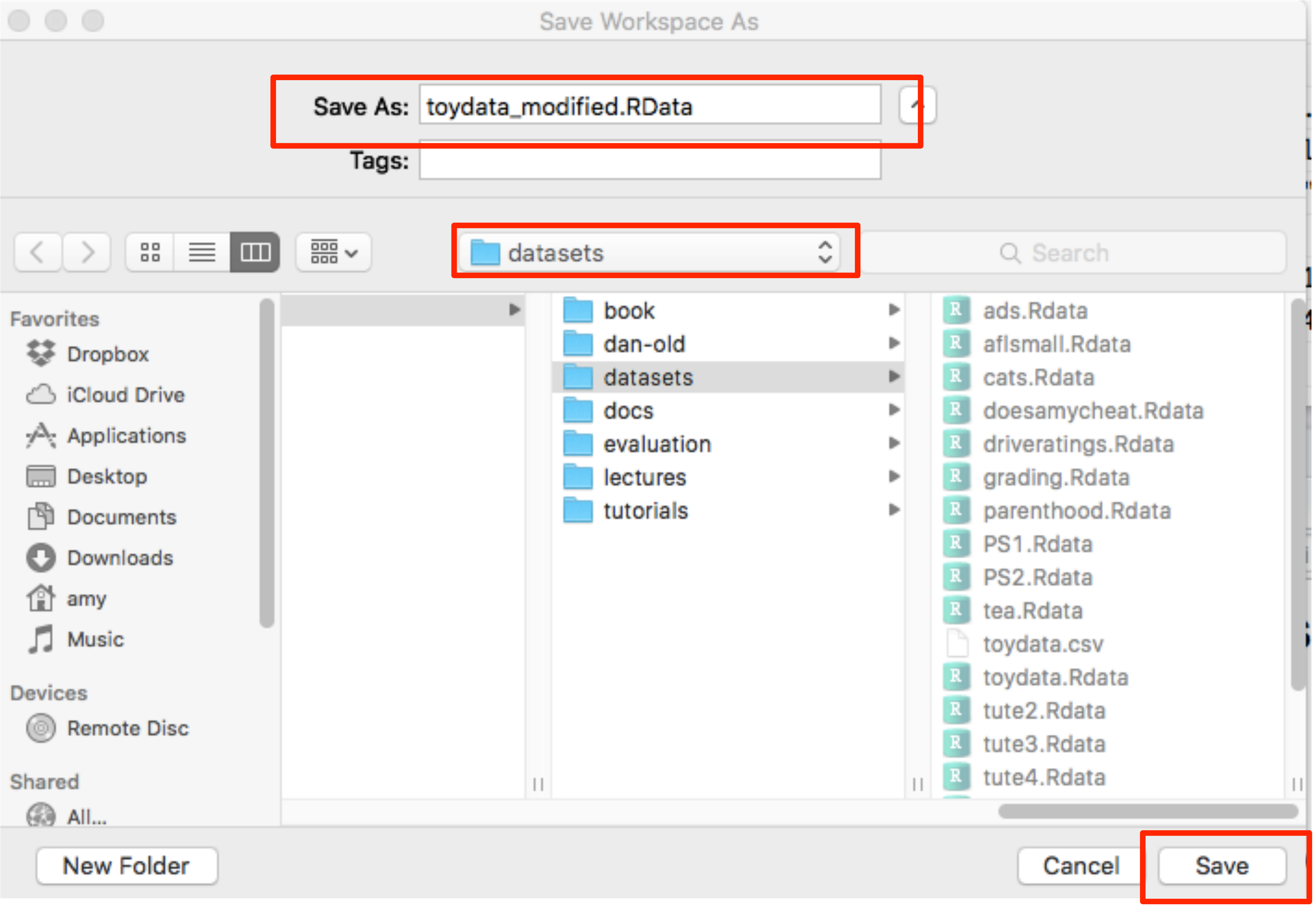
Data

expt	12 obs. of 7 variables

Values

crossTabulati…	xtabs [1:2, 1:9] 0 1 0 1 1 0 1 0 2 1 …
myRegression	List of 13

Browse, type a filename, and click save



Save As: toydata_modified.RData

Now the file is saved

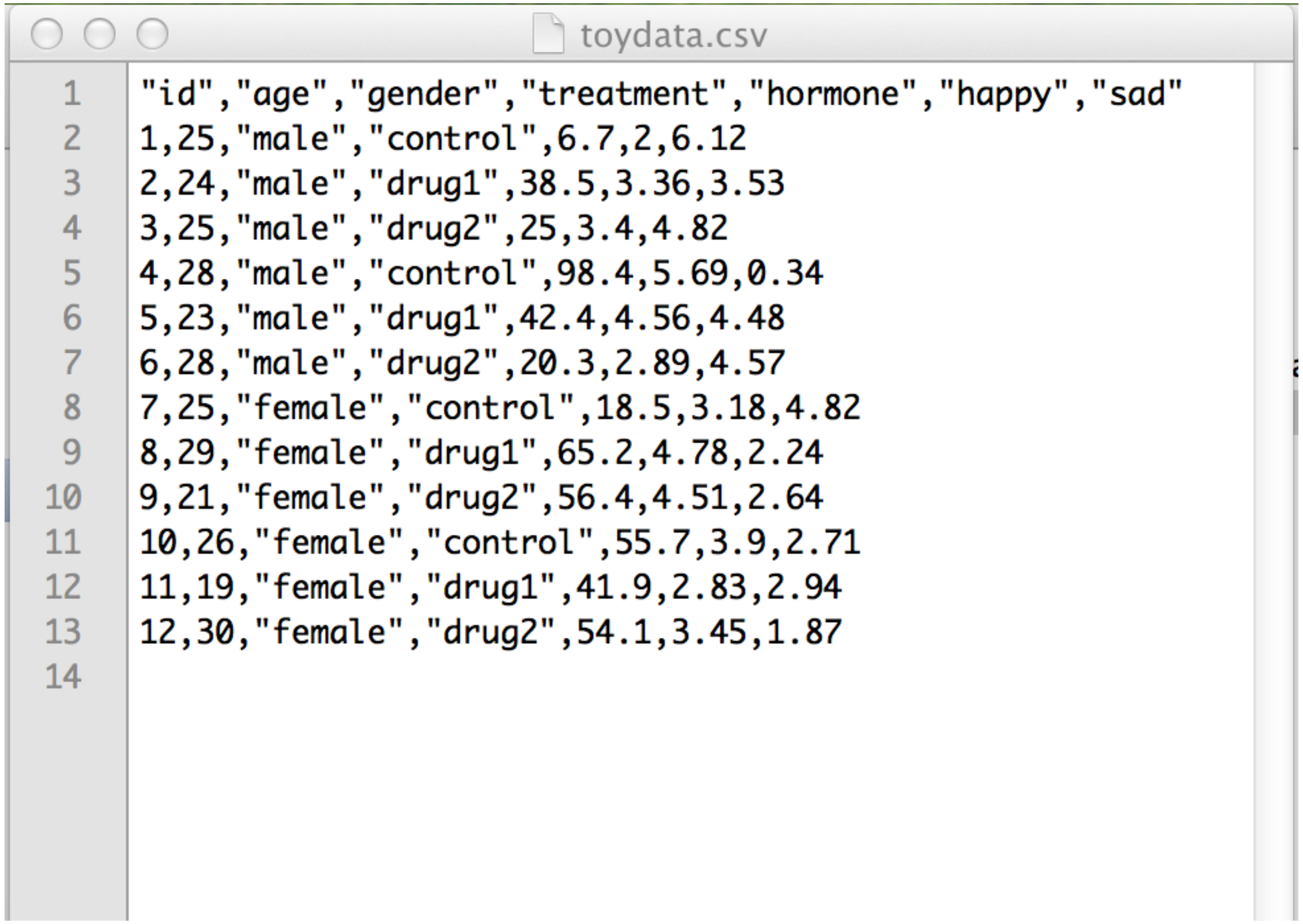
```
save.image(“~/Documents/teaching/2018/summerschool/datasets/toydata_modified.RData”)
```

As before, the actual command
shows up in the R console

Importing data from a text (“csv”) file

CSV is a standard format

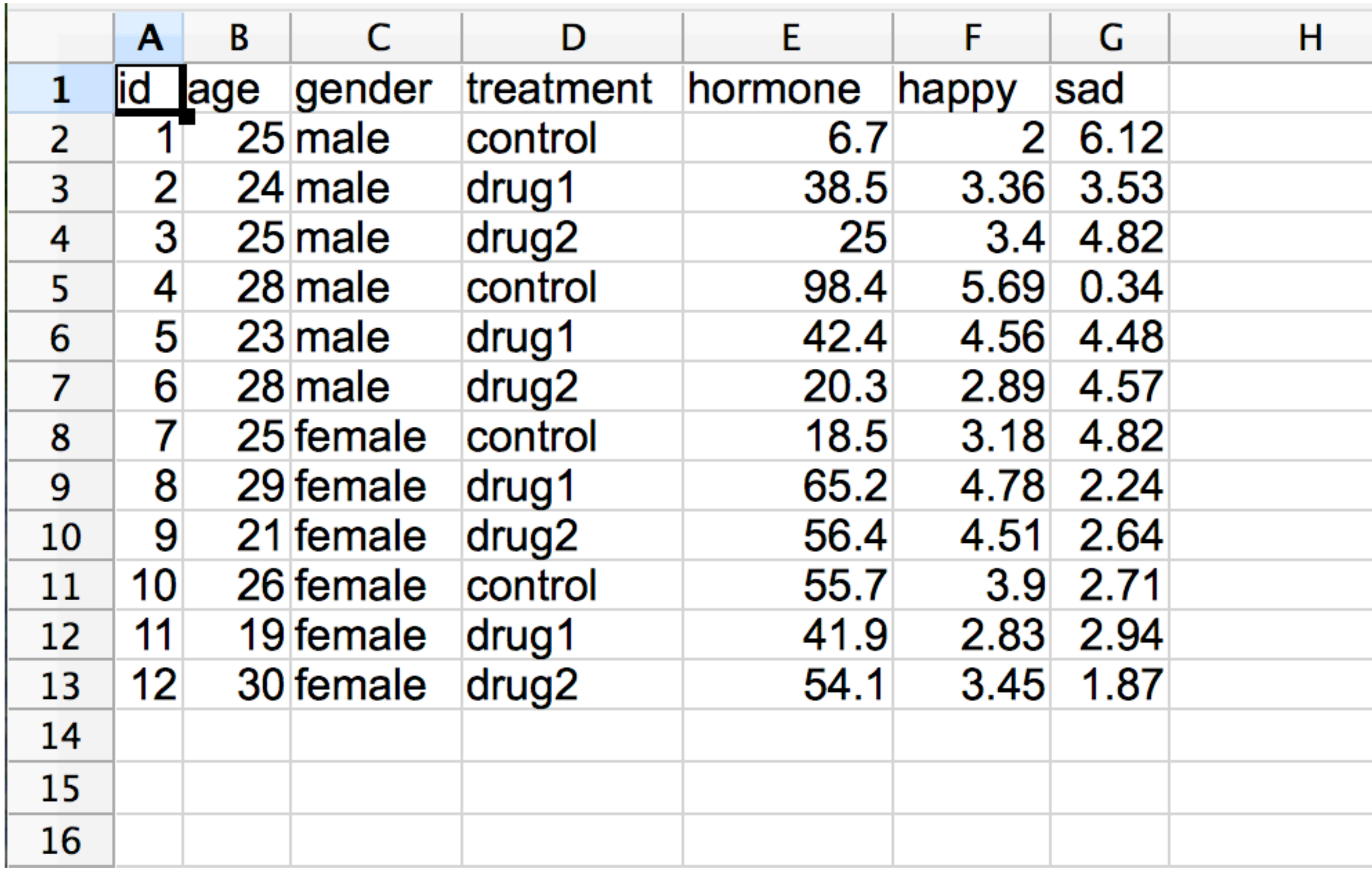
The raw data is just a plain text file: CSV stands for "comma separated value"



```
"id","age","gender","treatment","hormone","happy","sad"
1,25,"male","control",6.7,2,6.12
2,24,"male","drug1",38.5,3.36,3.53
3,25,"male","drug2",25,3.4,4.82
4,28,"male","control",98.4,5.69,0.34
5,23,"male","drug1",42.4,4.56,4.48
6,28,"male","drug2",20.3,2.89,4.57
7,25,"female","control",18.5,3.18,4.82
8,29,"female","drug1",65.2,4.78,2.24
9,21,"female","drug2",56.4,4.51,2.64
10,26,"female","control",55.7,3.9,2.71
11,19,"female","drug1",41.9,2.83,2.94
12,30,"female","drug2",54.1,3.45,1.87
```

CSV is a standard format

CSV files are usually opened by spreadsheets, and produce “rectangular” data like this...



	A	B	C	D	E	F	G	H
1	id	age	gender	treatment	hormone	happy	sad	
2	1	25	male	control	6.7	2	6.12	
3	2	24	male	drug1	38.5	3.36	3.53	
4	3	25	male	drug2	25	3.4	4.82	
5	4	28	male	control	98.4	5.69	0.34	
6	5	23	male	drug1	42.4	4.56	4.48	
7	6	28	male	drug2	20.3	2.89	4.57	
8	7	25	female	control	18.5	3.18	4.82	
9	8	29	female	drug1	65.2	4.78	2.24	
10	9	21	female	drug2	56.4	4.51	2.64	
11	10	26	female	control	55.7	3.9	2.71	
12	11	19	female	drug1	41.9	2.83	2.94	
13	12	30	female	drug2	54.1	3.45	1.87	
14								
15								
16								

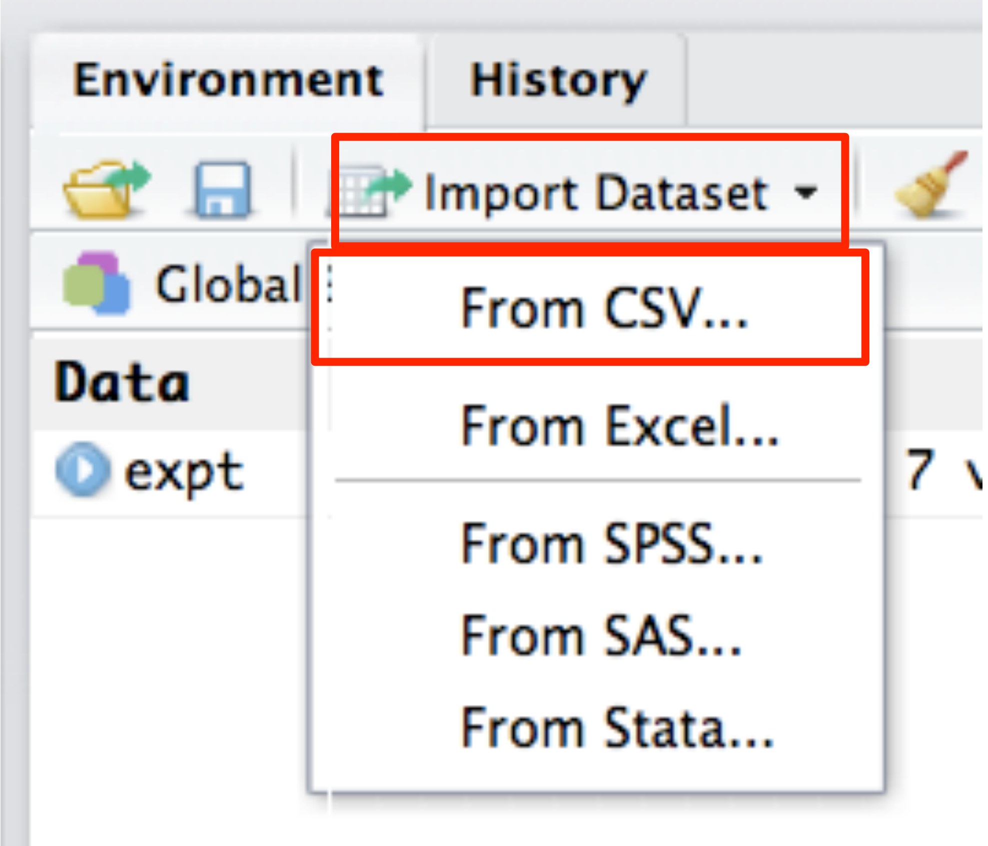
CSV is a standard format

In R, a CSV file gets imported as a data frame

```
> expt

   id age gender treatment hormone happy  sad
1   1  25   male   control     6.7  2.00 6.12
2   2  24   male     drug1    38.5  3.36 3.53
3   3  25   male     drug2    25.0  3.40 4.82
4   4  28   male   control    98.4  5.69 0.34
5   5  23   male     drug1    42.4  4.56 4.48
6   6  28   male     drug2    20.3  2.89 4.57
7   7  25 female   control    18.5  3.18 4.82
8   8  29 female     drug1    65.2  4.78 2.24
9   9  21 female     drug2    56.4  4.51 2.64
10 10  26 female   control    55.7  3.90 2.71
11 11  19 female     drug1    41.9  2.83 2.94
12 12  30 female     drug2    54.1  3.45 1.87
```

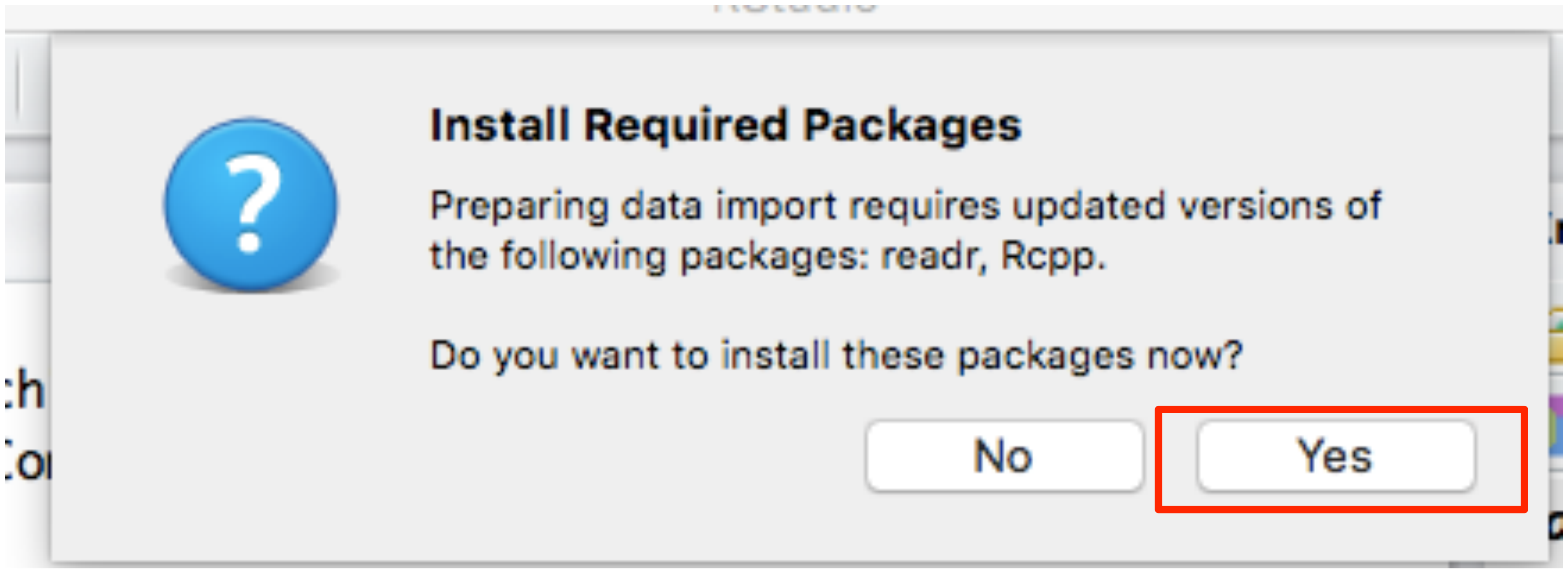
Importing CSV data using Rstudio

Click on this…



Importing CSV data using Rstudio

You may have to install some other packages…



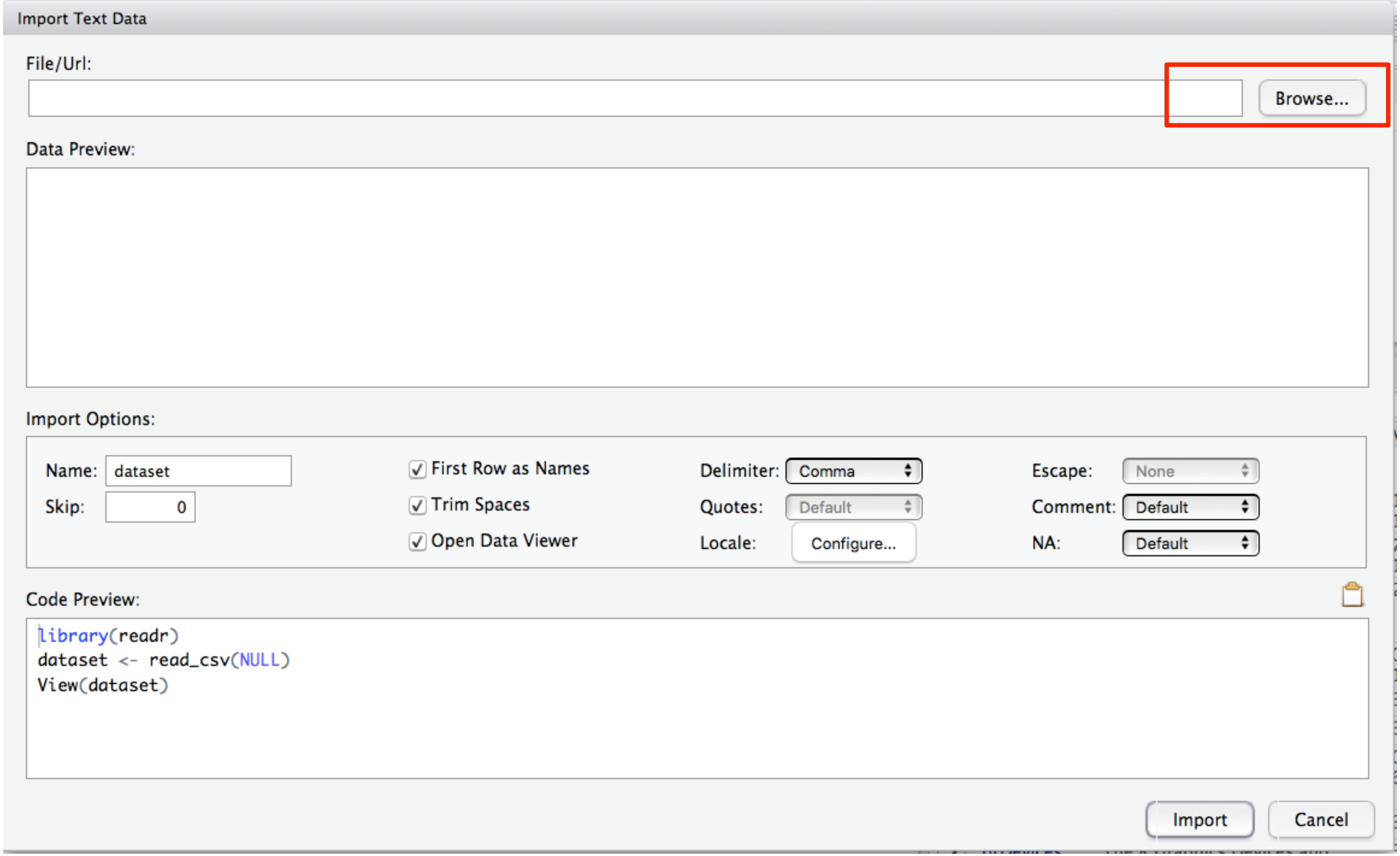
Install Required Packages

Preparing data import requires updated versions of the following packages: readr, Rcpp.

Do you want to install these packages now?

Importing CSV data using Rstudio

Once they're installed, browse over to the file you want…



Name: dataset

Skip: 0

☑ First Row as Names

☑ Trim Spaces

☑ Open Data Viewer

Delimiter: Comma

Quotes: Default

Locale: Configure...

Escape: None

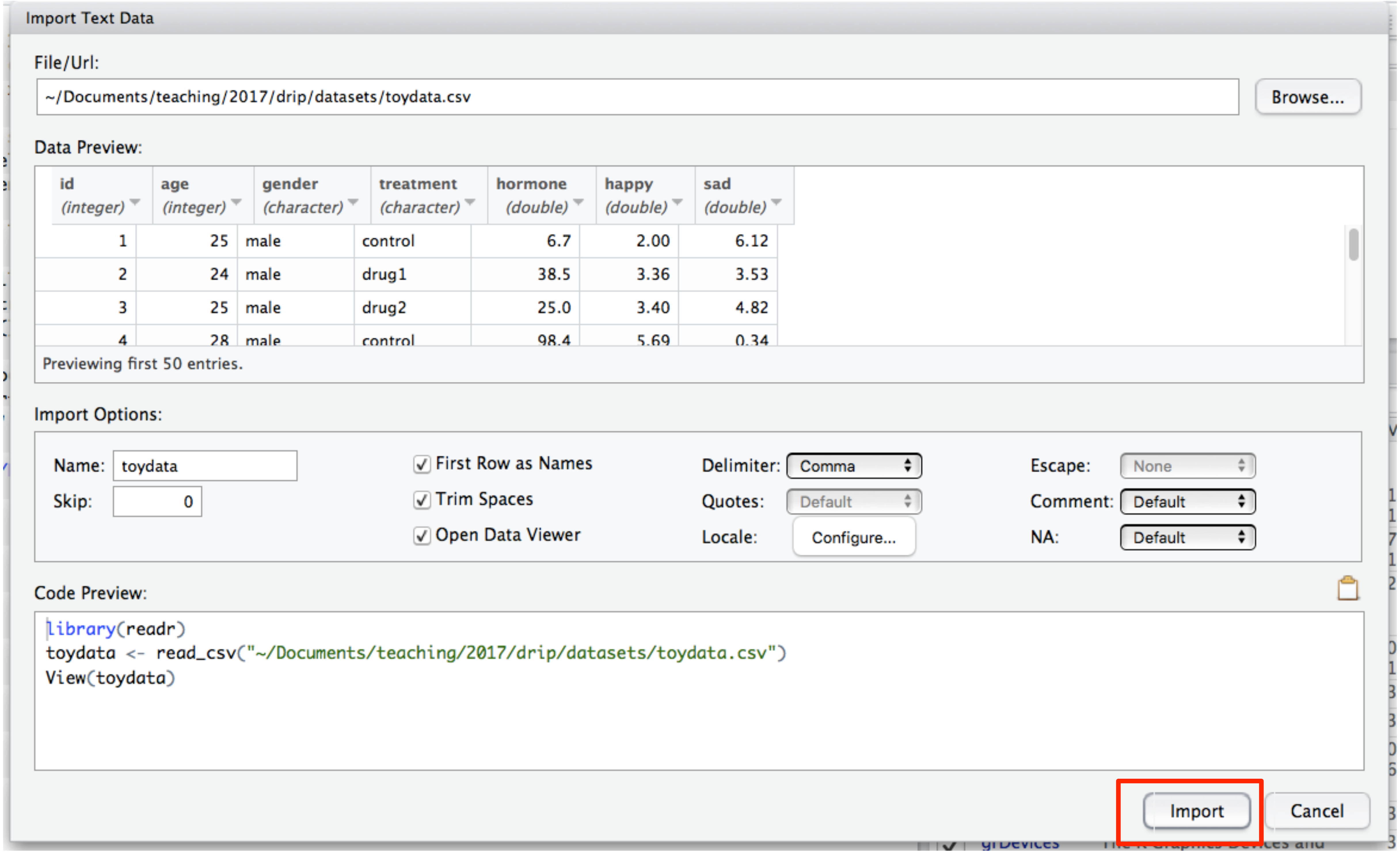
Comment: Default

NA: Default

```
library(readr)
dataset <- read_csv(NULL)
View(dataset)
```

Importing CSV data using Rstudio

When you see it, go ahead and “Import”



File/Url: ~/Documents/teaching/2017/drip/datasets/toydata.csv

Data Preview:

id (integer)	age (integer)	gender (character)	treatment (character)	hormone (double)	happy (double)	sad (double)
1	25	male	control	6.7	2.00	6.12
2	24	male	drug1	38.5	3.36	3.53
3	25	male	drug2	25.0	3.40	4.82
4	28	male	control	98.4	5.69	0.34

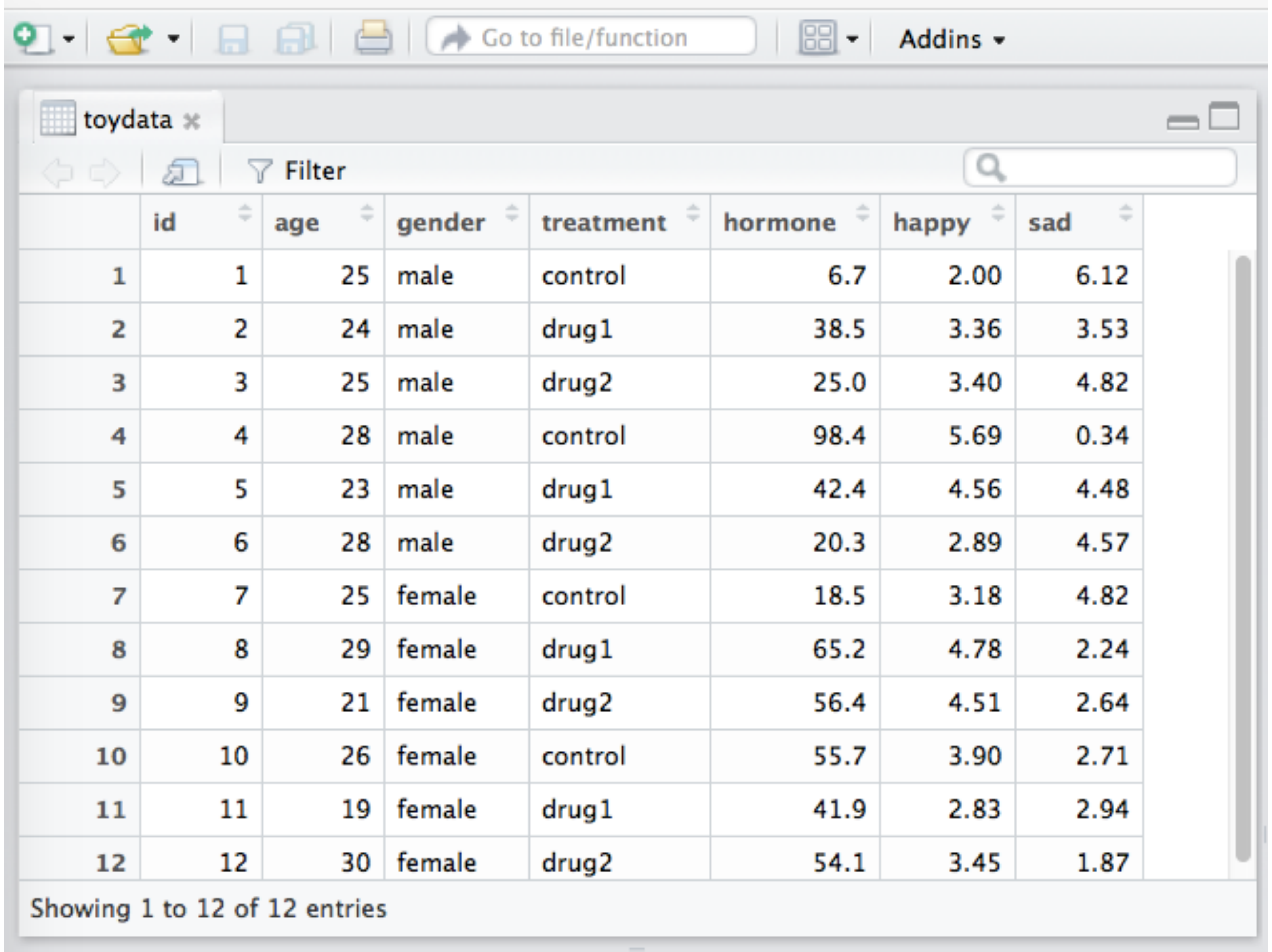
Previewing first 50 entries.

Import Options:

Name: toydata
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)
toydata <- read_csv("~/Documents/teaching/2017/drip/datasets/toydata.csv")
View(toydata)
```



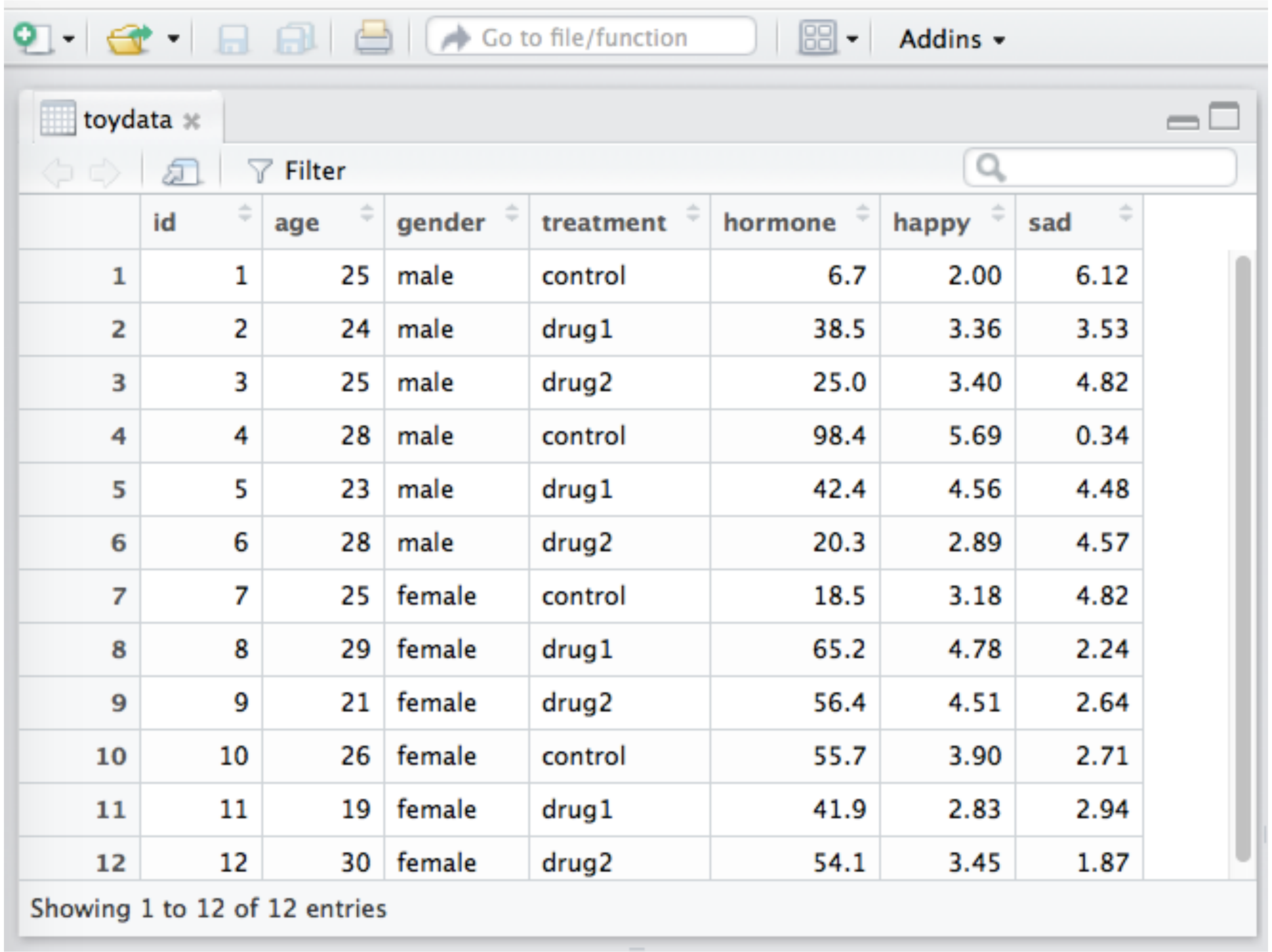
	id	age	gender	treatment	hormone	happy	sad
1	1	25	male	control	6.7	2.00	6.12
2	2	24	male	drug1	38.5	3.36	3.53
3	3	25	male	drug2	25.0	3.40	4.82
4	4	28	male	control	98.4	5.69	0.34
5	5	23	male	drug1	42.4	4.56	4.48
6	6	28	male	drug2	20.3	2.89	4.57
7	7	25	female	control	18.5	3.18	4.82
8	8	29	female	drug1	65.2	4.78	2.24
9	9	21	female	drug2	56.4	4.51	2.64
10	10	26	female	control	55.7	3.90	2.71
11	11	19	female	drug1	41.9	2.83	2.94
12	12	30	female	drug2	54.1	3.45	1.87

Showing 1 to 12 of 12 entries

Rstudio opens a tab showing you the contents of the data frame you just imported

These are the actual R commands that Rstudio used to import the data

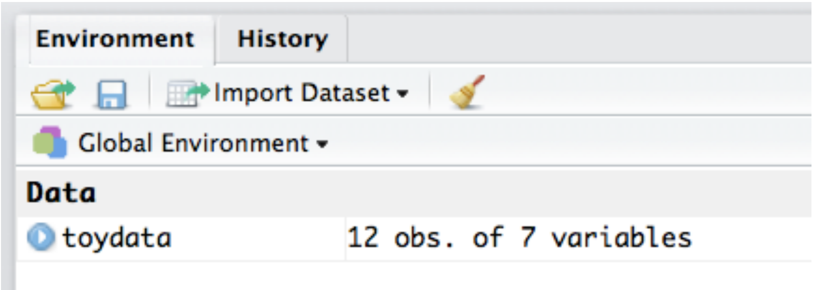
```
> toydata <- read_csv(“~/Documents/teaching/2017/drip/datasets/toydata.csv”)
>   View(toydata)
```



	id	age	gender	treatment	hormone	happy	sad
1	1	25	male	control	6.7	2.00	6.12
2	2	24	male	drug1	38.5	3.36	3.53
3	3	25	male	drug2	25.0	3.40	4.82
4	4	28	male	control	98.4	5.69	0.34
5	5	23	male	drug1	42.4	4.56	4.48
6	6	28	male	drug2	20.3	2.89	4.57
7	7	25	female	control	18.5	3.18	4.82
8	8	29	female	drug1	65.2	4.78	2.24
9	9	21	female	drug2	56.4	4.51	2.64
10	10	26	female	control	55.7	3.90	2.71
11	11	19	female	drug1	41.9	2.83	2.94
12	12	30	female	drug2	54.1	3.45	1.87

Showing 1 to 12 of 12 entries

And there it is in the workspace!

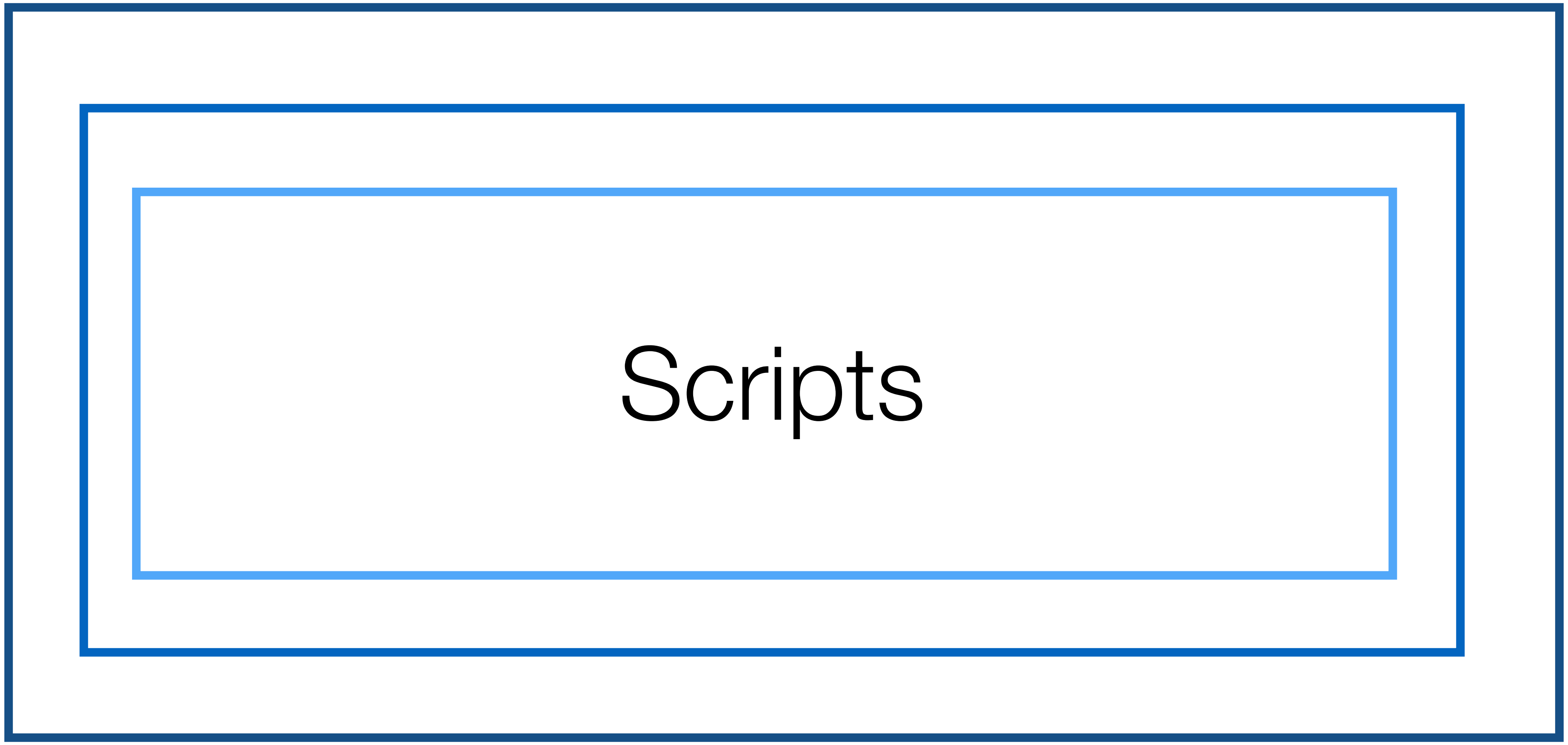


Data

toydata 12 obs. of 7 variables

These are the actual R commands that Rstudio used to import the data

```
> toydata <- read_csv(“~/Documents/teaching/2017/drip/datasets/toydata.csv”)
>   View(toydata)
```



Scripts

Working with data

- What do we know how do to?
  - Load data from .Rdata files and .csv files
  - Type commands to get R to make output
  - Save data / R output to .Rdata files
  - Install and load packages to extend R functionality

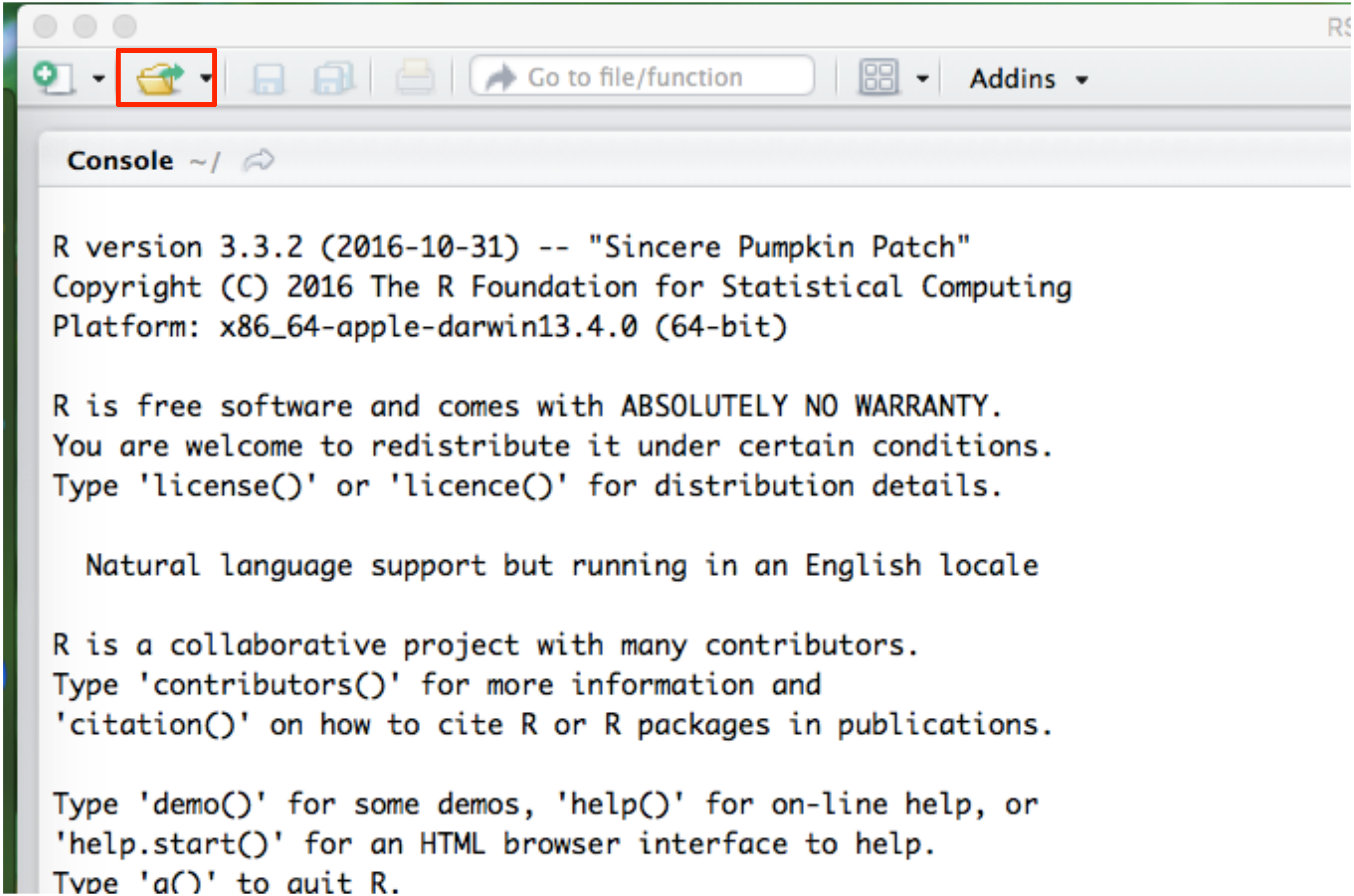
- What's missing?
  - How to save a collection of R commands to run later
  - i.e. scripts

Scripts

- What is an R script?
  - R scripts are text files, and have a .R extension
  - They contain a sequence of R commands that R will execute when the script is “sourced” (i.e., run)

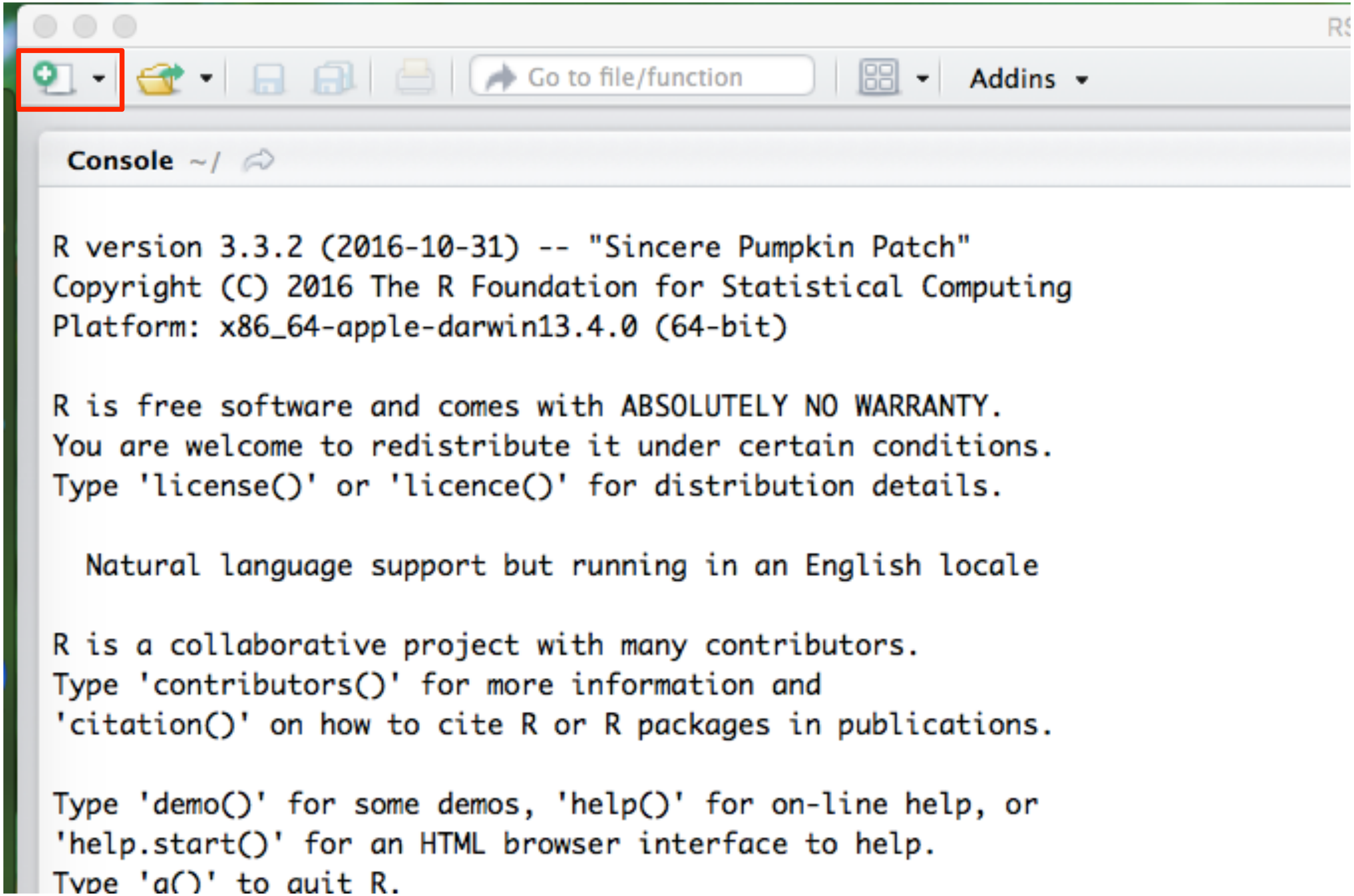
- How do I use scripts?
  - Type (or paste) R commands into the text file
  - Save the script (usually in the same folder as the data)
  - Use the “source” button to run it.

Click here to open a saved script



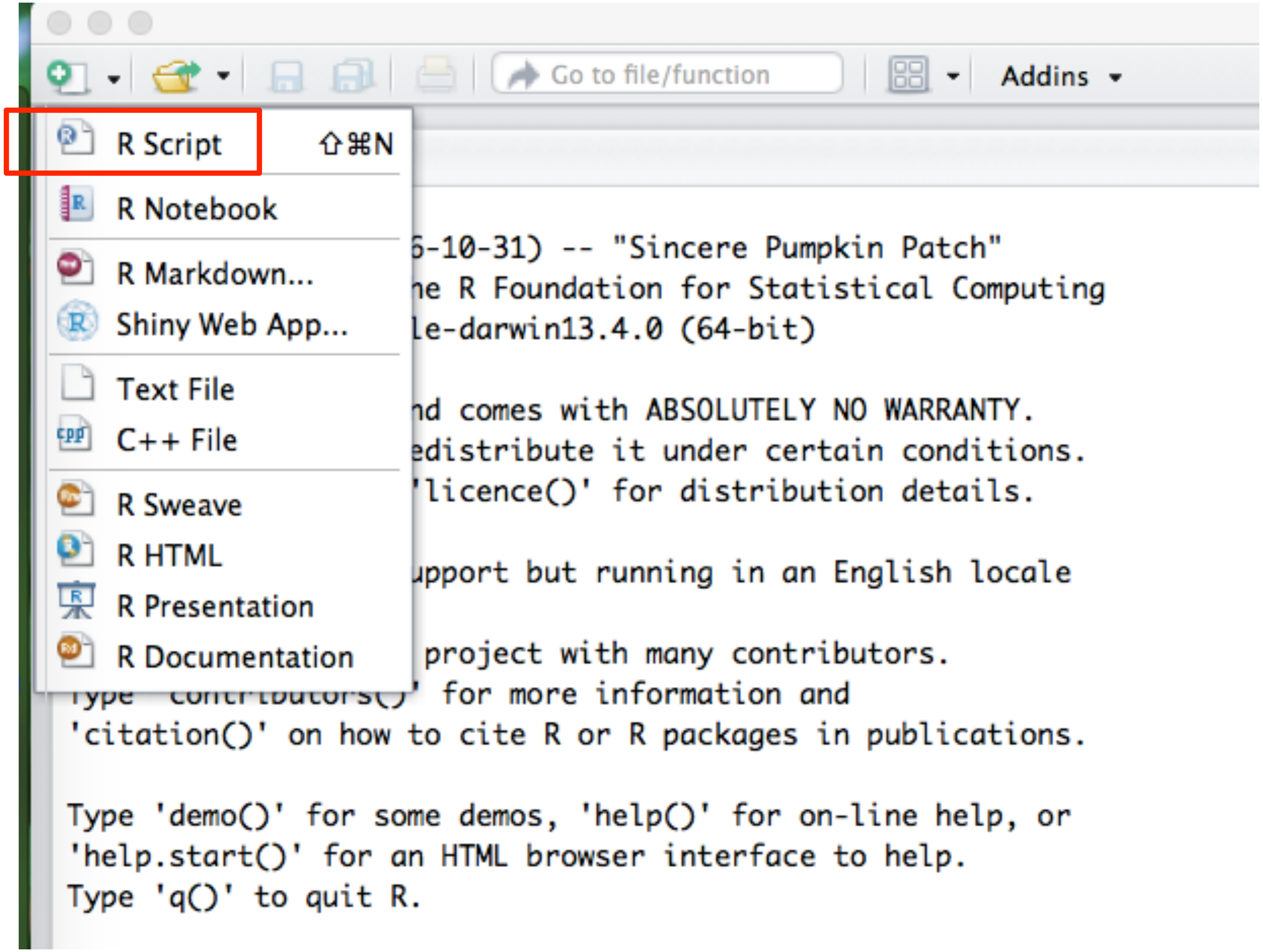
```
R version 3.3.2 (2016-10-31) -- "Sincere Pumpkin Patch"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (64-bit)

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.

  Natural language support but running in an English locale

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.
```

Or here to create a new one



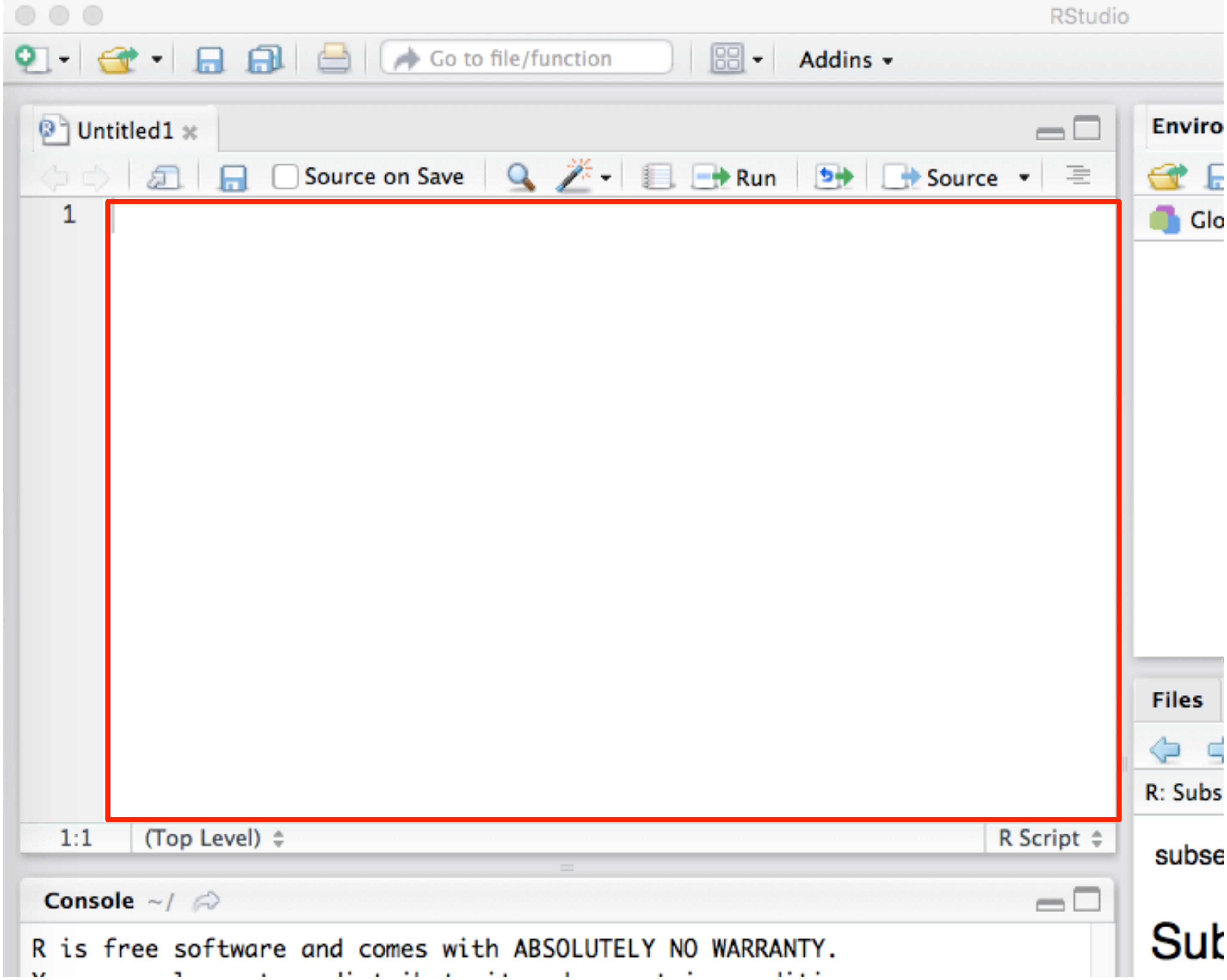
```
R version 3.3.2 (2016-10-31) -- "Sincere Pumpkin Patch"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (64-bit)

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.

  Natural language support but running in an English locale

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.
```

Or here to create a new one



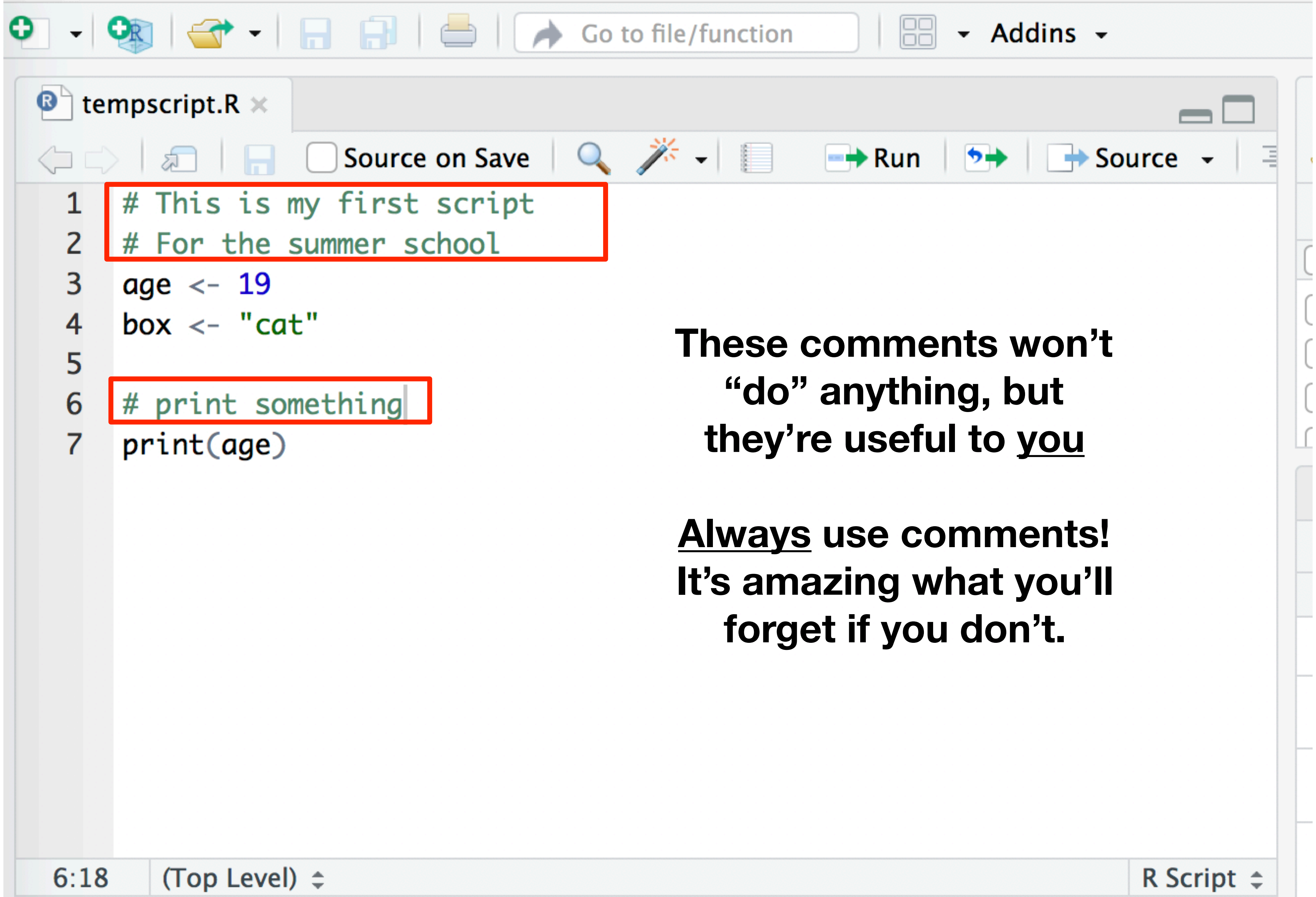
```
6-10-31) -- "Sincere Pumpkin Patch"
he R Foundation for Statistical Computing
le-darwin13.4.0 (64-bit)

nd comes with ABSOLUTELY NO WARRANTY.
edistribute it under certain conditions.
'licence()' for distribution details.

upport but running in an English locale

 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.
```

An empty script…



```
R is free software and comes with ABSOLUTELY NO WARRANTY.
```

Type some R commands here…

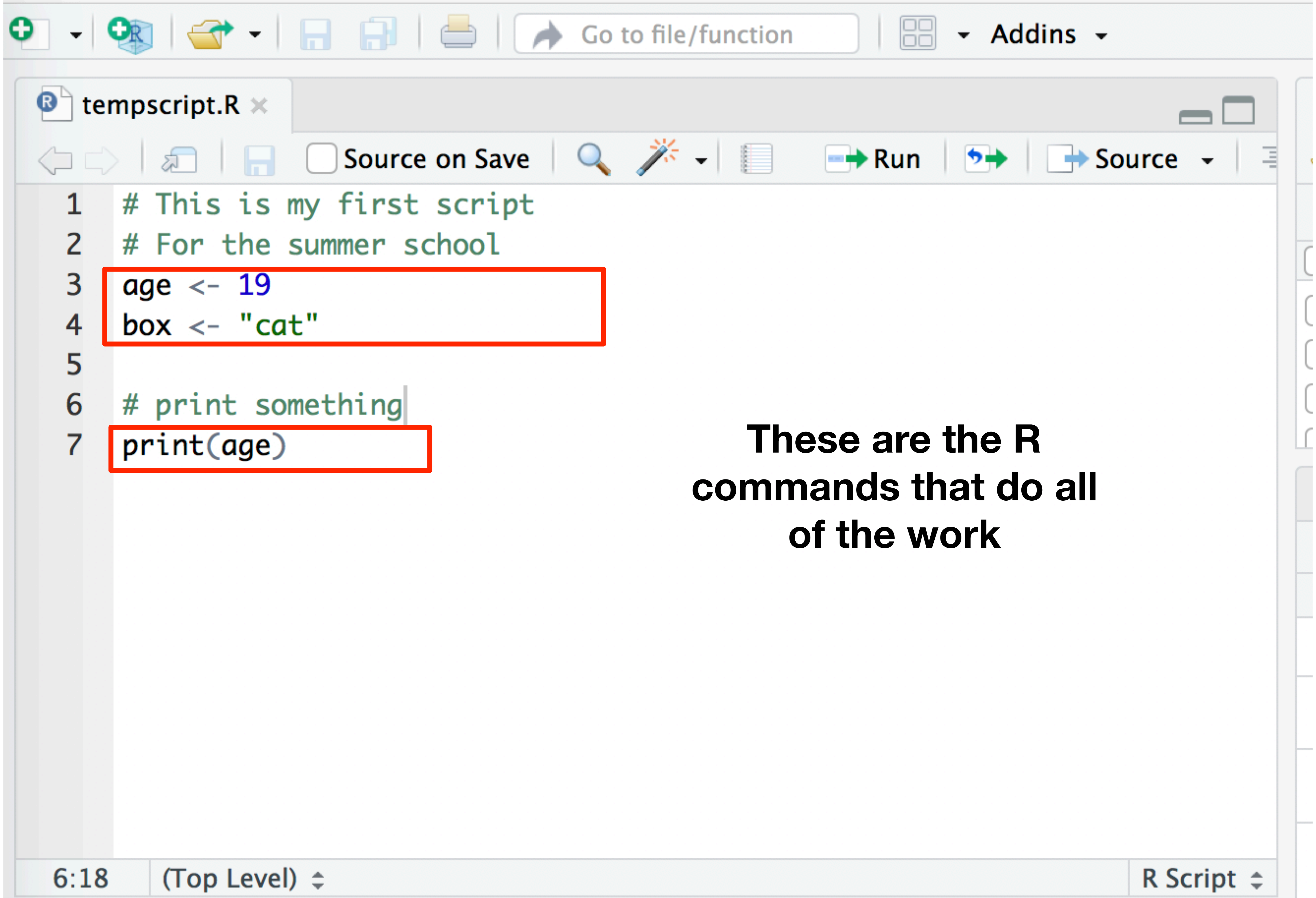


```r
# This is my first script
# For the summer school
age <- 19
box <- "cat"

# print something
print(age)
```

These comments won't "do" anything, but they're useful to you

Always use comments! It's amazing what you'll forget if you don't.

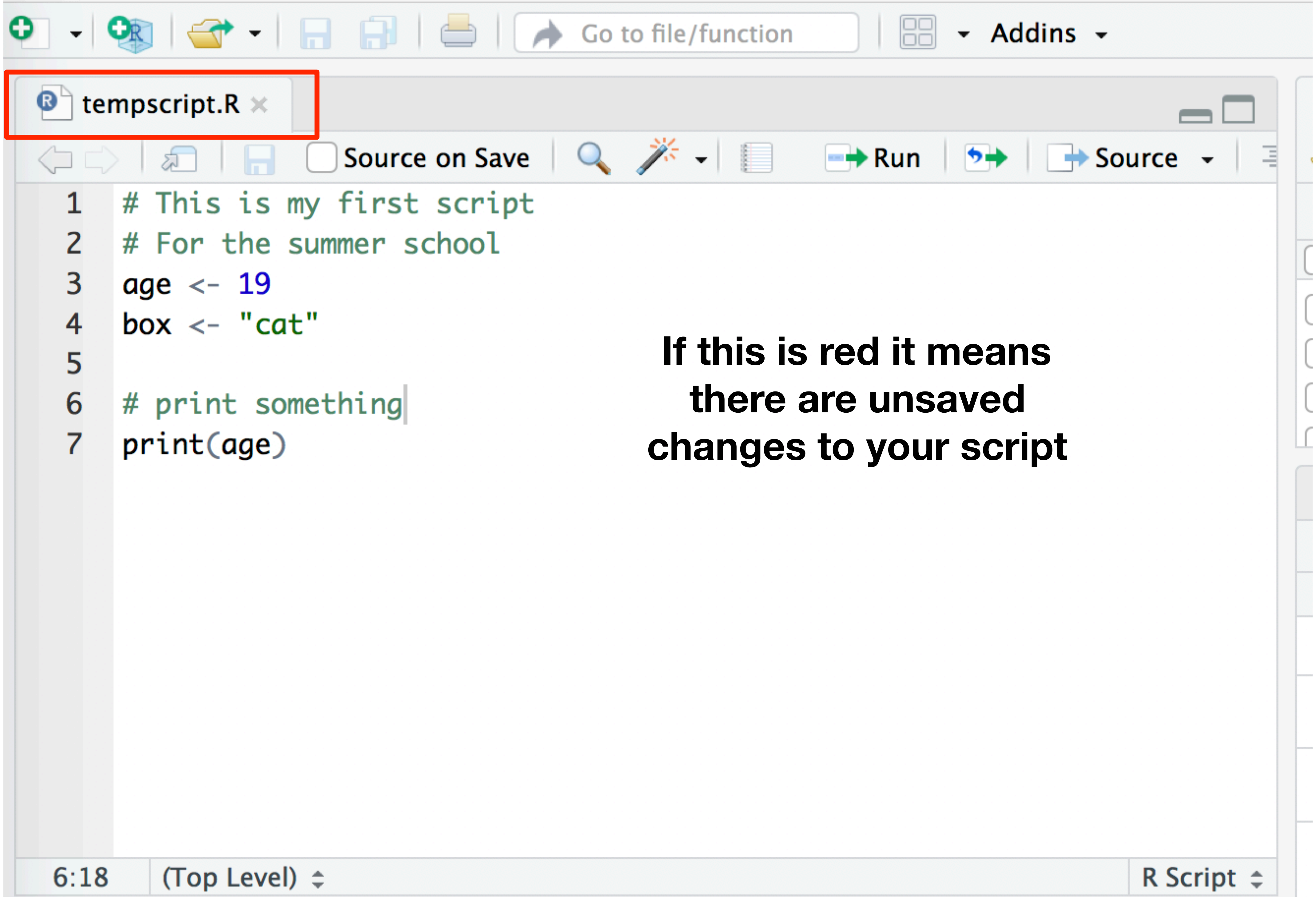
Type some R commands here…



```
# This is my first script
# For the summer school
age <- 19
box <- "cat"

# print something
print(age)
```

These are the R commands that do all of the work

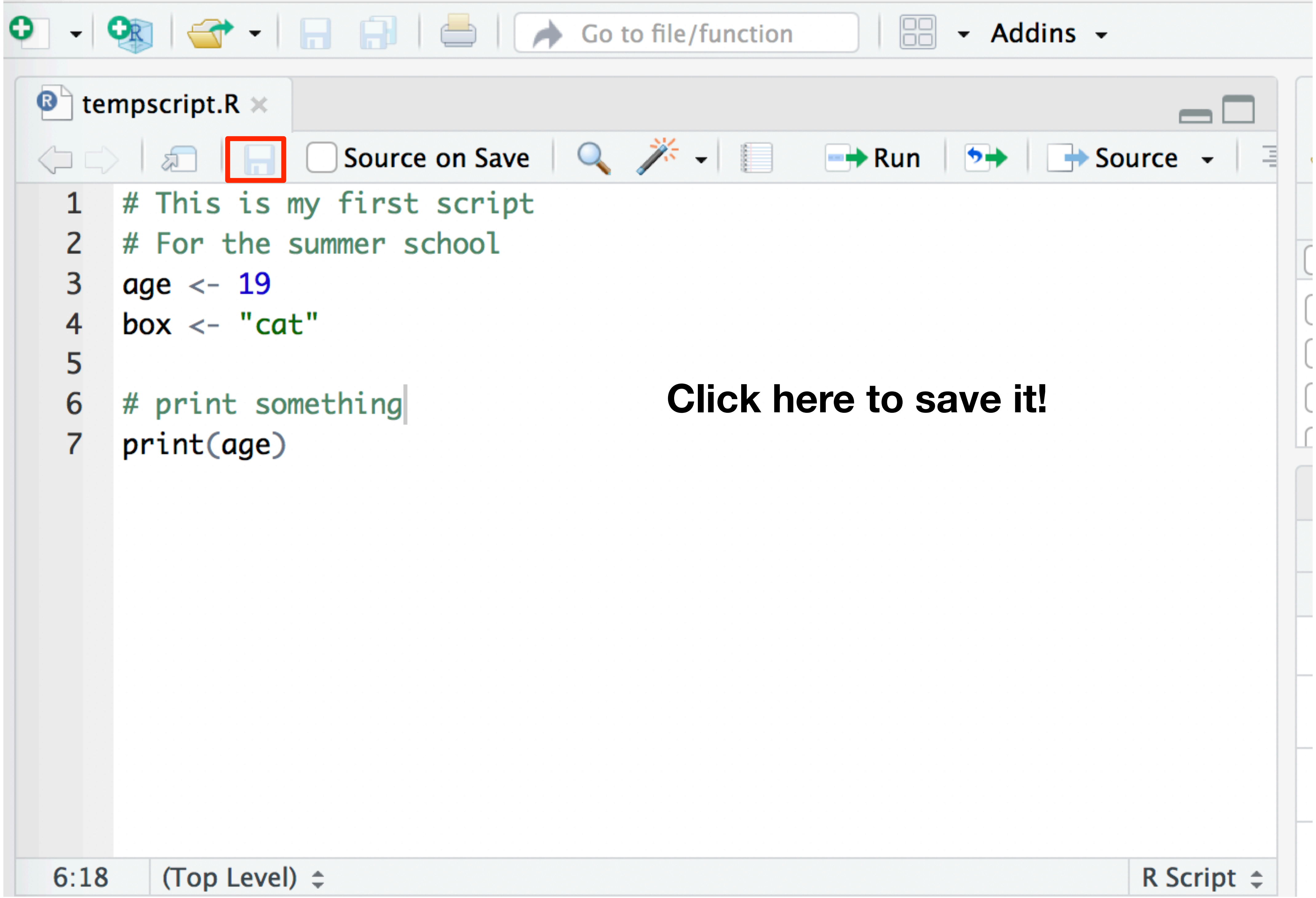
Type some R commands here…



```
# This is my first script
# For the summer school
age <- 19
box <- "cat"

# print something
print(age)
```

If this is red it means there are unsaved changes to your script

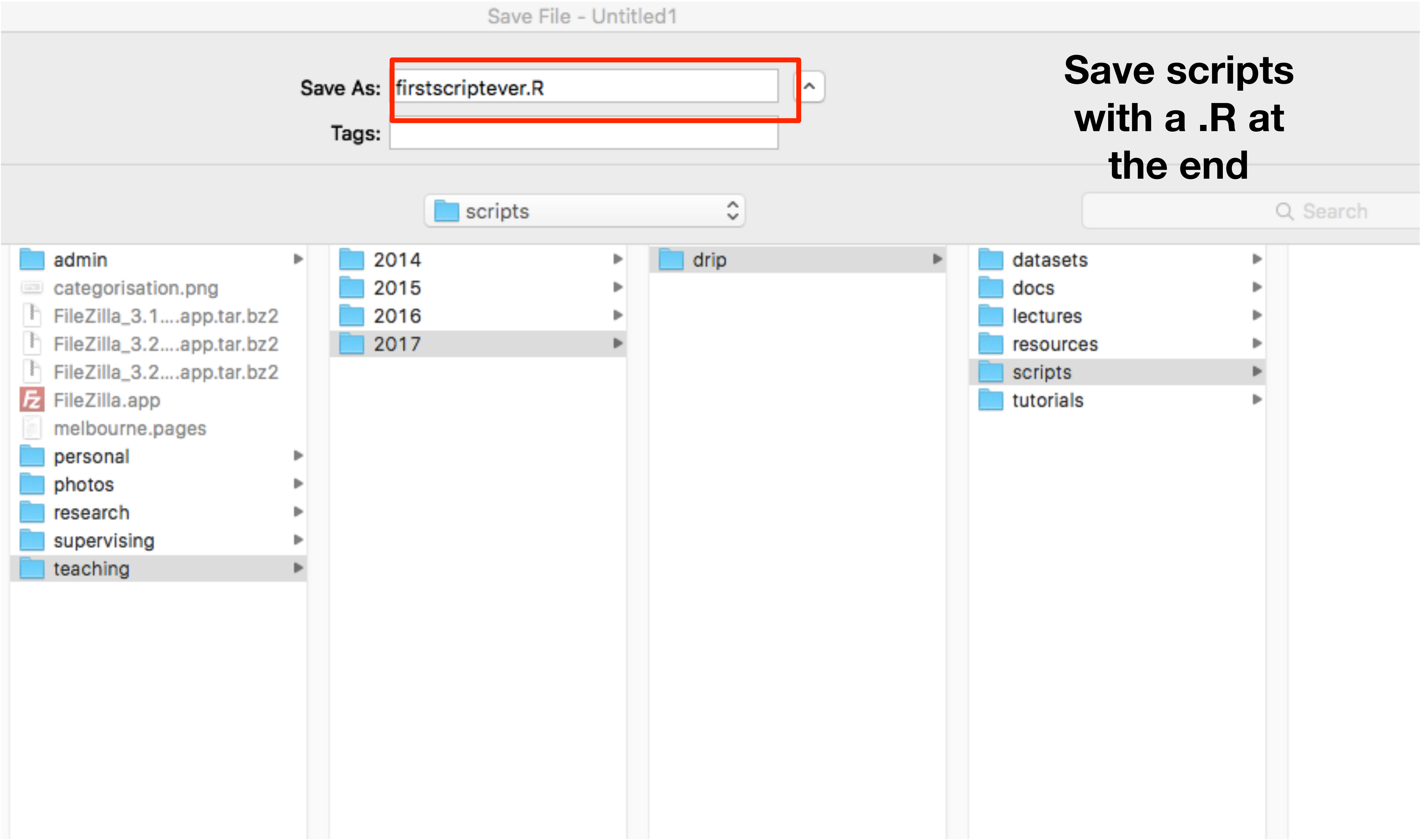
Type some R commands here…



```r
# This is my first script
# For the summer school
age <- 19
box <- "cat"

# print something
print(age)
```

Click here to save it!

Hey look, another save window!



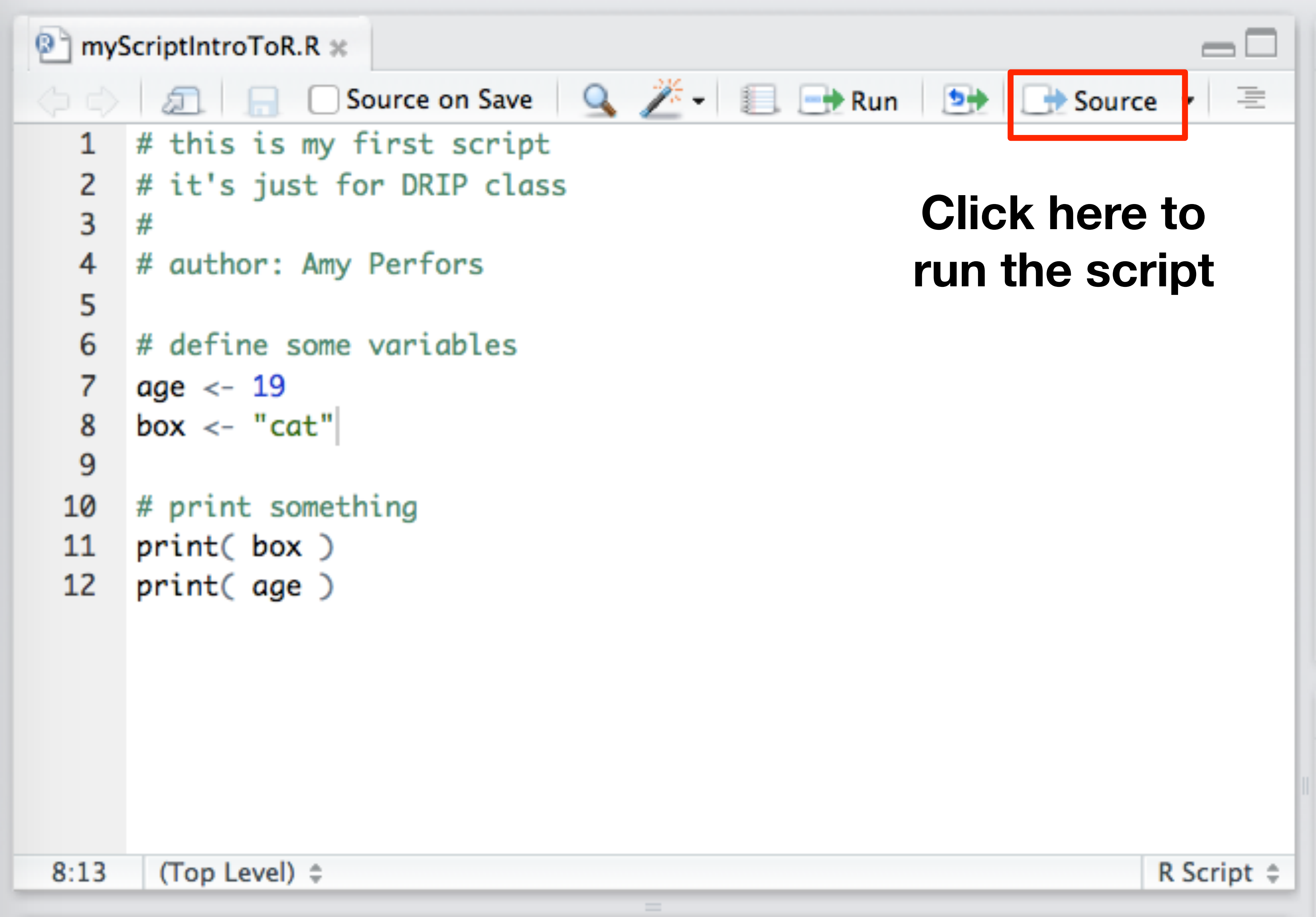
Save File - Untitled1

Save As: firstscriptever.R

Tags:

scripts

Save scripts with a .R at the end



```
# this is my first script
# it's just for DRIP class
#
# author: Amy Perfors

# define some variables
age <- 19
box <- "cat"

# print something
print( box )
print( age )
```

Click here to run the script

Scripts “run” from
top to bottom

```
# this is my first script
# for the summer school

# define some variables
age <- 19
box <- "cat"

# print something
print (age)
```

What does R do?

nothing; these are comments

```
# this is my first script
# for the summer school

# define some variables
```

What does R do?

create a variable called age with the value 19

```
# this is my first script
# for the summer school

# define some variables
age <- 19
```

What does R do?

create a variable called box with the value “cat”

```
# this is my first script
# for the summer school

# define some variables
age <- 19
box <- "cat"
```

What does R do?

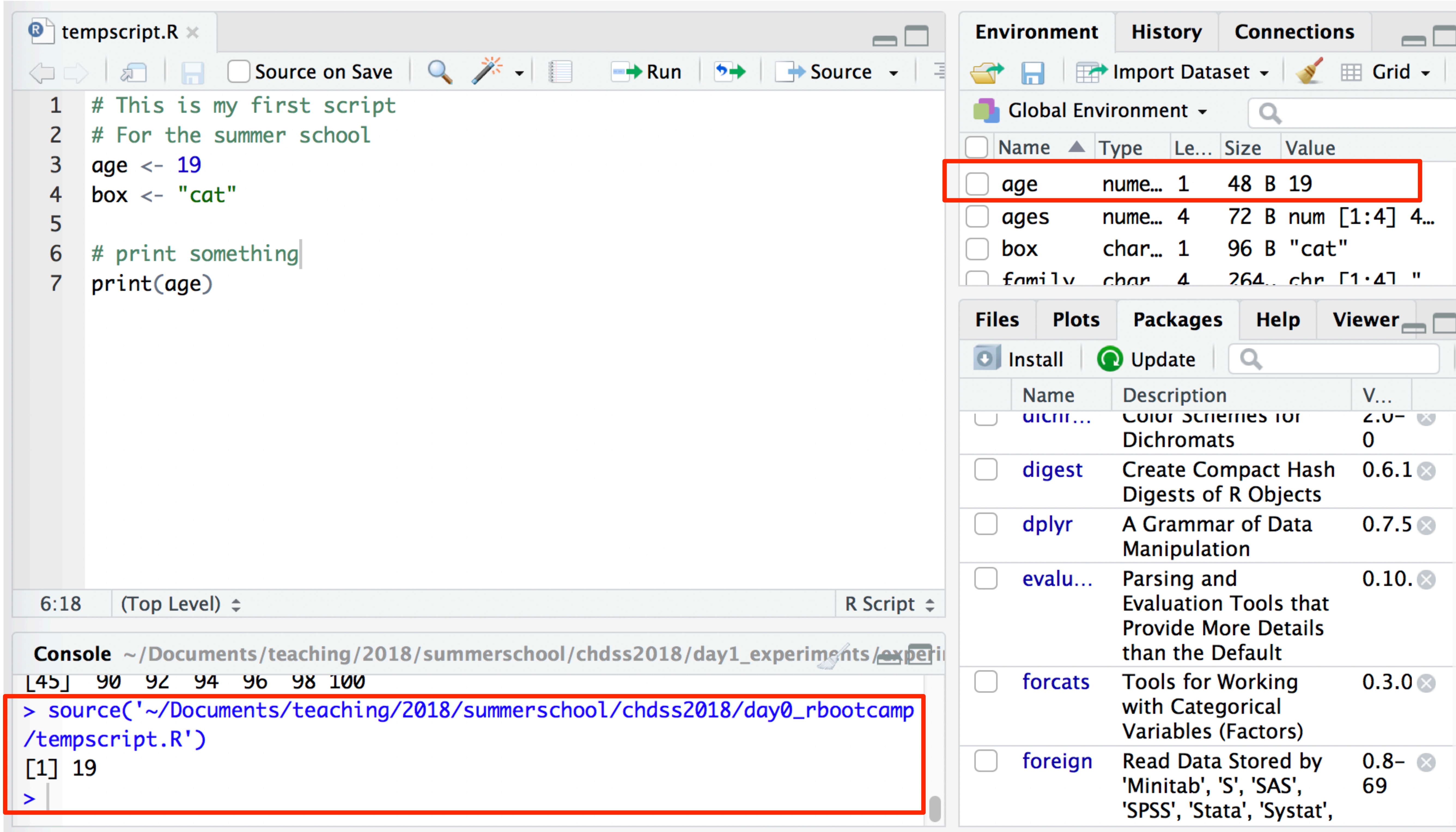
```
# this is my first script
# for the summer school

# define some variables
age <- 19
box <- "cat"

# print something
```

nothing; this is an empty line and a comment

What does R do?

```
# this is my first script
# for the summer school

# define some variables
age <- 19
box <- "cat"

# print something
print (age)
```

print the value in the variable age



```
# This is my first script
# For the summer school
age <- 19
box <- "cat"

# print something
print(age)
```

Name	Type	Le...	Size	Value
age	nume…	1	48 B	19
ages	nume…	4	72 B	num [1:4] 4…
box	char…	1	96 B	"cat"
family	char…	4	264…	chr [1:4] "

Name	Description	V...
dichr…	Color Schemes for Dichromats	2.0-0
digest	Create Compact Hash Digests of R Objects	0.6.1
dplyr	A Grammar of Data Manipulation	0.7.5
evalu…	Parsing and Evaluation Tools that Provide More Details than the Default	0.10.
forcats	Tools for Working with Categorical Variables (Factors)	0.3.0
foreign	Read Data Stored by 'Minitab', 'S', 'SAS', 'SPSS', 'Stata', 'Systat',	0.8-69

```
[45]  90  92  94  96  98 100
> source('~/Documents/teaching/2018/summerschool/chdss2018/day0_rbootcamp
/tempscript.R')
[1] 19
>
```

Things have happened!

Help

Suppose you want to know more about a function…

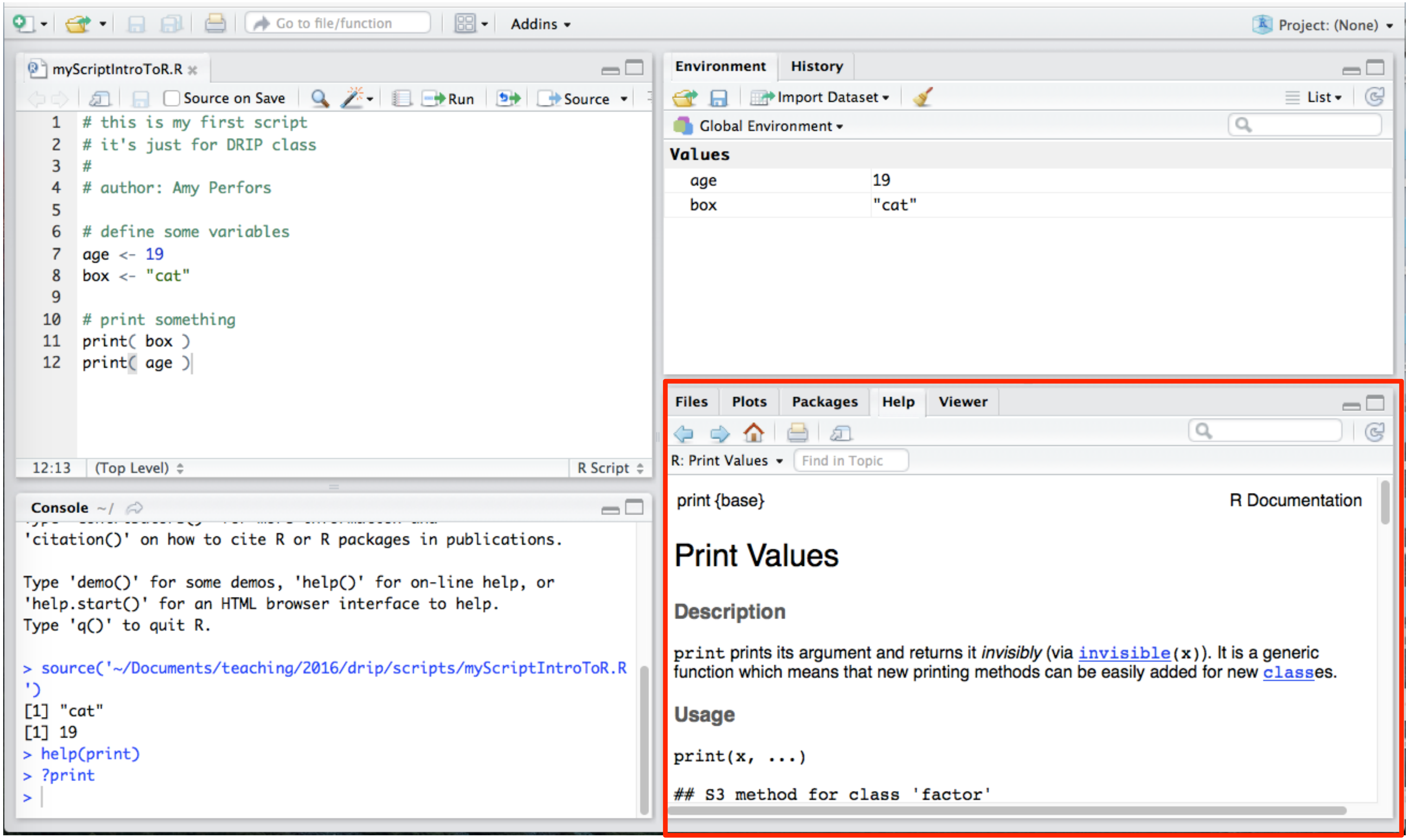
```
# print something
print( box )
print( age )
```

Every function comes with documentation

`help(print)` or `?print`

R documentation

When you type `help()`, it shows up in the lower right panel



```r
# this is my first script
# it's just for DRIP class
#
# author: Amy Perfors

# define some variables
age <- 19
box <- "cat"

# print something
print( box )
print( age )
```

Values

age	19
box	"cat"

```
'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.

> source('~/Documents/teaching/2016/drip/scripts/myScriptIntroToR.R
')
[1] "cat"
[1] 19
> help(print)
> ?print
>
```

print {base} R Documentation

Print Values

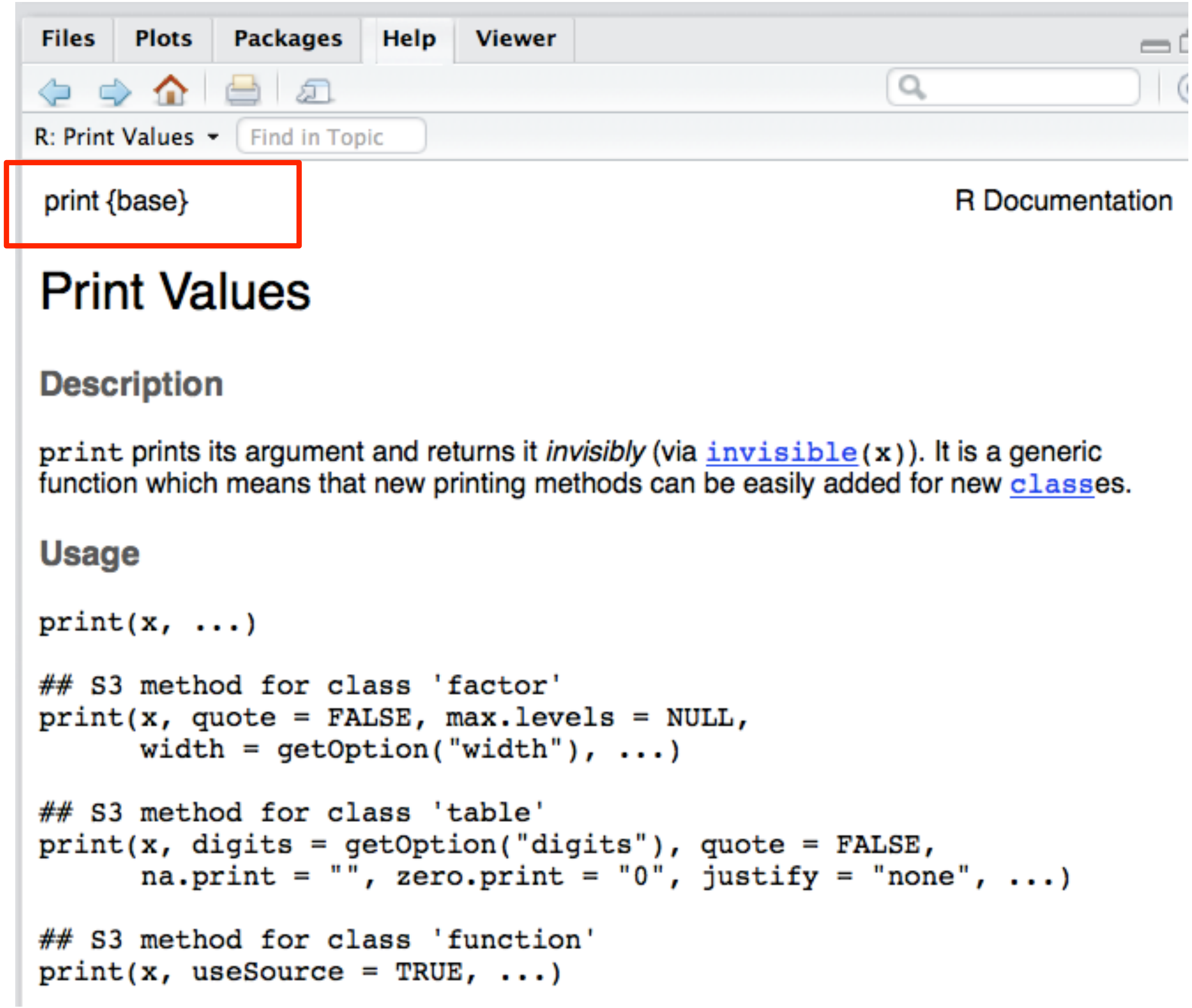
Description

`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'
```

R documentation



print {base}

R Documentation

Print Values

Description

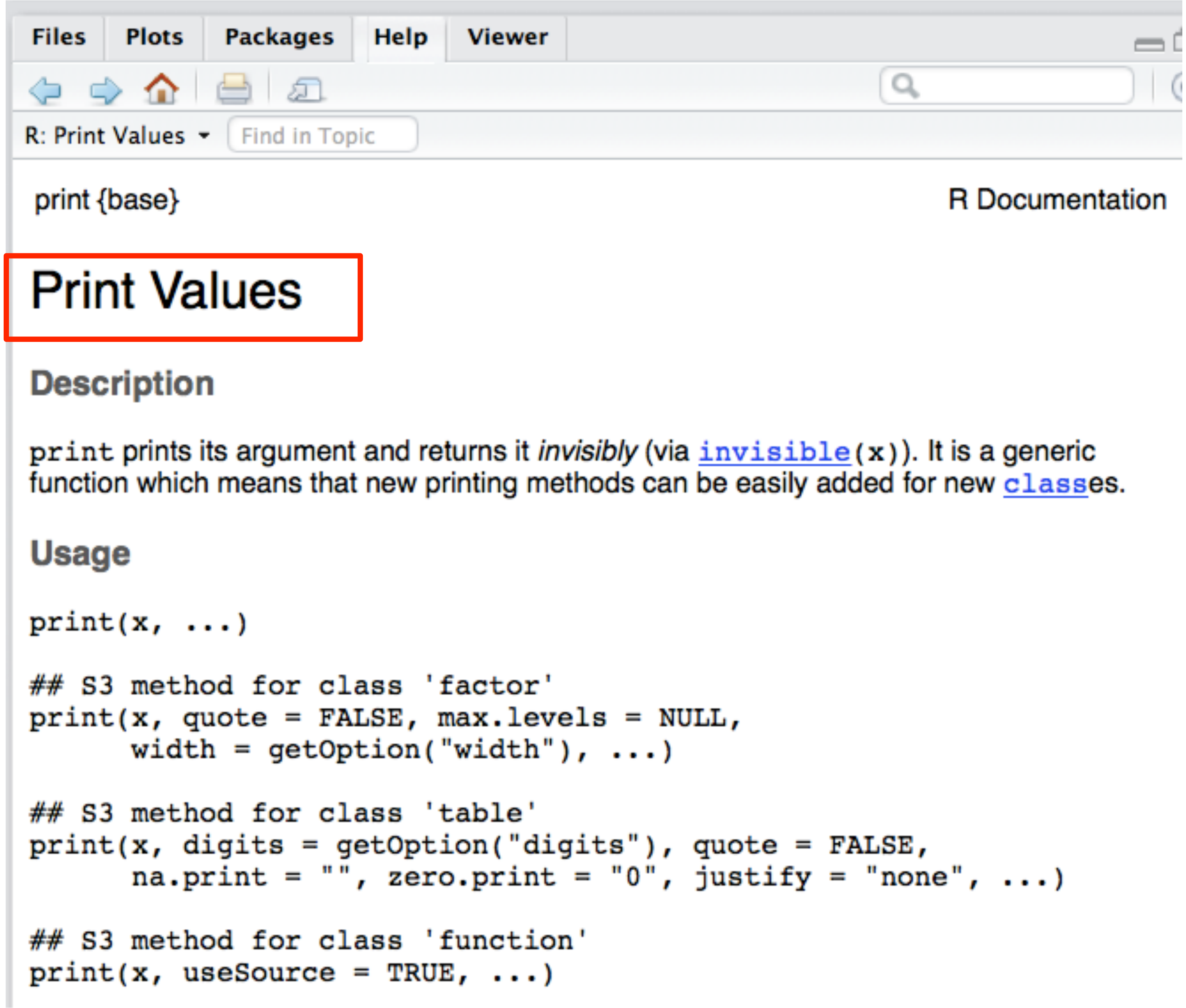
`print` prints its argument and returns it *invisibly* (via `invisible(x)`). It is a generic function which means that new printing methods can be easily added for new `class`es.

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", justify = "none", ...)

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

tells you what function the documentation is for

R documentation



print {base} R Documentation

Print Values

Description

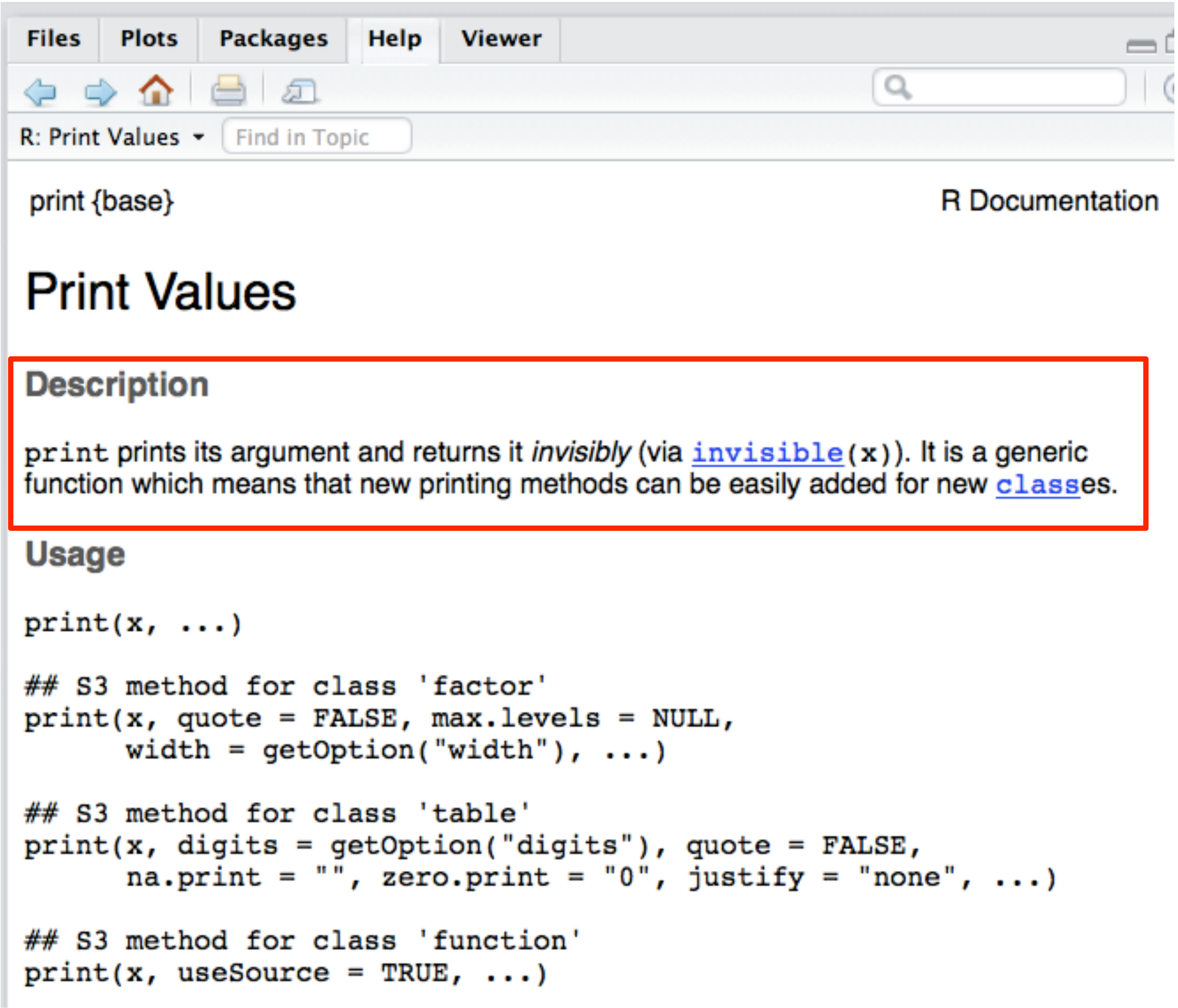
`print` prints its argument and returns it *invisibly* (via `invisible(x)`). It is a generic function which means that new printing methods can be easily added for new `class`es.

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", justify = "none", ...)

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

quickly describes what the function does

R documentation



print {base} R Documentation

Print Values

Description

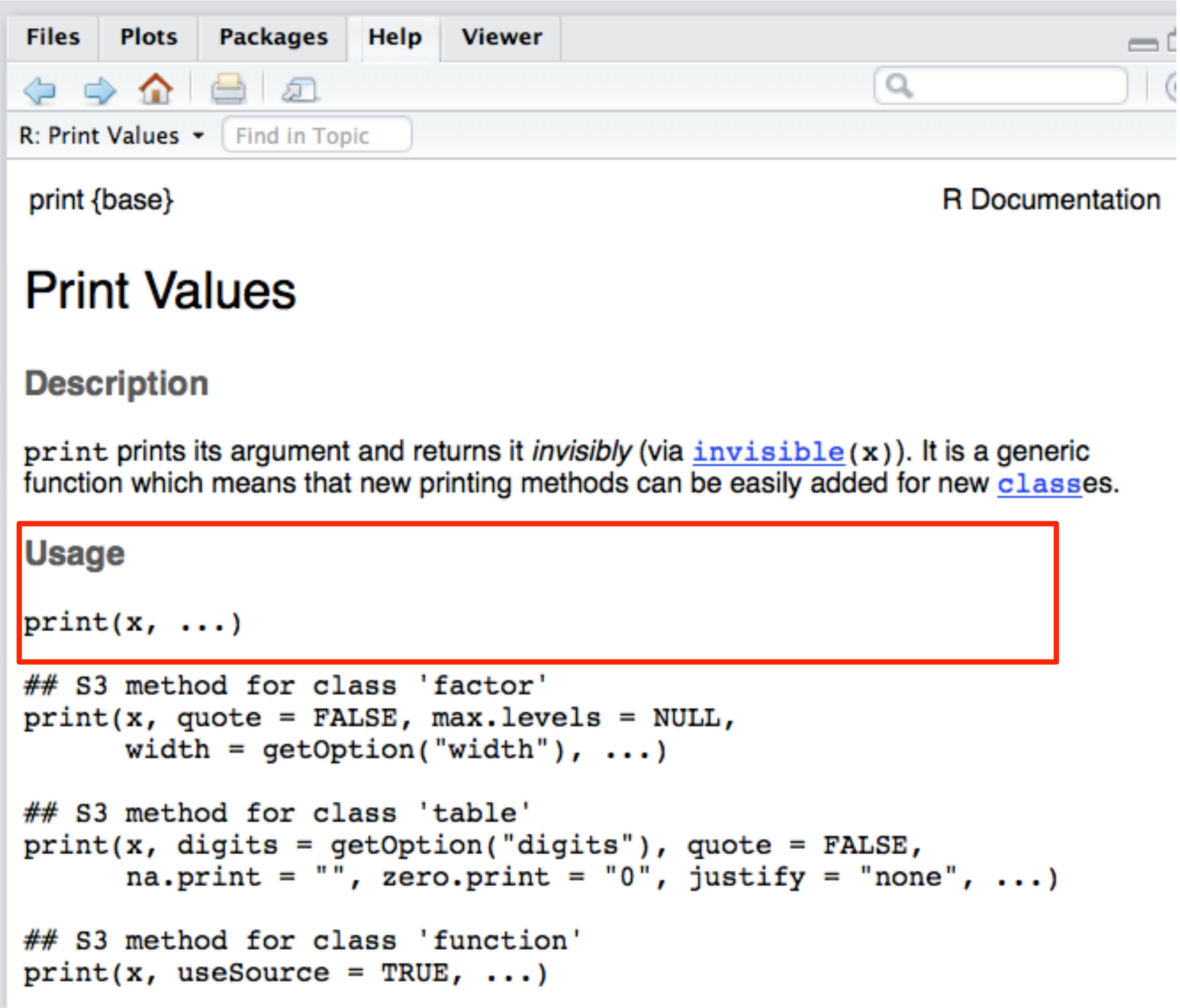
`print` prints its argument and returns it *invisibly* (via `invisible(x)`). It is a generic function which means that new printing methods can be easily added for new `class`es.

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", justify = "none", ...)

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

a longer
description of
what the
function does

R documentation



print {base}　R Documentation

Print Values

Description

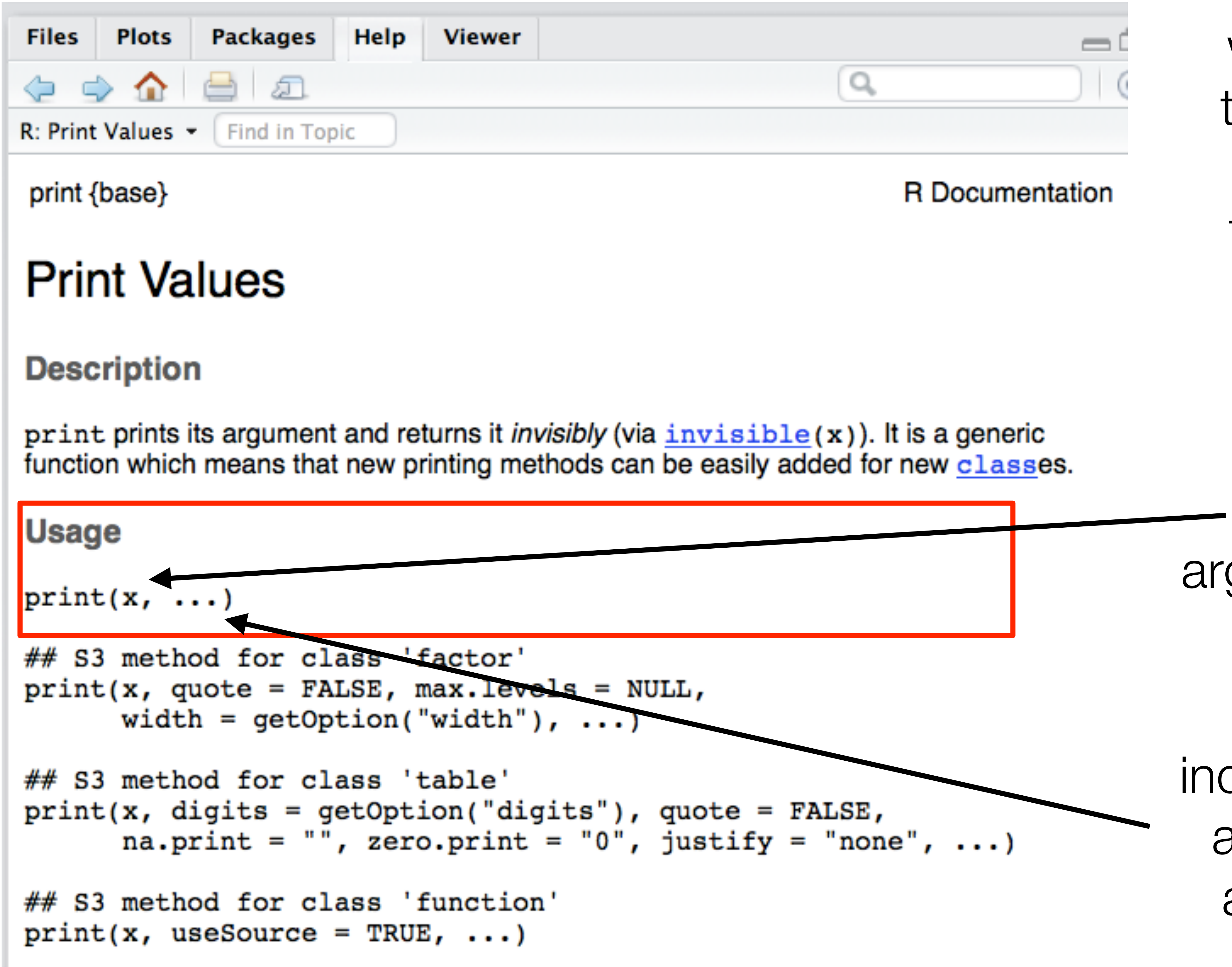
`print` prints its argument and returns it *invisibly* (via `invisible(x)`). It is a generic function which means that new printing methods can be easily added for new `class`es.

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", justify = "none", ...)

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

what you have to type in order to get the function to run

R documentation



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

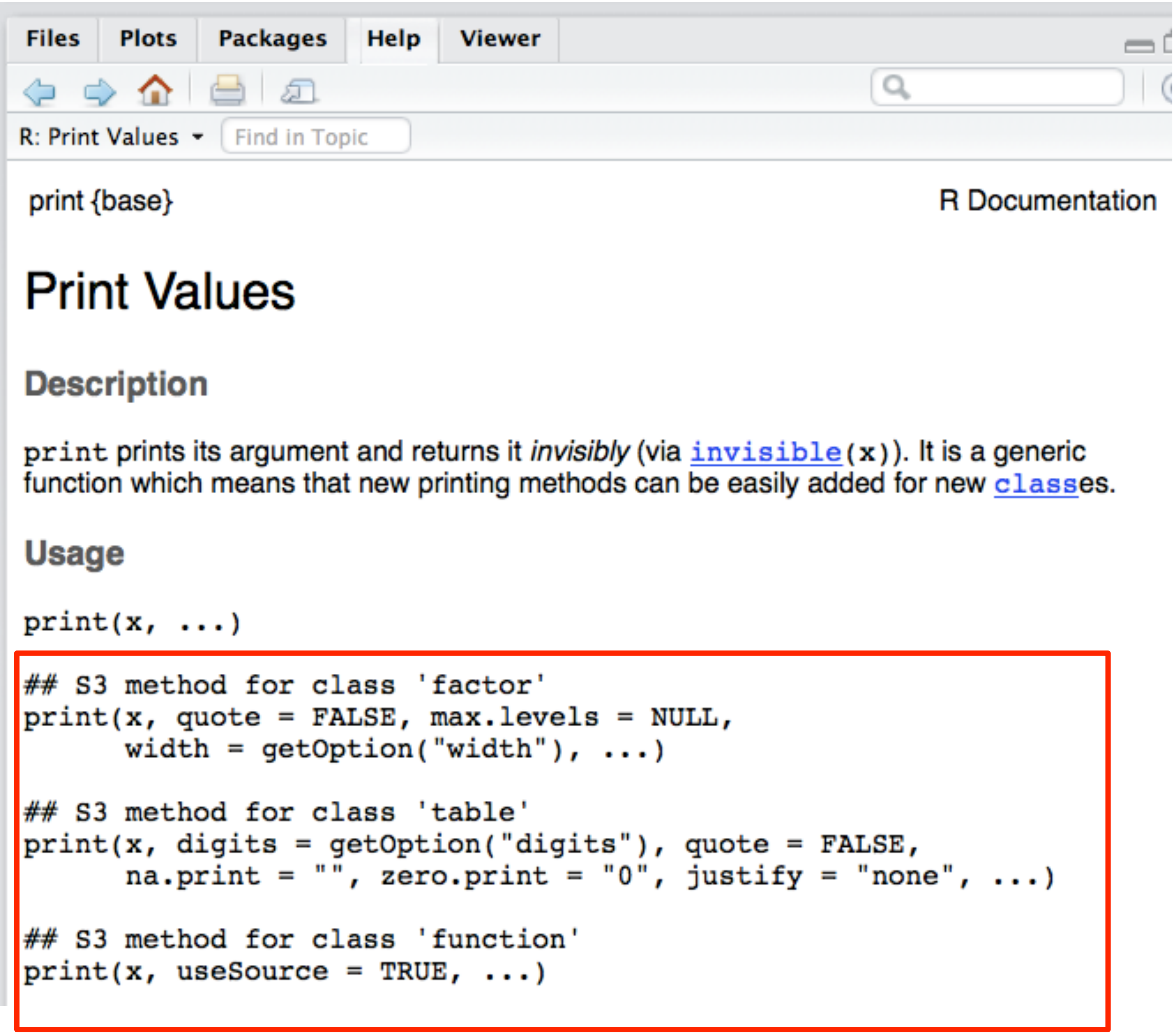
```
print(x, ...)

## 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", justify = "none", ...)

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

what you have
to type in order
to get the
function to run

which
arguments are
obligatory

indicates there
are optional
arguments

R documentation



print {base} R Documentation

Print Values

Description

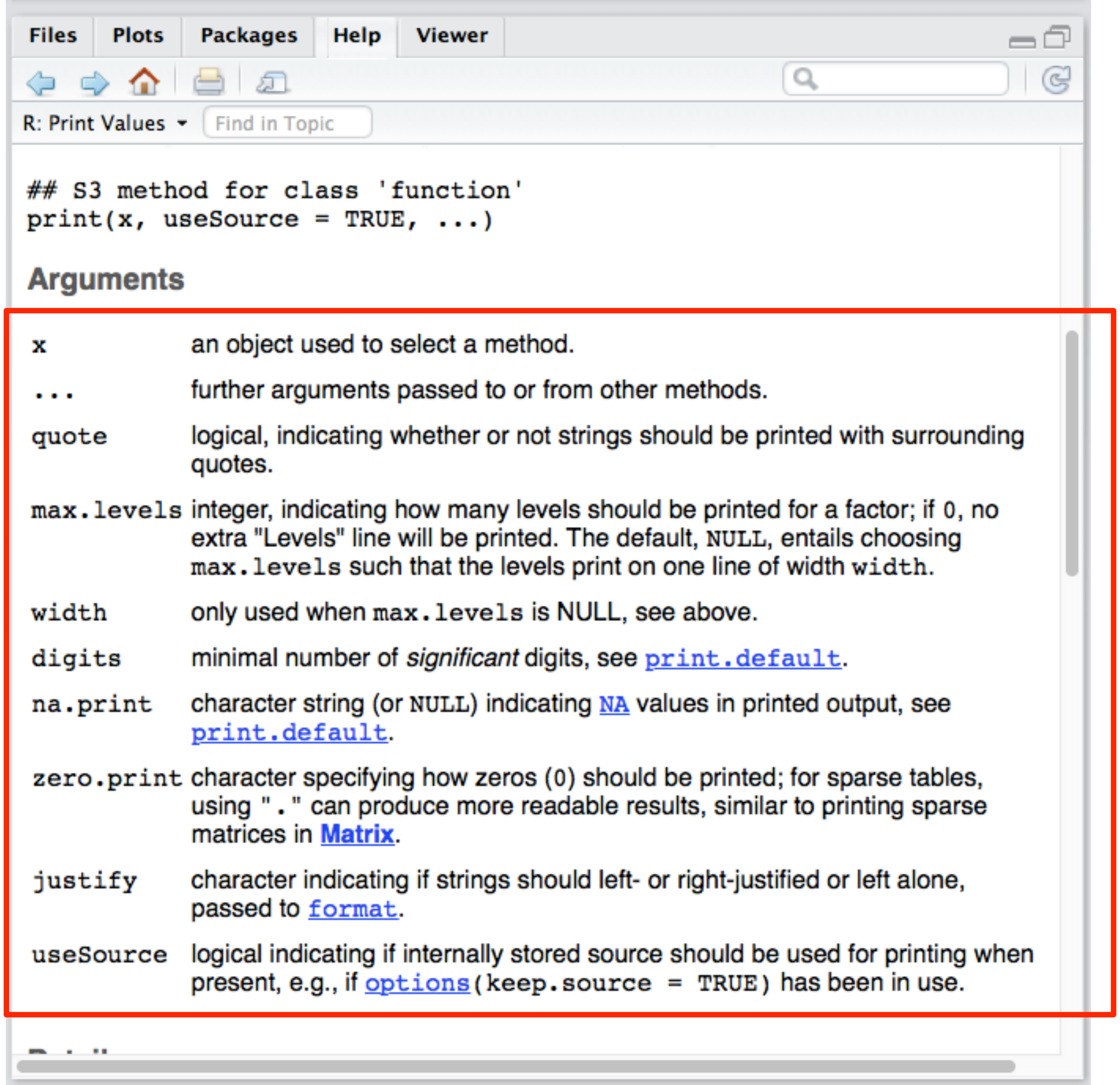
`print` prints its argument and returns it *invisibly* (via `invisible(x)`). It is a generic function which means that new printing methods can be easily added for new `class`es.

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", justify = "none", ...)

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

don’t worry
about this!

…scrolling down…



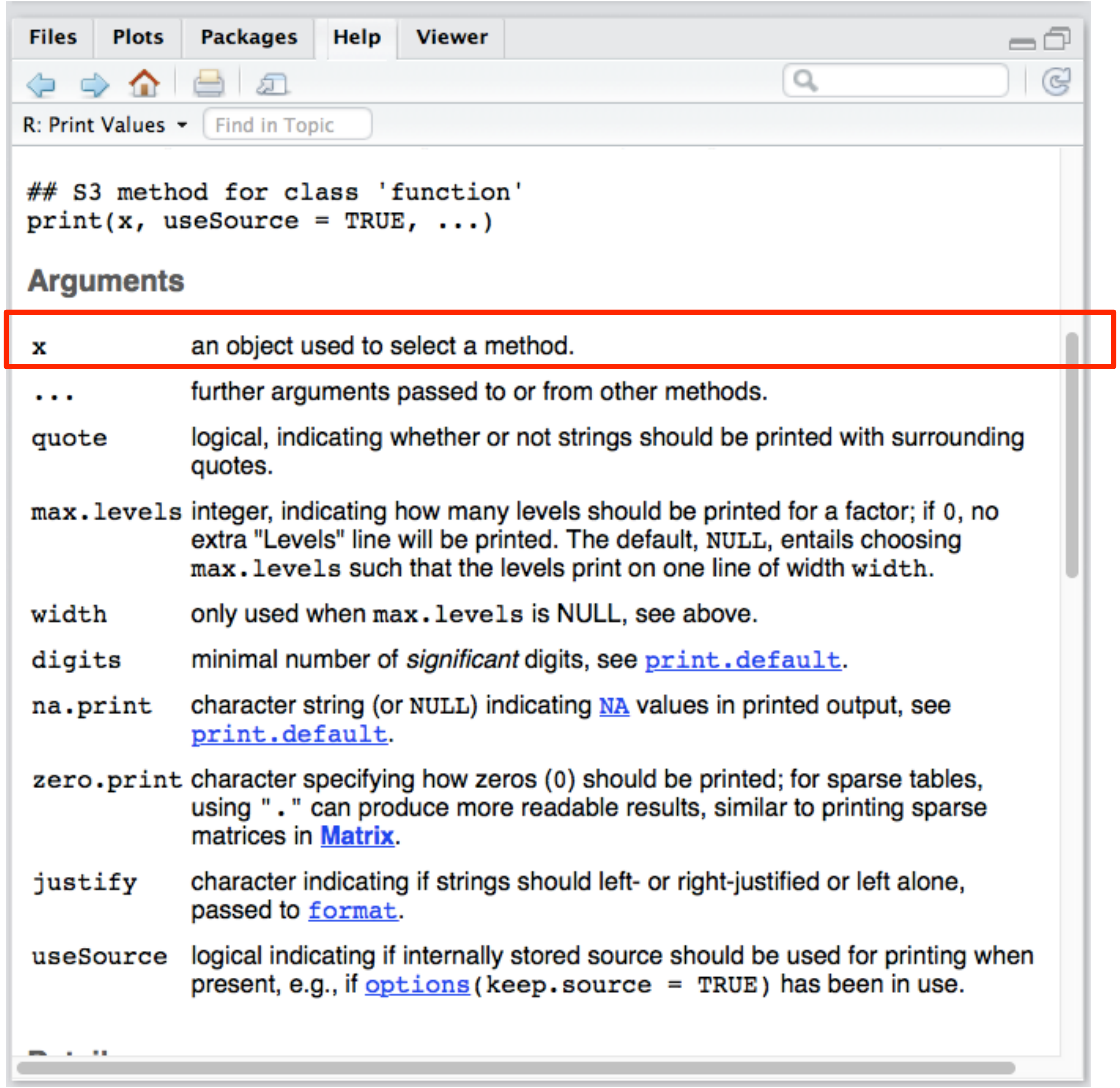
```
## S3 method for class 'function'
print(x, useSource = TRUE, ...)
```

Arguments

`x`	an object used to select a method.
`...`	further arguments passed to or from other methods.
`quote`	logical, indicating whether or not strings should be printed with surrounding quotes.
`max.levels`	integer, indicating how many levels should be printed for a factor; if `0`, no extra "Levels" line will be printed. The default, `NULL`, entails choosing `max.levels` such that the levels print on one line of width `width`.
`width`	only used when `max.levels` is NULL, see above.
`digits`	minimal number of *significant* digits, see `print.default`.
`na.print`	character string (or `NULL`) indicating NA values in printed output, see `print.default`.
`zero.print`	character specifying how zeros (`0`) should be printed; for sparse tables, using `"."` can produce more readable results, similar to printing sparse matrices in **Matrix**.
`justify`	character indicating if strings should left- or right-justified or left alone, passed to `format`.
`useSource`	logical indicating if internally stored source should be used for printing when present, e.g., if `options(keep.source = TRUE)` has been in use.

here it tells you what it needs to take as an argument

…scrolling down…



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

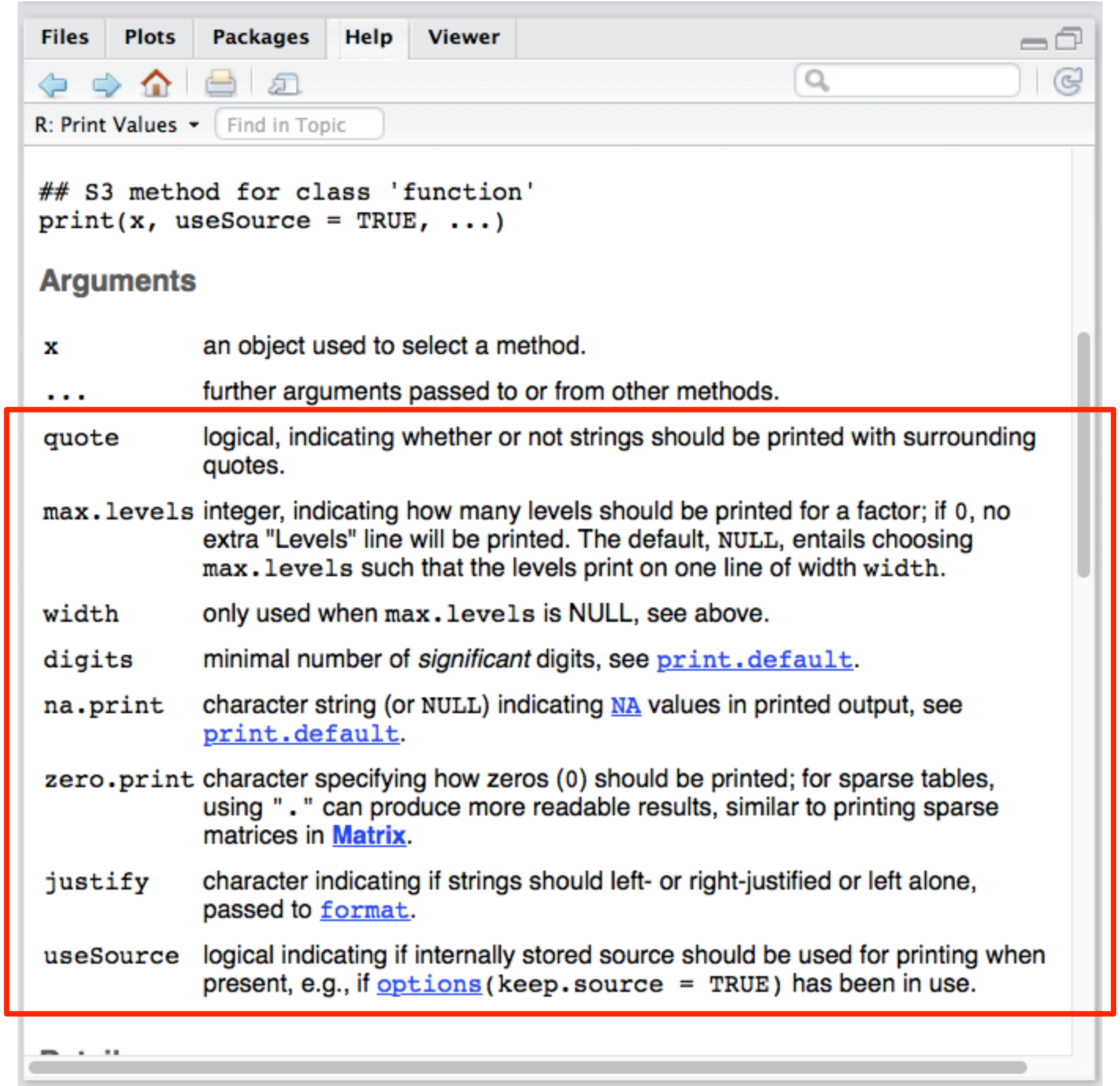
Arguments

`x`	an object used to select a method.
`...`	further arguments passed to or from other methods.
`quote`	logical, indicating whether or not strings should be printed with surrounding quotes.
`max.levels`	integer, indicating how many levels should be printed for a factor; if `0`, no extra "Levels" line will be printed. The default, `NULL`, entails choosing `max.levels` such that the levels print on one line of width `width`.
`width`	only used when `max.levels` is NULL, see above.
`digits`	minimal number of *significant* digits, see `print.default`.
`na.print`	character string (or `NULL`) indicating `NA` values in printed output, see `print.default`.
`zero.print`	character specifying how zeros (`0`) should be printed; for sparse tables, using `"."` can produce more readable results, similar to printing sparse matrices in **Matrix**.
`justify`	character indicating if strings should left- or right-justified or left alone, passed to `format`.
`useSource`	logical indicating if internally stored source should be used for printing when present, e.g., if `options(keep.source = TRUE)` has been in use.

remember this was something you had to include

(in this case, it is the object that is printed)

…scrolling down…



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

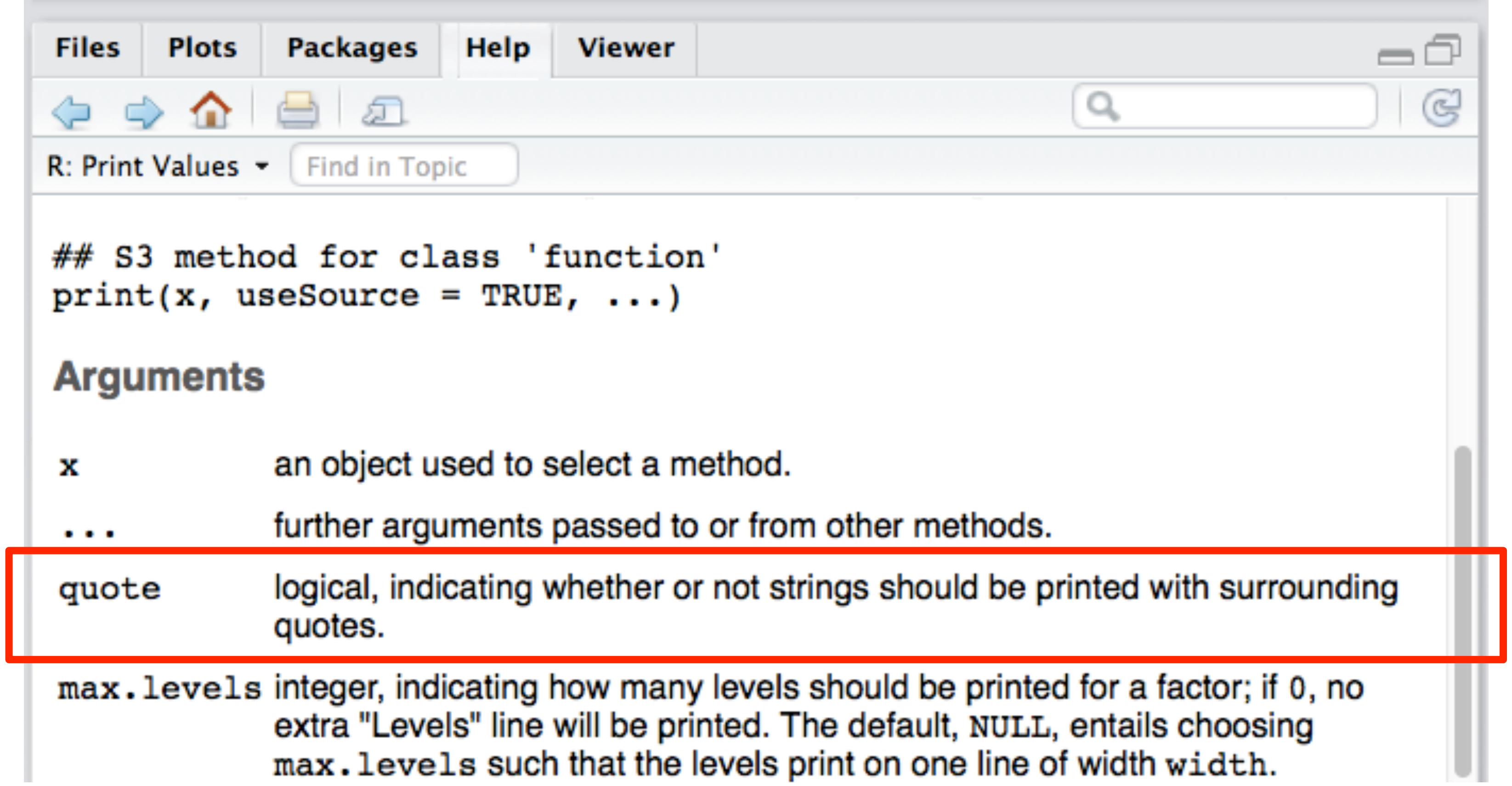
Arguments

`x`	an object used to select a method.
`...`	further arguments passed to or from other methods.
`quote`	logical, indicating whether or not strings should be printed with surrounding quotes.
`max.levels`	integer, indicating how many levels should be printed for a factor; if `0`, no extra "Levels" line will be printed. The default, `NULL`, entails choosing `max.levels` such that the levels print on one line of width `width`.
`width`	only used when `max.levels` is NULL, see above.
`digits`	minimal number of *significant* digits, see `print.default`.
`na.print`	character string (or `NULL`) indicating NA values in printed output, see `print.default`.
`zero.print`	character specifying how zeros (`0`) should be printed; for sparse tables, using `"."` can produce more readable results, similar to printing sparse matrices in **Matrix**.
`justify`	character indicating if strings should left- or right-justified or left alone, passed to `format`.
`useSource`	logical indicating if internally stored source should be used for printing when present, e.g., if `options(keep.source = TRUE)` has been in use.

these are other things you *might* want to specify but don't need to

unless told otherwise you can probably ignore most of them

…scrolling down…

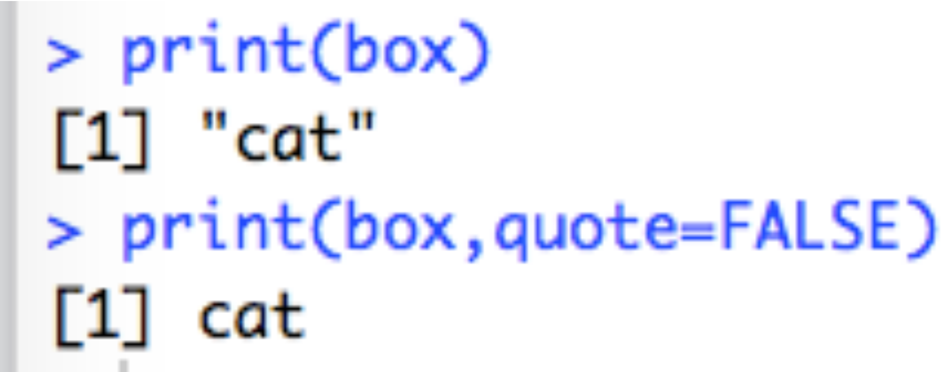


```
## S3 method for class 'function'
print(x, useSource = TRUE, ...)
```

Arguments

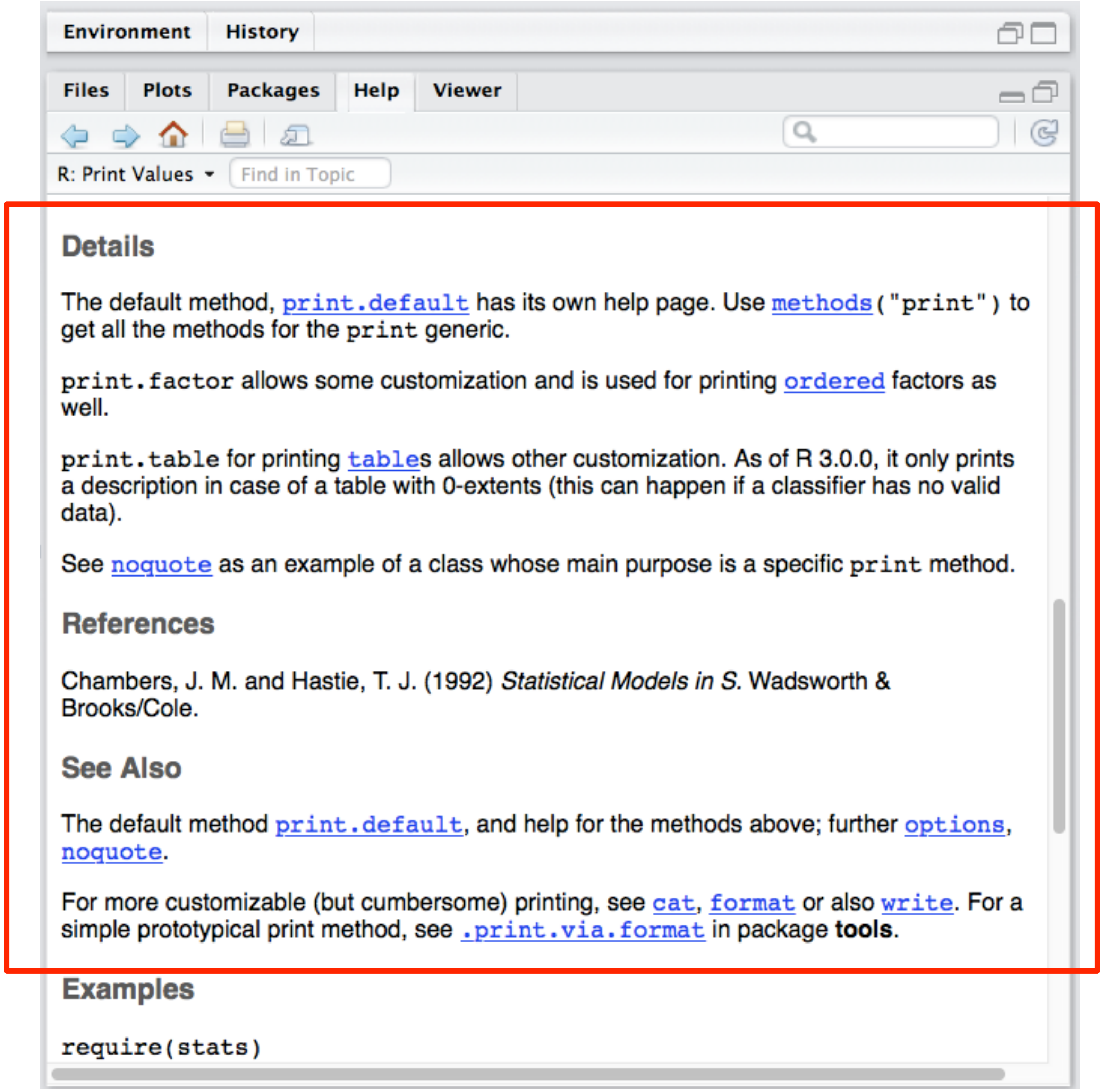
`x`	an object used to select a method.
`...`	further arguments passed to or from other methods.
`quote`	logical, indicating whether or not strings should be printed with surrounding quotes.
`max.levels`	integer, indicating how many levels should be printed for a factor; if `0`, no extra "Levels" line will be printed. The default, `NULL`, entails choosing `max.levels` such that the levels print on one line of width `width`.

but it also never hurts to play around!



```
> print(box)
[1] "cat"
> print(box,quote=FALSE)
[1] cat
```

… scrolling even more…



Details

The default method, `print.default` has its own help page. Use `methods("print")` to get all the methods for the `print` generic.

`print.factor` allows some customization and is used for printing `ordered` factors as well.

`print.table` for printing `table`s allows other customization. As of R 3.0.0, it only prints a description in case of a table with 0-extents (this can happen if a classifier has no valid data).

See `noquote` as an example of a class whose main purpose is a specific `print` method.

References

Chambers, J. M. and Hastie, T. J. (1992) *Statistical Models in S.* Wadsworth & Brooks/Cole.

See Also

The default method `print.default`, and help for the methods above; further `options`, `noquote`.

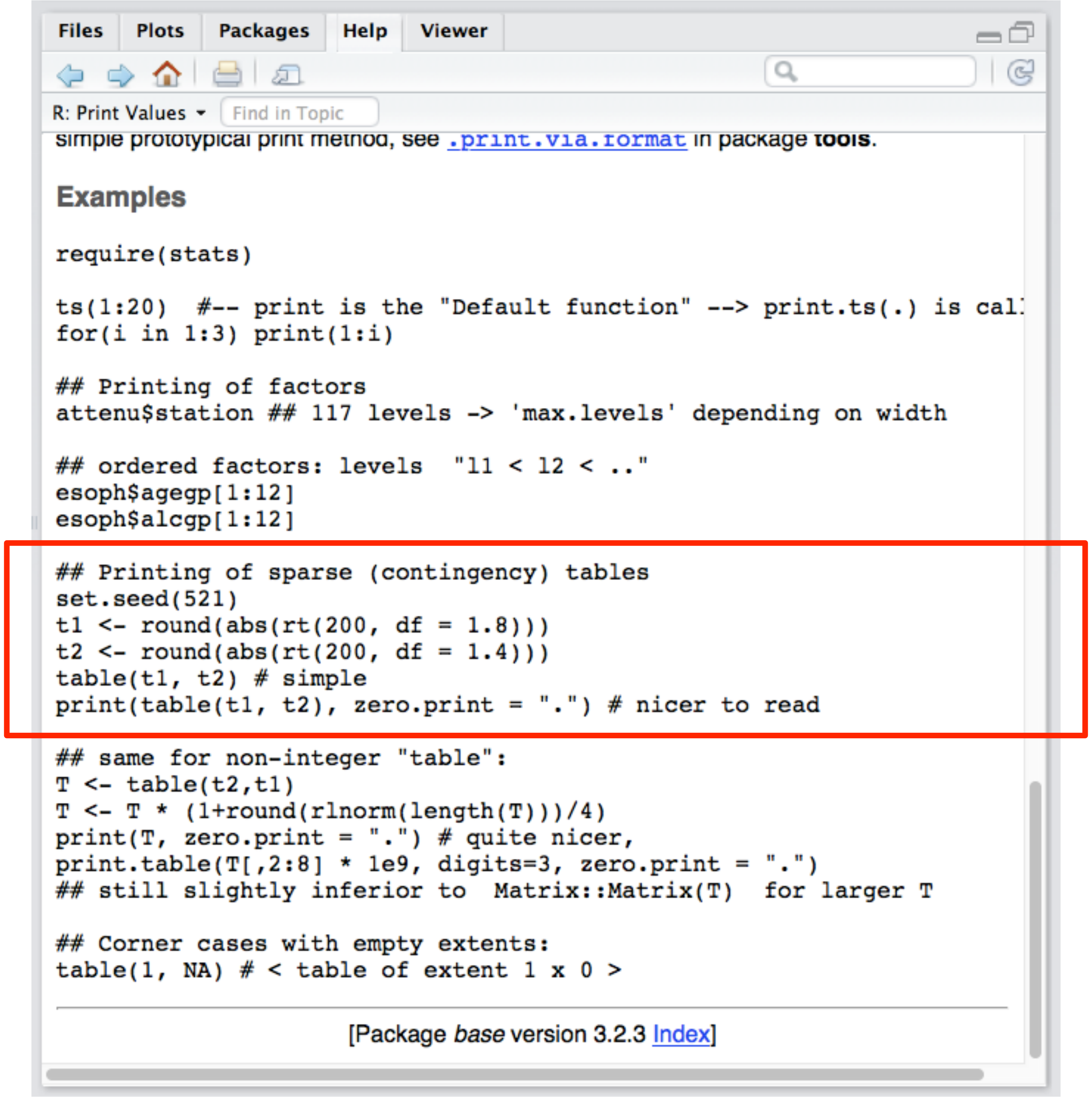
For more customizable (but cumbersome) printing, see `cat`, `format` or also `write`. For a simple prototypical print method, see `.print.via.format` in package **tools**.

Examples

```
require(stats)
```

you can pretty much ignore all of this (it's far advanced of what you'll need in DRIP)

the end of the scrolling…



simple prototypical print method, see `.print.via.format` in package **tools**.

Examples

```
require(stats)

ts(1:20)  #-- print is the "Default function" --> print.ts(.) is cal
for(i in 1:3) print(1:i)

## Printing of factors
attenu$station ## 117 levels -> 'max.levels' depending on width

## ordered factors: levels  "l1 < l2 < .."
esoph$agegp[1:12]
esoph$alcgp[1:12]

## Printing of sparse (contingency) tables
set.seed(521)
t1 <- round(abs(rt(200, df = 1.8)))
t2 <- round(abs(rt(200, df = 1.4)))
table(t1, t2) # simple
print(table(t1, t2), zero.print = ".") # nicer to read

## same for non-integer "table":
T <- table(t2,t1)
T <- T * (1+round(rlnorm(length(T)))/4)
print(T, zero.print = ".") # quite nicer,
print.table(T[,2:8] * 1e9, digits=3, zero.print = ".")
## still slightly inferior to  Matrix::Matrix(T)  for larger T

## Corner cases with empty extents:
table(1, NA) # < table of extent 1 x 0 >
```

[Package *base* version 3.2.3 Index]

These can be useful to make sense of how to use some of the optional arguments.

But if they are confusing it’s because it’s almost certainly not something you need to understand!

Exercises

1. Write a script which begins with two variables, `weightInKilos` and `sizeInCm`. Set those to a reasonable weight and size. Then convert the kilos to pounds (1 kilo = 2.2 pounds) and cm to inches (2.54 cm = 1 inch) and save those values in new variables. Print the new variables out. Save your script as `conversion.R` and run it.

2. Write a script which loads the `toyData` dataset, creates two subset datasets (one with males, one with females) and for each one prints out the people with happiness greater than 3.0. Save your script as `happyAnalysis.R` and run it. Clear your entire workspace and then run it again.

Intro to R cheat sheet

8 Packages: 5000+ available online

install	**load**
put on computer	make available to R
install.packages(“lsr”)	library(“lsr”)

9 data and data frames

`expt`

load data from menu or with `load()`)

```
   id age gender treatment hormone happy  sad
1   1  25   male   control     6.7  2.00 6.12
2   2  24   male     drug1    38.5  3.36 3.53
3   3  25   male     drug2    25.0  3.40 4.82
4   4  28   male   control    98.4  5.69 0.34
5   5  23   male     drug1    42.4  4.56 4.48
6   6  28   male     drug2    20.3  2.89 4.57
7   7  25 female   control    18.5  3.18 4.82
8   8  29 female     drug1    65.2  4.78 2.24
9   9  21 female     drug2    56.4  4.51 2.64
10 10  26 female   control    55.7  3.90 2.71
11 11  19 female     drug1    41.9  2.83 2.94
12 12  30 female     drug2    54.1  3.45 1.87
```

10 data manipulation

`expt$age` selects the variable `age`

`expt$age[1]` or `expt[1,”age”]`

selects the first case of `age`

`expt$over25 <- expt$age > 25`

creates a new variable called `over25` which is true if `age` is over 25

`expt$over25 <- NULL`

removes the variable `over25`

`expt[ c(1,4,7), c(“age”,“gender”)`

selects rows 1,4,7 and age/gender columns

`subset (expt, gender==“male”)`

select all males out of dataset

`class(expt$gender)`

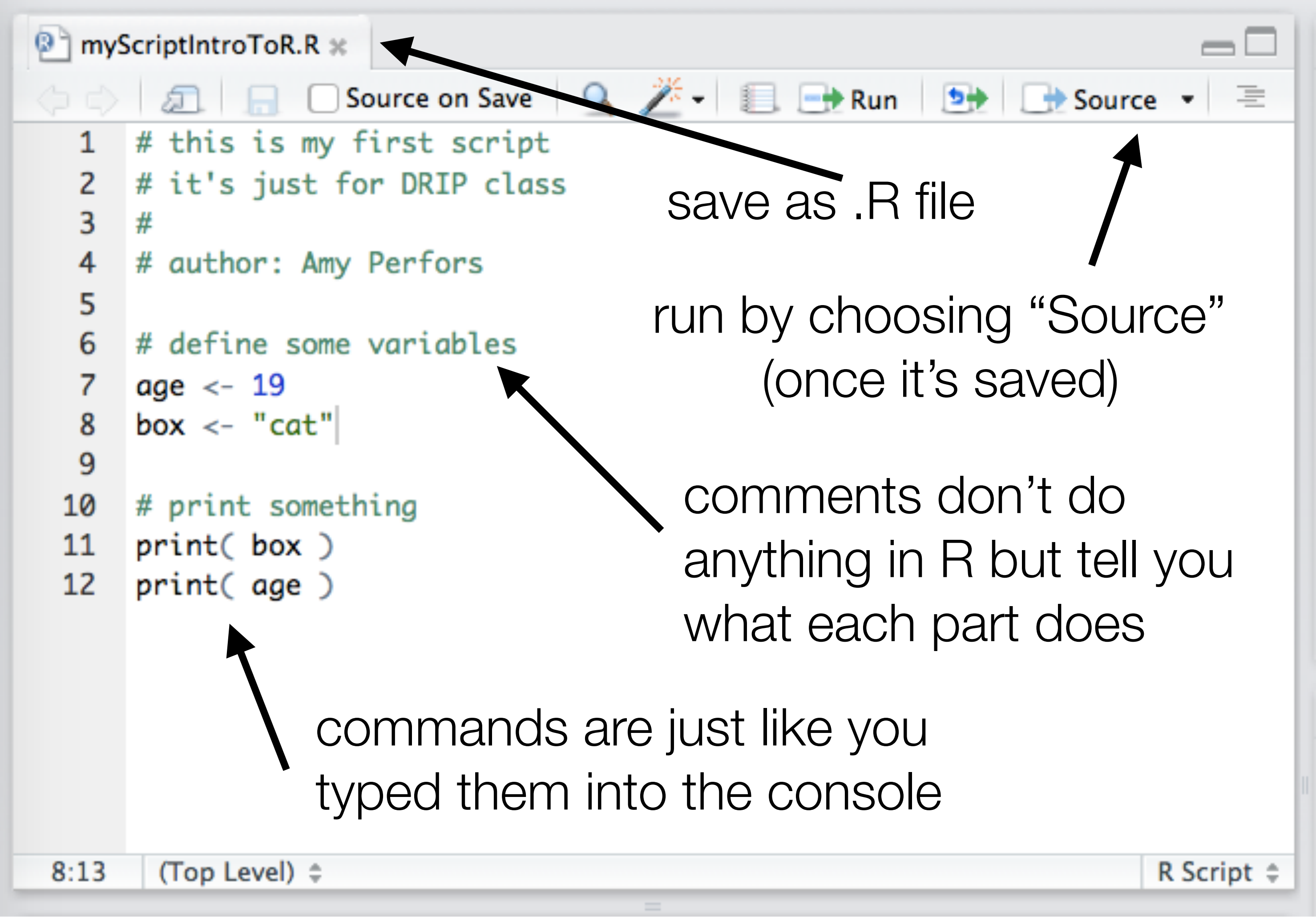
tells you gender is a nominal scale variable

Intro to R cheat sheet

11 Saving and importing

- Save as .RData, using menu or `save.image()`
- Can load .csv, using menu or `read.csv()`

12 Scripts let you run and save series of commands



```
# this is my first script
# it's just for DRIP class
#
# author: Amy Perfors

# define some variables
age <- 19
box <- "cat"

# print something
print( box )
print( age )
```

save as .R file

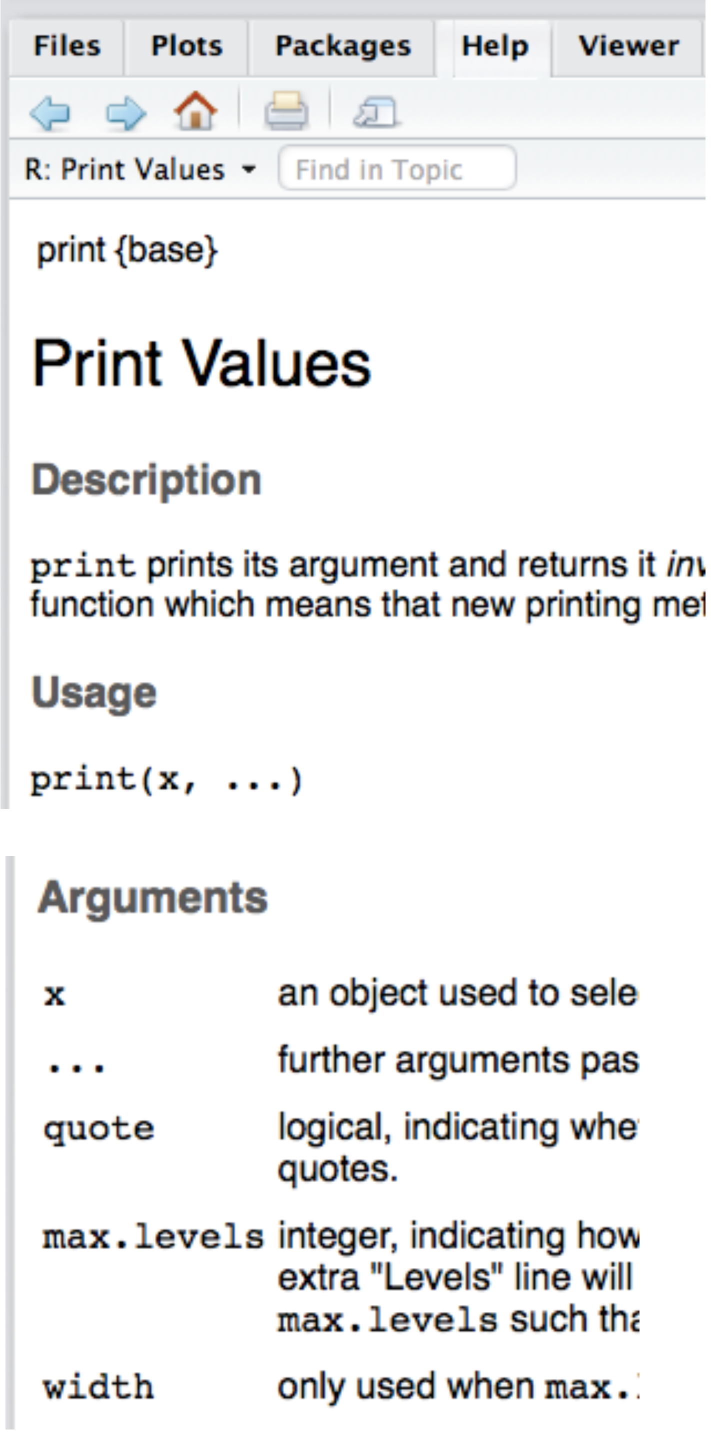
run by choosing “Source” (once it’s saved)

comments don’t do anything in R but tell you what each part does

commands are just like you typed them into the console

13 `help(functionName)`

e.g. `help(print)`



print {base}

Print Values

Description

`print` prints its argument and returns it *inv*
function which means that new printing me

Usage

`print(x, ...)`

Arguments

`x`	an object used to sele
`...`	further arguments pas
`quote`	logical, indicating whe quotes.
`max.levels`	integer, indicating how extra "Levels" line will `max.levels` such tha
`width`	only used when `max.`