

R Bootcamp (continued)

Amy Perfors

This time

More useful things: data, data, data

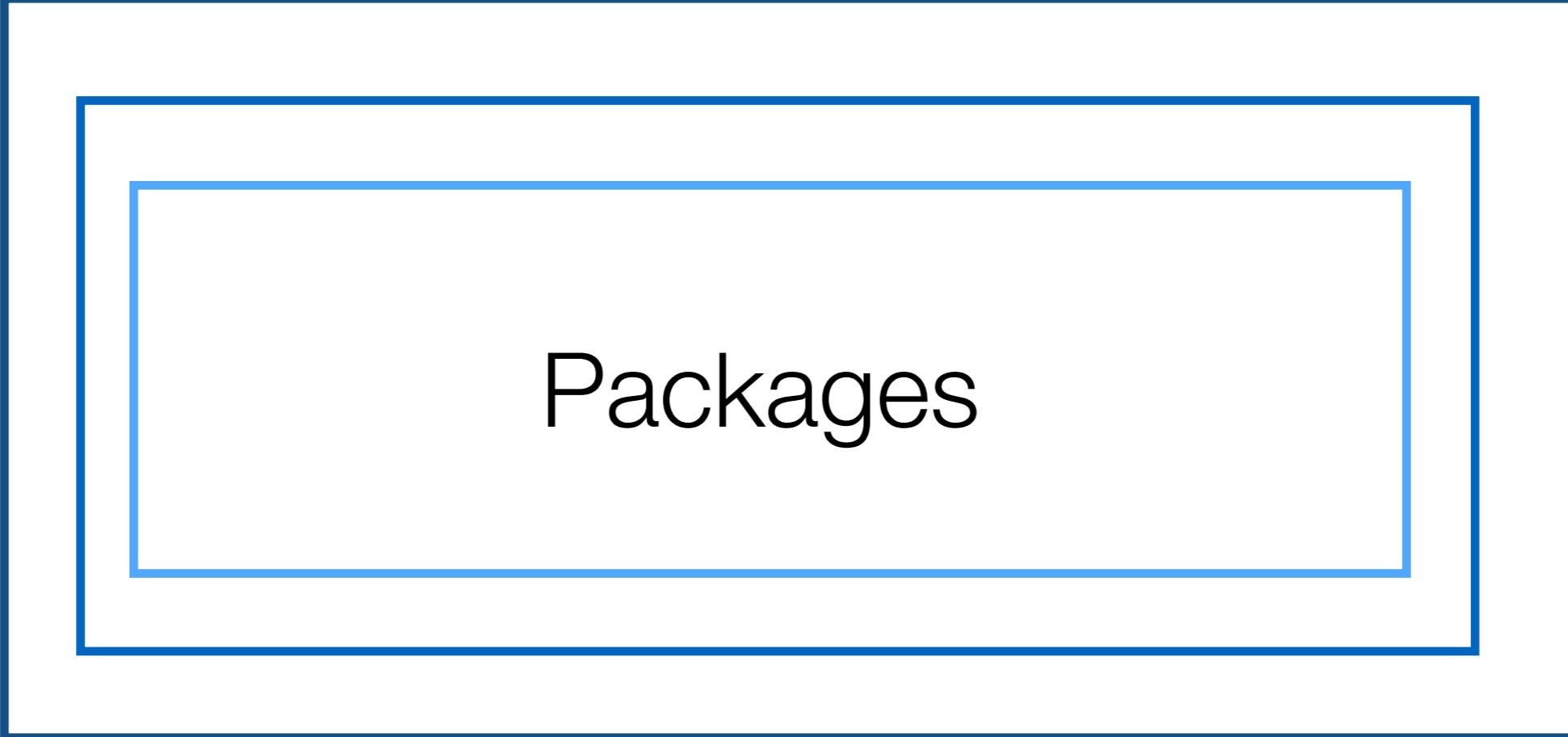
Before we begin:

You should have downloaded the following datasets:

[toydata.RData](#) and [toydata.csv](#)

They are 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

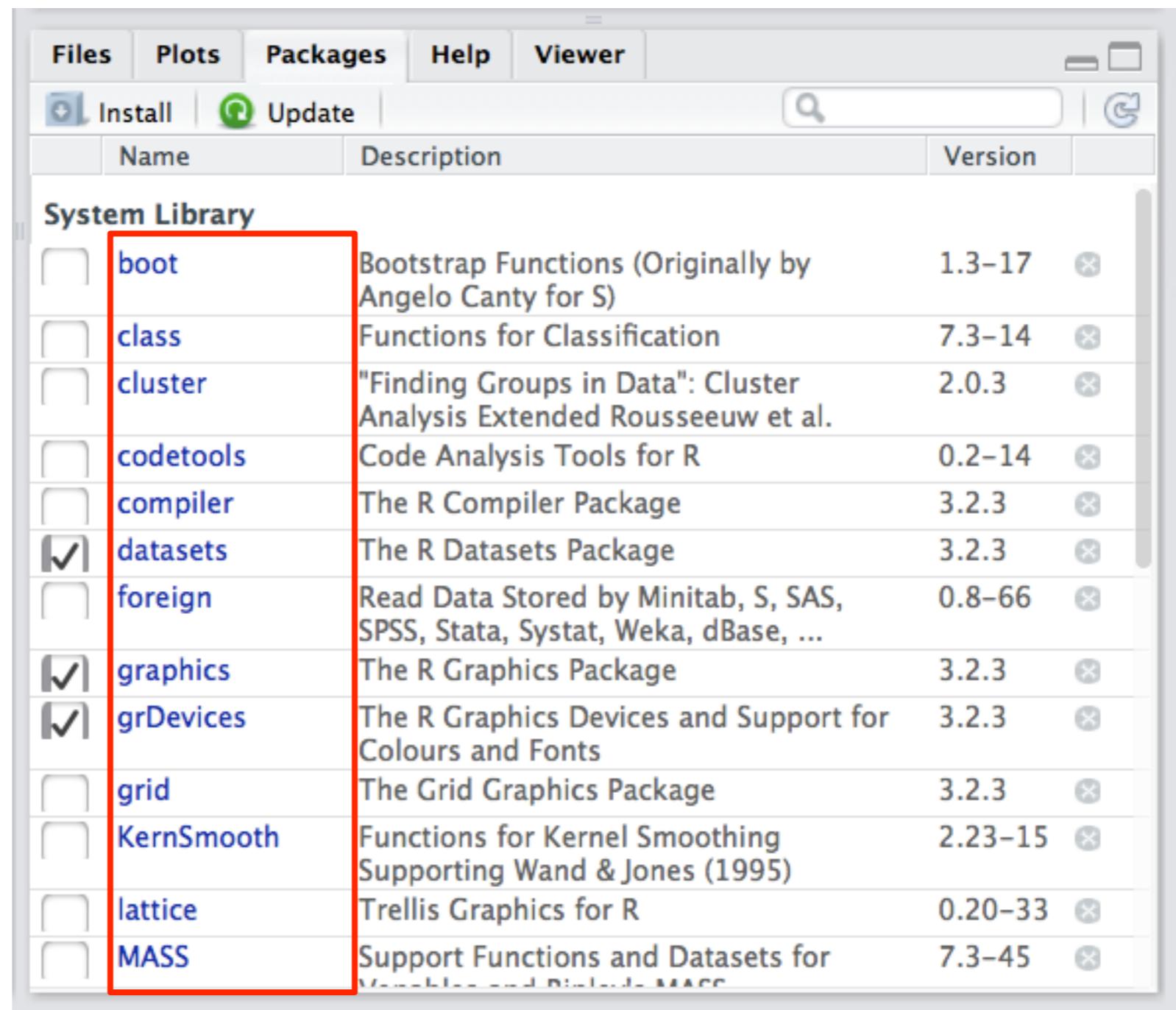
(lower right part
of RStudio)

The screenshot shows the RStudio interface with the 'Packages' tab selected, highlighted by a red box. Below the tabs, there are two buttons: 'Install' and 'Update'. A search bar and a refresh icon are also present. The main area is titled 'System Library' and displays a list of R packages. Each package entry includes the name, description, version, and a remove icon. Several packages have checkmarks next to their names, indicating they are selected or installed.

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
<input checked="" type="checkbox"/> datasets	The R Datasets Package	3.2.3
foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
<input checked="" type="checkbox"/> graphics	The R Graphics Package	3.2.3
<input checked="" type="checkbox"/> 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 Venables and Ripley's MASS	7.3-45

The Rstudio “packages” panel

These are the names of the packages that are installed



The screenshot shows the RStudio interface with the "Packages" tab selected in the top navigation bar. Below the navigation bar, there are two buttons: "Install" and "Update". A search bar and a refresh icon are also present. The main area is titled "System Library" and displays a list of packages. Each package entry includes the name, description, version, and a remove button. The "datasets" package is checked, while others like "graphics" and "grDevices" are checked and highlighted with a red border.

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

The Rstudio “packages” panel

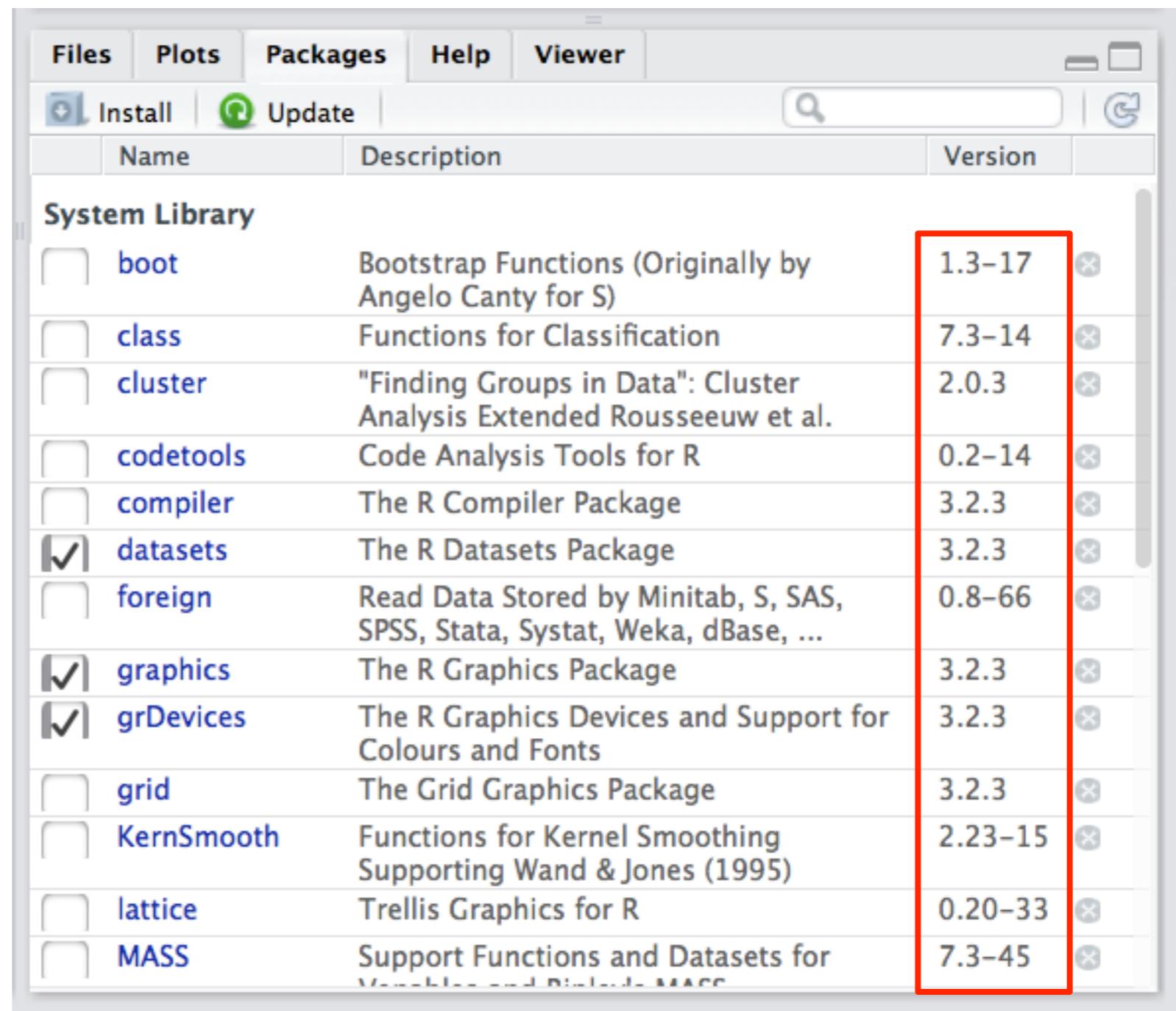
This describes what the package does

The screenshot shows the RStudio interface with the "Packages" tab selected in the top menu bar. Below the menu, there are two buttons: "Install" and "Update". A search bar and a refresh icon are also present. The main area is titled "System Library" and displays a list of packages. The "datasets" package is selected, indicated by a checked checkbox icon to its left. Its details are highlighted with a red box, showing the following information:

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
<input checked="" type="checkbox"/> datasets	The R Datasets Package	3.2.3
foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
<input checked="" type="checkbox"/> graphics	The R Graphics Package	3.2.3
<input checked="" type="checkbox"/> 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 MASS	7.3-45

The Rstudio “packages” panel

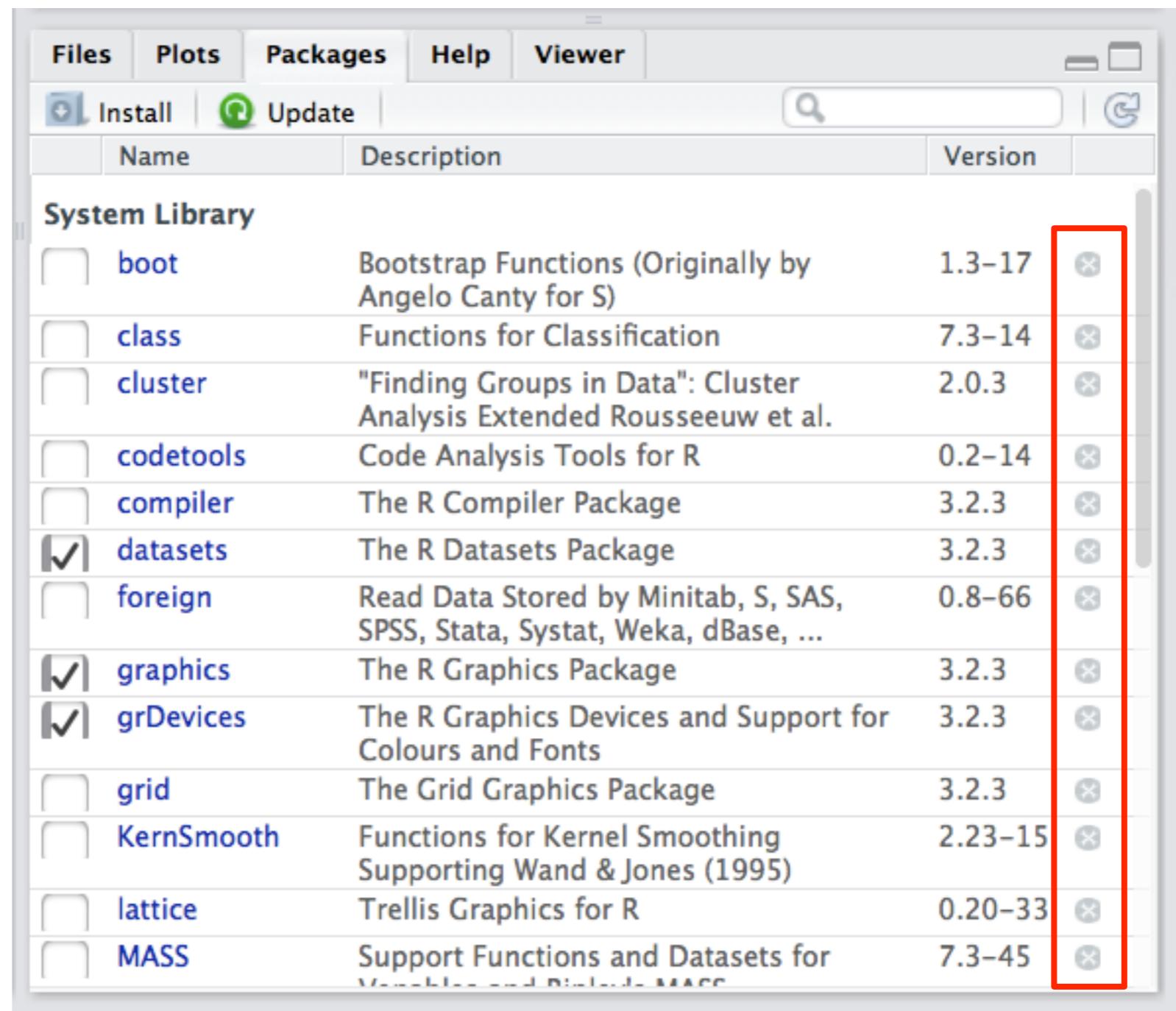
This tells you what version you have



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

The Rstudio “packages” panel

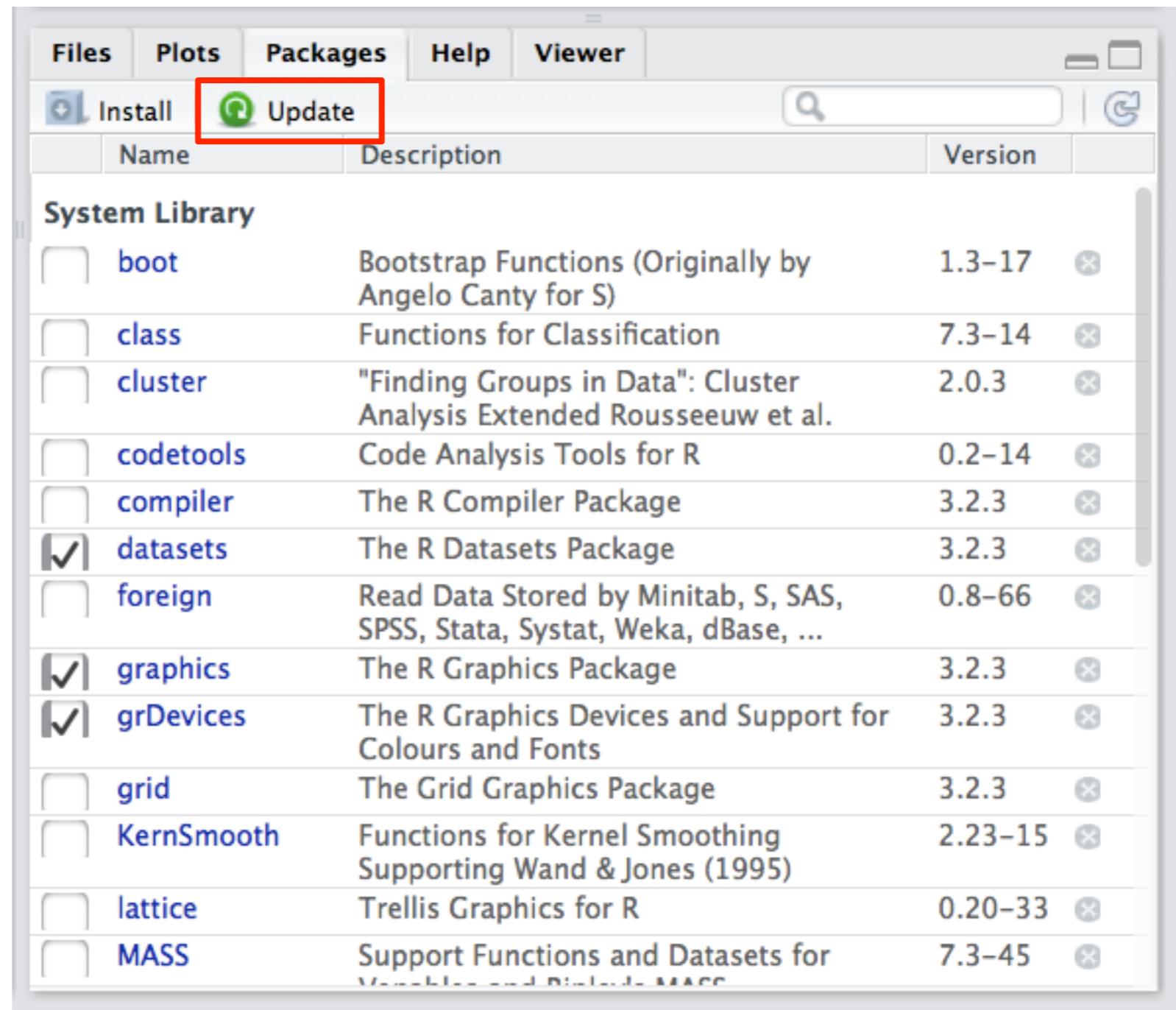
Clicking this will
uninstall the
package



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

The Rstudio “packages” panel

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

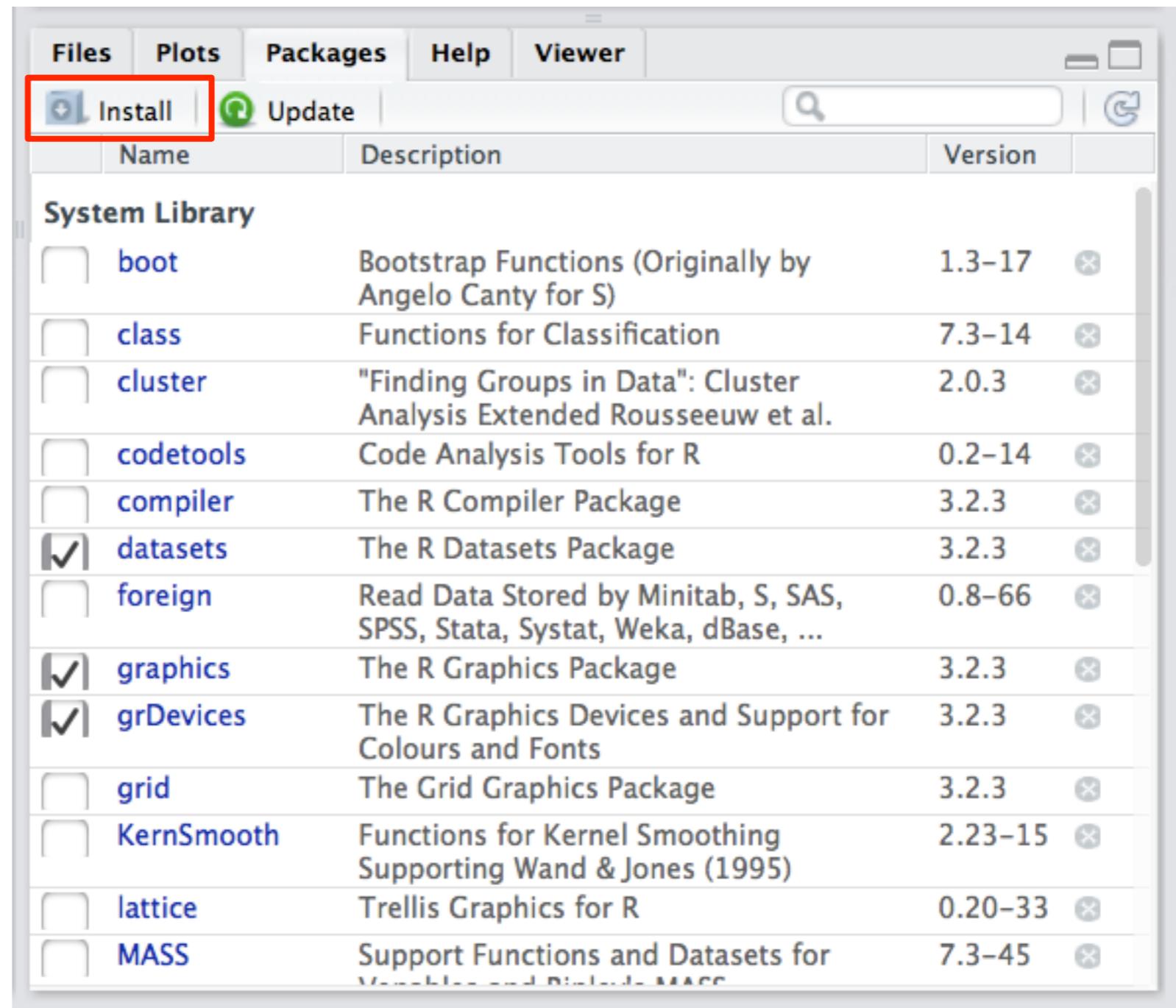


The screenshot shows the RStudio interface with the "Packages" tab selected in the top navigation bar. A red box highlights the "Update" button in the toolbar below the tabs. The main area displays a table titled "System Library" with columns for Name, Description, and Version. Several packages are listed, including "boot", "class", "cluster", "codetools", "compiler", "datasets", "foreign", "graphics", "grDevices", "grid", "KernSmooth", "lattice", and "MASS". The "graphics", "grDevices", and "grid" packages are checked, indicating they have updates available. The "datasets" package is also checked but appears to be up-to-date.

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
<input checked="" type="checkbox"/> datasets	The R Datasets Package	3.2.3
foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
<input checked="" type="checkbox"/> graphics	The R Graphics Package	3.2.3
<input checked="" type="checkbox"/> 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 Venables and Ripley's MASS	7.3-45

The Rstudio “packages” panel

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



The screenshot shows the RStudio interface with the "Packages" tab selected in the top navigation bar. Below the navigation bar, there are two buttons: "Install" (highlighted with a red box) and "Update". A search bar and a refresh icon are also present. The main area is titled "System Library" and displays a list of R packages. Each package entry includes a checkbox, the package name, its description, its version, and a remove icon. Several packages have checkboxes checked, including "datasets", "foreign", "graphics", "grDevices", "grid", and "MASS".

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
<input checked="" type="checkbox"/> datasets	The R Datasets Package	3.2.3
<input checked="" type="checkbox"/> foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
<input checked="" type="checkbox"/> graphics	The R Graphics Package	3.2.3
<input checked="" type="checkbox"/> grDevices	The R Graphics Devices and Support for Colours and Fonts	3.2.3
<input checked="" type="checkbox"/> grid	The Grid Graphics Package	3.2.3
KernSmooth	Functions for Kernel Smoothing Supporting Wand & Jones (1995)	2.23-15
<input checked="" type="checkbox"/> lattice	Trellis Graphics for R	0.20-33
<input checked="" type="checkbox"/> MASS	Support Functions and Datasets for Venables and Ripley's MASS	7.3-45

The Rstudio “packages” panel

Click here to load
or unload a
package

loaded

unloaded

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

Let's load the MASS package

(just click on it)

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	⊗
<input checked="" type="checkbox"/> 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



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

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

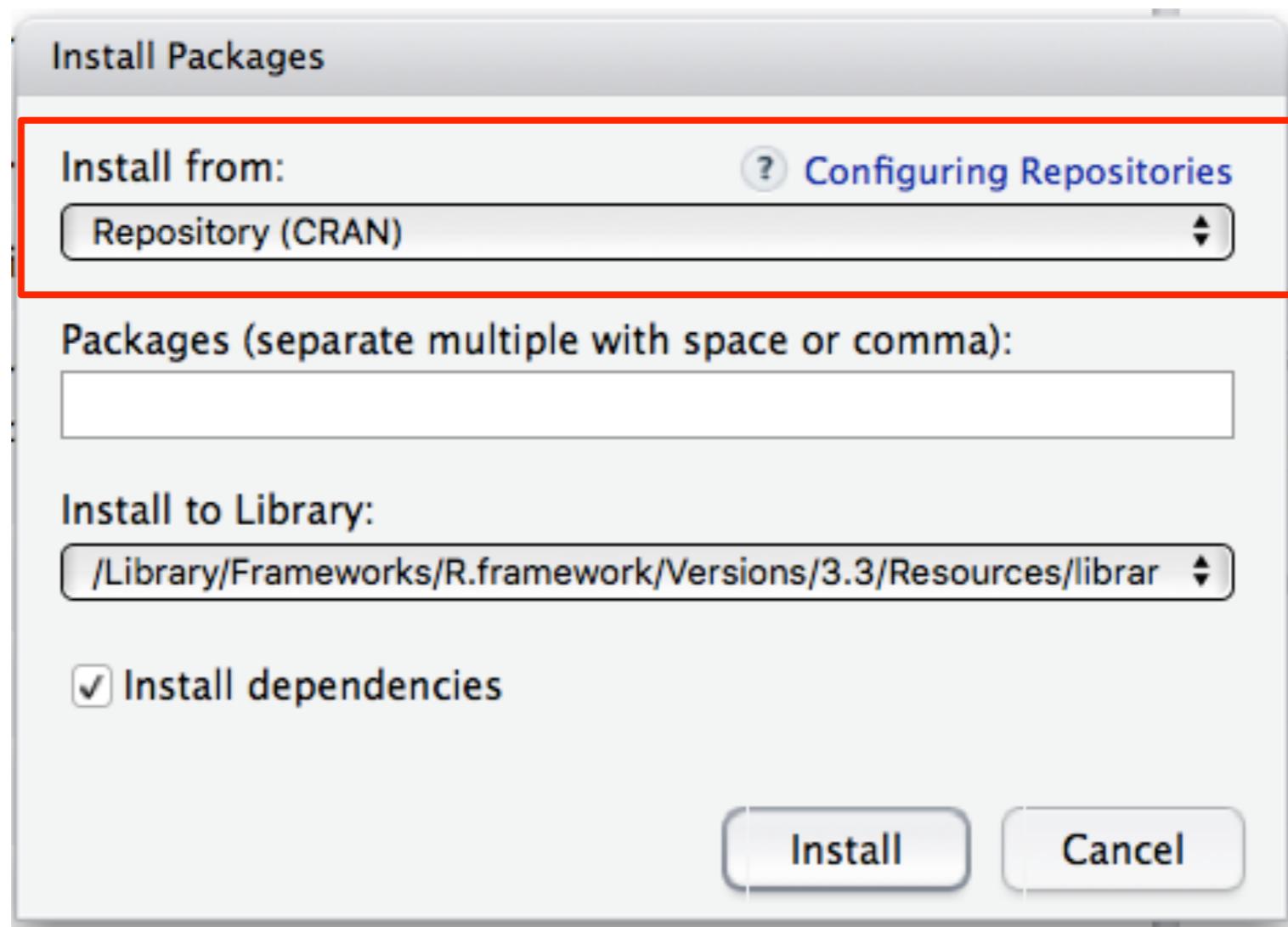
The screenshot shows the RStudio interface with the 'Packages' tab selected. The 'Install' button in the top navigation bar is highlighted with a red box and a red arrow. Below the navigation bar is a search bar and a refresh icon. The main area is titled 'System Library' and lists various R packages with their descriptions and versions. Some packages have checkmarks next to them, indicating they are selected or installed. The packages listed include:

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
<input checked="" type="checkbox"/> datasets	The R Datasets Package	3.2.3
foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
<input checked="" type="checkbox"/> graphics	The R Graphics Package	3.2.3
<input checked="" type="checkbox"/> 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 Venables and Ripley's MASS	7.3-45

Installing packages

Where to install it
from?

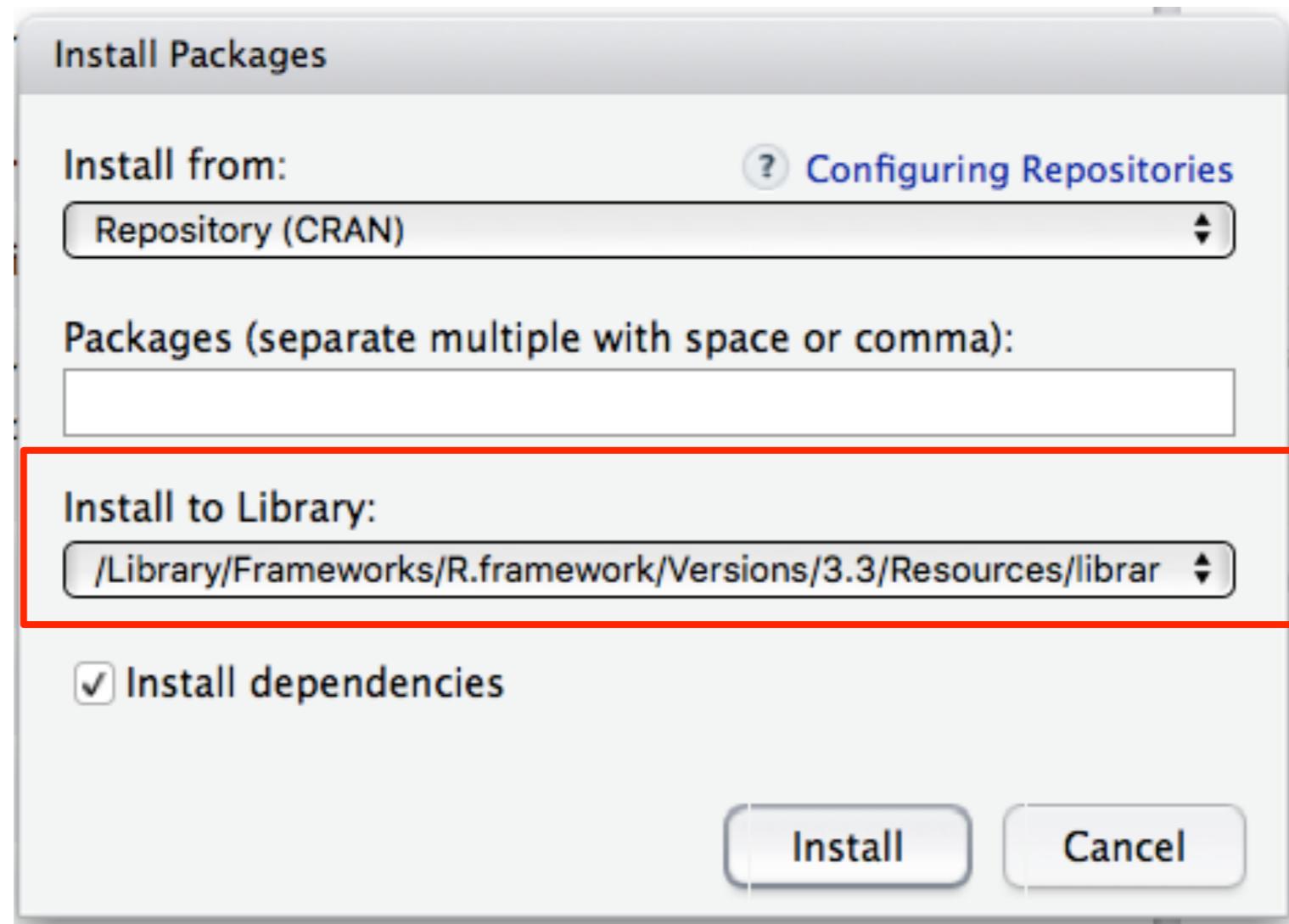
(ignore this:
default is fine)



Installing packages

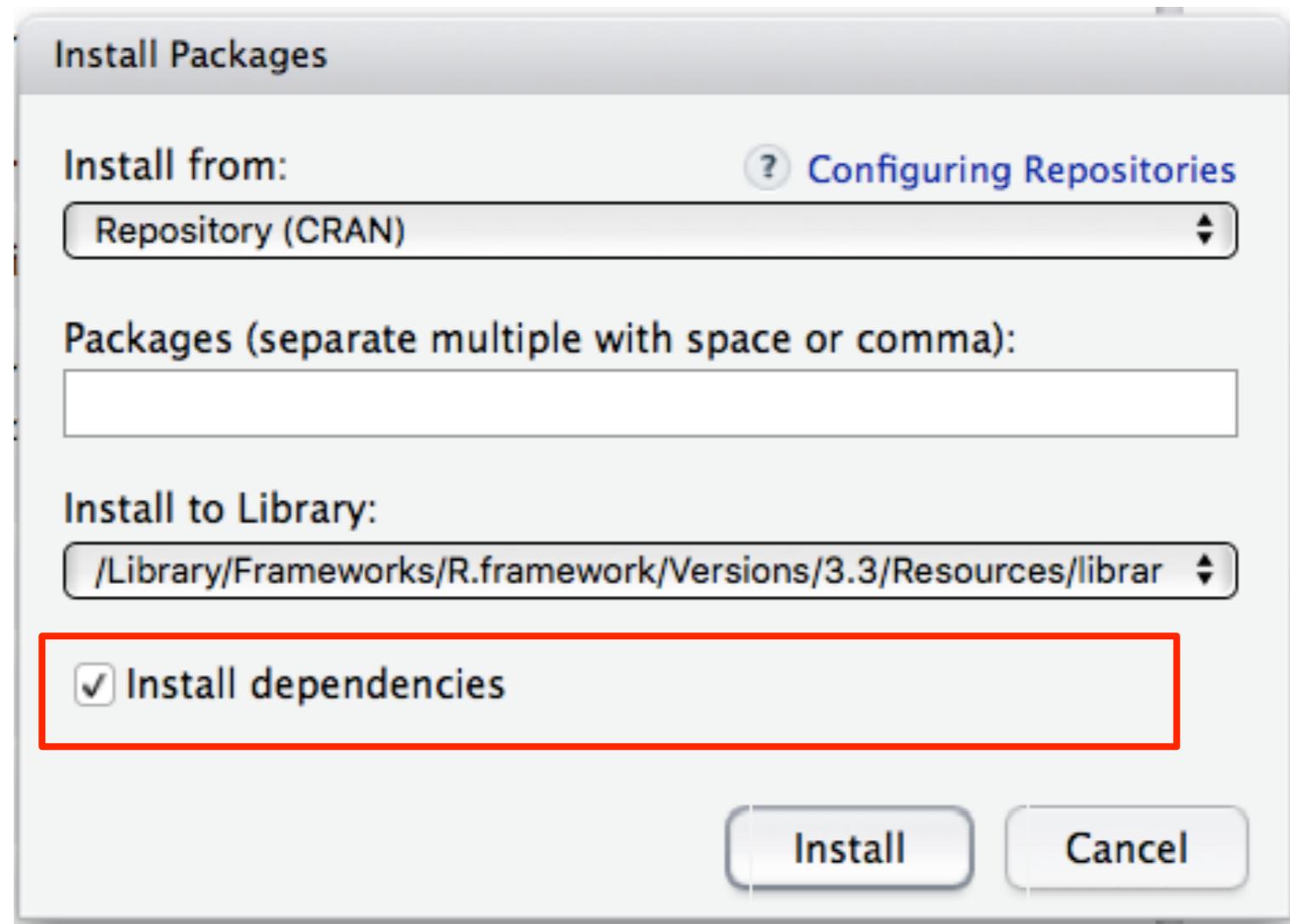
Where should
packages be
stored?

(default is also
fine)



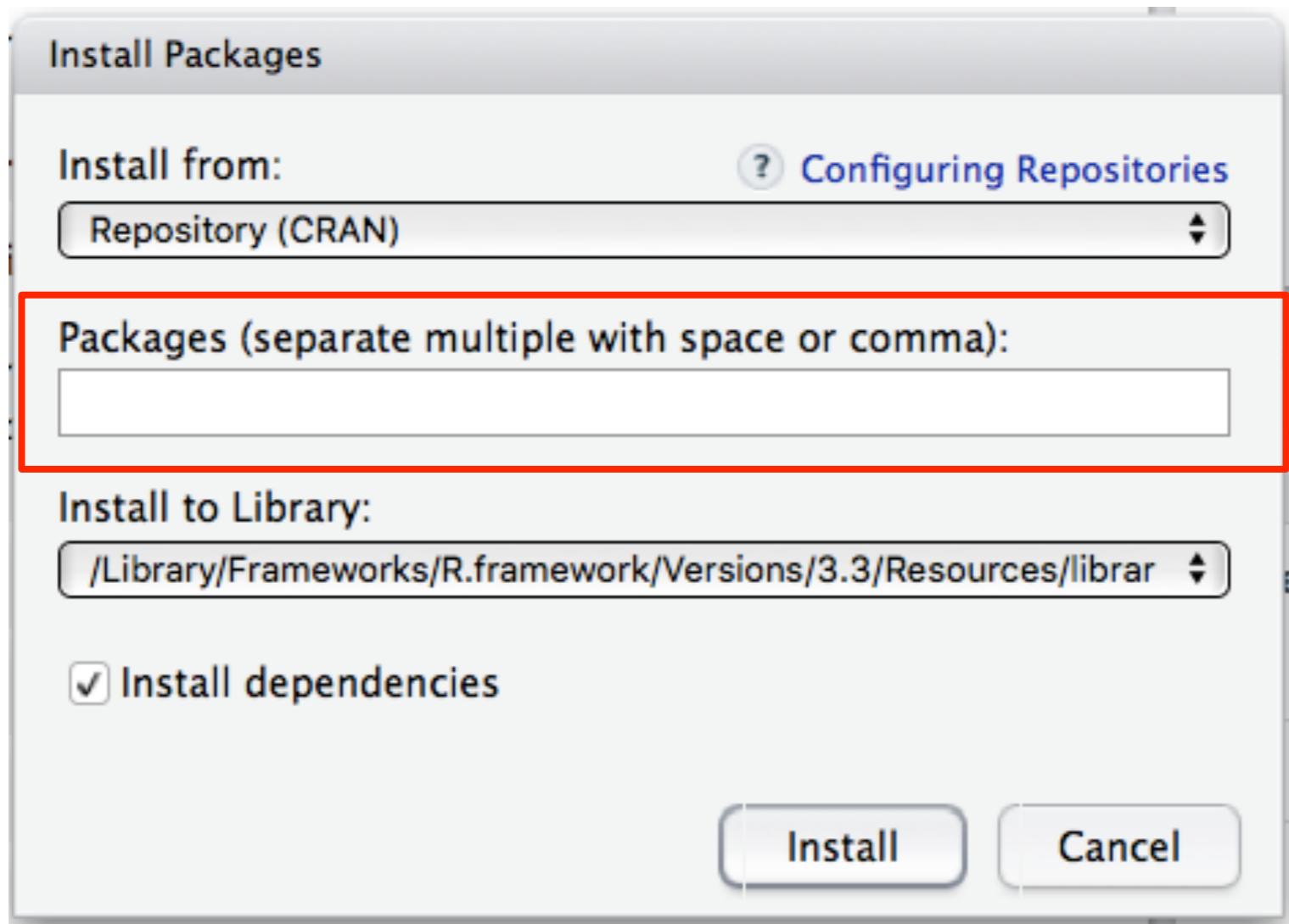
Installing packages

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



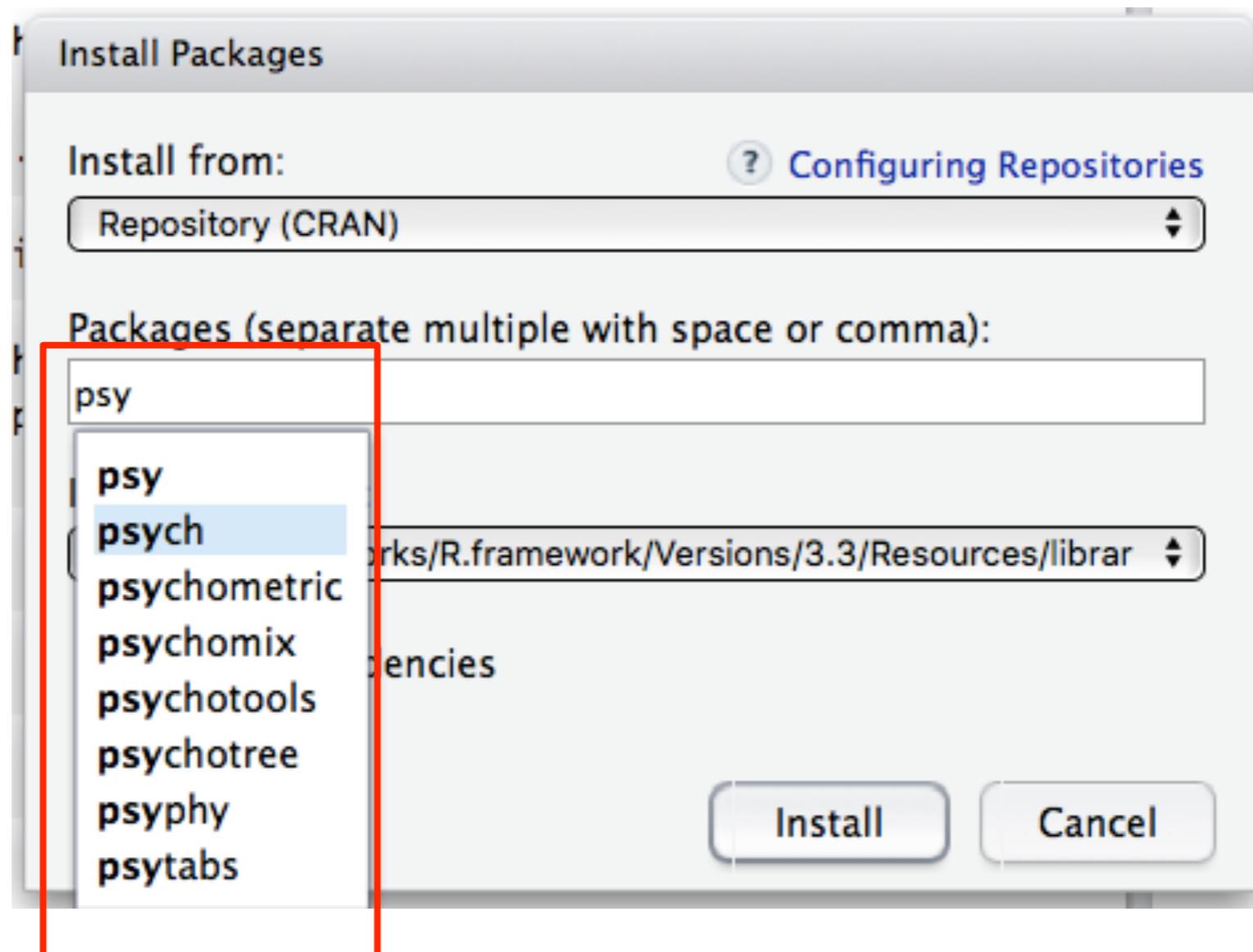
Installing packages

Which packages to install? This is the important bit!

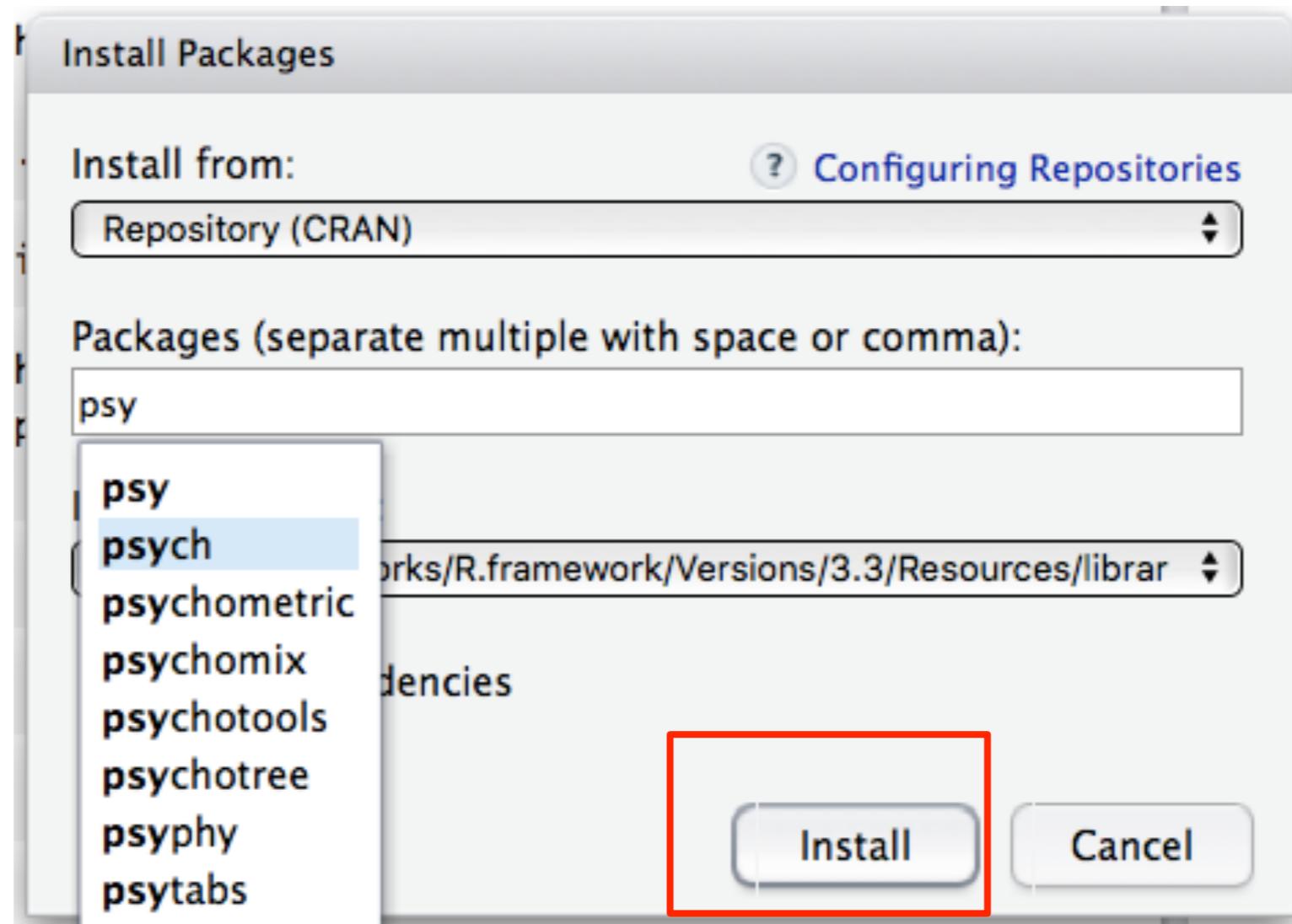


Installing packages

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



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 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      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//RtmppeTAmh8 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 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 --::--- --::--- --::--- 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.

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 )
```

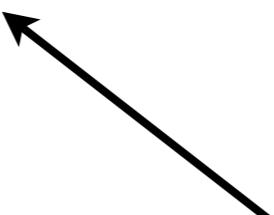
Attaching package: ‘car’

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

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

logit

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)

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

The screenshot shows the RStudio interface with the 'Environment' tab selected. Below it, the 'Values' section displays two variables: 'ages' (a numeric vector [1:3] with values 12, 67, 32) and 'box' (a character string "cat"). Both the 'Environment' tab and the 'Values' table are highlighted with red boxes. An arrow points from the code block above to the 'values' table in the RStudio interface.

Value	Type	Content
ages	num [1:3]	12 67 32
box	"cat"	

When I create variables,
they appear in the
environment panel

The Rstudio “environment” panel

The screenshot shows the RStudio Environment panel. At the top, there are tabs for "Environment" (selected) and "History". Below the tabs are several icons: a folder, a file, an import dataset icon, and a paintbrush. To the right of these icons is a dropdown menu labeled "Grid" with a red box around it. Underneath the icons is a section titled "Global Environment". A search bar is also present. The main area displays a table with the following data:

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

A large black arrow points upwards from the bottom of the "Table" view towards the "Grid" view, indicating a transition between the two.

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 →

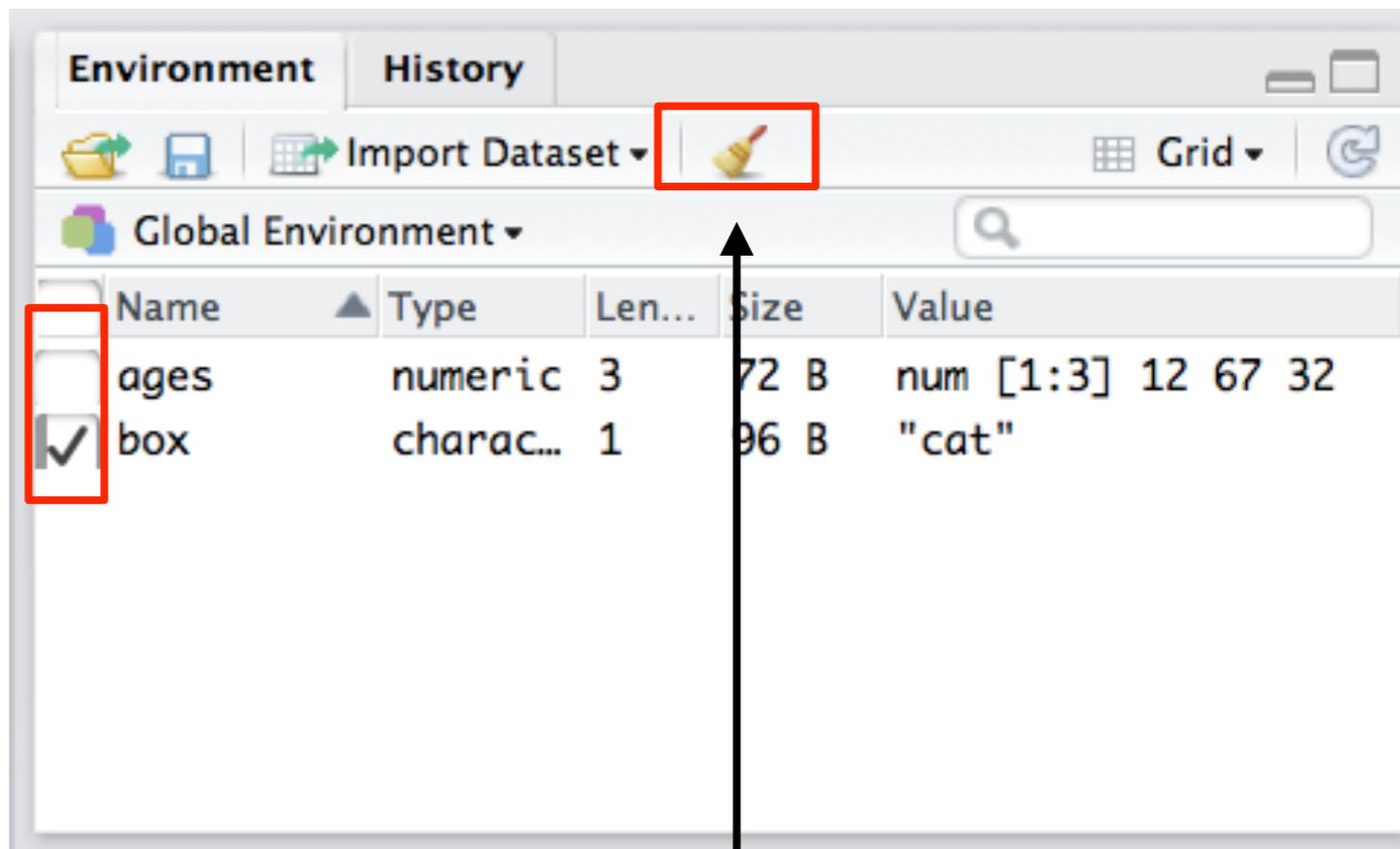
What kind of information is stored in this variable? (e.g., numeric)

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?



The screenshot shows the RStudio Environment pane. At the top, there are tabs for 'Environment' (selected) and 'History'. Below the tabs are icons for 'New Project', 'Import Dataset', and a trash can icon (highlighted with a red box). The title bar says 'Global Environment'. A search bar is on the right. The main area displays a table of variables:

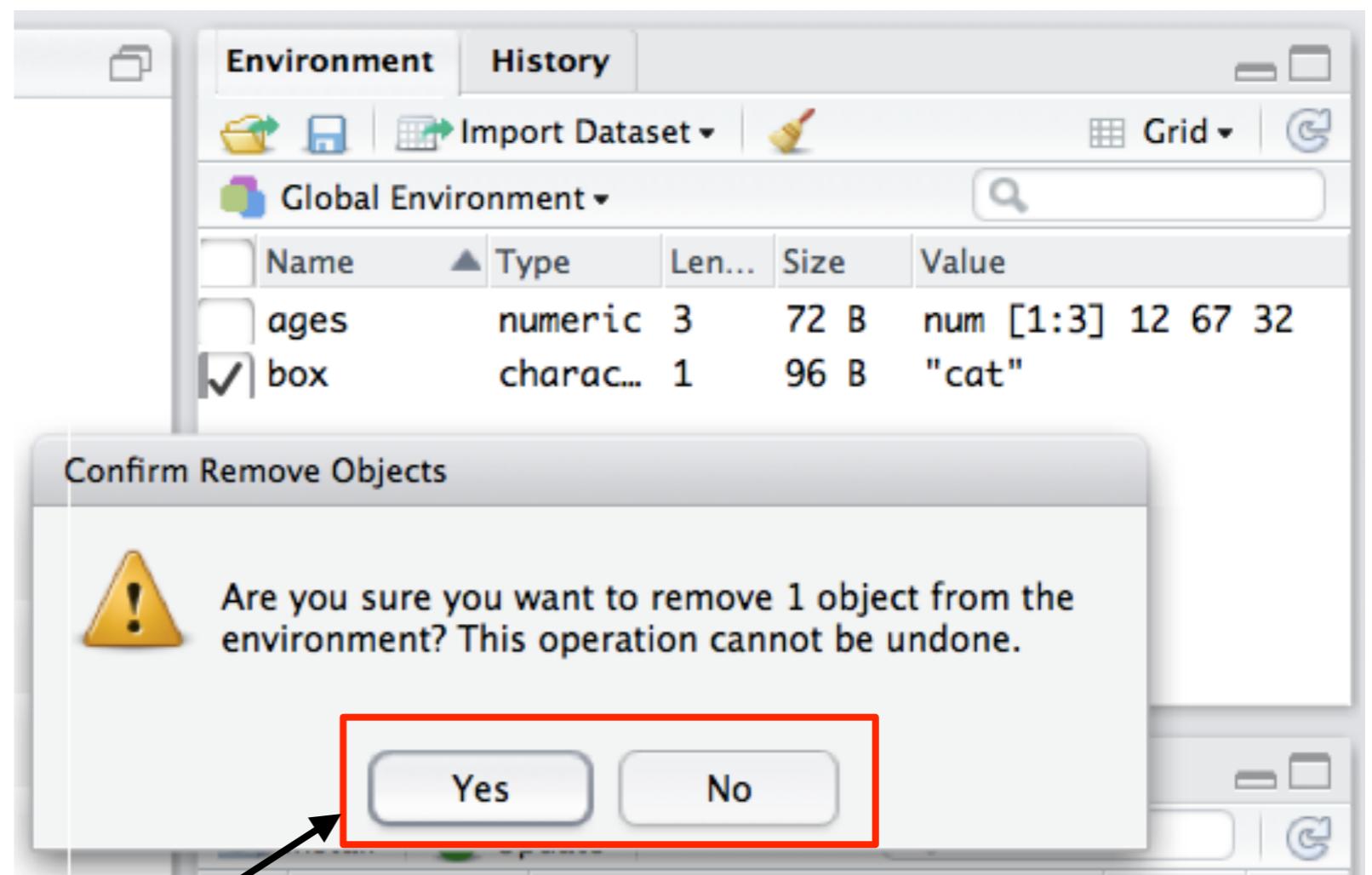
Name	Type	Len...	Size	Value
ages	numeric	3	72 B	num [1:3] 12 67 32
box	charac...	1	96 B	"cat"

A red box highlights the checkbox column for the 'box' variable. A black arrow points from the text 'Click here to select variables' to this checkbox. Another black arrow points from the text 'Now click here to delete the selected variable(s)' to the trash can icon at the top.

Click here to
select variables

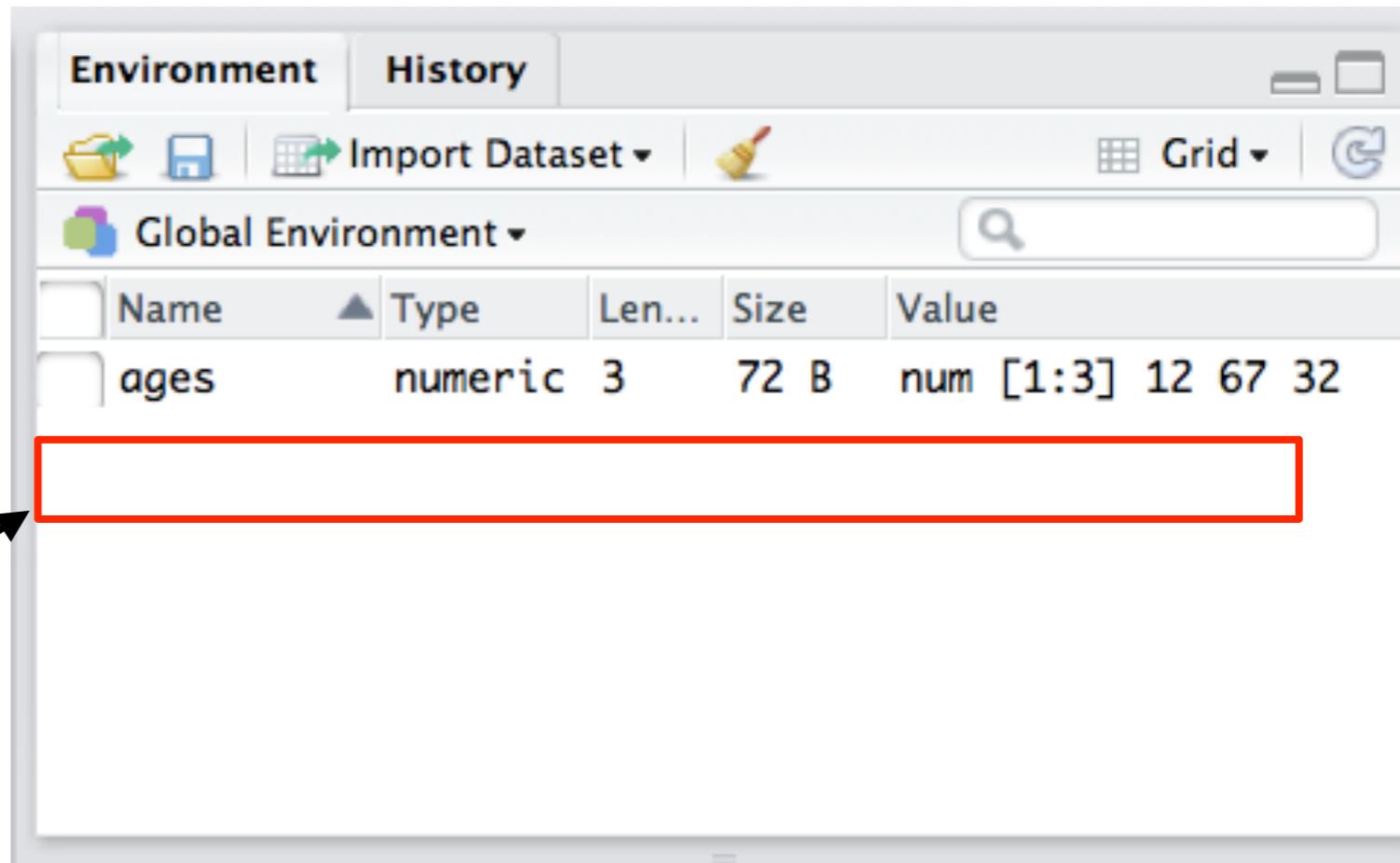
Now click here to delete
the selected variable(s)

Getting rid of variables?



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

Getting rid of variables?



The screenshot shows the RStudio interface with the 'Environment' tab selected. The 'Global Environment' dropdown is set to 'Global Environment'. A table lists variables: 'ages' is a numeric vector of length 3, size 72 B, with values [1:3] 12 67 32. A red rectangle highlights the row for 'ages', and a black arrow points from the text in the adjacent paragraph to this highlighted row.

	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 package called `ggplot2`.

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

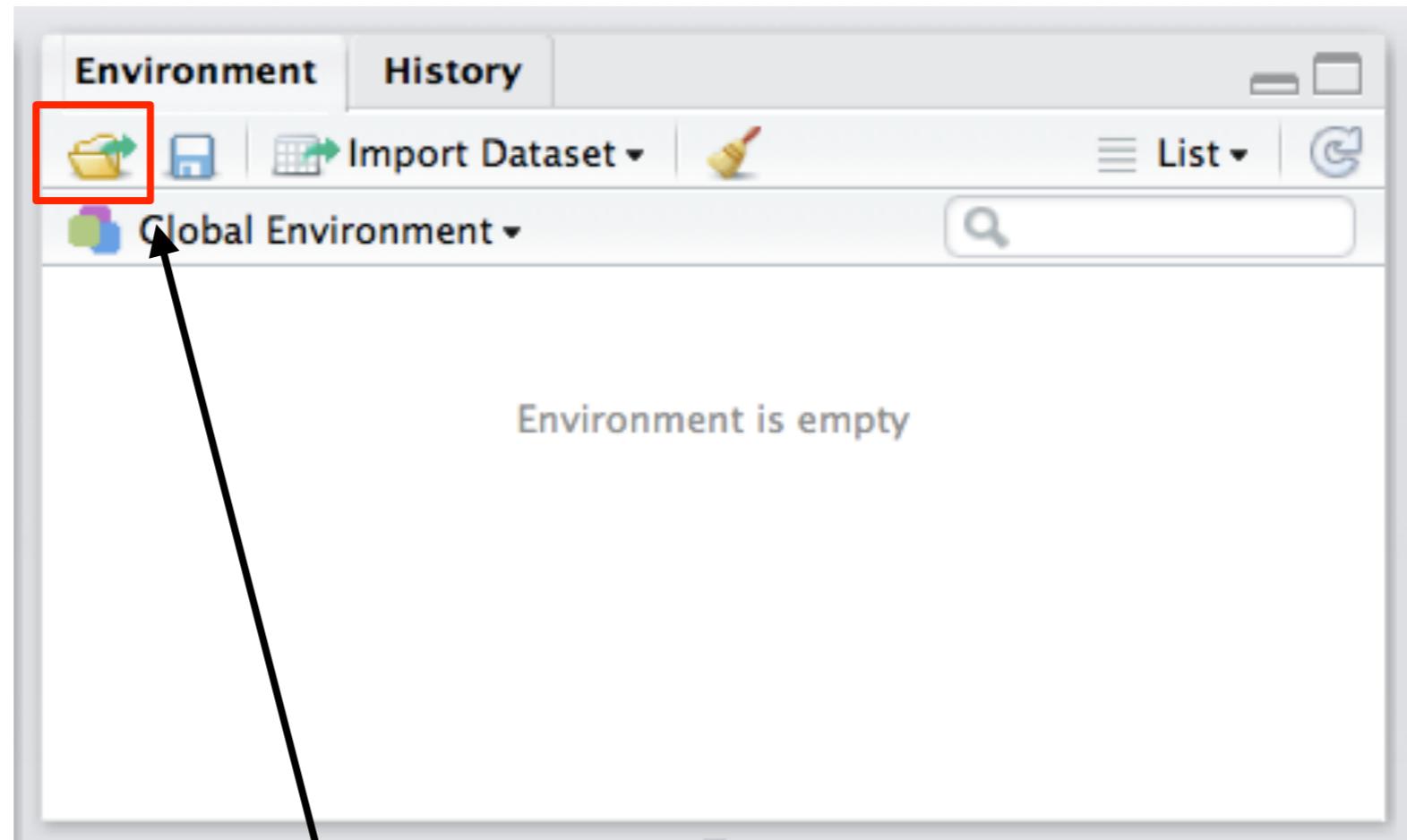
What does it mean to load data?

- 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

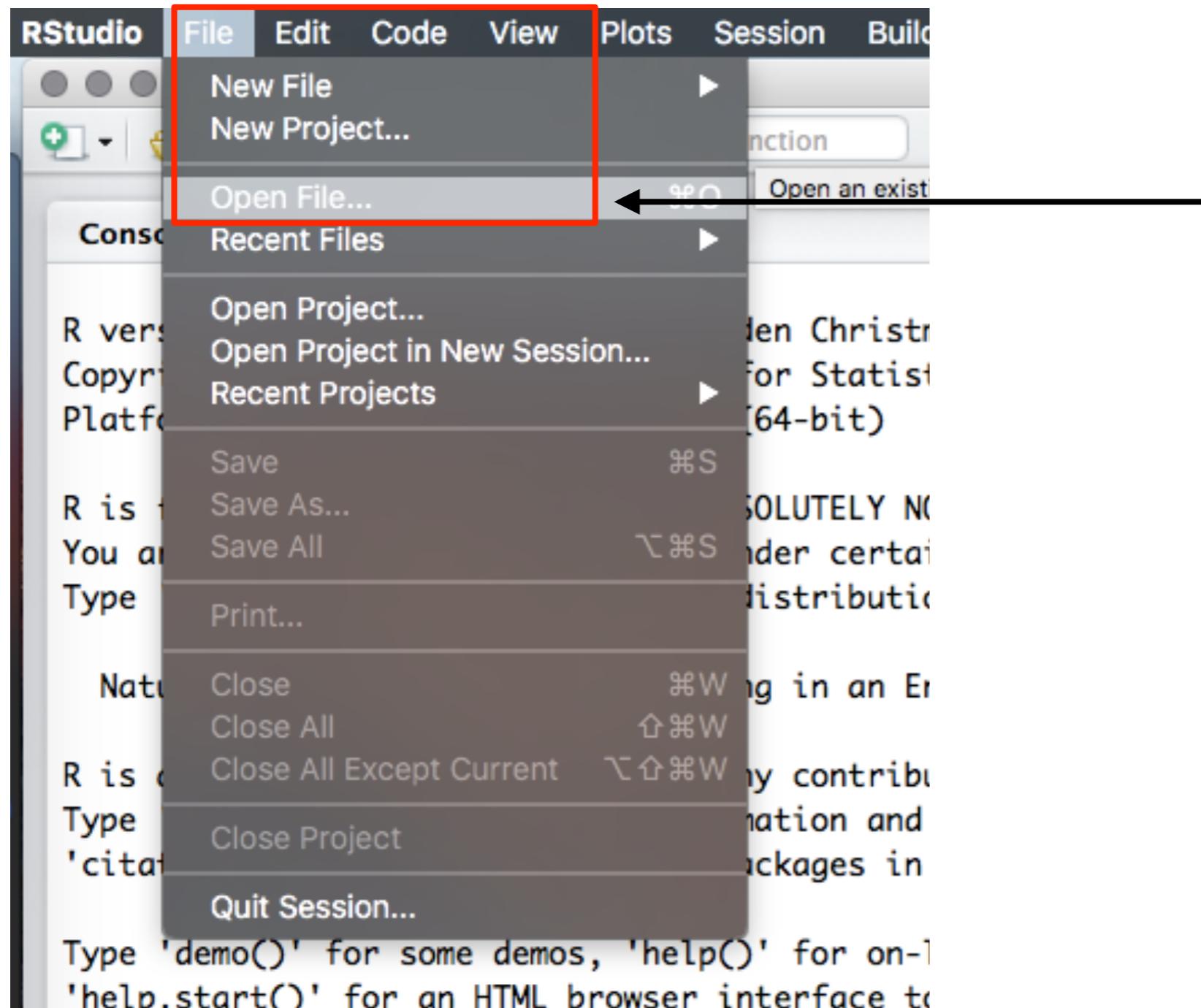
- The primary file format used by R is .Rdata (though it can also load Excel, csv, etc)
 - .Rdata files are saved workspaces
 - They contain whatever data sets, variables, functions etc that the workspace included when the file was created
- How to load an .Rdata file?
 - Hard(er) way: use the `load()` 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



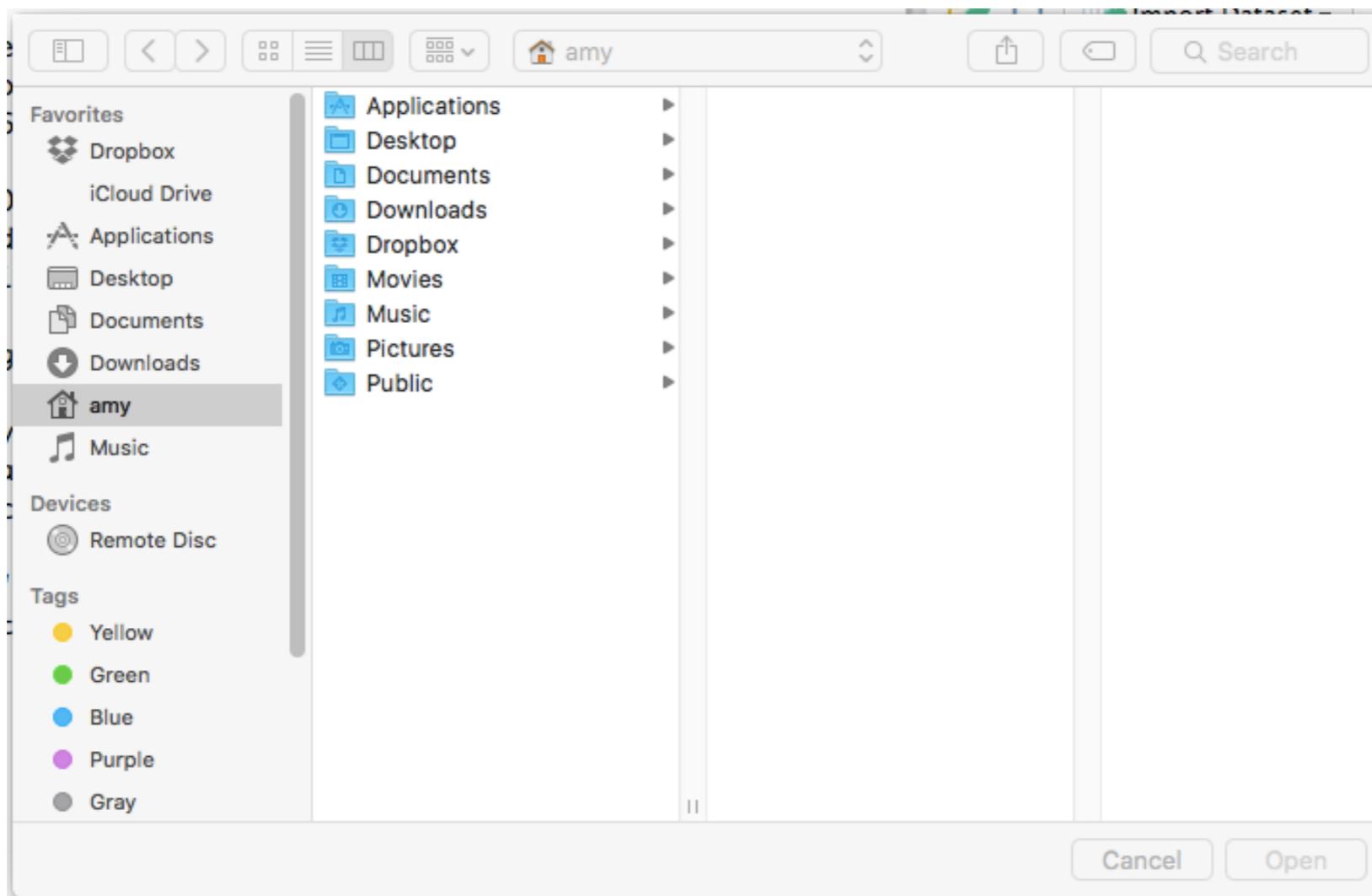
This is the “file open” button

Using Rstudio to load Rdata files



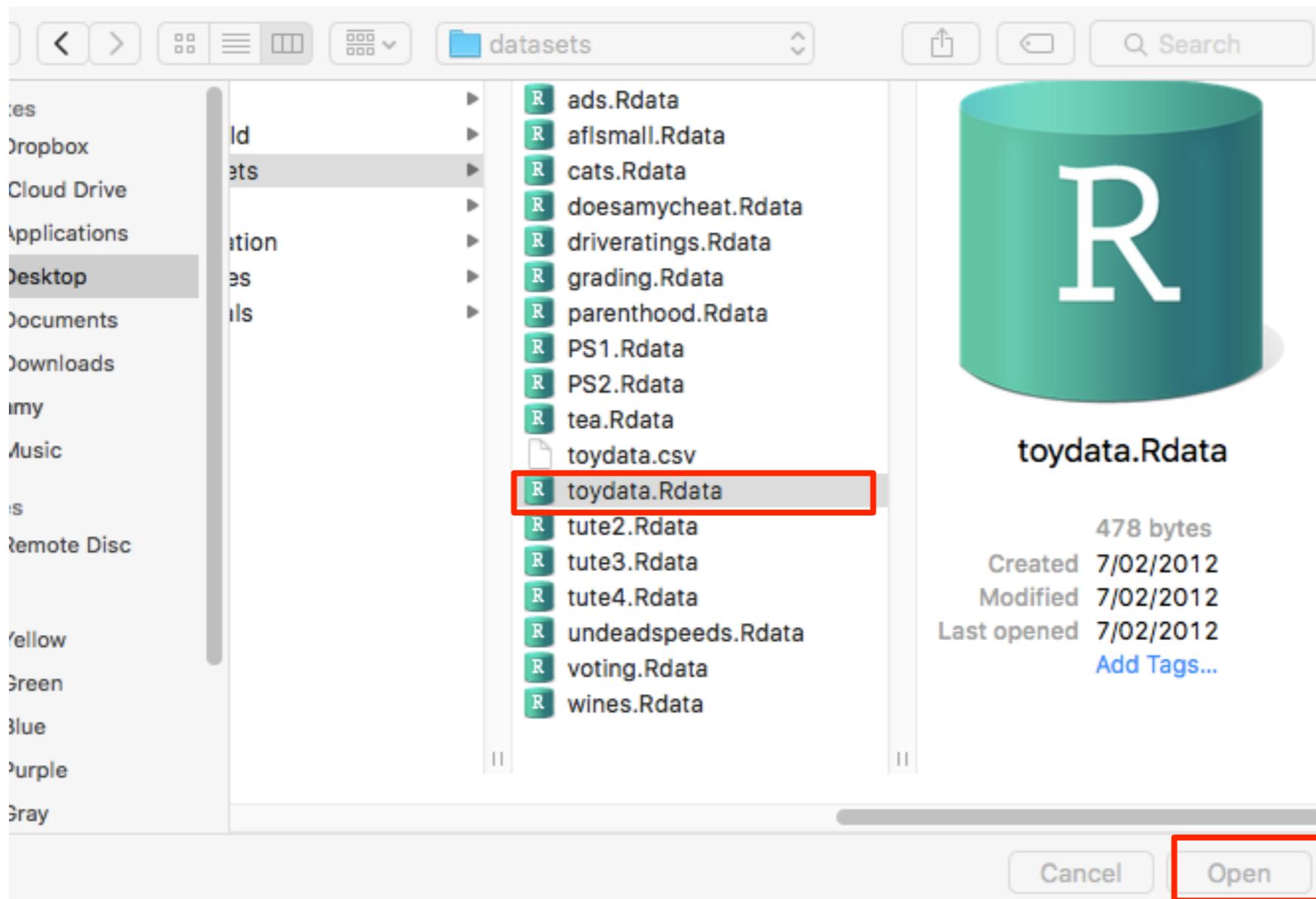
You can also use the File menu to do the same thing if you want to...

This opens a file open dialog box...



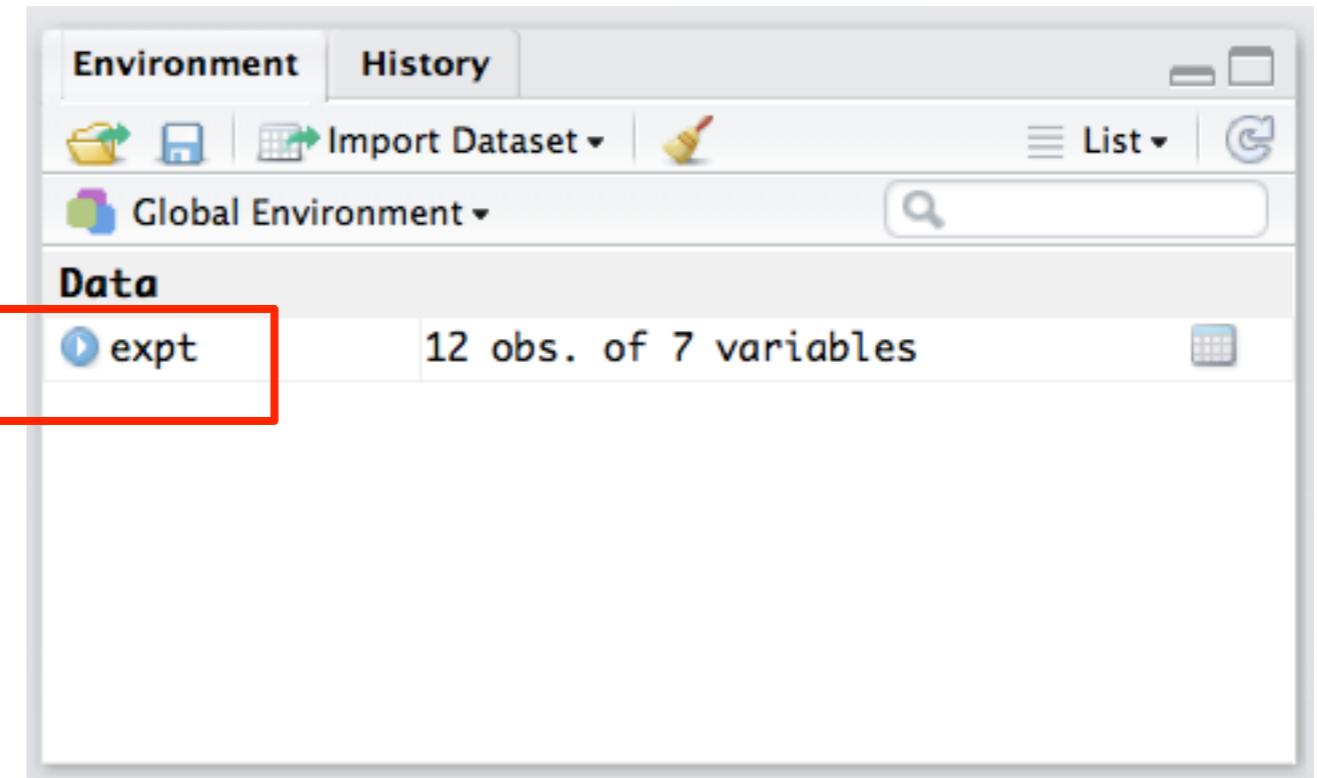
It will look different on different operating systems... it will look like a familiar Windows thing on a Windows computer, a standard Mac thing on a Mac computer etc etc...

Browse for the file you want, and open:



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

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



```
> load("~/Documents/teaching/2019/summerschool/datasets/toydata.Rdata")
```

A command like this will appear in the R console
(the command is what actually 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`

	<code>id</code>	<code>age</code>	<code>gender</code>	<code>treatment</code>	<code>hormone</code>	<code>happy</code>	<code>sad</code>
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
```

	<code>id</code>	<code>age</code>	<code>gender</code>	<code>treatment</code>	<code>hormone</code>	<code>happy</code>	<code>sad</code>
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...

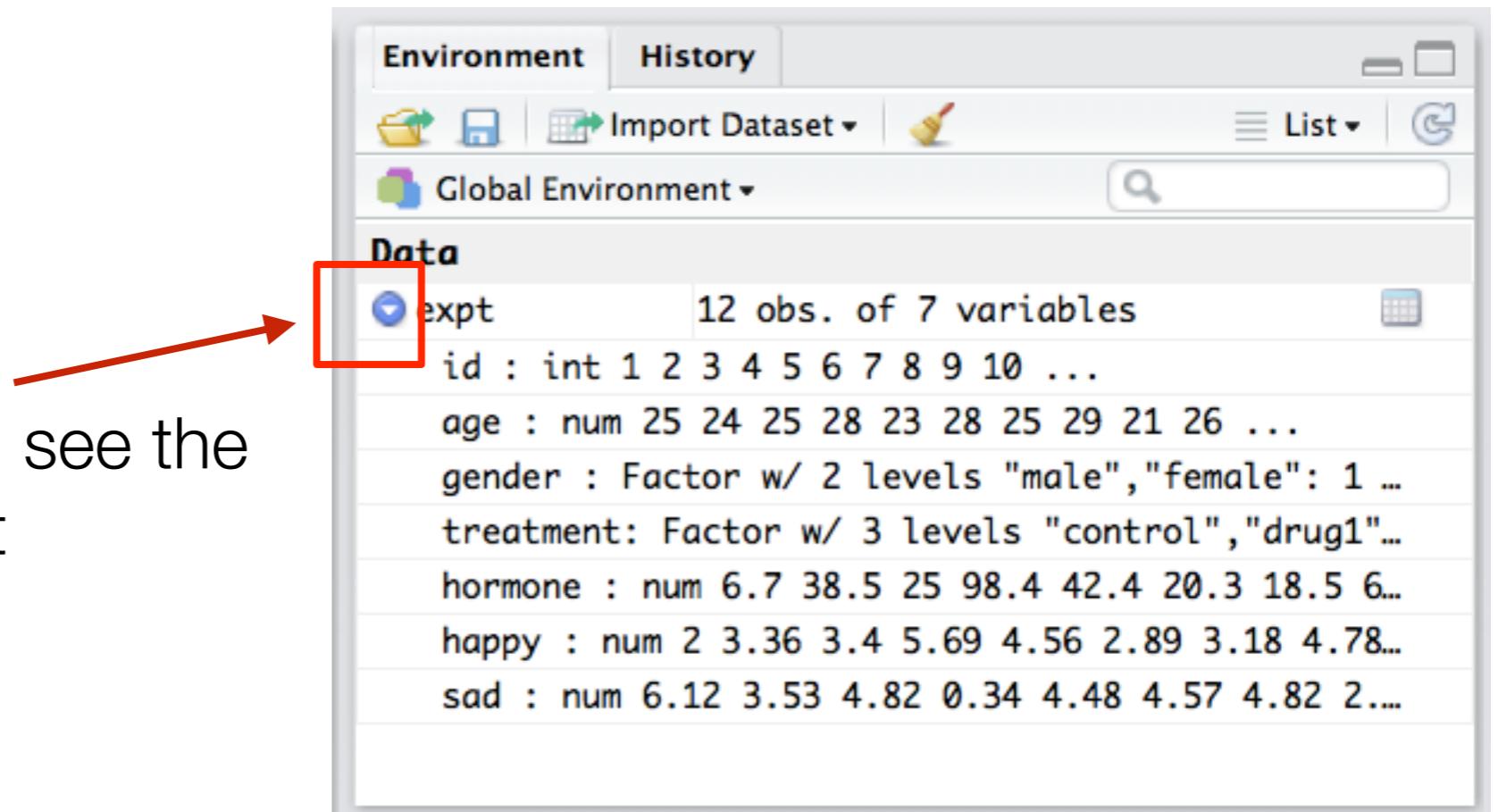
```
> expt$happy
```

```
[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

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.

The screenshot shows the RStudio interface with the 'expt' dataset loaded. A red arrow points from the 'treatment' column header in the left panel to the 'expt' entry in the 'Data' pane of the right panel.

Environment pane (left):

	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

Console pane:

```
* DONE (ggplot2)
```

The downloaded source packages are in

```
'/private/var/folders/rm/q1q1mvp12fv75l41jkm4gz7w000gn/T/RtmpetAmh8 downloaded_packages'
```

Data pane (right):

```
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....
```

System Library pane:

Package	Description	Version
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")
```

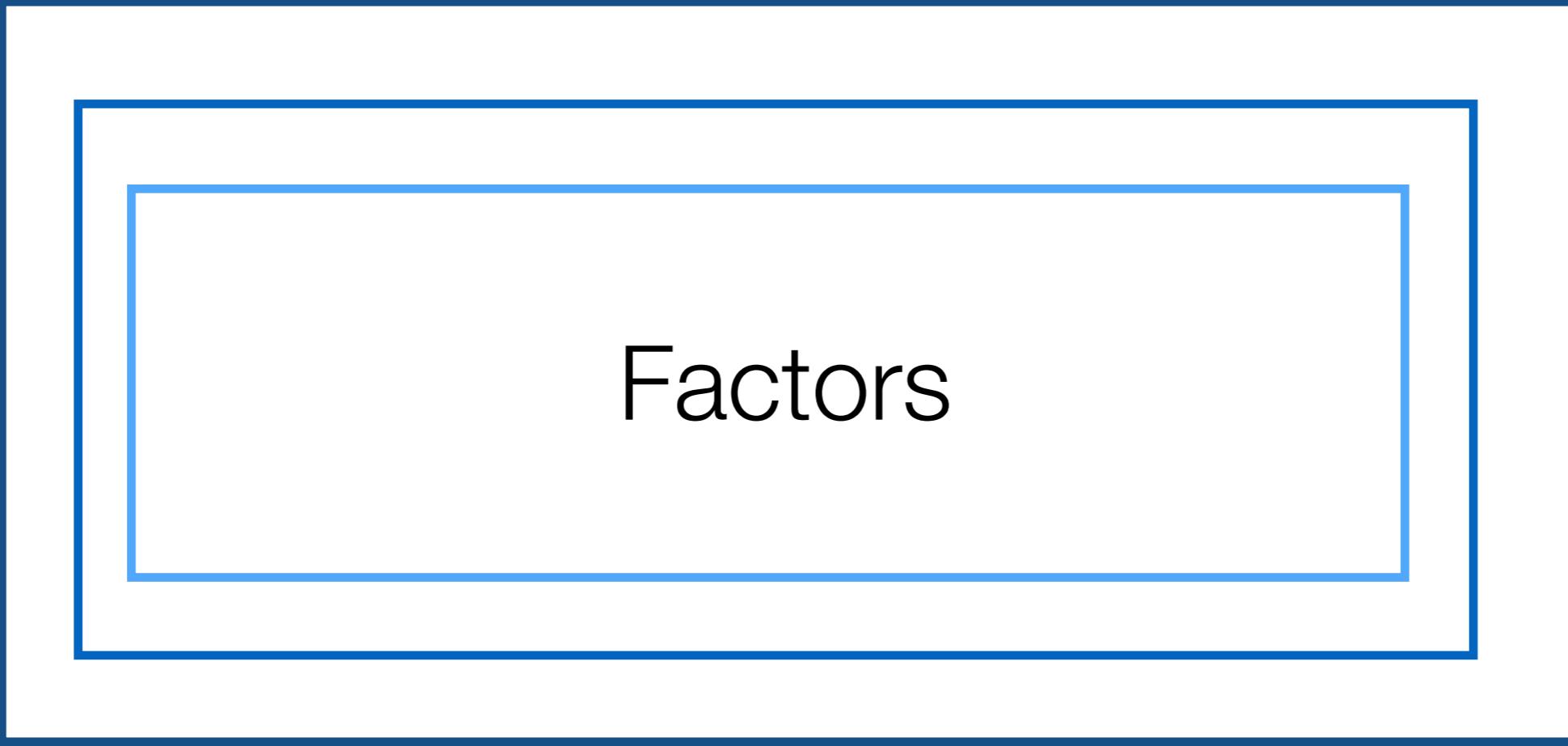
This is the name of the data frame that contains only males

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!

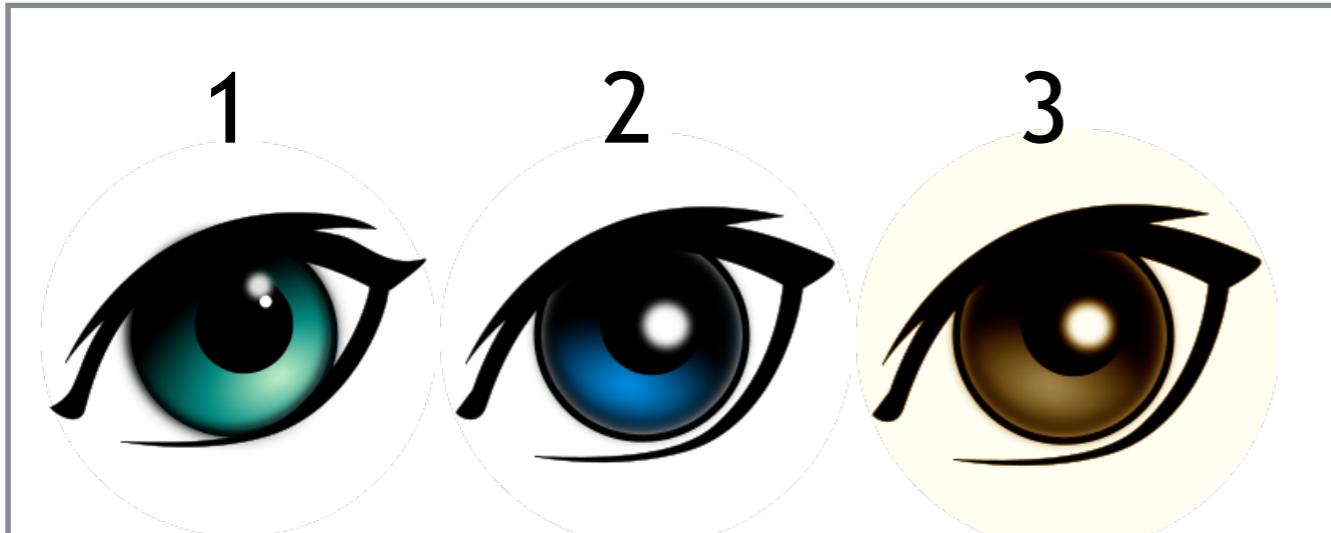
expt\$gender is actually a “factor”...

```
> expt$gender
```

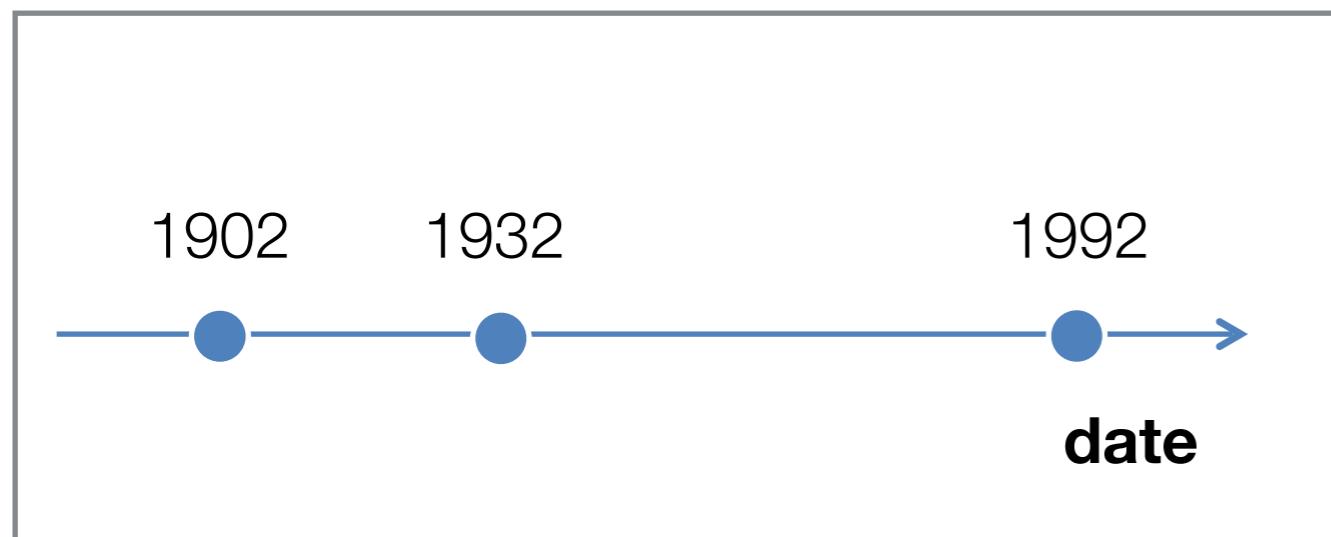
```
[1] male   male   male   male   male   male   female  
[8] female female female female female  
Levels: female male
```

```
> 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



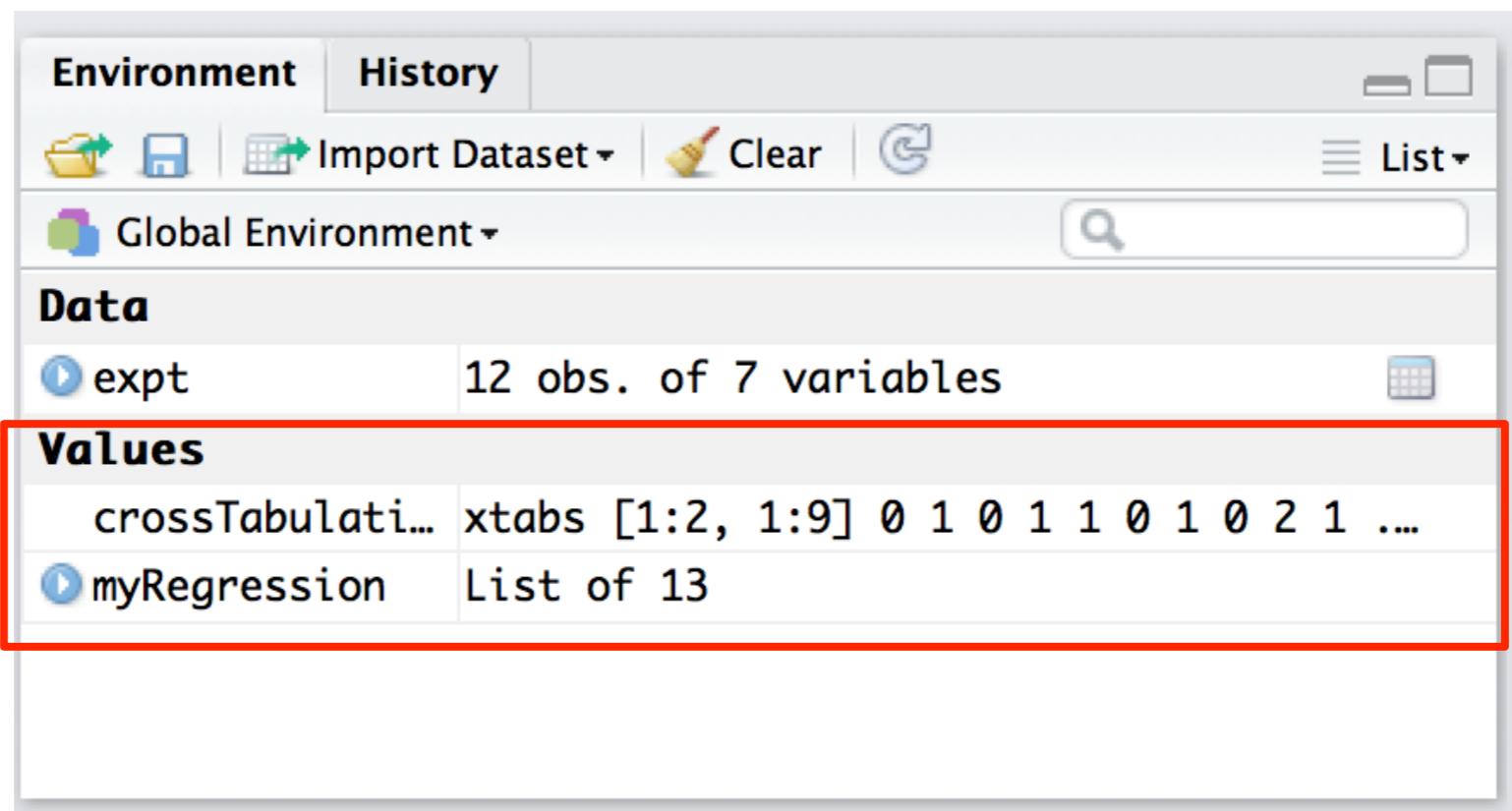
Ordinal scale data are stored as **ordered factors**

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]

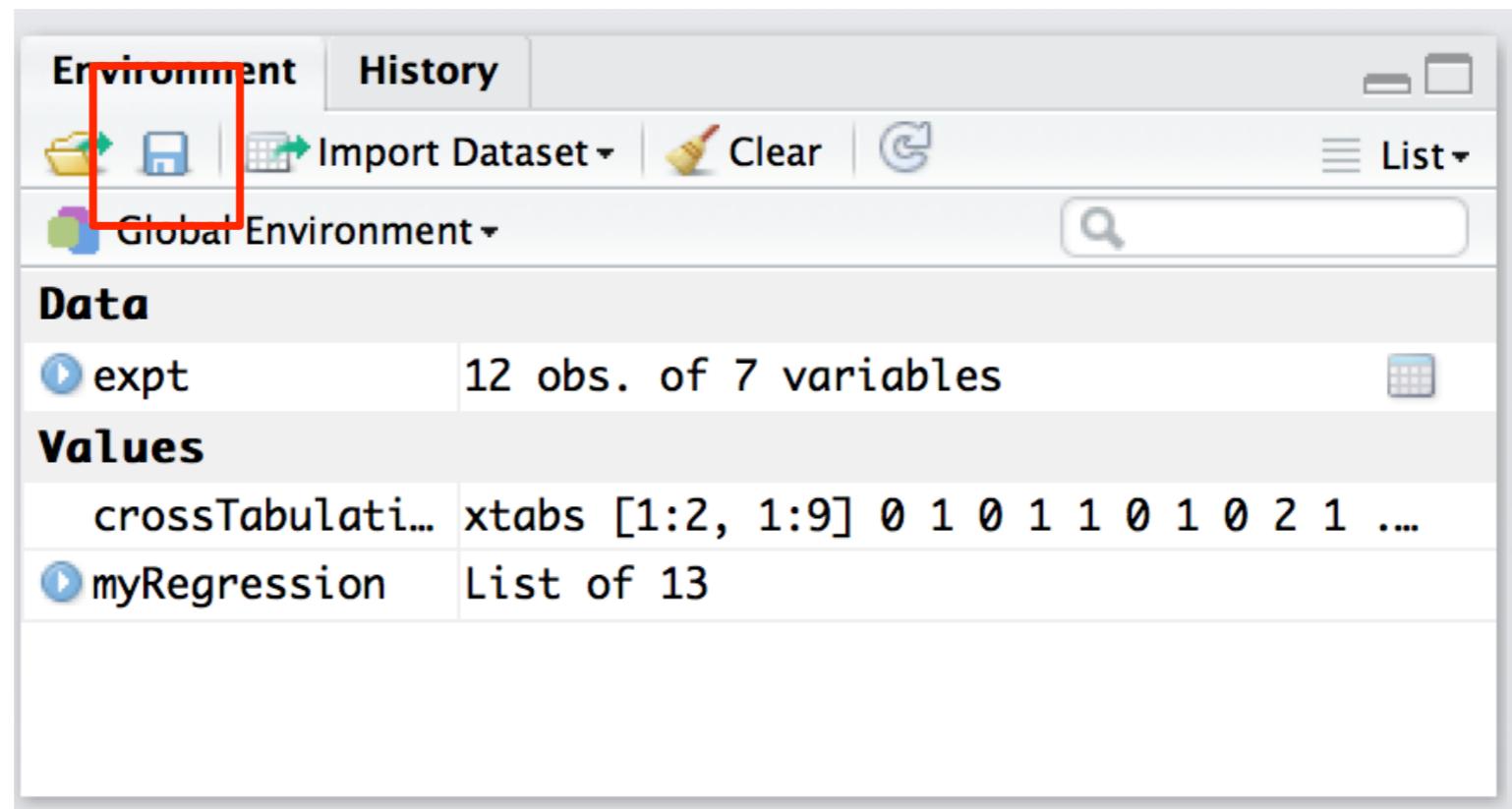
Saving your current
variables to a file

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

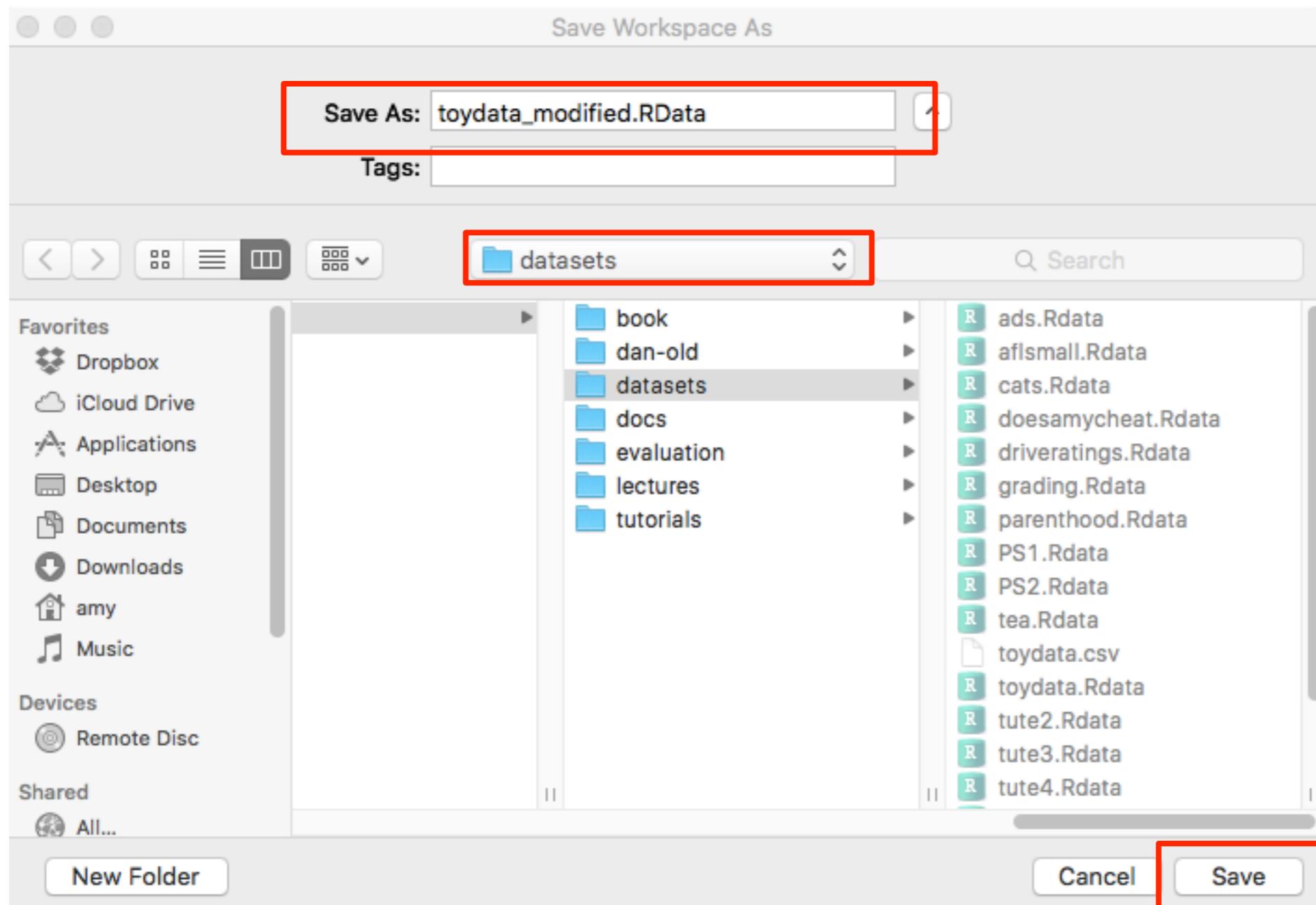


I must have done some work, there's all this new stuff in the workspace!

The save button is your friend



Browse, type a filename, and click save



Now the file is saved

```
save.image("~/Documents/teaching/2019/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”

	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		

CSV is a standard format

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

CSV is a standard format

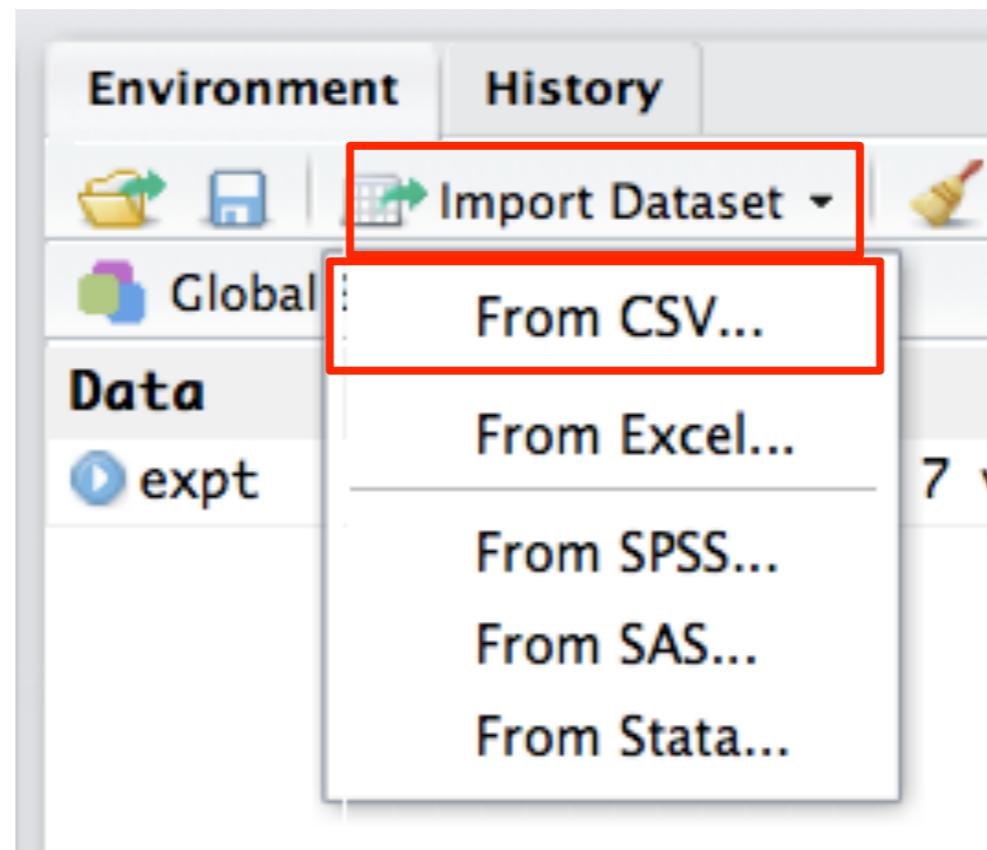
```
> expt
```

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		

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

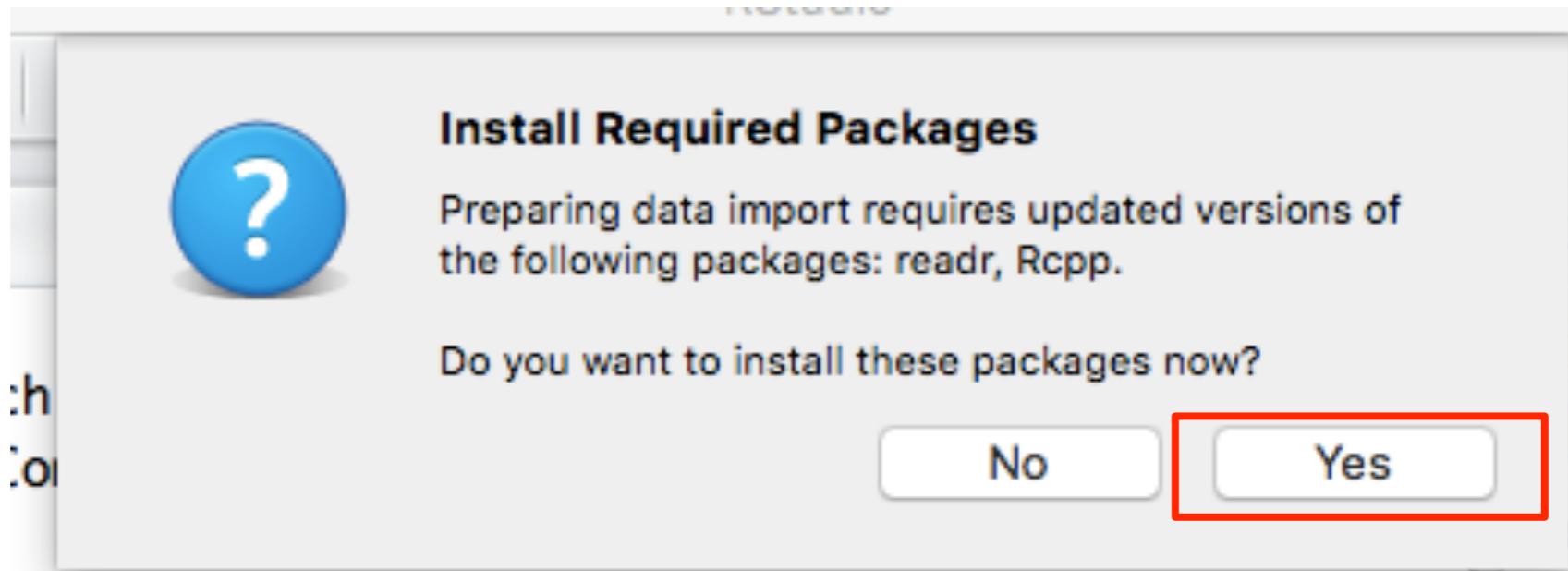
Importing CSV data using Rstudio

Click on this...



Importing CSV data using Rstudio

You may have to install some other packages...



Importing CSV data using Rstudio

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

Import Text Data

File/Url:

Data Preview:

Import Options:

Name: <input type="text" value="dataset"/>	<input checked="" type="checkbox"/> First Row as Names	Delimiter: <input type="button" value="Comma"/>	Escape: <input type="button" value="None"/>
Skip: <input type="text" value="0"/>	<input checked="" type="checkbox"/> Trim Spaces	Quotes: <input type="button" value="Default"/>	Comment: <input type="button" value="Default"/>
	<input checked="" type="checkbox"/> Open Data Viewer	Locale: <input type="button" value="Configure..."/>	NA: <input type="button" value="Default"/>

Code Preview:

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

Importing CSV data using Rstudio

When you see it, go ahead and “Import”

Import Text Data

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: <input type="text" value="toydata"/>	<input checked="" type="checkbox"/> First Row as Names	Delimiter: <input type="button" value="Comma ▾"/>	Escape: <input type="button" value="None ▾"/>
Skip: <input type="text" value="0"/>	<input checked="" type="checkbox"/> Trim Spaces	Quotes: <input type="button" value="Default ▾"/>	Comment: <input type="button" value="Default ▾"/>
	<input checked="" type="checkbox"/> Open Data Viewer	Locale: <input type="button" value="Configure..."/>	NA: <input type="button" value="Default ▾"/>

Code Preview:

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

The screenshot shows the RStudio interface with the 'toydata' data frame open in a viewer panel. The panel has a header with tabs for 'View', 'Text', and 'Code'. Below the header is a search bar and a 'Filter' button. The main area displays a table with 12 rows and 8 columns. The columns are labeled: id, age, gender, treatment, hormone, happy, and sad. The data shows a mix of male and female participants aged 19 to 30, assigned to control or drug1/drug2 treatments, with hormone levels ranging from 6.7 to 98.4.

	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/2019/summerschool/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!

Environment History
Import Dataset Global Environment
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/2019/summerschool/datasets/toydata.csv")
> View(toydata)
```



Scripts

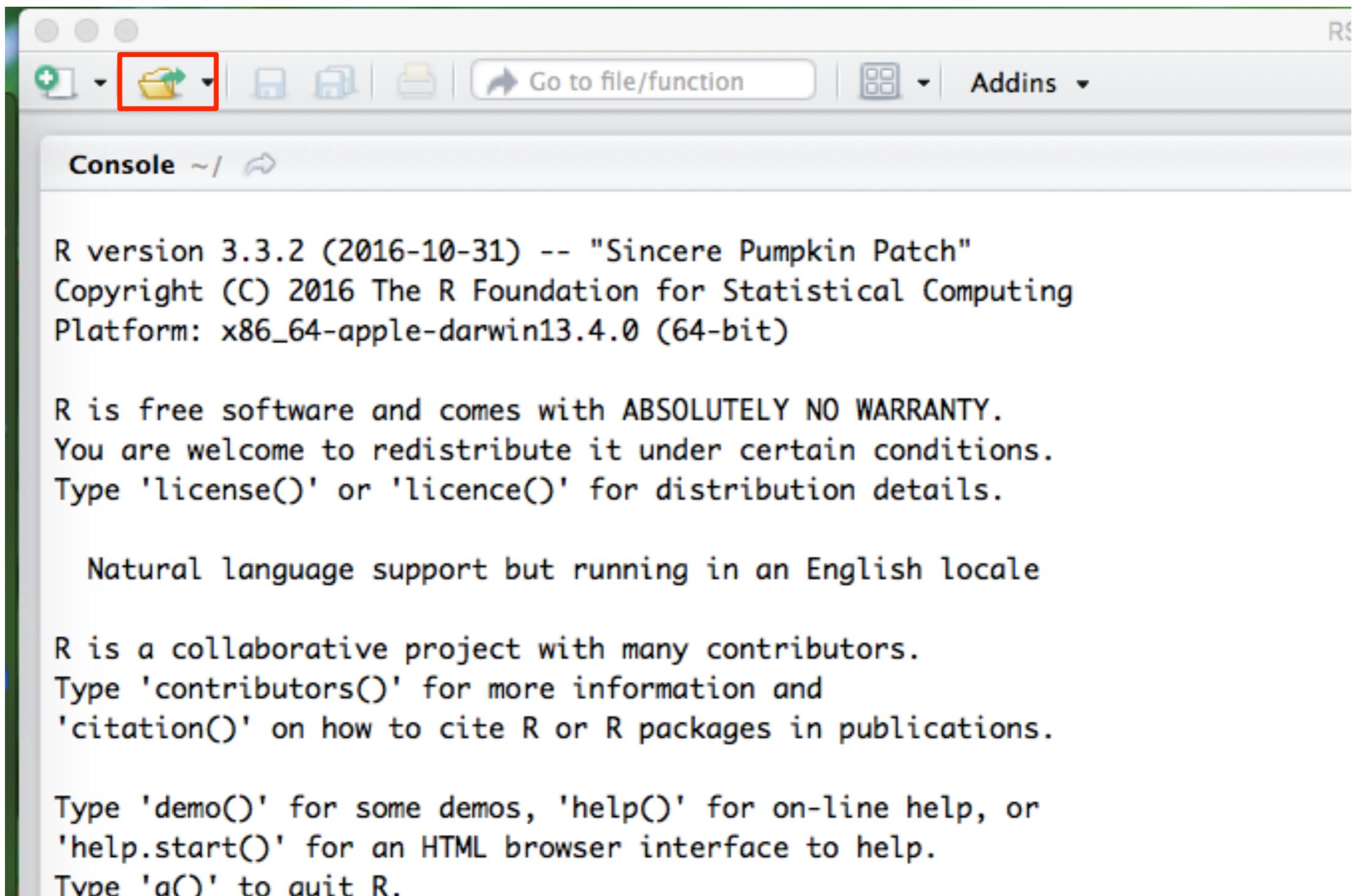
Working with data

- 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

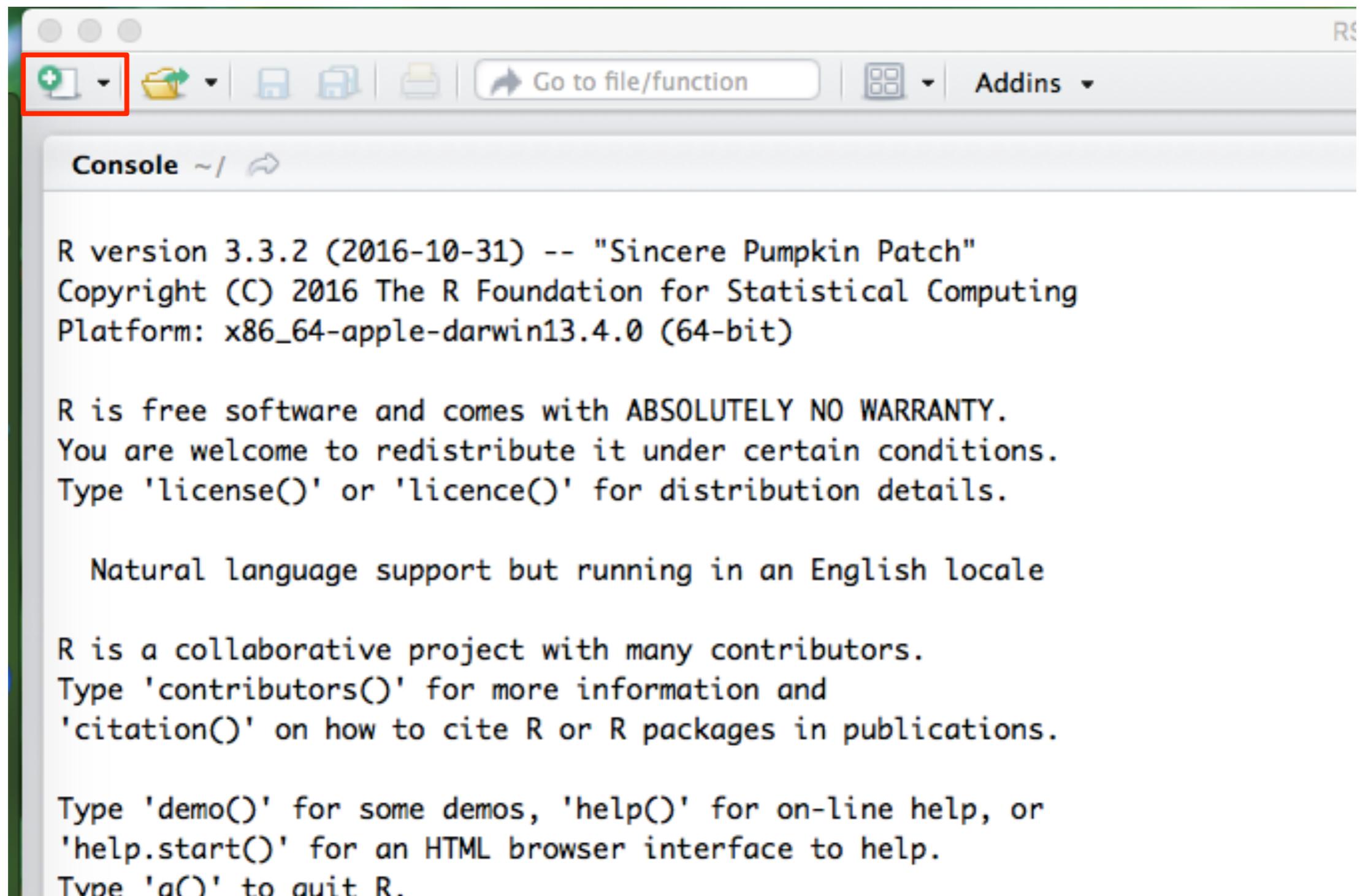
Scripts

- 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.

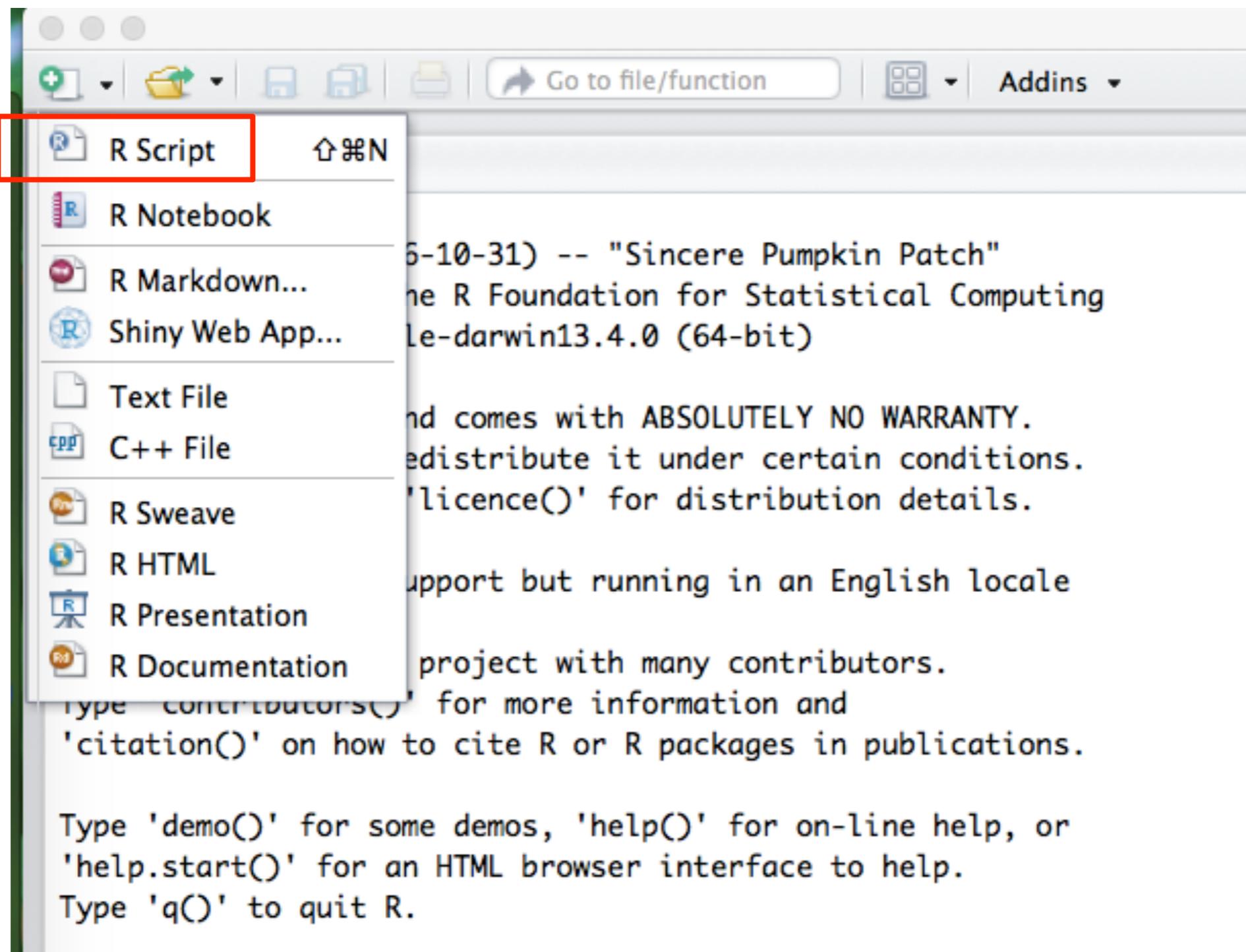
Click here to open a saved script



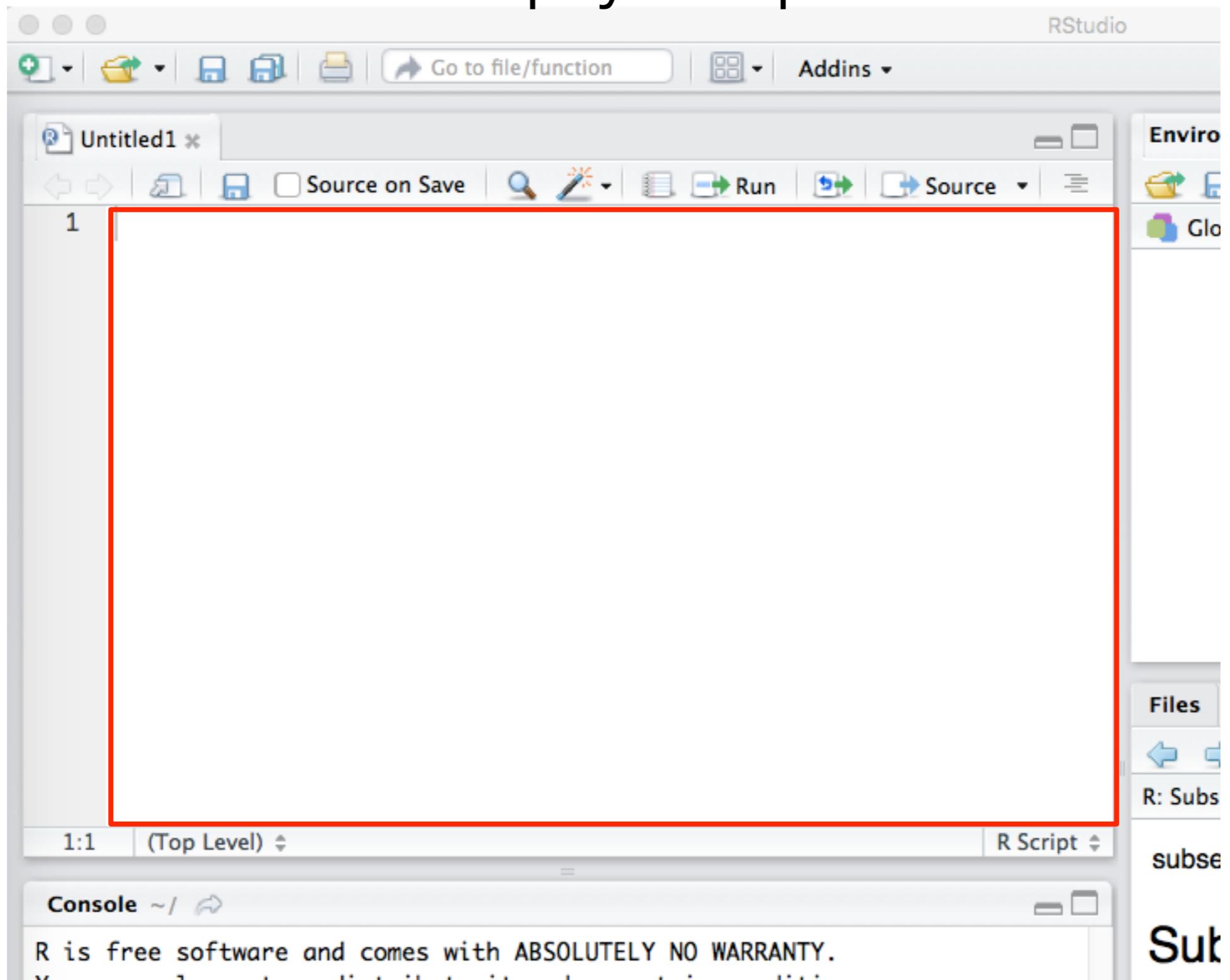
Or here to create a new one



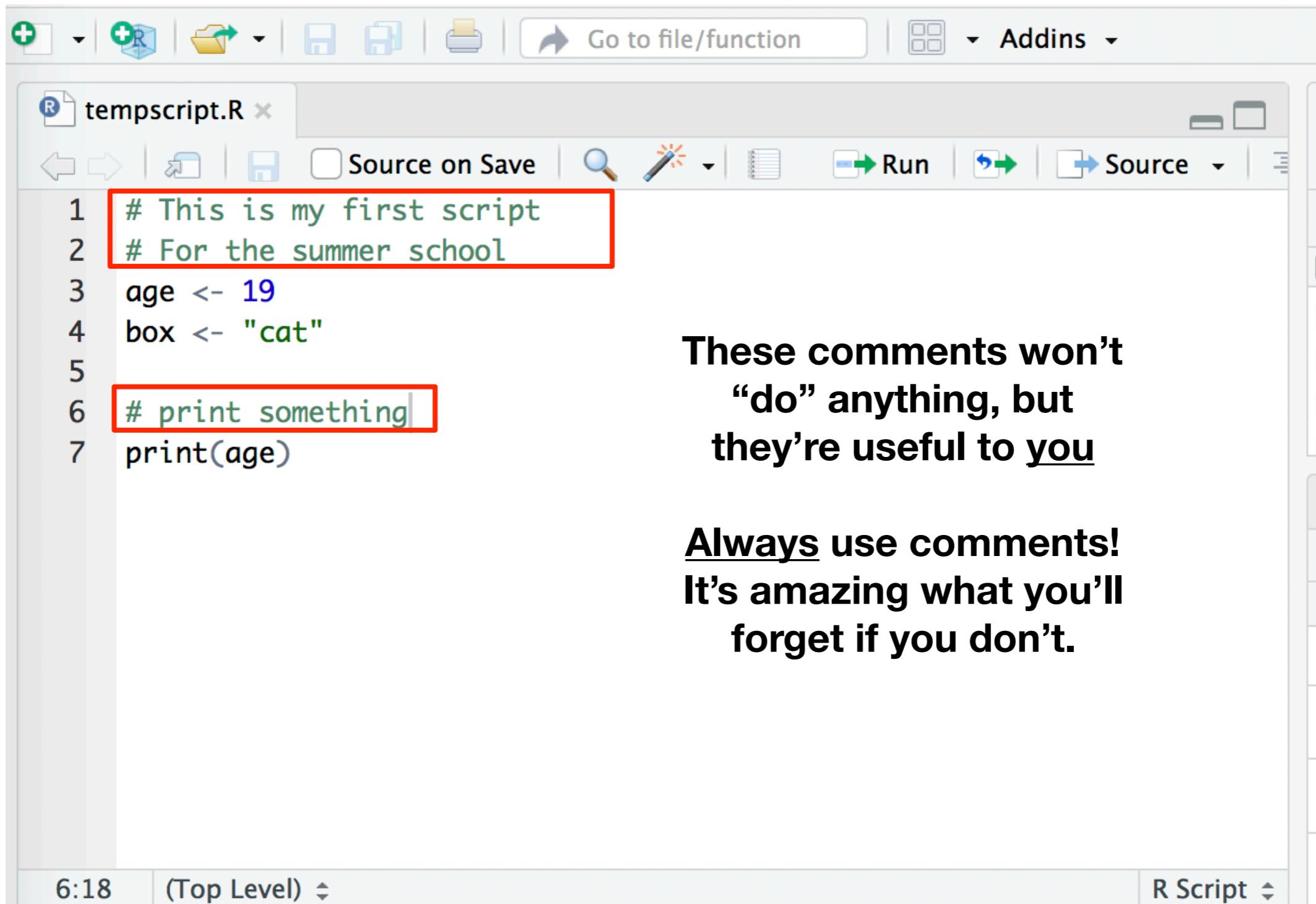
Or here to create a new one



An empty script...



Type some R commands here...



The screenshot shows the RStudio IDE interface. The top menu bar includes standard icons for file operations like Open, Save, Print, and Go to file/function, followed by an Addins dropdown. Below the menu is a toolbar with various icons for navigation, search, and execution. A script file titled "tempscript.R" is open in the main editor area. The code contains the following R commands:

```
1 # This is my first script
2 # For the summer school
3 age <- 19
4 box <- "cat"
5
6 # print something
7 print(age)
```

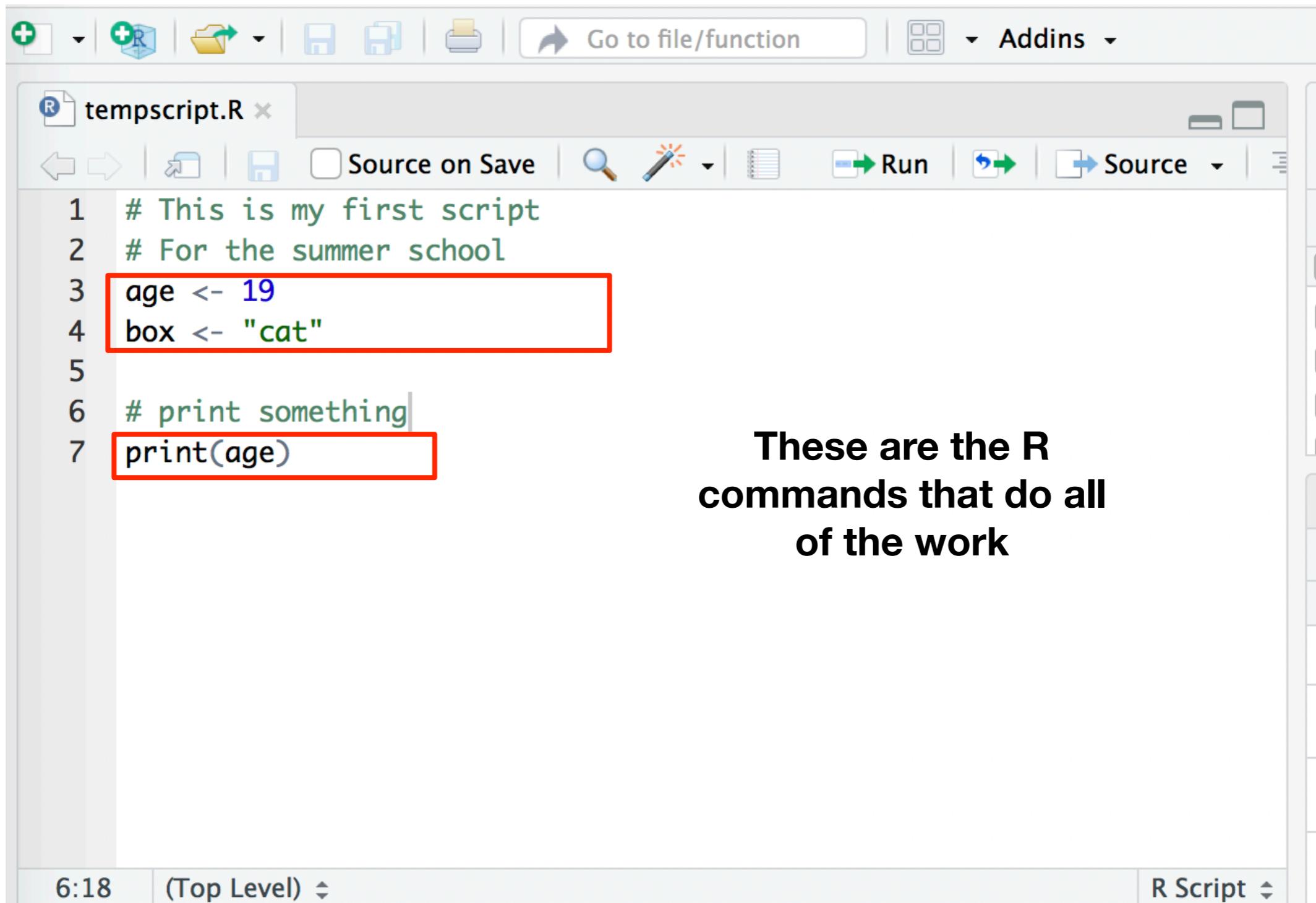
Two specific lines of code are highlighted with red boxes: the first two lines and the line "# print something". To the right of the editor, there is explanatory text.

**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.**

At the bottom of the RStudio window, the status bar displays the time "6:18" and the current mode "(Top Level) ▾", along with a tab indicator for "R Script" ▾.

Type some R commands here...

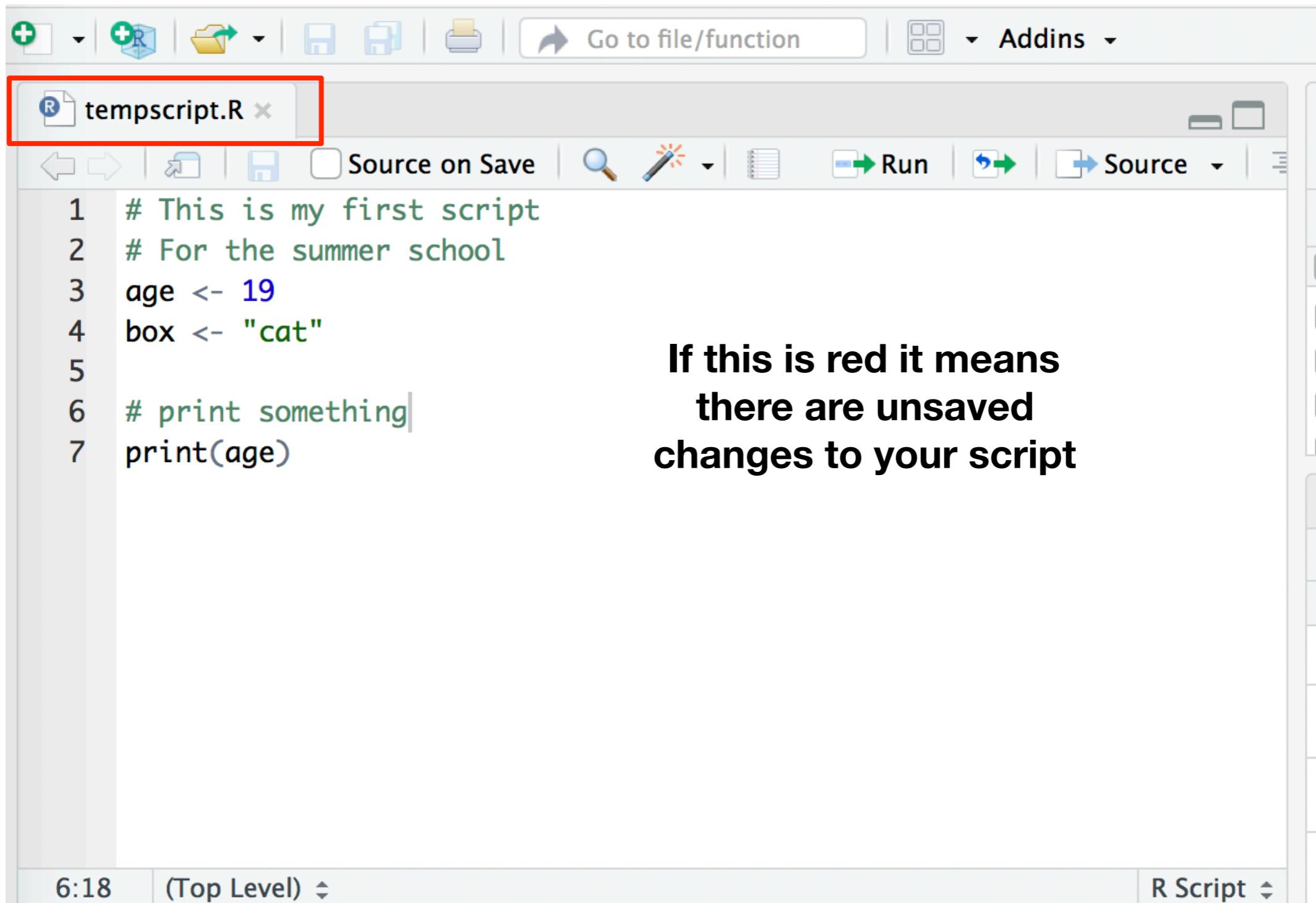


```
# This is my first script
# For the summer school
age <- 19
box <- "cat"

# print something
print(age)
```

These are the R commands that do all of the work

Type some R commands here...



The screenshot shows the RStudio interface with an R script file open. The title bar shows the file name "tempscript.R". A red box highlights the title bar area. The main editor window contains the following R code:

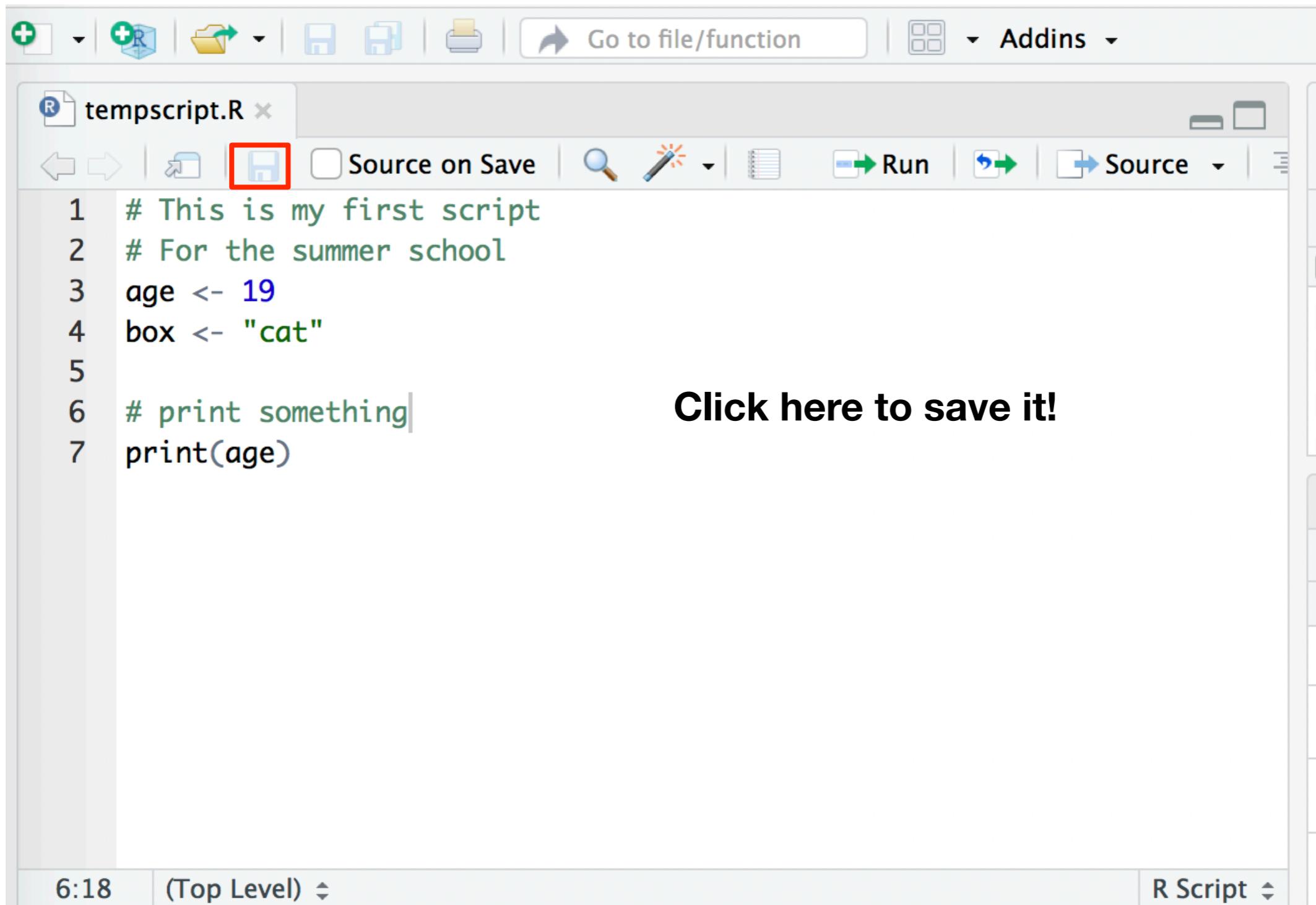
```
1 # This is my first script
2 # For the summer school
3 age <- 19
4 box <- "cat"
5
6 # print something
7 print(age)
```

To the right of the code, there is a text block with the following message:

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

The status bar at the bottom left shows the time "6:18" and the current mode "(Top Level)". The status bar at the bottom right shows "R Script".

Type some R commands here...

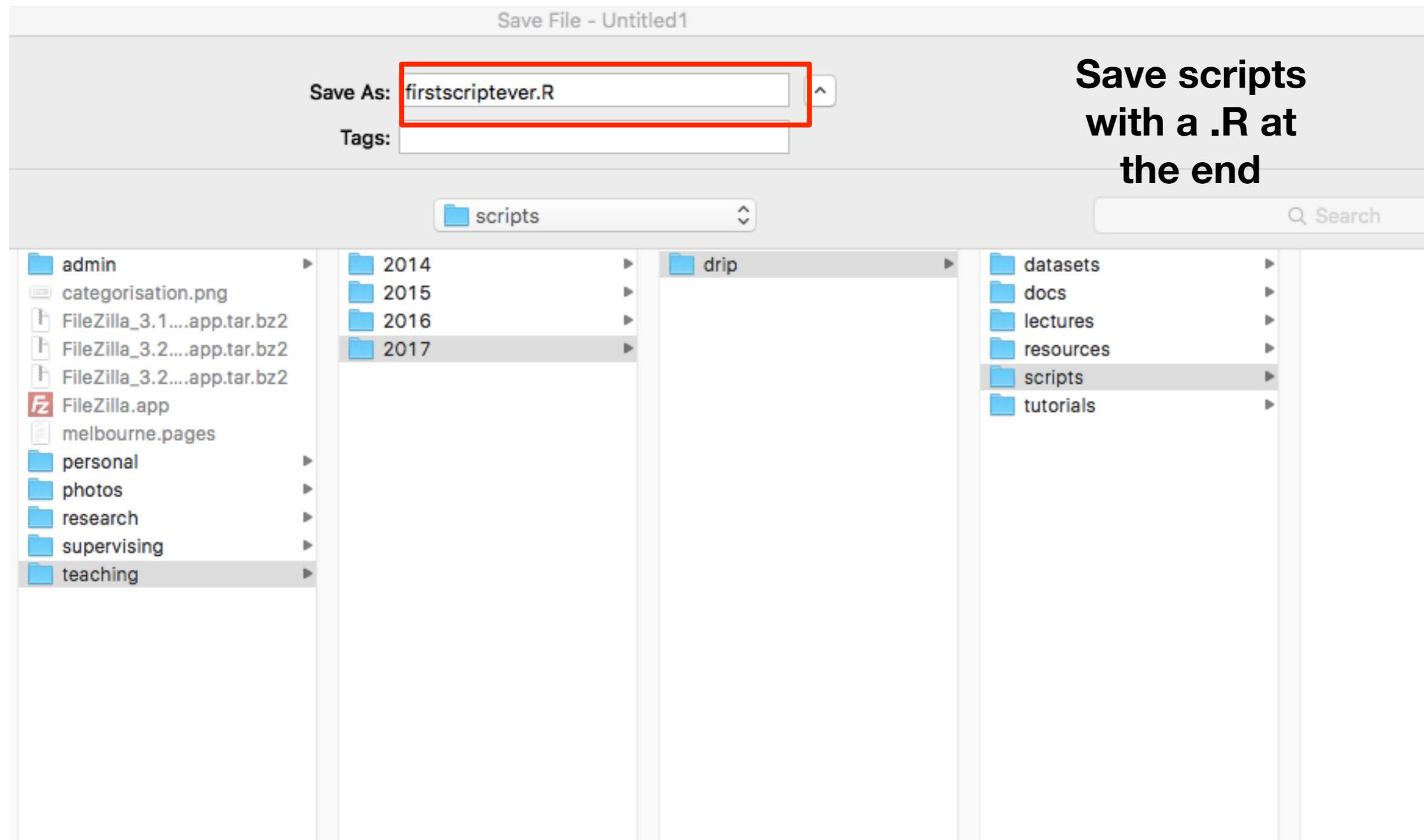


The screenshot shows a software interface for editing R scripts. At the top, there's a toolbar with various icons for file operations like saving, printing, and running. Below the toolbar, a tab bar shows a file named "tempscript.R". The main area contains the following R code:

```
1 # This is my first script
2 # For the summer school
3 age <- 19
4 box <- "cat"
5
6 # print something
7 print(age)
```

A red box highlights the "Save" icon in the toolbar, which is a standard Windows-style save icon. To the right of the code, there's a large, bold, black text overlay that reads "Click here to save it!". At the bottom of the editor window, there are status bars showing "6:18" and "(Top Level)" on the left, and "R Script" on the right.

Hey look, another save window!



The screenshot shows the RStudio interface with the following details:

- Title Bar:** myScriptIntroToR.R *
- Toolbar:** Includes icons for back, forward, file, and search, followed by a "Source on Save" checkbox, a magnifying glass icon, a pencil icon, a notebook icon, a "Run" button, and a "Source" button. The "Source" button is highlighted with a red box.
- Code Editor:** Displays the following R script:

```
1 # this is my first script
2 # it's just for DRIP class
3 #
4 # author: Amy Perfors
5
6 # define some variables
7 age <- 19
8 box <- "cat"
9
10 # print something
11 print( box )
12 print( age )
```
- Status Bar:** Shows "8:13" and "(Top Level) ▾" on the left, and "R Script ▾" on the right.

Text Overlay: The text "Click here to run the script" is displayed in bold black font on the right side of the window, positioned above the status bar.

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?

nothing; this is an
empty line and a
comment

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

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

# print something
```

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

The screenshot shows the RStudio interface with the following components:

- Script Editor (tempscript.R):** Contains the following R code:

```
1 # This is my first script
2 # For the summer school
3 age <- 19
4 box <- "cat"
5
6 # print something
7 print(age)
```
- Environment Browser:** Shows the global environment with the following objects:

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] "
- Console:** Displays the output of running the script:

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

The command `source('~/Documents/teaching/2018/summerschool/chdss2018/day0_rbootcamp/tempscript.R')` is highlighted with a red box.

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

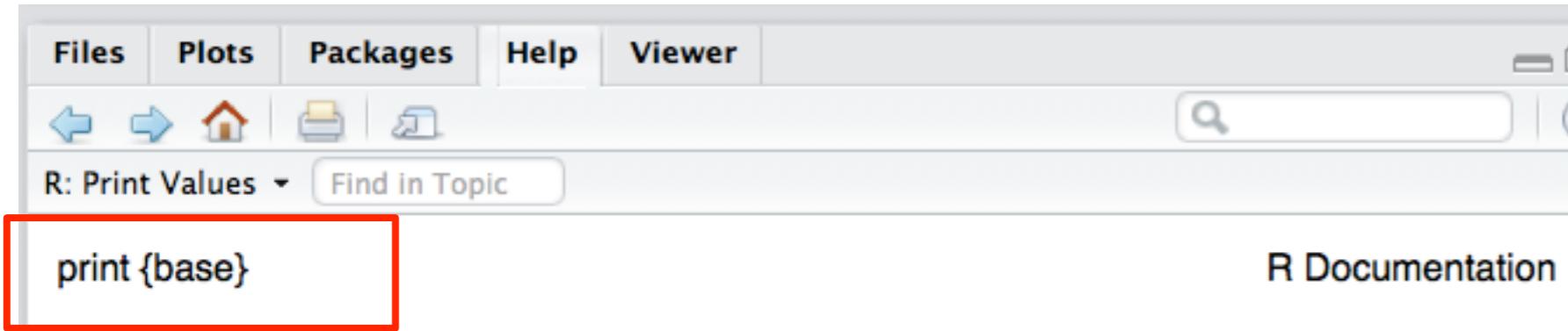
The screenshot shows the RStudio interface with the following panels:

- Code Editor:** Displays the script `myScriptIntroToR.R` containing R code.
- Environment:** Shows the global environment with variables `age` (19) and `box` ("cat").
- Console:** Displays the output of running the script, including help messages and the results of `print()` calls.
- Viewer:** Contains the help page for the `print` function, which is highlighted with a red border.

Help Page Content (Viewer Panel):

- Header:** Files, Plots, Packages, Help, Viewer (with Help selected).
- Title:** R: 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 [classes](#).
- Usage:** `print(x, ...)`
- Note:** ## S3 method for class 'factor'

R documentation



The screenshot shows the RStudio interface with the 'Viewer' tab selected. The title bar includes 'Files', 'Plots', 'Packages', 'Help', 'Viewer', and a search bar. Below the title bar, there are icons for back, forward, home, and other navigation functions. The main content area displays the documentation for the 'print' function. A red box highlights the code 'print {base}' in the 'Description' section. The page title is 'Print Values' and the subtitle is 'R Documentation'. The 'Description' section contains the following text: '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 [classes](#).'. The 'Usage' section shows the R code for the 'print' function:

```
print(x, ...)

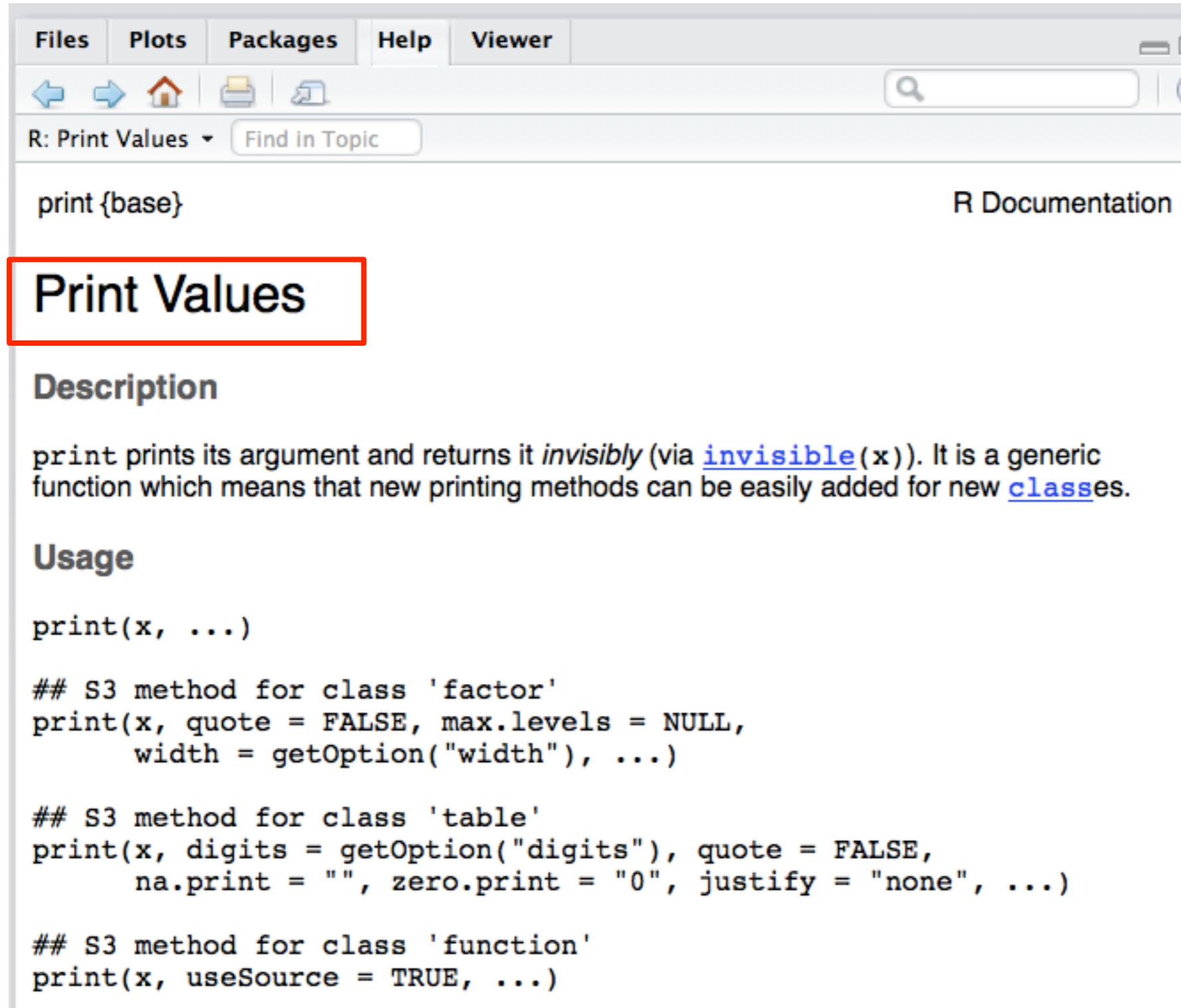
## S3 method for class 'factor'
print(x, quote = FALSE, max.levels = NULL,
      width = getOption("width"), ...)

## 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



The screenshot shows the RStudio interface with the 'Viewer' tab selected. The title bar includes 'Files', 'Plots', 'Packages', 'Help', 'Viewer', and a search bar. Below the title bar, there are navigation icons and a dropdown menu 'R: Print Values'. A red box highlights the title 'Print Values'.

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 [classes](#).

Usage

```
print(x, ...)

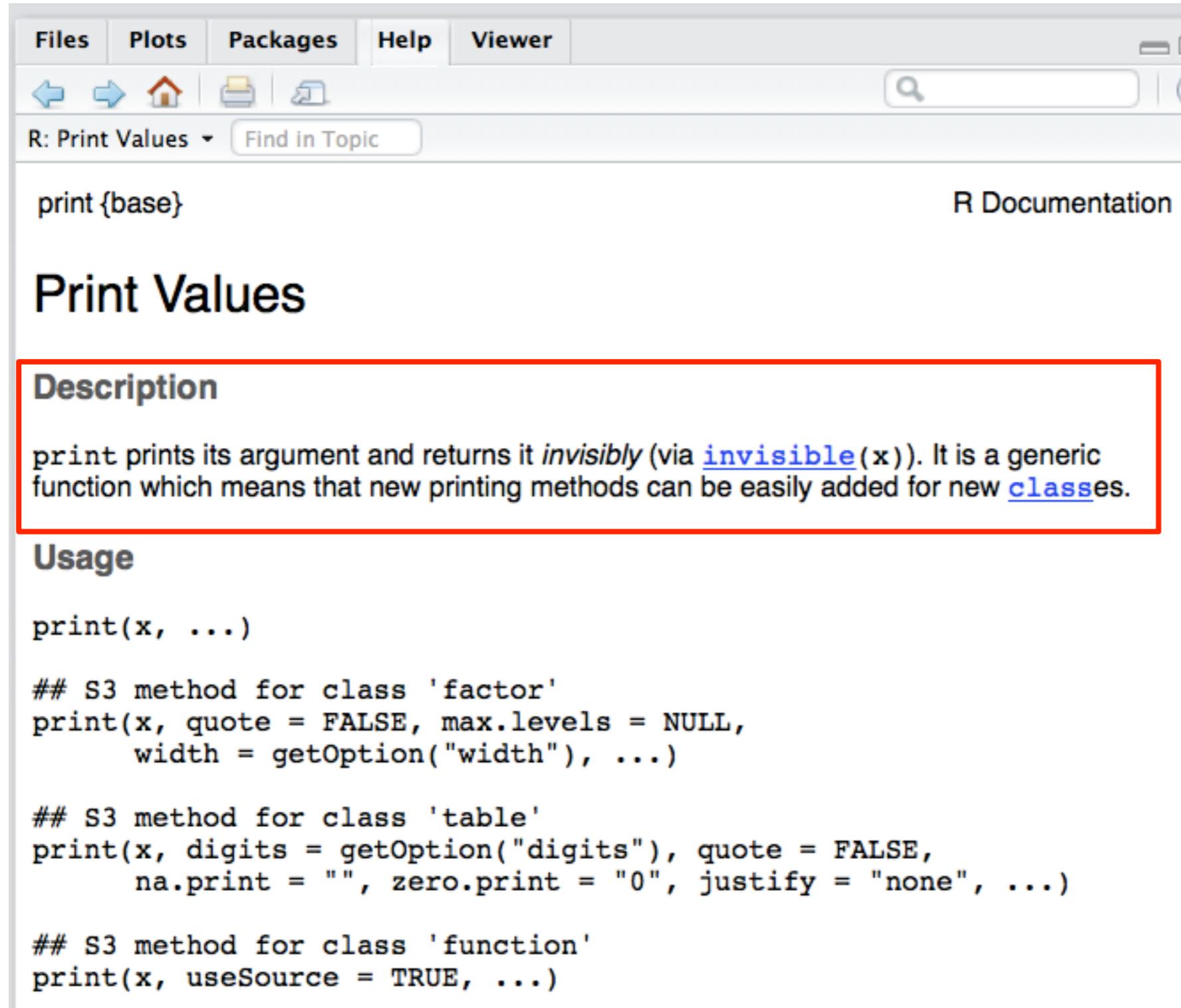
## S3 method for class 'factor'
print(x, quote = FALSE, max.levels = NULL,
      width = getOption("width"), ...)

## 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



The screenshot shows the R Help Viewer interface. The title bar includes tabs for Files, Plots, Packages, Help, and Viewer, along with standard window controls. Below the tabs, there are icons for back, forward, search, and help. The main menu bar shows "R: Print Values" and a "Find in Topic" button. The main content area displays the documentation for the `print` function. The title is "Print Values". The "Description" section is highlighted with a red border and contains the text: "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 `classes`". The "Usage" section follows, containing several examples of the `print` function for different classes: factor, table, and function.

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 `classes`.

Usage

```
print(x, ...)

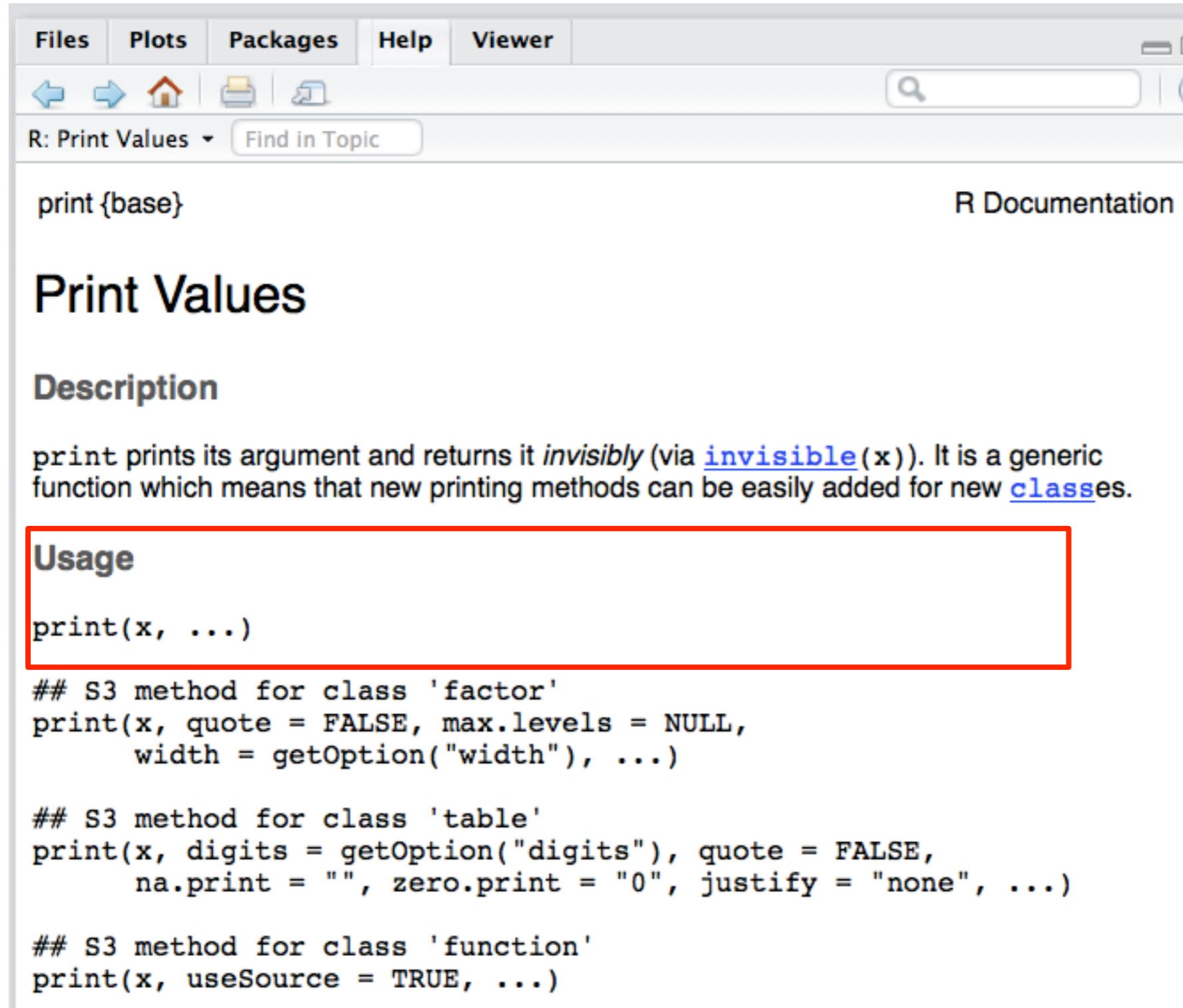
## S3 method for class 'factor'
print(x, quote = FALSE, max.levels = NULL,
      width =getOption("width"), ...)

## 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



The screenshot shows the RStudio interface with the 'Viewer' tab selected. The title bar includes 'Files', 'Plots', 'Packages', 'Help', 'Viewer', and a search bar. Below the title bar, there are navigation icons and a dropdown menu 'R: Print Values'. A 'Find in Topic' button is also present. The main content area displays the documentation for the 'print' function. The title 'Print Values' is bolded. The 'Description' section explains that '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 `classes`. The 'Usage' section is highlighted with a red border and contains the base usage: `print(x, ...)`. Below this, three S3 method definitions are shown for 'factor', 'table', and 'function' classes.

```
print {base}

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

The screenshot shows the RStudio interface with the 'Viewer' tab selected. The title bar says 'R: Print Values'. The main content area displays the R documentation for the 'print' function. The 'Usage' section is highlighted with a red box and has two arrows pointing to it from the explanatory text on the right.

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 [classes](#).

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", ...)
```

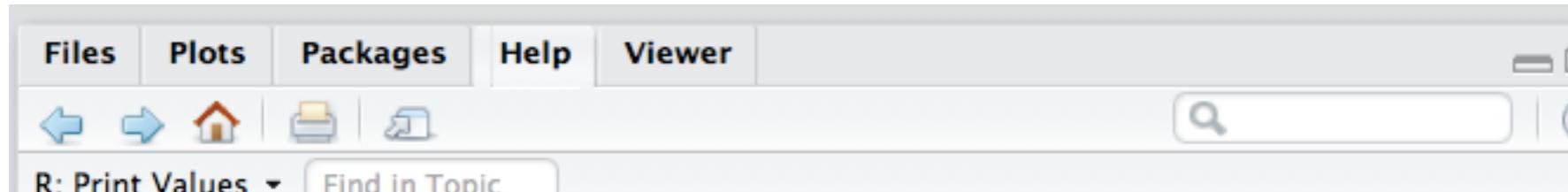
```
## S3 method for class 'function'  
print(x, useSource = TRUE, ...)
```

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

which
arguments are
obligatory

indicates there
are optional
arguments

R documentation



The screenshot shows the RStudio interface with the 'Viewer' tab selected. The title bar says 'R: Print Values'. The main content area displays the documentation for the 'print' function, specifically the base version. It includes the source code for the S3 methods for 'factor', 'table', and 'function' classes, which are highlighted with a red box.

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 [classes](#).

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...

The screenshot shows the R Help Viewer window. The title bar includes 'Files', 'Plots', 'Packages', 'Help', and 'Viewer'. Below the title bar are standard icons for back, forward, search, and help. The main content area has a header 'R: Print Values' with a 'Find in Topic' button. The text area begins with a comment line and then defines the 'print' function:

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

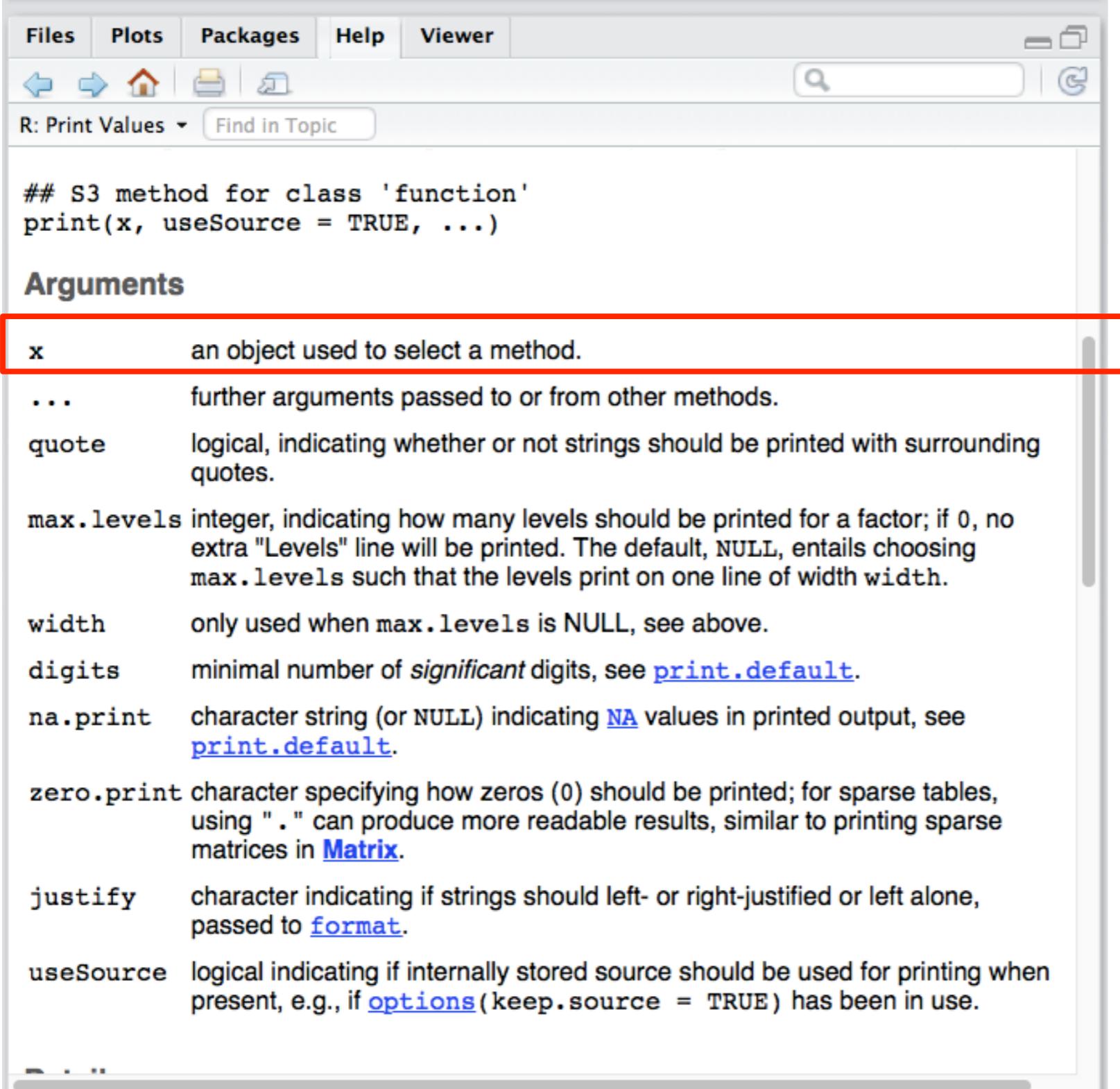
Arguments

The 'Arguments' section is highlighted with a red border. It lists the following parameters:

- 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...



The screenshot shows the R Help Viewer window. The title bar includes 'Files', 'Plots', 'Packages', 'Help', and 'Viewer'. Below the title bar are standard icons for navigating between help pages. The main content area has a search bar and a 'Find in Topic' button. The topic is 'R: Print Values'. The 'Arguments' section is highlighted with a red border. The arguments listed are:

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

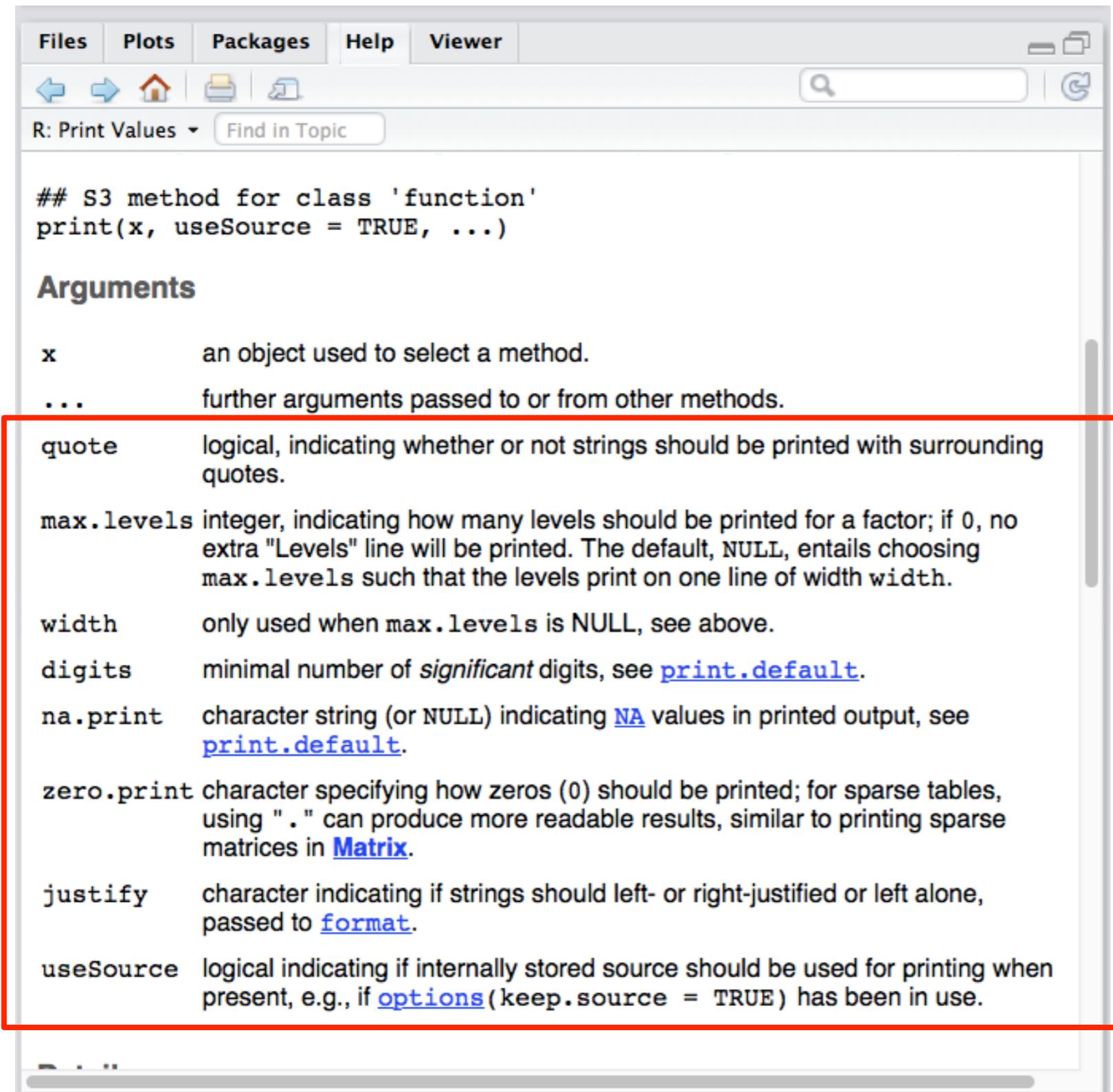
Arguments

- 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...



The screenshot shows the R Help Viewer window. The title bar says "R: Print Values". The main content area displays the documentation for the "print" function. At the top, it shows the S3 method for the "function" class:

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

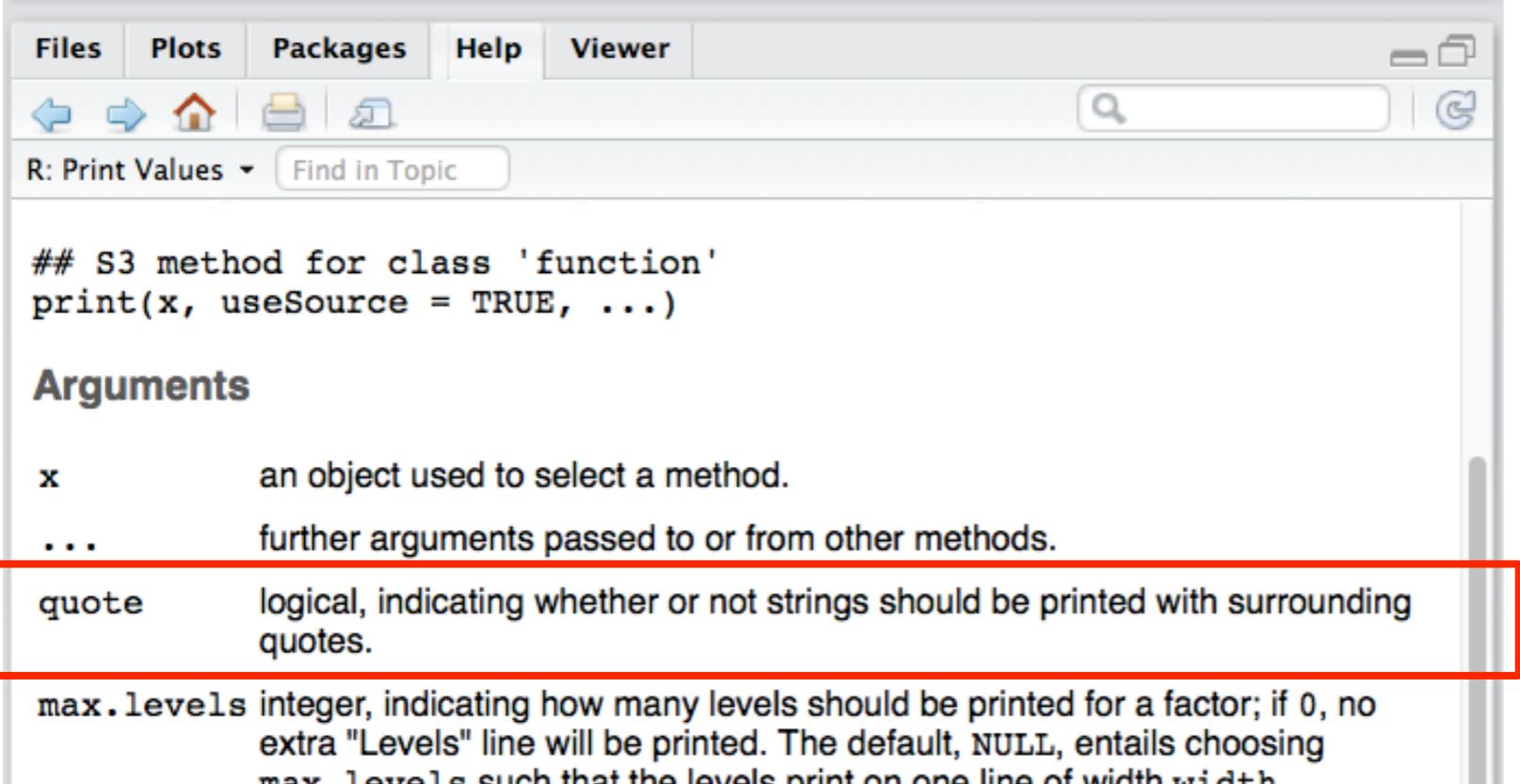
Below this is the "Arguments" section, which lists various parameters with their descriptions. A red box highlights several of these arguments:

- 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...



The screenshot shows the R Help Viewer window. The title bar includes 'Files', 'Plots', 'Packages', 'Help', and 'Viewer' tabs, with 'Viewer' selected. Below the tabs is a toolbar with icons for back, forward, home, and search. The main area displays the documentation for the 'print' function. The code section starts with a comment: '## S3 method for class 'function''. Below it is the function definition: 'print(x, useSource = TRUE, ...)'. The 'Arguments' section follows, with three entries: 'x' (an object used to select a method), '...' (further arguments passed to or from other methods), and 'quote' (logical, indicating whether or not strings should be printed with surrounding quotes). The 'quote' entry is highlighted with a red rectangular box. The 'max.levels' argument is also partially visible at the bottom.

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

Arguments

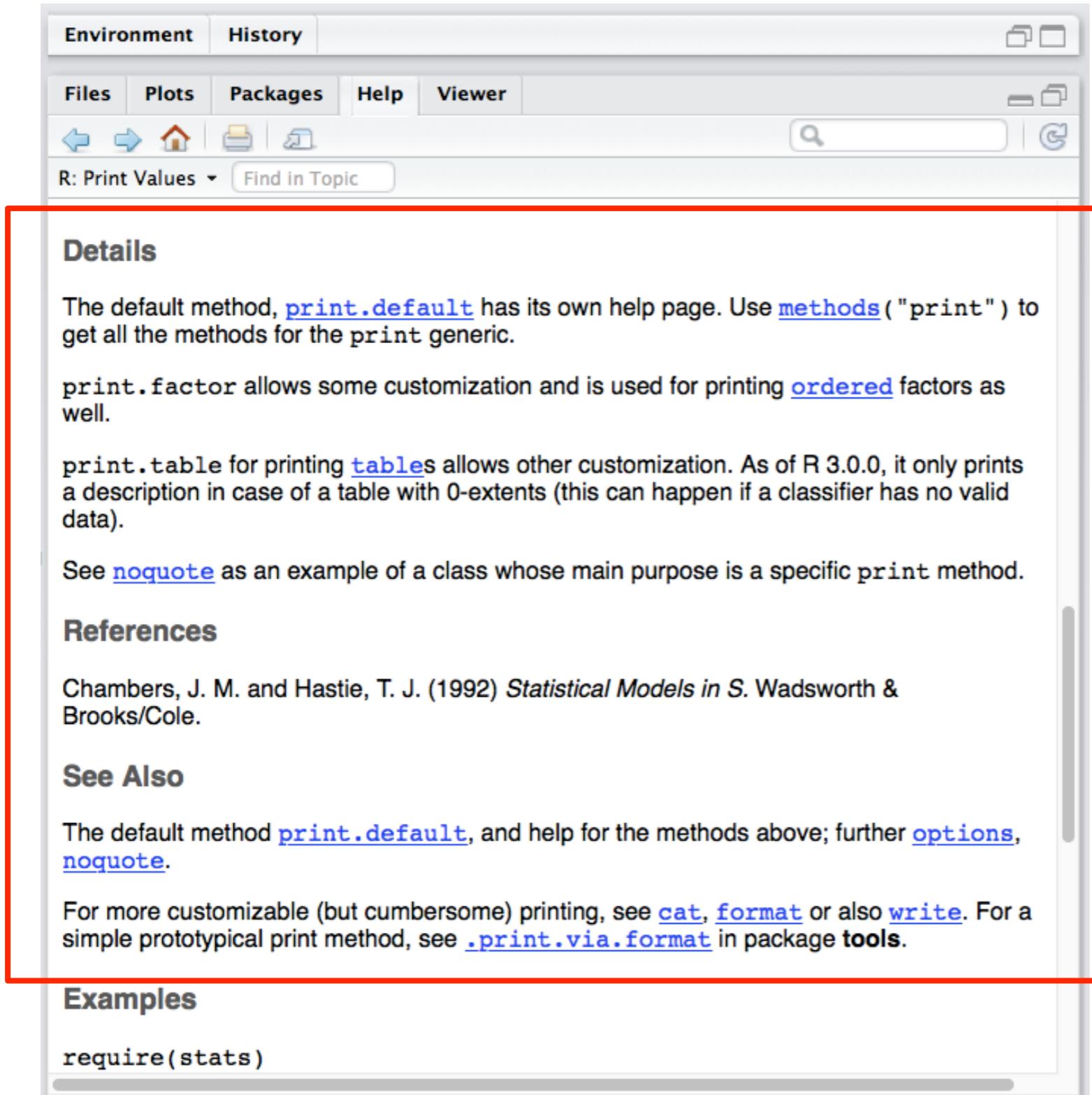
  x      an object used to select a method.
  ...
  quote   logical, indicating whether or not strings should be printed with surrounding
          quotes.

  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.
```

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

but it also never
hurts to play
around!

... scrolling even more...



The screenshot shows the R help viewer interface. The title bar says "R: Print Values". The menu bar includes "Environment", "History", "Files", "Plots", "Packages", "Help", and "Viewer". Below the menu is a toolbar with icons for back, forward, search, and help. The main content area has a red border and contains the following text:

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 [tables](#) 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](#).

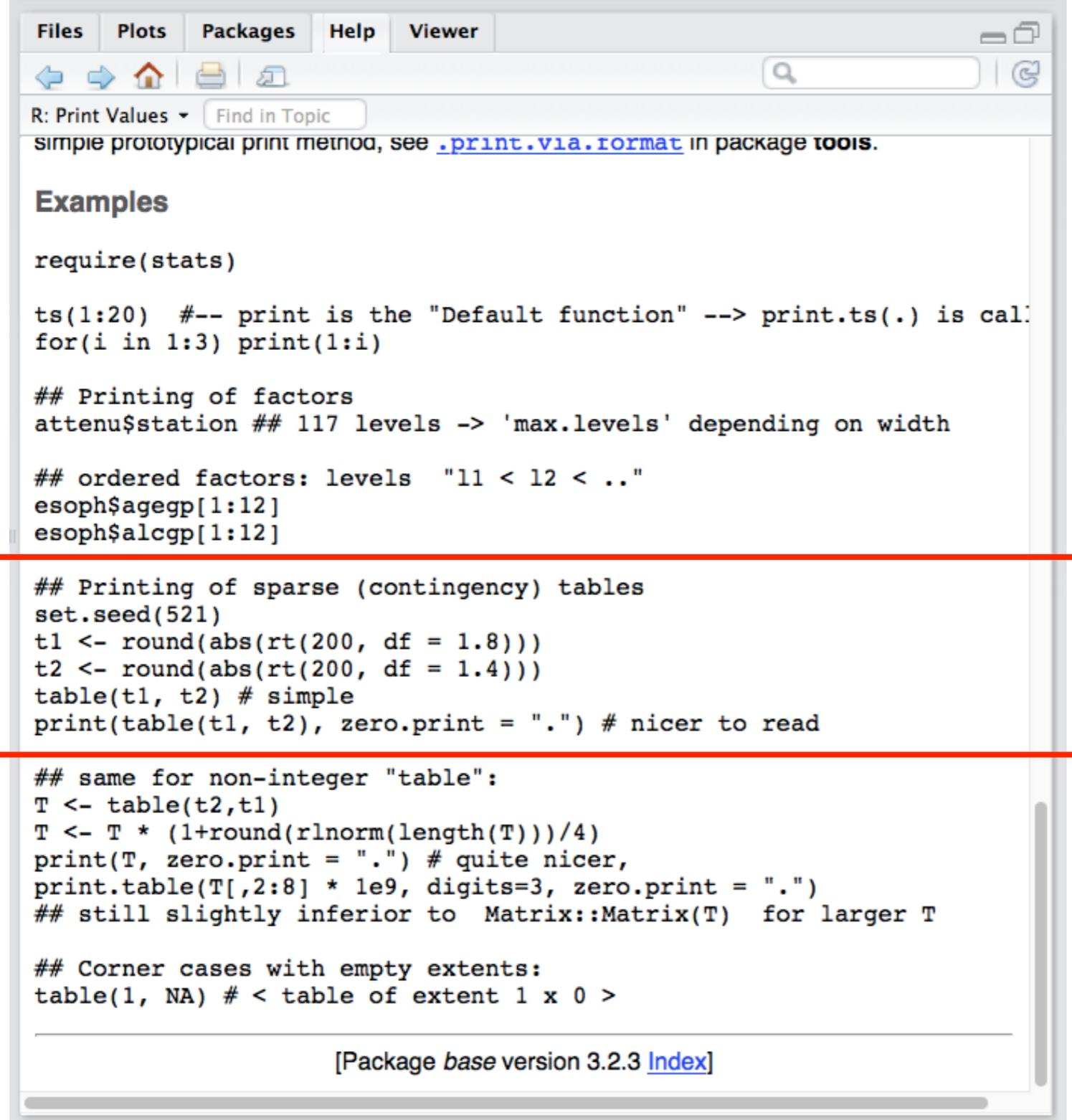
For more customizable (but cumbersome) printing, see [cat](#), [format](#) or also [write](#). For a simple prototypical print method, see [.print.via.format](#) in package [tools](#).

Examples

```
require(stats)
```

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

the end of the scrolling...



The screenshot shows the RStudio interface with the 'Viewer' tab selected. The title bar says 'R: Print Values'. The main content area displays the help page for the 'print' function, specifically the 'print.ts' method. It includes examples of printing vectors, factors, ordered factors, sparse contingency tables, non-integer tables, and corner cases like empty extent tables.

```
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 called
for(i in 1:3) print(1:i)

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

## ordered factors: levels "11 < 12 < .."
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

expt

data and data frames

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

	<code>id</code>	<code>age</code>	<code>gender</code>	<code>treatment</code>	<code>hormone</code>	<code>happy</code>	<code>sad</code>
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

```
myScriptIntroToR.R x
1 # this is my first script
2 # it's just for DRIP class
3 #
4 # author: Amy Perfors
5 #
6 # define some variables
7 age <- 19
8 box <- "cat"
9
10 # print something
11 print( box )
12 print( age )
```

commands are just like you typed them into the console

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

run by choosing "Source" (once it's saved)

save as .R file

13

`help(functionName)`
e.g. `help(print)`

