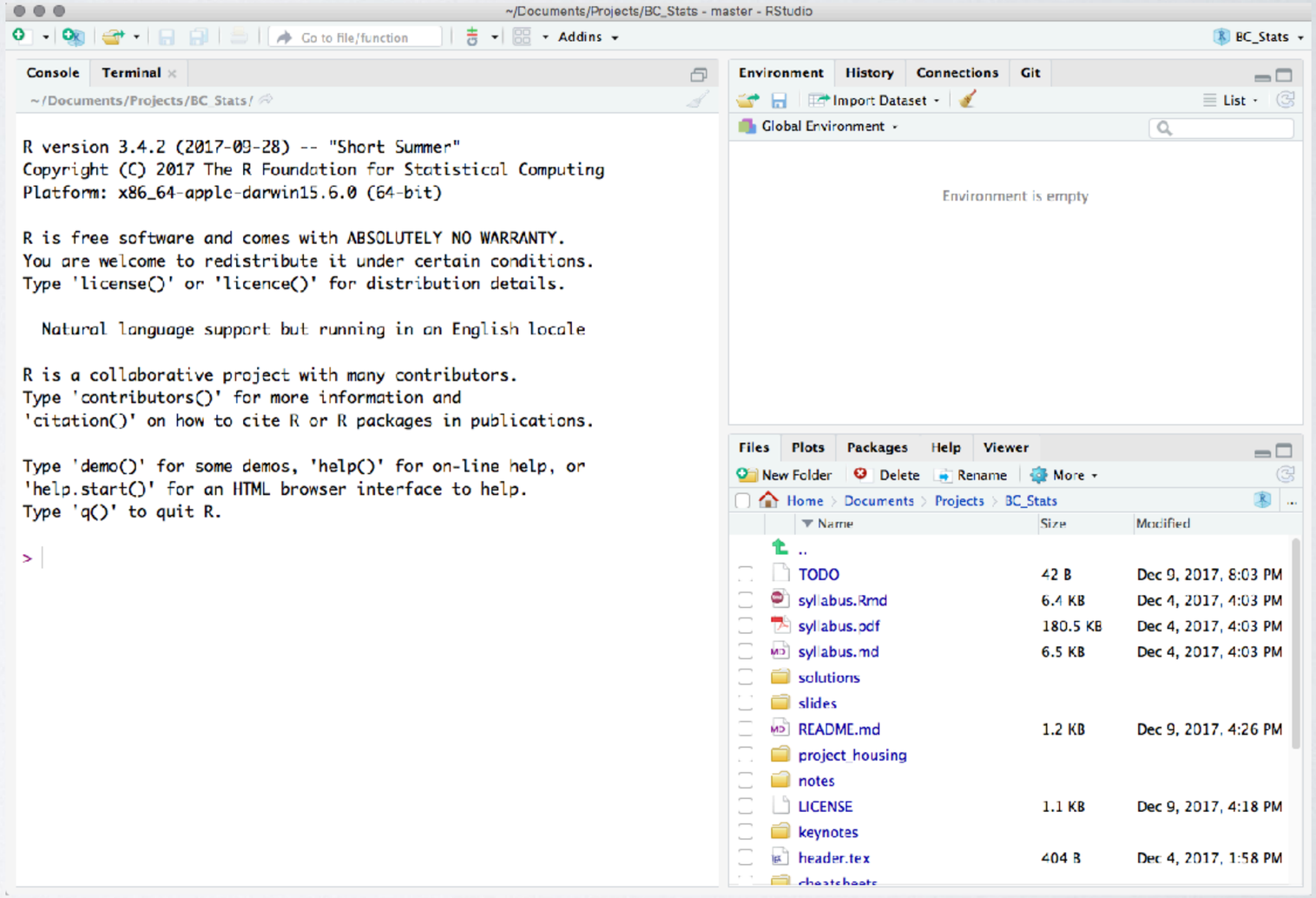
Getting Started

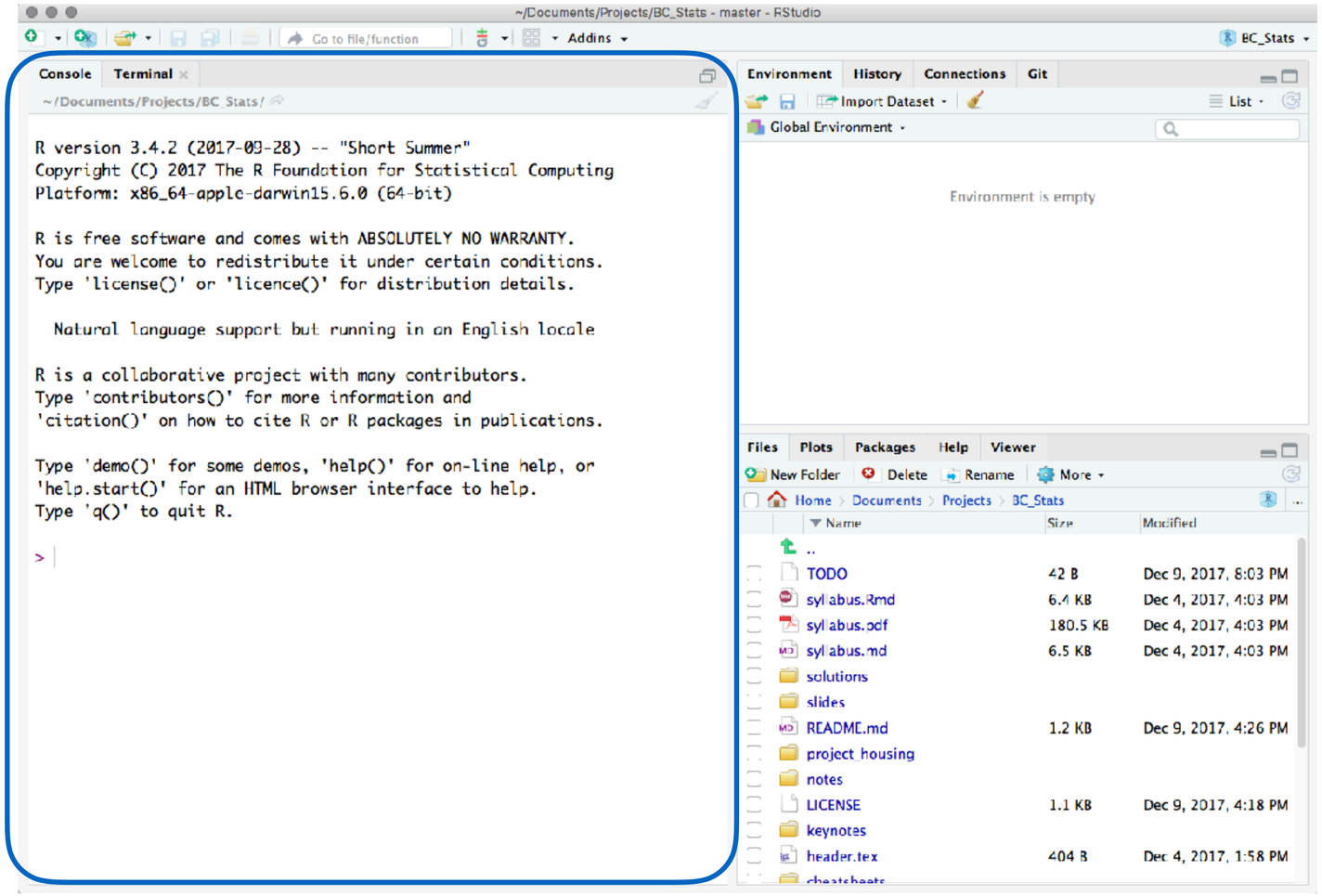
with R and RStudio

Your Turn

If you haven't already, open up the r_intro_bc_stats.Rproj file.

This should open up RStudio.





Console

R is awaiting your instructions

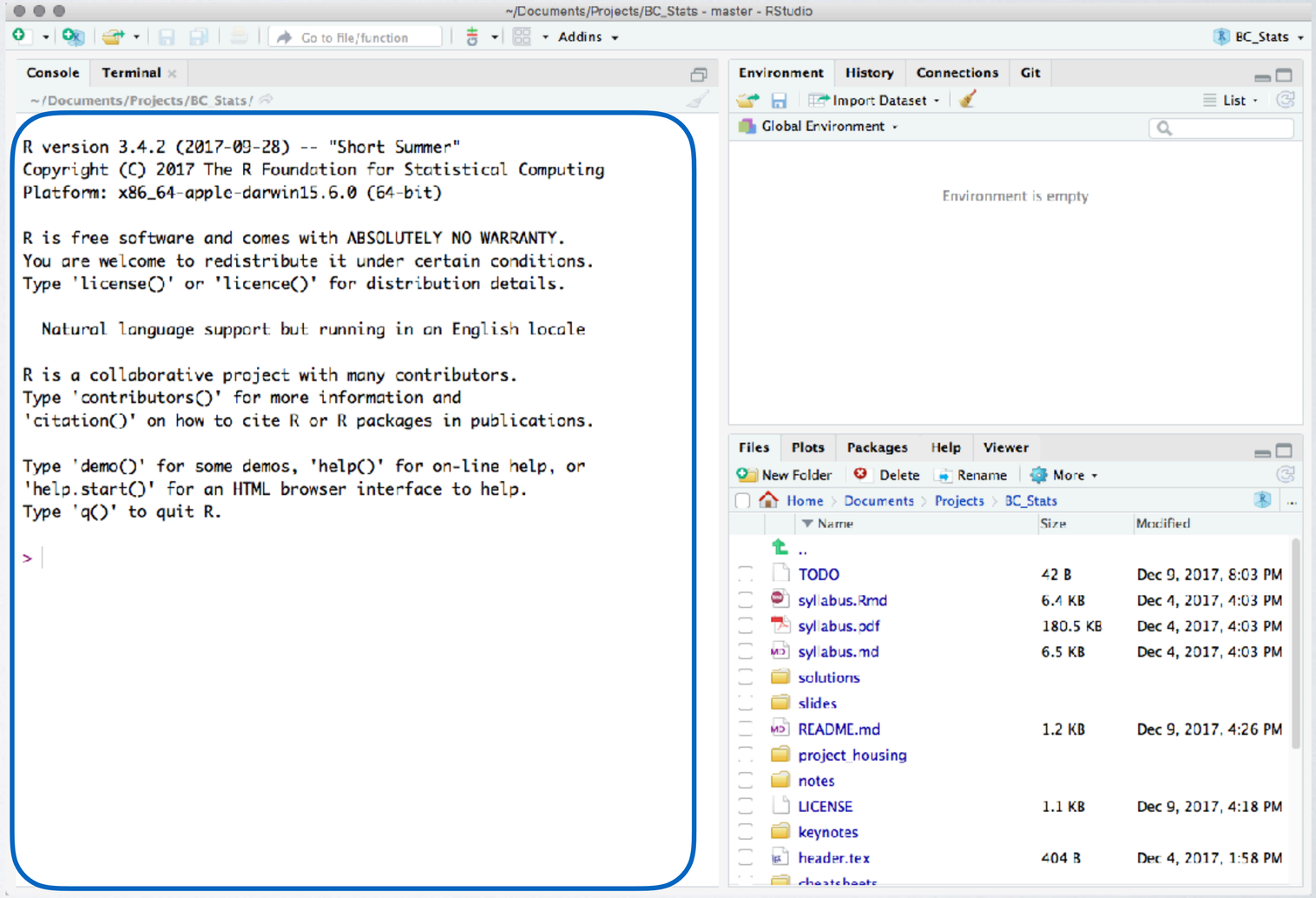
You can type code here, hit Enter, and R will evaluate it

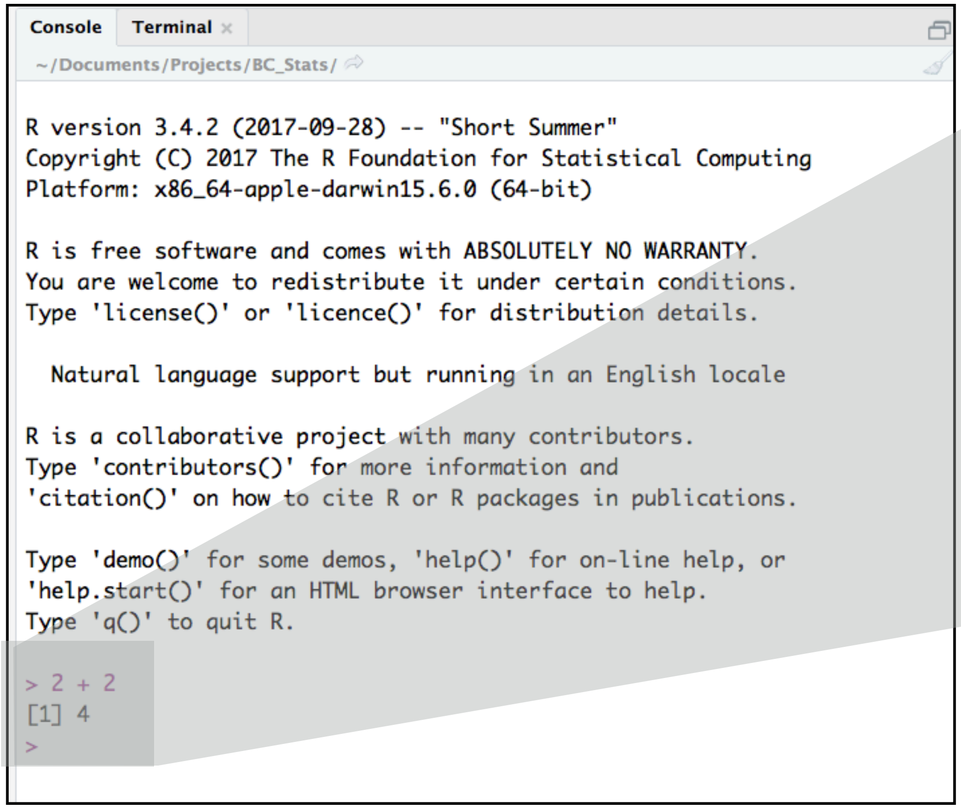
Your Turn

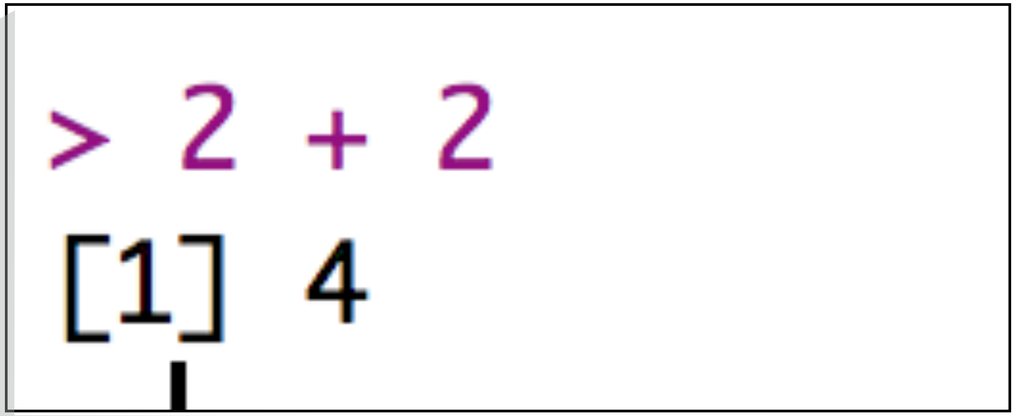
Into the console, type:

```
2 + 2
```

Then hit Enter



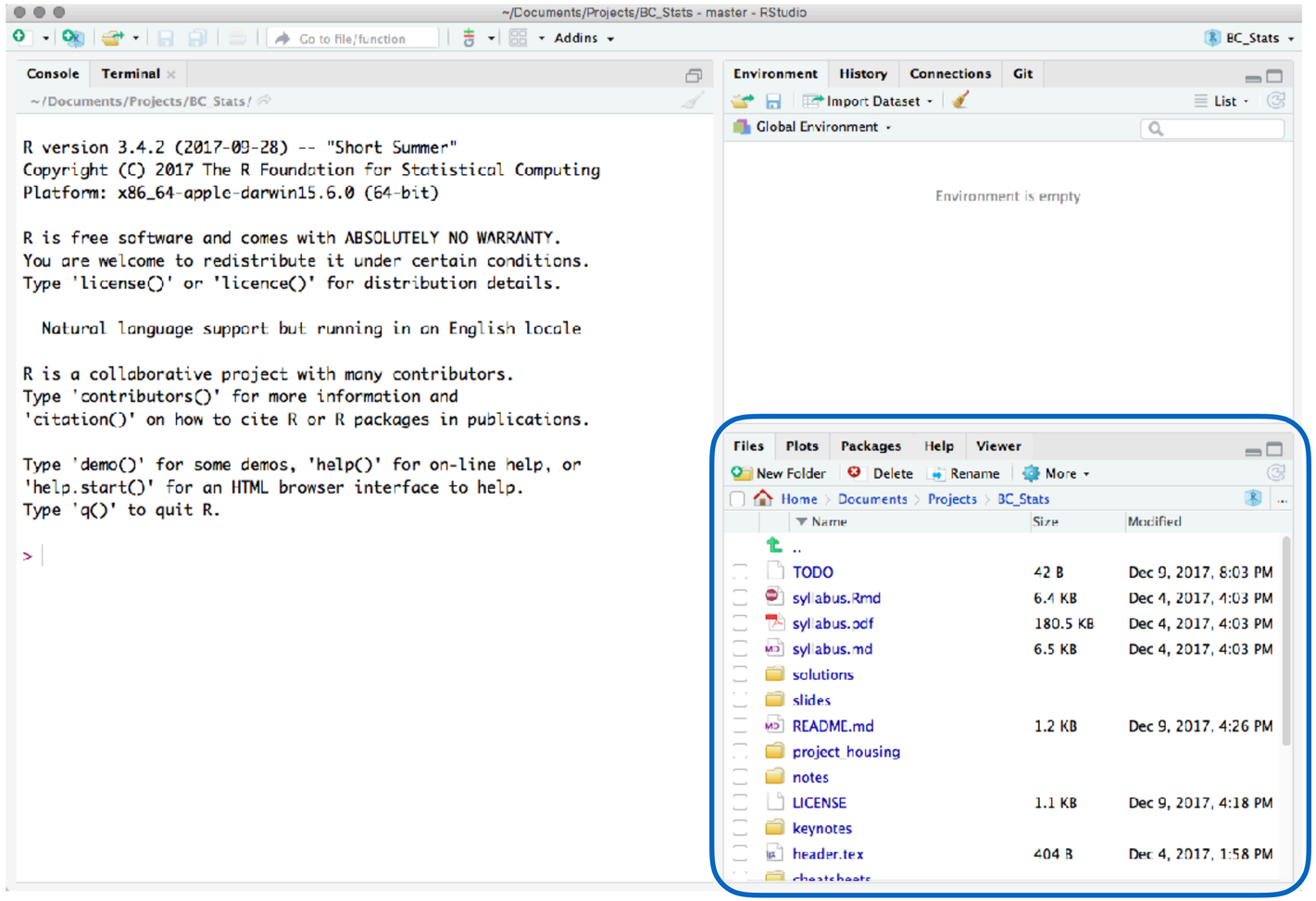




You asked, and R answered,

but this is ephemeral.

In practice you'll keep your R code in a document.



Files Pane

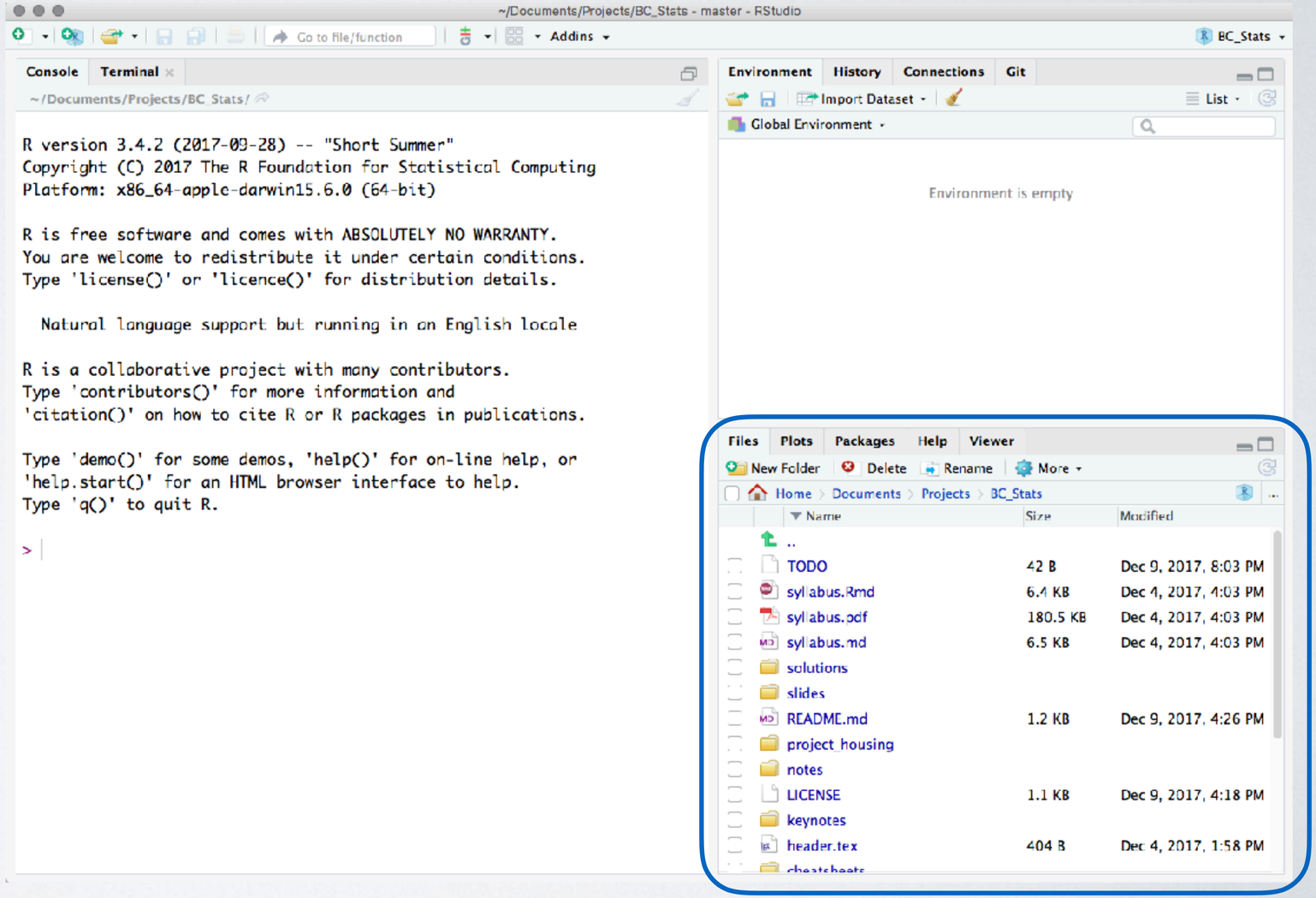
Files in your current working directory.

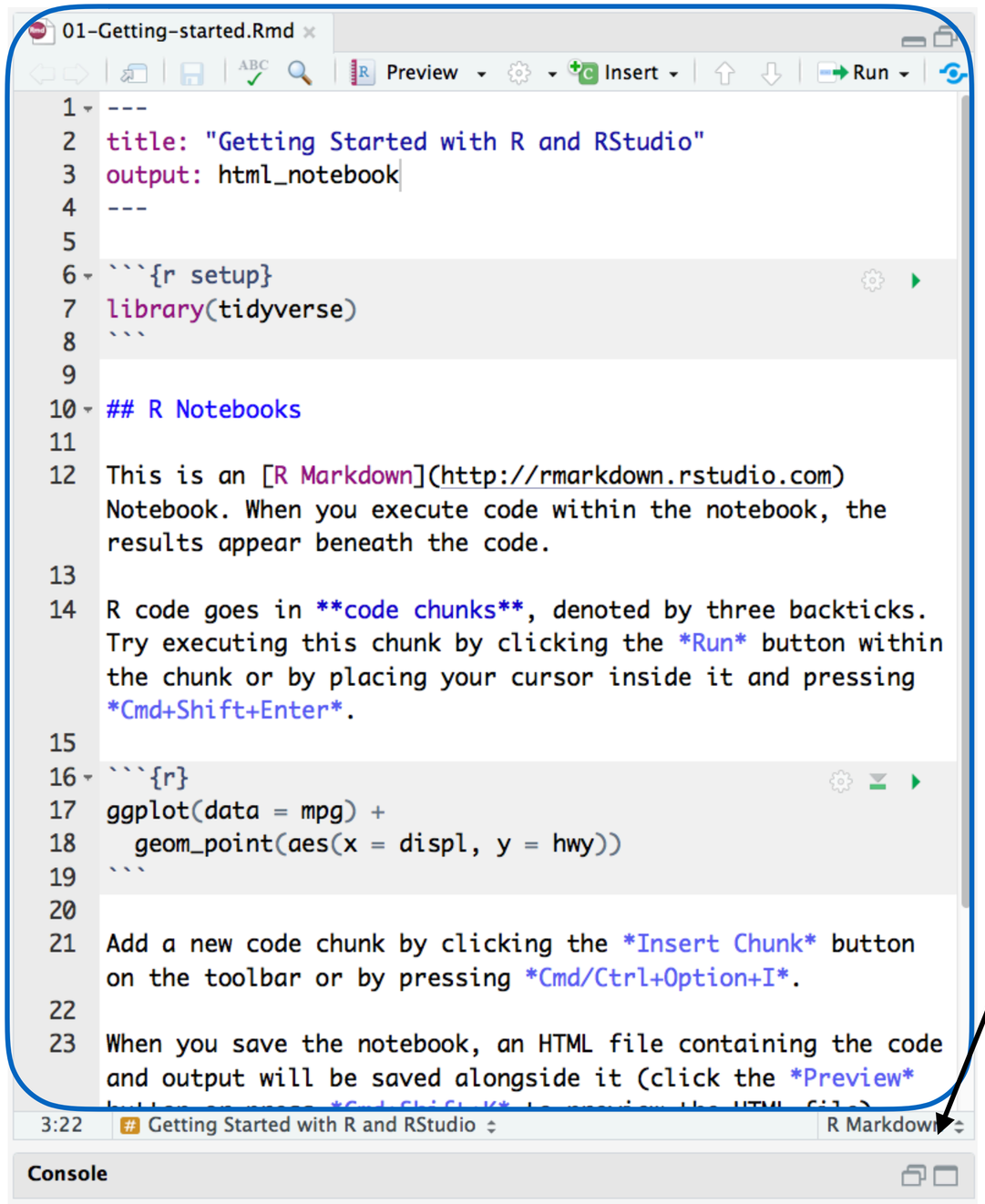
Your Turn

In the Files Pane,

Find 01-Getting-started.Rmd

Click on it's name to open the file.

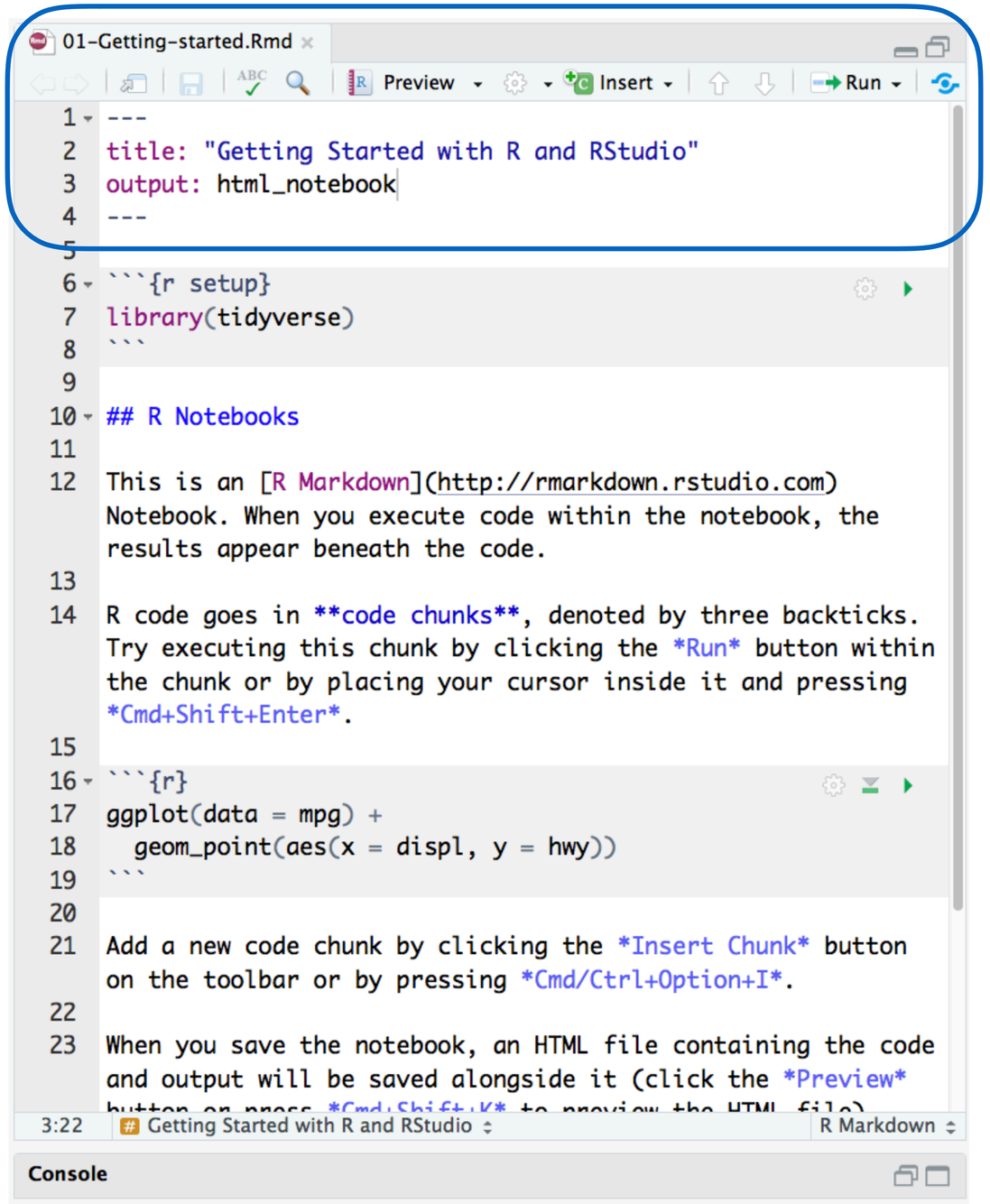




Source Pane

Documents open up here.

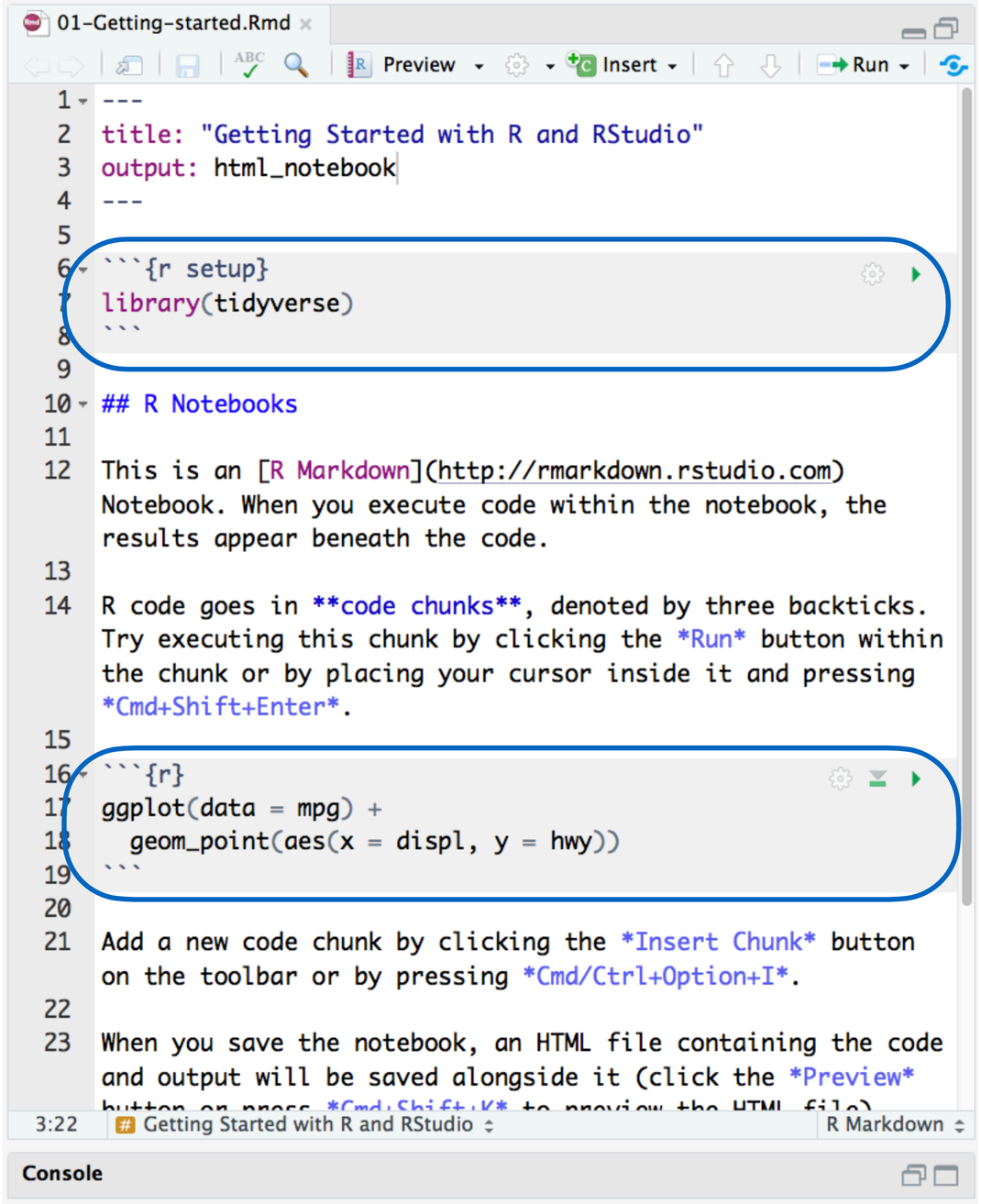
The console is still here but it's been minimized



R notebooks

An authoring format for Data Science

01-Getting-started.Rmd is an R notebook



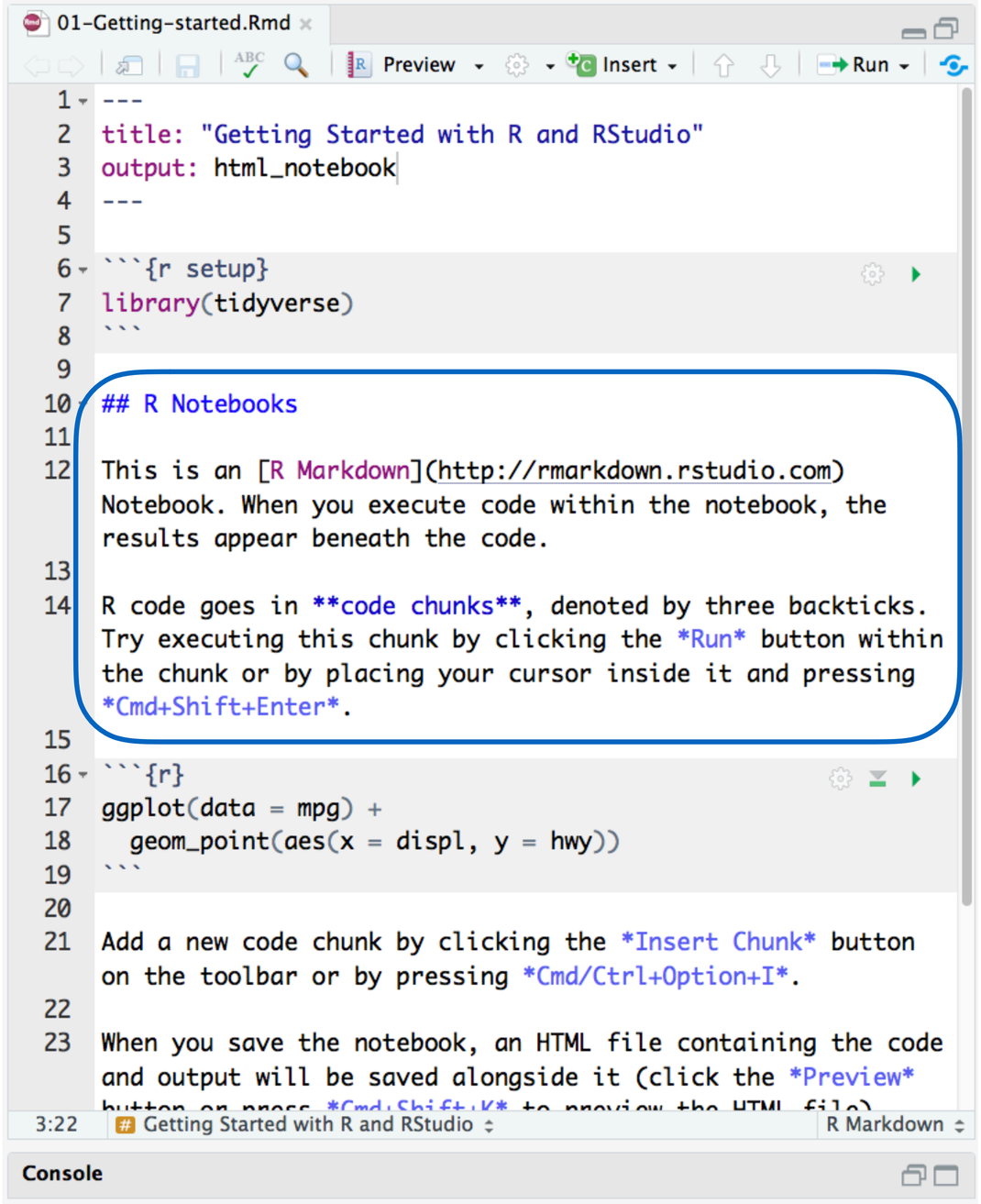
R notebooks

An authoring format for Data Science

01-Getting-started.Rmd is an R notebook

Integrates:

- Code



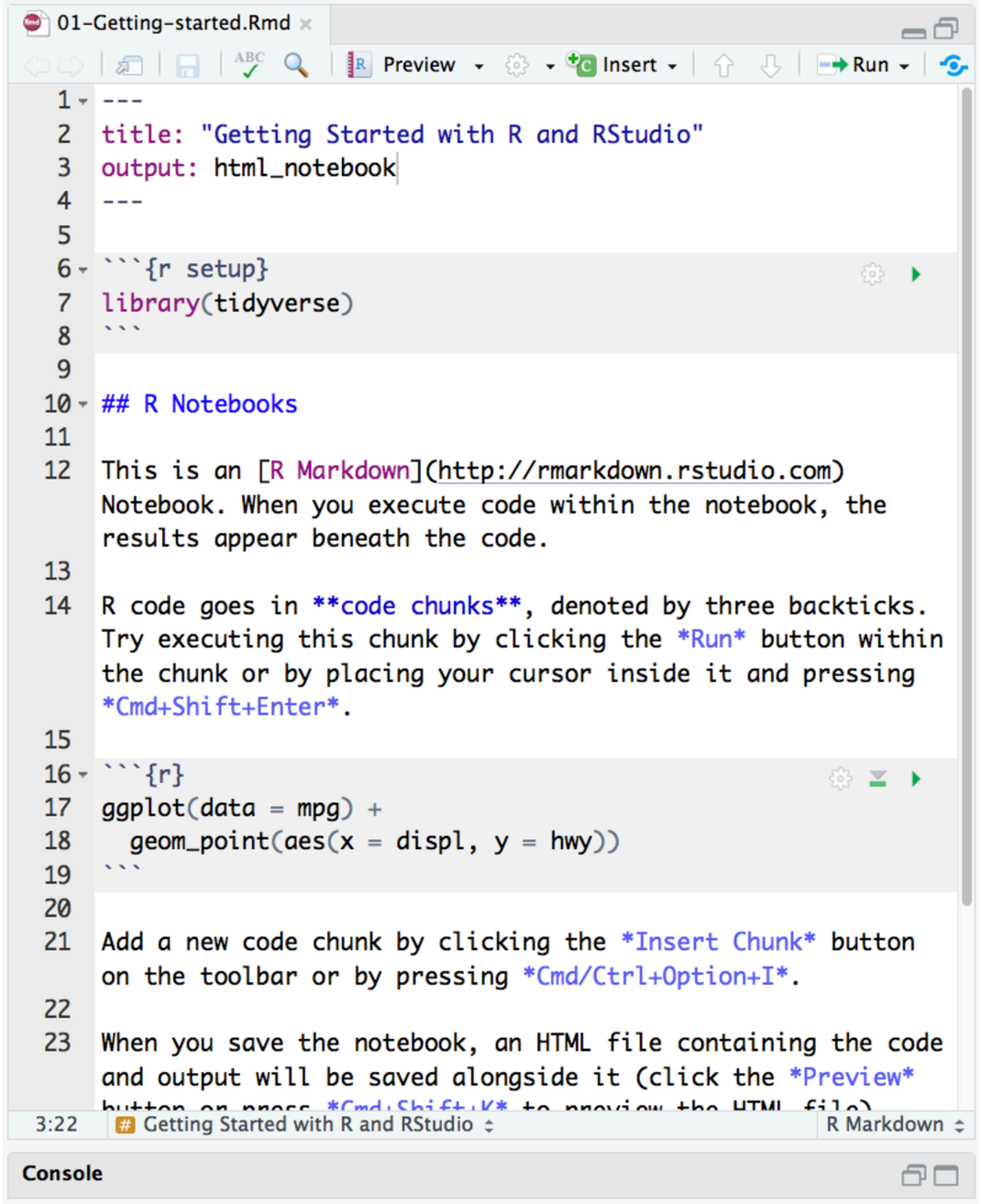
R notebooks

An authoring format for Data Science

01-Getting-started.Rmd is an R notebook

Integrates:

- Code
- Text



R notebooks

An authoring format for Data Science

01-Getting-started.Rmd is an R notebook

Integrates:

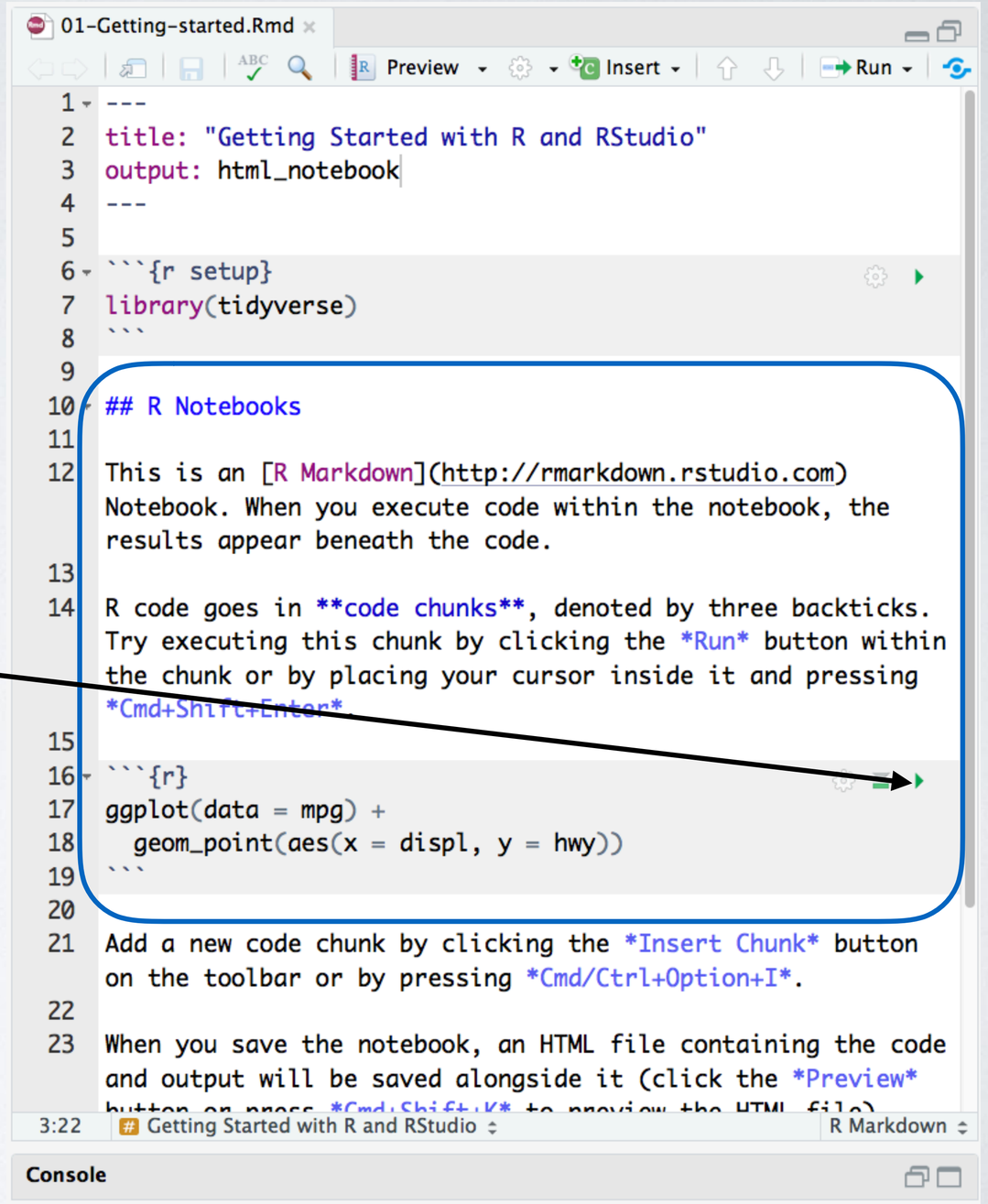
- Code
- Text
- Output

Your Turn

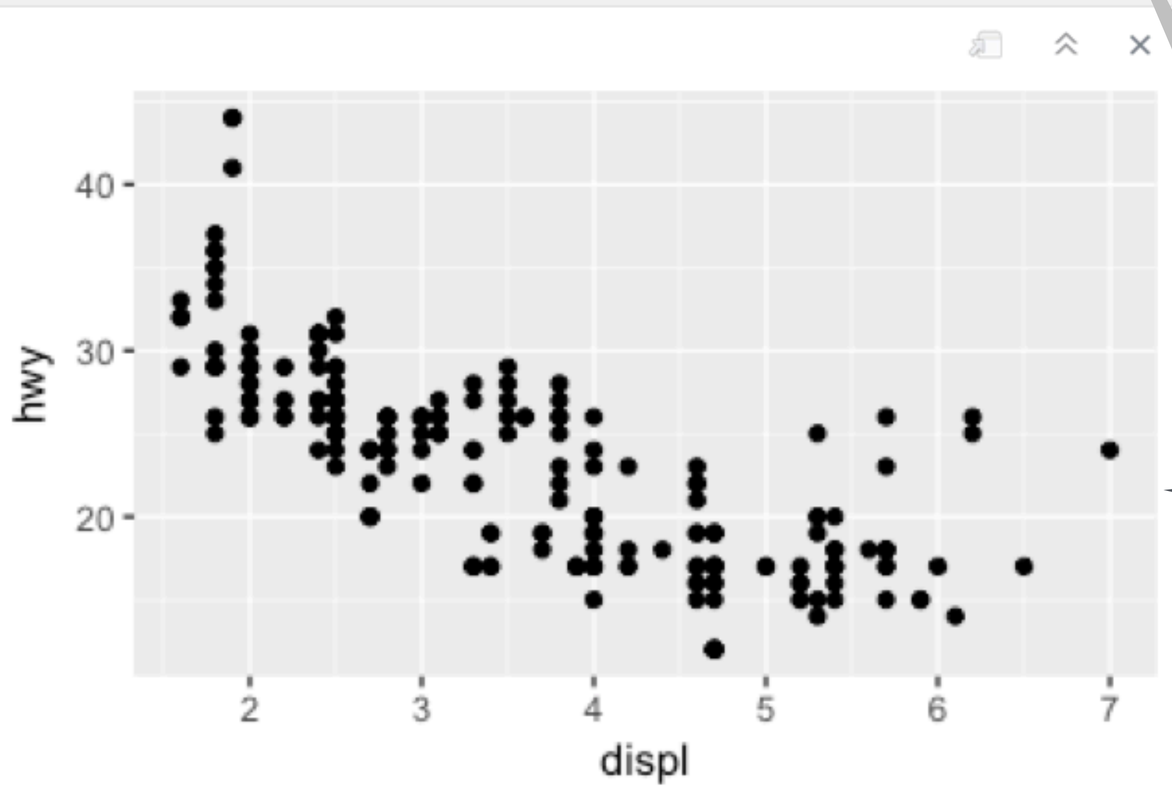
Read the instructions.

Run the code by hitting the play button,

or using the keyboard shortcut.



````
```{r}
ggplot(data = mpg) +
  geom_point(aes(x = displ, y = hwy))
```
````



Click to run code in chunk

Click to run all code chunks above

Code result

Your Turn

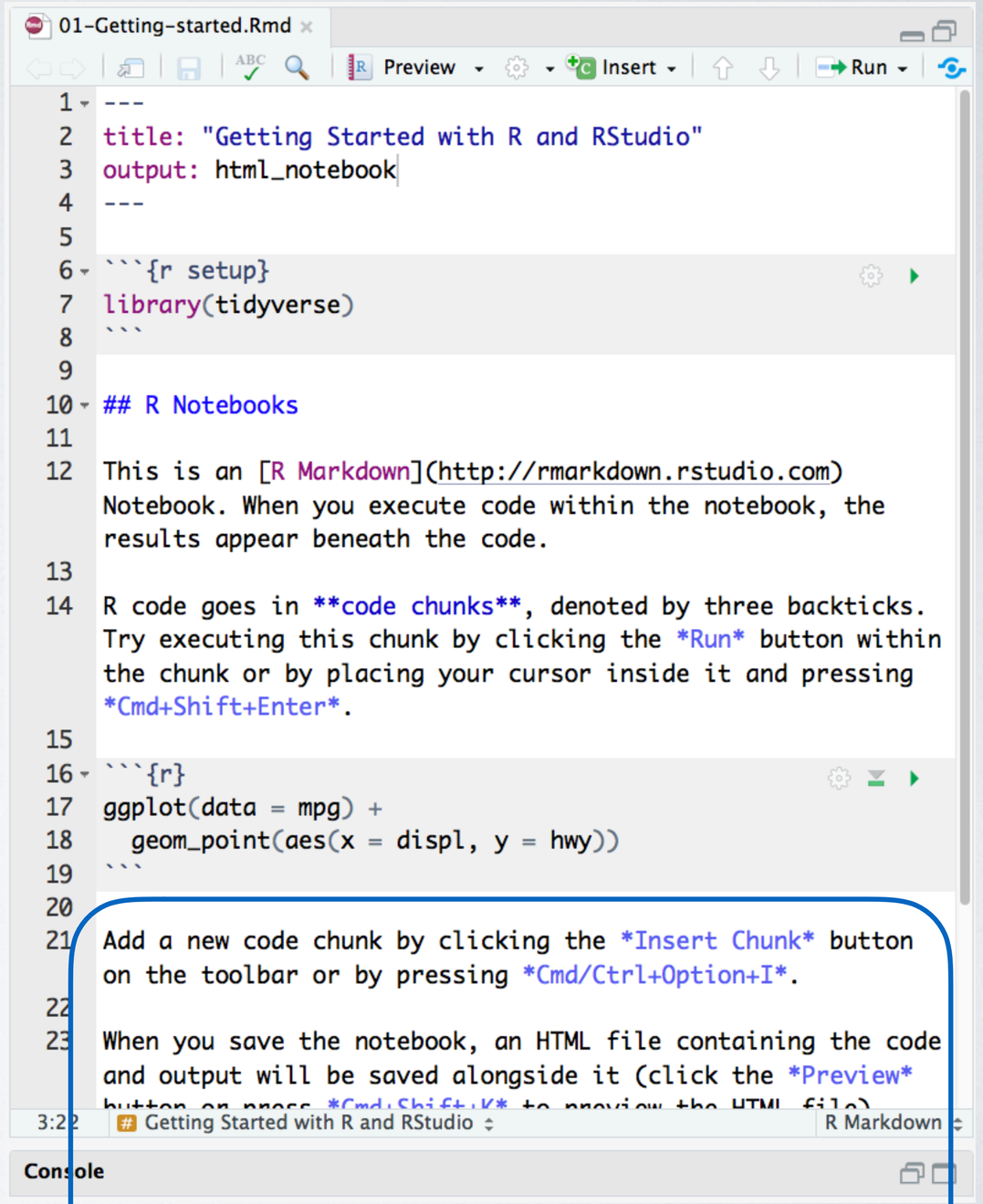
Keep going!

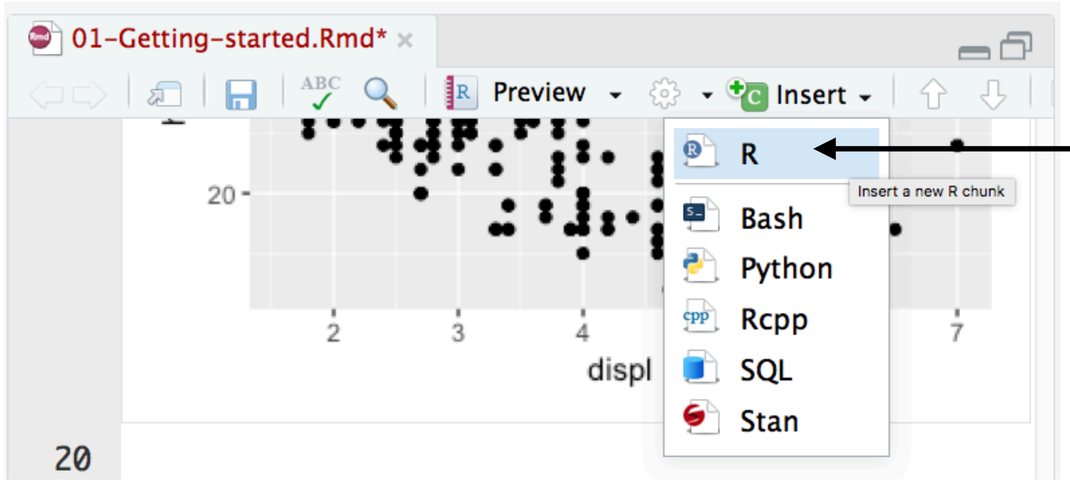
Add a new code chunk.

Put:

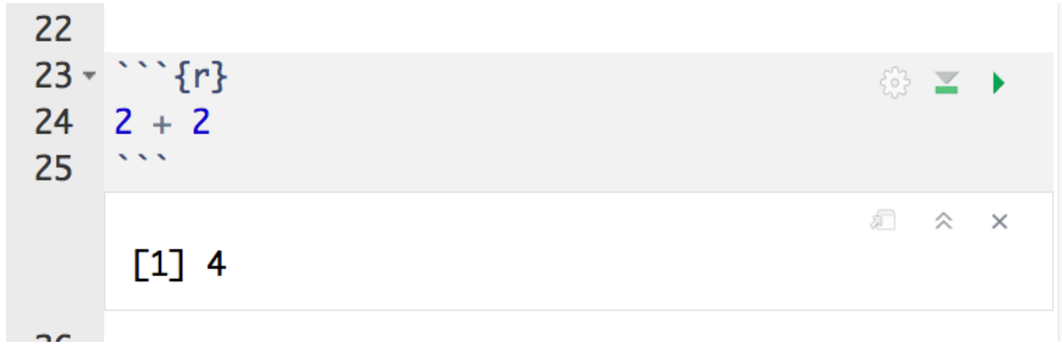
```
2 + 2
```

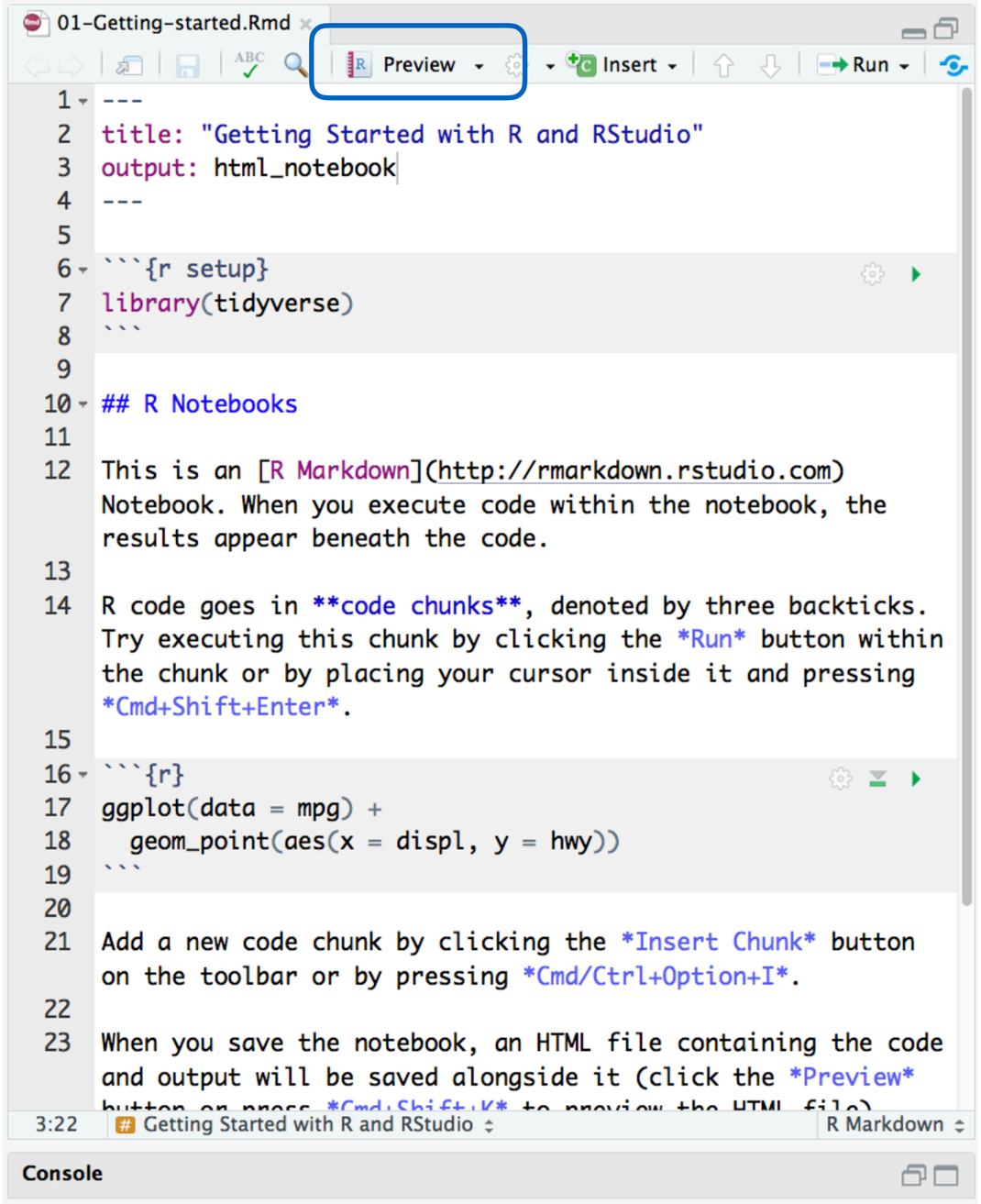
in your new chunk and run it.





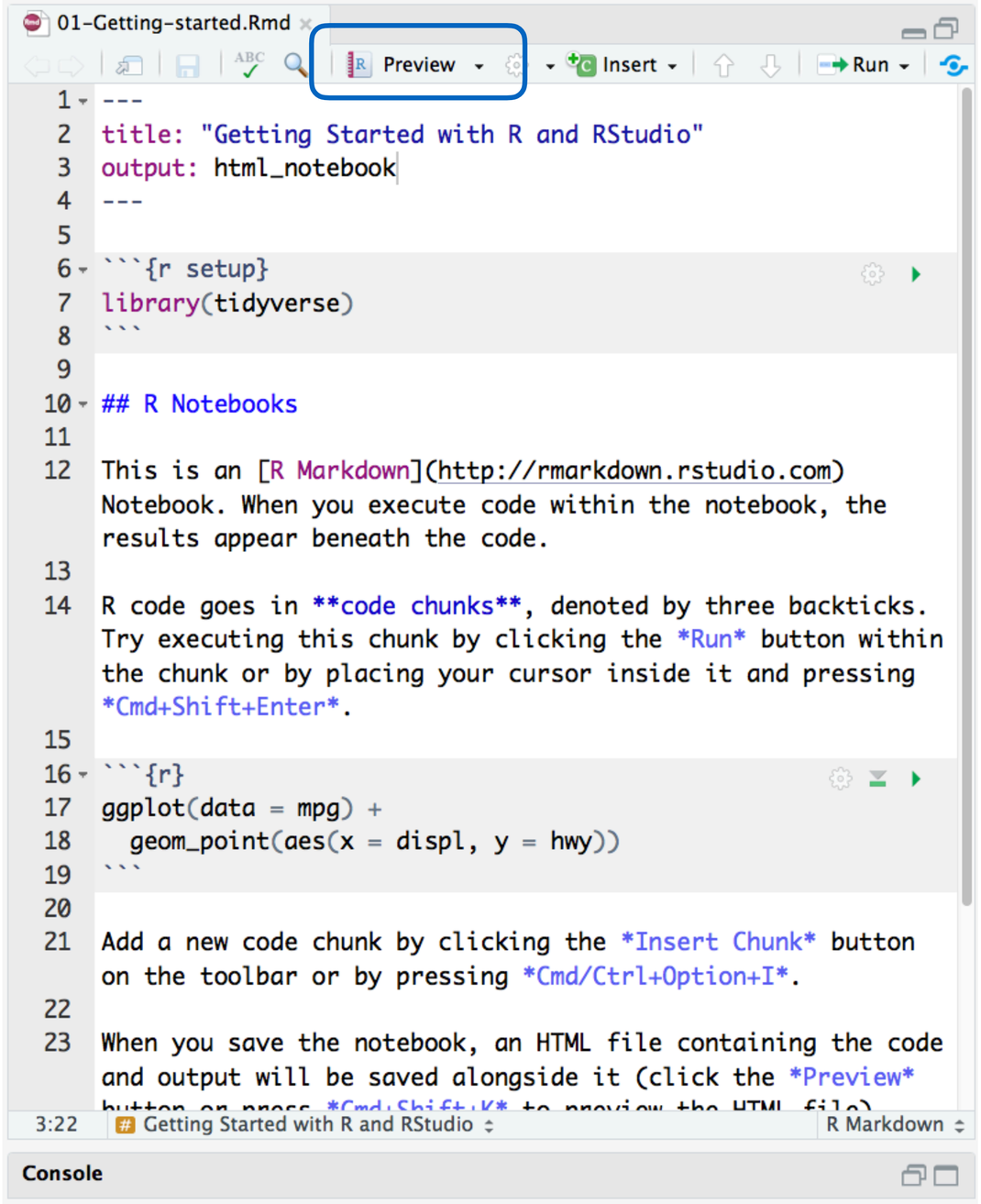
Insert new code chunk





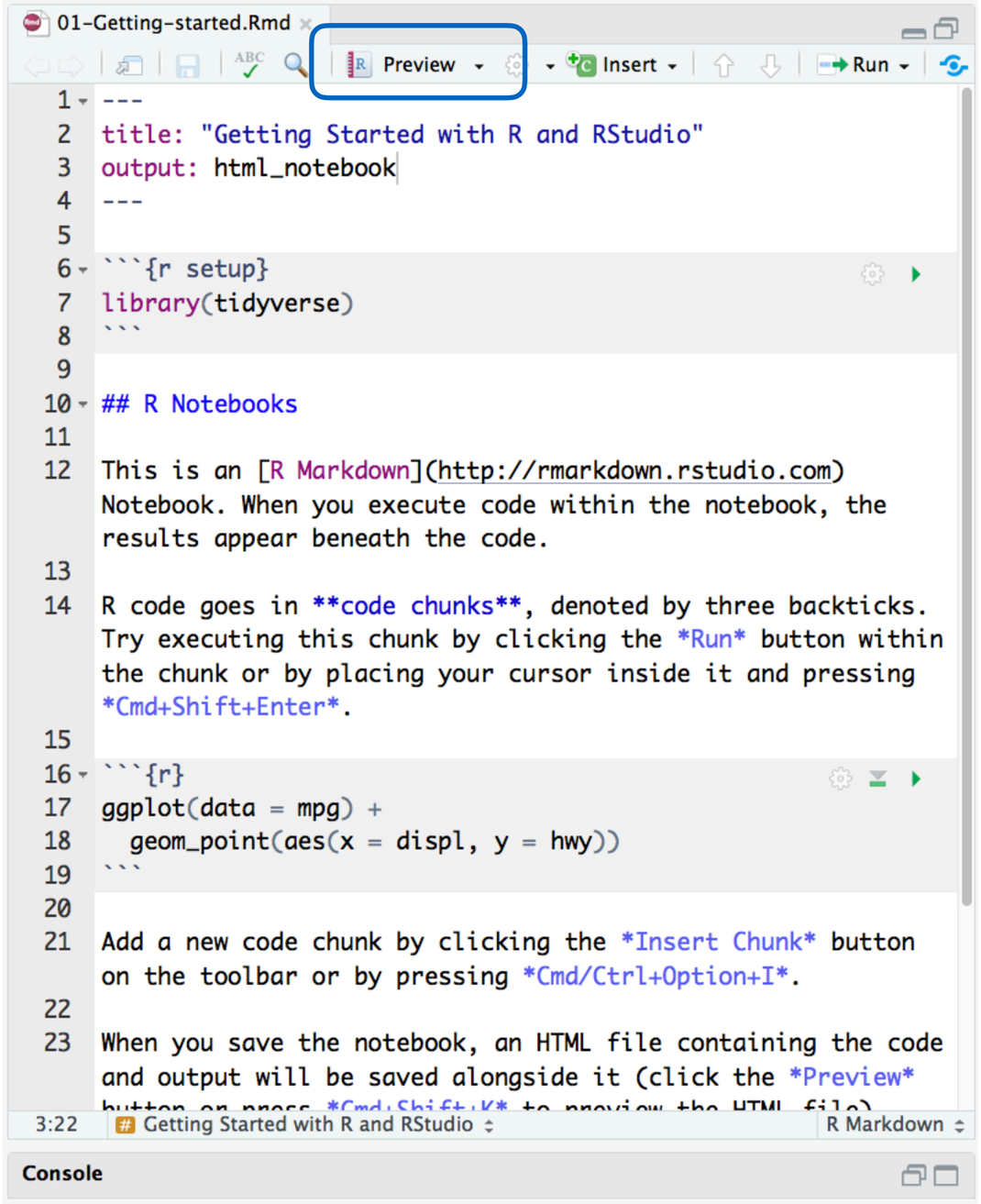
R notebooks

Keep a record of things you have run, you can see an HTML version, by hitting preview.



R notebooks

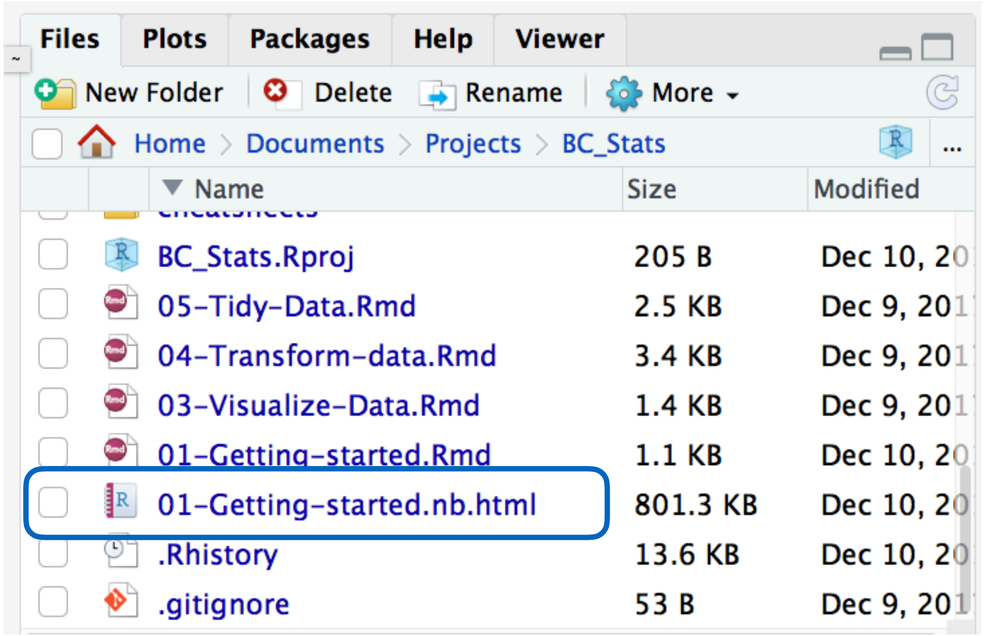
Keep a record of things you have run, you can see an HTML version, by hitting preview.



R notebooks

The result is stand-alone,

you can email it to your friends.



R Notebooks

An easy way to combine R code and narrative

Useful for us:

- I'll provide starter code
- You can complete "Your Turns"
- At the end, a useful record for you

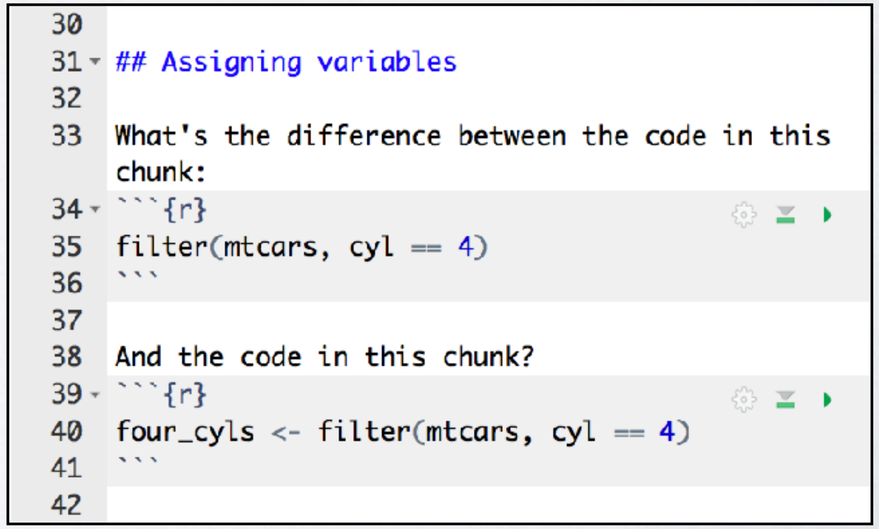
Your Turn

1. Spot the difference

```
filter(mtcars, cyl == 4)
```

```
four_cyls <- filter(mtcars, cyl == 4)
```

2. Find the chunks in the notebook and run them, what's different about what happens?



```
filter(mtcars, cyl == 4)
```

Does something (?) and output is displayed

```
four_cyls <- filter(mtcars, cyl == 4)
```

Does the exact same thing but output is stored in variable `four_cyls`

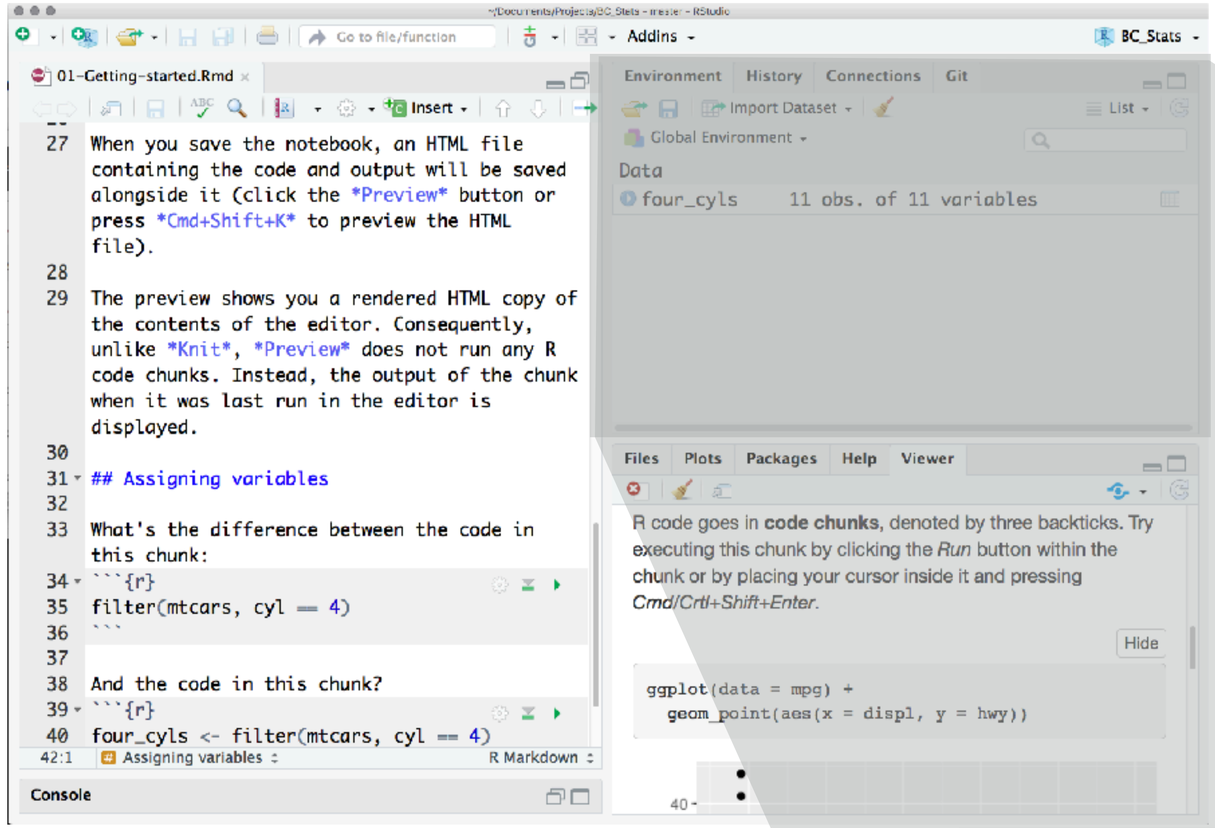
Assignment

```
four_cyls <- filter(mtcars, cyl == 4)
```

**Assign the output from the right hand side,
to a variable with the name on the left hand side**

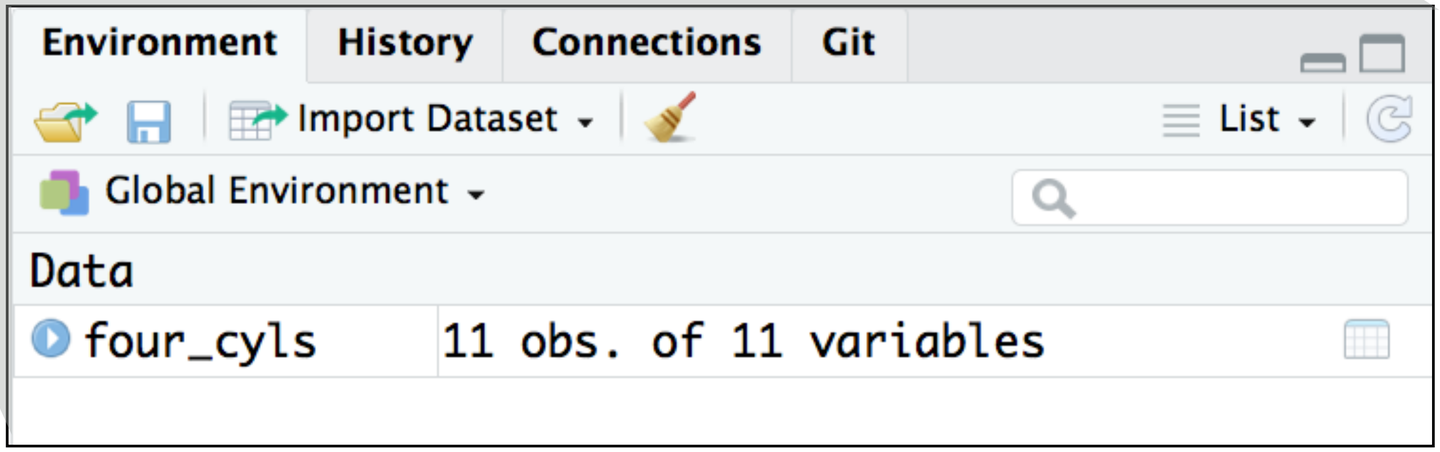
`<-`

The assignment operator, 'less than symbol' 'dash'



Environment

All the variables you have stored

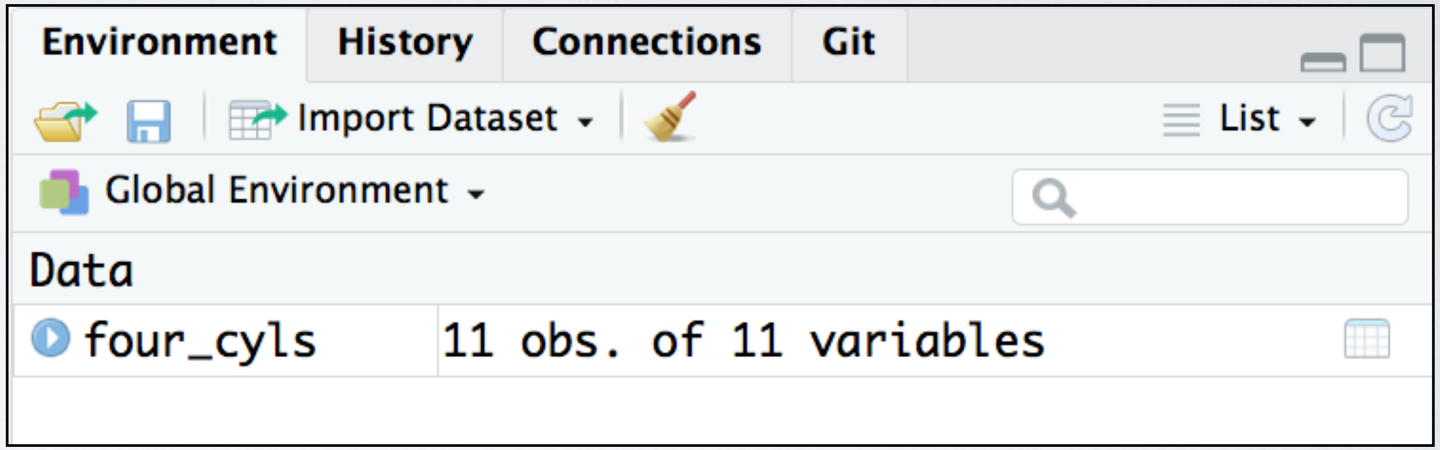


Your Turn

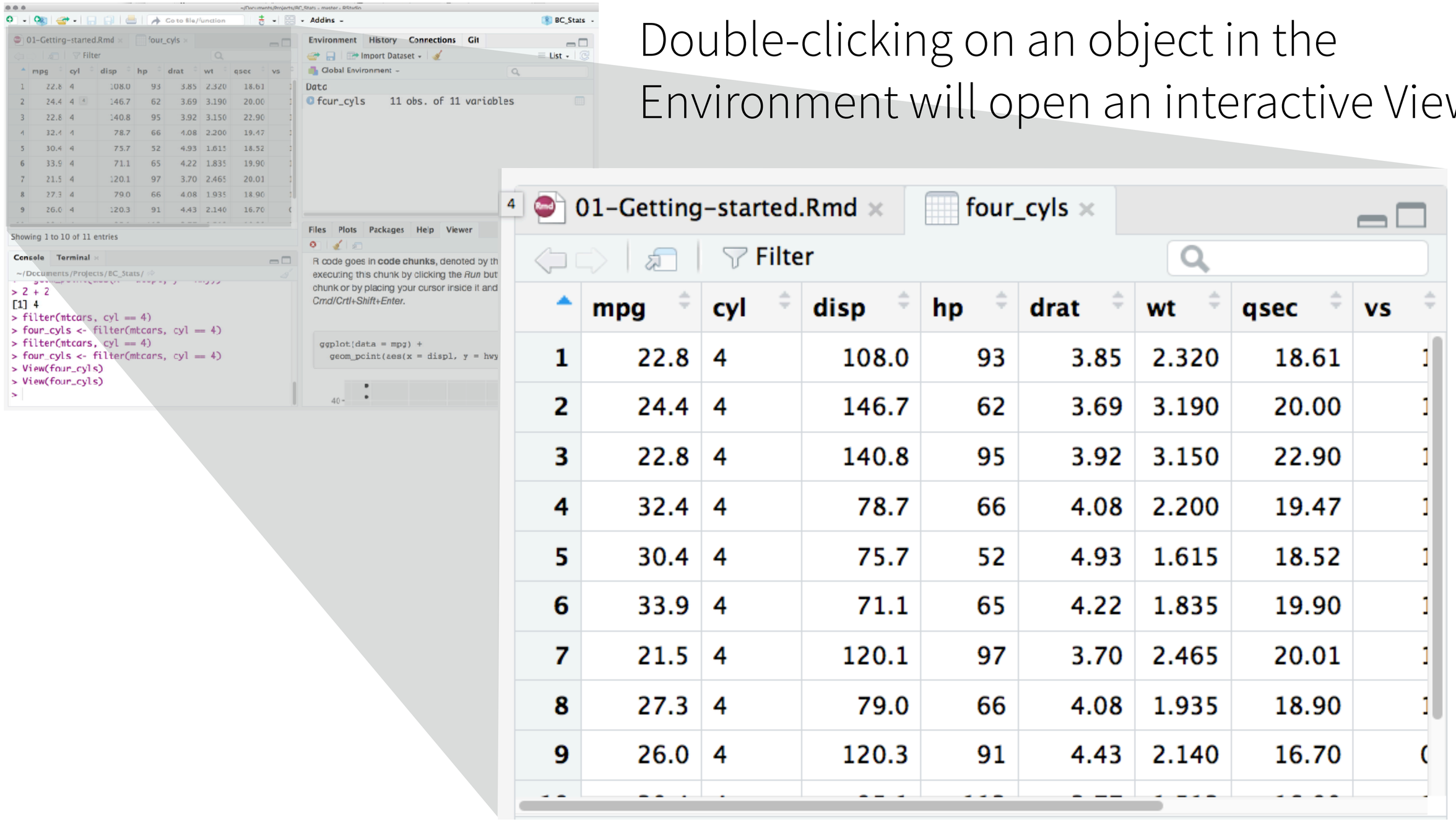
Find `four_cyls` in the environment pane.

Double-click on the name `four_cyls`.

What happens? What is this object?



Double-clicking on an object in the Environment will open an interactive Viewer



Basic Syntax

```
four_cyls <- filter(mtcars, cyl == 4)
```

Functions **do** things in R

Functions take arguments, and output results

If you want to keep the output, you assign it to a variable

Basic Syntax

```
four_cyls <- filter(mtcars, cyl == 4)
```

To *call* a function, you need to know it's name,

and the arguments it is expecting.

Arguments are comma separated and go inside the parenthesis following the function name.

?

for help on functions

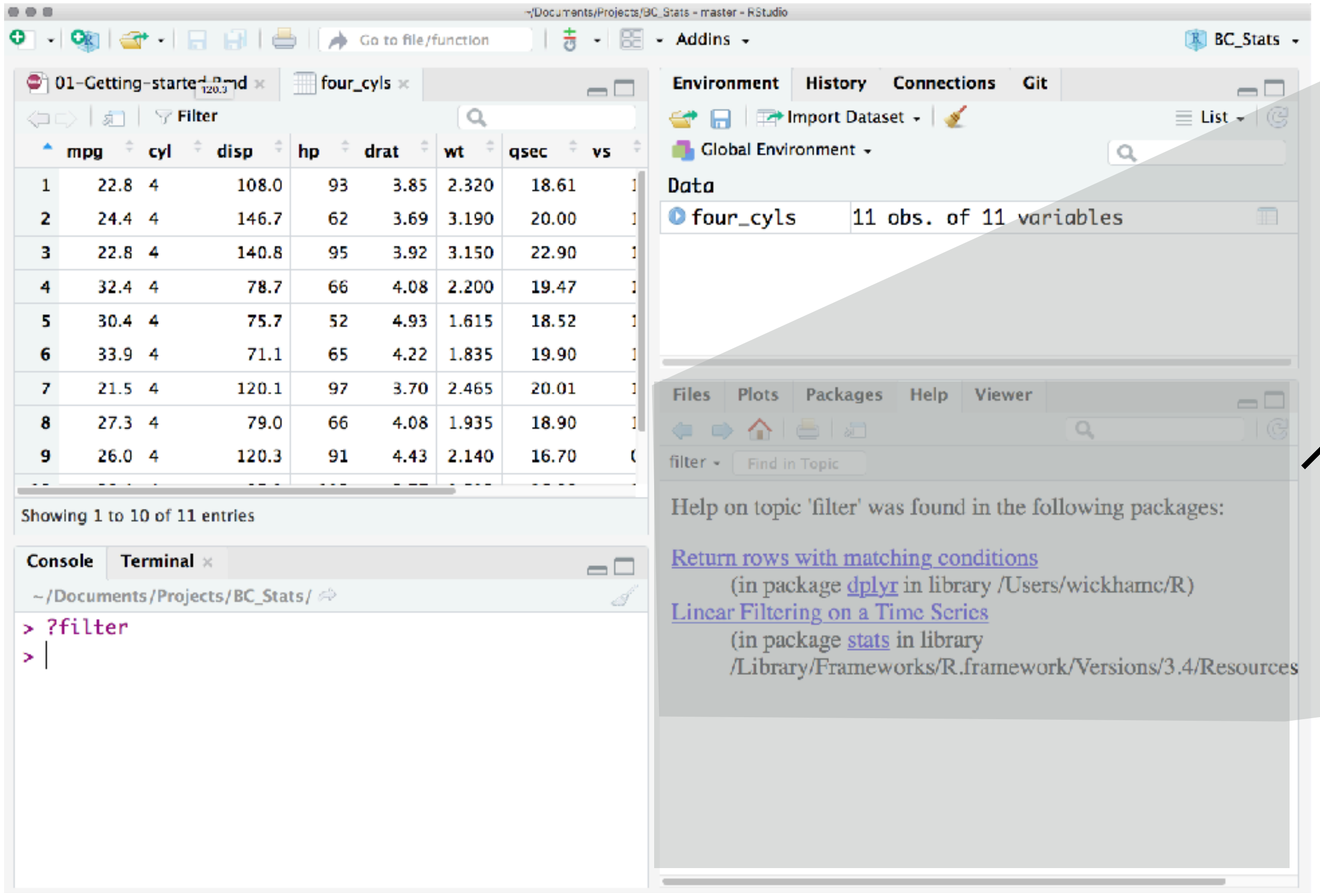
To get help on a function in R, on the **Console**, type

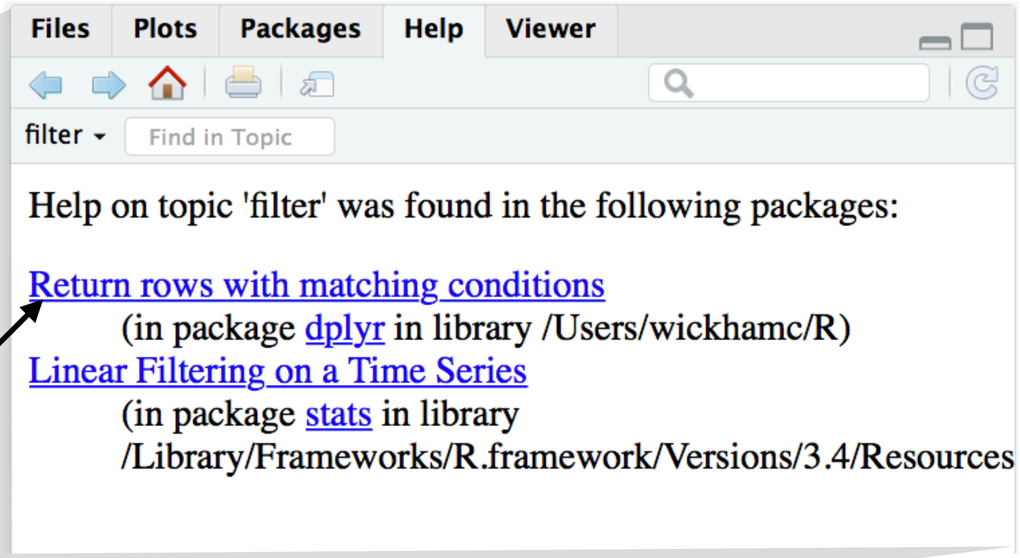
```
?function_name
```

Your Turn

Try

```
?filter
```





filter {dplyr} R Documentation

Return rows with matching conditions

Description

Use `filter()` find rows/cases where conditions are true. Unlike base subsetting, rows where the condition evaluates to `NA` are dropped.

Usage

```
filter(.data, ...)
```

Arguments

filter {dplyr} R Documentation

Return rows with matching conditions

Description

Use `filter()` find rows/cases where conditions are true. Unlike base subsetting, rows where the condition evaluates to `NA` are dropped.

Usage

```
filter(.data, ...)
```

Arguments

Arguments

A named argument `.data`

`...` Some other unnamed arguments

filter {dplyr} R Documentation

Return rows with matching conditions

Description

Use `filter()` find rows/cases where conditions are true. Unlike base subsetting, rows where the condition evaluates to `NA` are dropped.

Usage

```
filter(.data, ...)
```

Arguments

`.data`	A tbl. All main verbs are S3 generics and provide methods for `tbl_df()`, `dtplyr::tbl_dt()` and `dbplyr::tbl_dbi()`.
`...`	Logical predicates defined in terms of the variables in `.data`. Multiple conditions are combined with `&`. Only rows where the condition evaluates to `TRUE` are kept. These arguments are automatically quoted and evaluated in the context of the data frame. They support unquoting and splicing. See `vignette("programming")` for an introduction to these concepts.

Details

Note that dplyr is not yet smart enough to optimise filtering optimisation on grouped datasets that don't need grouped calculations. For this reason, filtering is often considerably faster on `ungroup()`ed data.

Naming arguments is optional,

unnamed arguments are used in the order they are given

```
four_cyls <- filter(mtcars, cyl == 4)
```

the first argument should take the value `mtcars`

```
four_cyls <- filter(.data = mtcars, cyl == 4)
```

the `.data` **argument should take the value** `mtcars`

Your Turn

Take a look at the next chunk.

1. What functions are being called?

2. What arguments do they take?

3. What values are being passed as which arguments?

````
46
47 ```{r}
48 heights <- pull(.data = starwars, var = height)
49 mean(heights, na.rm = TRUE)
50 ```
51
````

```
heights <- pull(.data = starwars, var = height)
```

pull() is being called with the `.data` argument being `starwars`, and the `var` argument being `height`

```
mean(heights, na.rm = TRUE)
```

mean() is being called with the `x` argument being `heights`, and the `na.rm` argument being `TRUE`

Common Syntax problem #1

Missing closing parenthesis or quotes

```
mean(heights
```

```
"Oops I'm stuck in a string
```

Common Syntax problem #2

Surrounding something in quotes, when R expected something else (or vice versa), e.g.

```
> mean("heights")
[1] NA
Warning message:
In mean.default("heights") :
  argument is not numeric or logical: returning NA
```

Your Turn

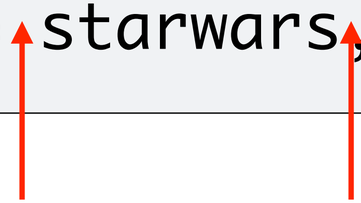
There are three chunks under "Syntax gone wrong".

Run each, read the error message, and try to fix the syntax.

```
sd(pull(.data = starwars, var = weight))
```

```
my_name <- "Charlotte"
```

```
pull(.data = starwars, var = height)
```



Getting Started

with R and RStudio

Summary

Use R Notebooks to keep together code, output and text.

If you need to use output, assign it to a variable <-

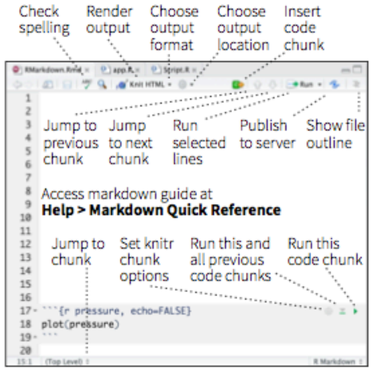
Get help on a function with ?

Watch your (, "

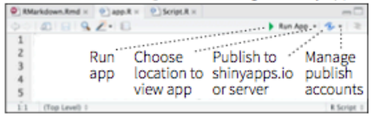
RStudio IDE : : CHEAT SHEET

Documents and Apps

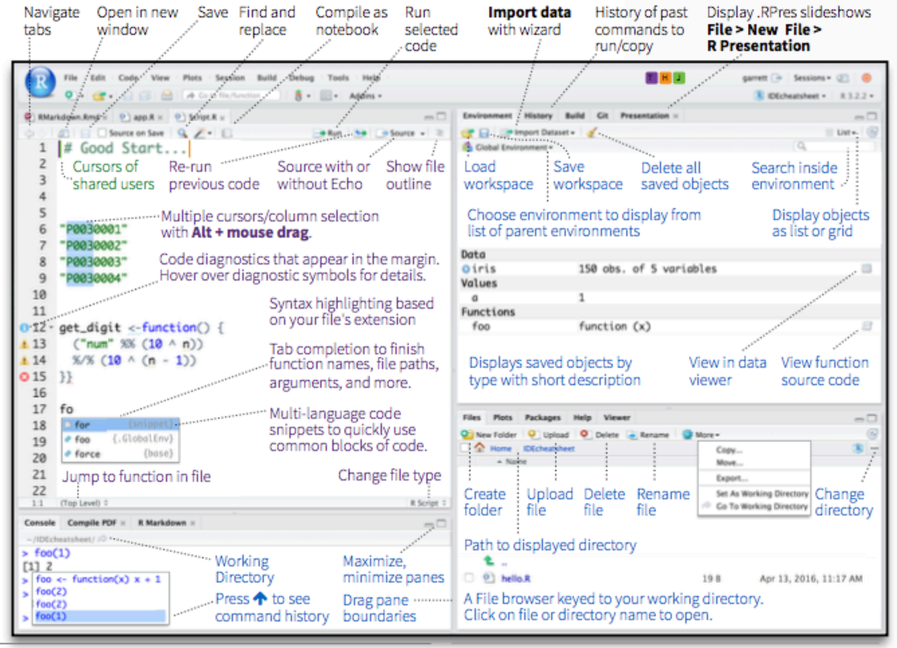
Open Shiny, R Markdown, knitr, Sweave, LaTeX, .Rd files and more in Source Pane



RStudio recognizes that files named **app.R**, **server.R**, **ui.R**, and **global.R** belong to a shiny app

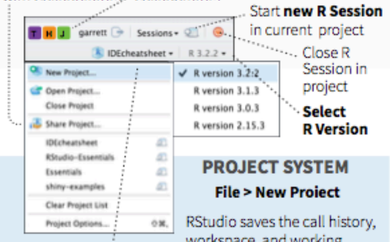


Write Code



R Support

Pro Features



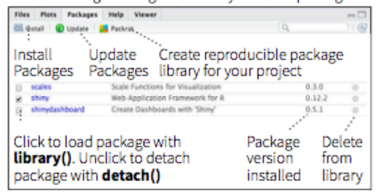
PROJECT SYSTEM

File > New Project

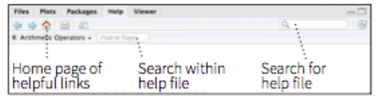
RStudio saves the call history, workspace, and working directory associated with a project. It reloads each when you re-open a project.

RStudio opens plots in a dedicated Plots pane

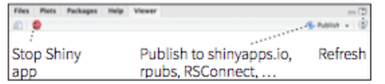
GUI Package manager lists every installed package



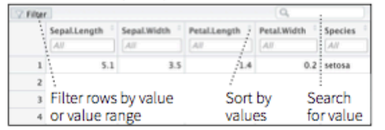
RStudio opens documentation in a dedicated Help pane



Viewer Pane displays HTML content, such as Shiny apps, RMarkdown reports, and interactive visualizations

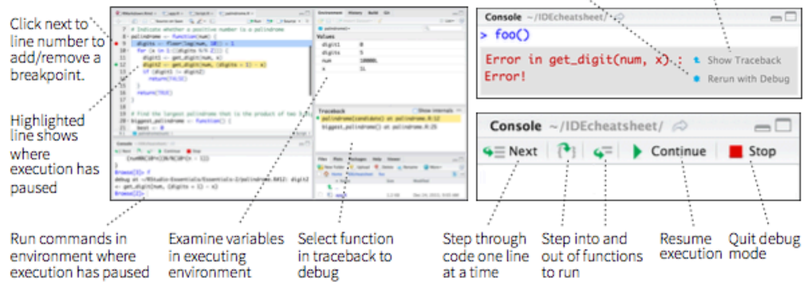


View(<data>) opens spreadsheet like view of data set



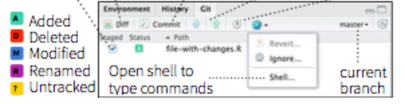
Debug Mode

Open with **debug()**, **browse()**, or a breakpoint. RStudio will open the debugger mode when it encounters a breakpoint while executing code.



Version Control with Git or SVN

Turn on at **Tools > Project Options > Git/SVN**

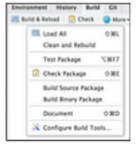


Package Writing

File > New Project > New Directory > R Package

Turn project into package,
Enable roxygen documentation with
Tools > Project Options > Build Tools

Roxygen guide at
Help > Roxygen Quick Reference



RStudio

1 LAYOUT

	Windows/Linux	Mac
Move focus to Source Editor	Ctrl+1	Ctrl+1
Move focus to Console	Ctrl+2	Ctrl+2
Move focus to Help	Ctrl+3	Ctrl+3
Show History	Ctrl+4	Ctrl+4
Show Files	Ctrl+5	Ctrl+5
Show Plots	Ctrl+6	Ctrl+6
Show Packages	Ctrl+7	Ctrl+7
Show Environment	Ctrl+8	Ctrl+8
Show Git/SVN	Ctrl+9	Ctrl+9
Show Build	Ctrl+0	Ctrl+0

2 RUN CODE

	Windows/Linux	Mac
Search command history	**Ctrl+↑**	**Cmd+↑**
Navigate command history	↑/↓	↑/↓
Move cursor to start of line	Home	Cmd+←
Move cursor to end of line	End	Cmd+→
Change working directory	Ctrl+Shift+H	Ctrl+Shift+H
Interrupt current command	**Esc**	**Esc**
Clear console	**Ctrl+L**	**Ctrl+L**
Quit Session (desktop only)	Ctrl+Q	Cmd+Q
Restart R Session	**Ctrl+Shift+F10**	**Cmd+Shift+F10**
Run current line/selection	**Ctrl+Enter**	**Cmd+Enter**
Run current (retain cursor)	Alt+Enter	Option+Enter
Run from current to end	Ctrl+Alt+E	Cmd+Option+E
Run the current function	Ctrl+Alt+F	Cmd+Option+F
Source a file	Ctrl+Alt+G	Cmd+Option+G
Source the current file	**Ctrl+Shift+S**	**Cmd+Shift+S**
Source with echo	Ctrl+Shift+Enter	Cmd+Shift+Enter

3 NAVIGATE CODE

	Windows /Linux	Mac
Goto File/Function	**Ctrl+.**	**Ctrl+.**
Fold Selected	Alt+L	Cmd+Option+L
Unfold Selected	Shift+Alt+L	Cmd+Shift+Option+L
Fold All	Alt+O	Cmd+Option+O
Unfold All	Shift+Alt+O	Cmd+Shift+Option+O
Go to line	Shift+Alt+G	Cmd+Shift+Option+G
Jump to	Shift+Alt+J	Cmd+Shift+Option+J
Switch to tab	Ctrl+Shift+.	Ctrl+Shift+.
Previous tab	Ctrl+F11	Ctrl+F11
Next tab	Ctrl+F12	Ctrl+F12
First tab	Ctrl+Shift+F11	Ctrl+Shift+F11
Last tab	Ctrl+Shift+F12	Ctrl+Shift+F12
Navigate back	Ctrl+F9	Cmd+F9
Navigate forward	Ctrl+F10	Cmd+F10
Jump to Brace	Ctrl+P	Ctrl+P
Select within Braces	Ctrl+Shift+Alt+E	Ctrl+Shift+Option+E
Use Selection for Find	Ctrl+F3	Cmd+E
Find in Files	Ctrl+Shift+F	Cmd+Shift+F
Find Next	Win: F3, Linux: Ctrl+G	Cmd+G
Find Previous	W: Shift+F3, L:	Cmd+Shift+G
Jump to Word	Ctrl+←/→	Option+←/→
Jump to Start/End	Ctrl+↑/↓	Cmd+↑/↓
Toggle Outline	Ctrl+Shift+O	Cmd+Shift+O

RStudio

4 WRITE CODE

	Windows /Linux	Mac
Attempt completion	**Tab or Ctrl+Space**	**Tab or Cmd+Space**
Navigate candidates	↑/↓	↑/↓
Accept candidate	Enter, Tab, or →	Enter, Tab, or →
Dismiss candidates	Esc	Esc
Undo	Ctrl+Z	Cmd+Z
Redo	Ctrl+Shift+Z	Cmd+Shift+Z
Cut	Ctrl+X	Cmd+X
Copy	Ctrl+C	Cmd+C
Paste	Ctrl+V	Cmd+V
Select All	Ctrl+A	Cmd+A
Delete Line	Ctrl+D	Cmd+D
Select	Shift+[Arrow]	Shift+[Arrow]
Select Word	Ctrl+Shift+ ←/→	Option+Shift+ ←/→
Select to Line Start	Alt+Shift+←	Cmd+Shift+←
Select to Line End	Alt+Shift+→	Cmd+Shift+→
Select Page Up/Down	Shift+PageUp/Down	Shift+PageUp/Down
Select to Start/End	Shift+Alt+↑/↓	Cmd+Shift+↑/↓
Delete Word Left	Ctrl+Backspace	Ctrl+Opt+Backspace
Delete Word Right		Option+Delete
Delete to Line End		Ctrl+K
Delete to Line Start		Option+Backspace
Indent	Tab (at start of line)	Tab (at start of line)
Outdent	Shift+Tab	Shift+Tab
Yank line up to cursor	Ctrl+U	Ctrl+U
Yank line after cursor	Ctrl+K	Ctrl+K
Insert yanked text	Ctrl+Y	Ctrl+Y
Insert <-	**Alt+-**	**Option+-**
Insert %>%	**Ctrl+Shift+M**	**Cmd+Shift+M**
Show help for function	F1	F1
Show source code	F2	F2
New document	Ctrl+Shift+N	Cmd+Shift+N
New document (Chrome)	Ctrl+Alt+Shift+N	Cmd+Shift+Opt+N
Open document	Ctrl+O	Cmd+O
Save document	Ctrl+S	Cmd+S
Close document	Ctrl+W	Cmd+W
Close document (Chrome)	Ctrl+Alt+W	Cmd+Option+W
Close all documents	Ctrl+Shift+W	Cmd+Shift+W
Extract function	Ctrl+Alt+X	Cmd+Option+X
Extract variable	Ctrl+Alt+V	Cmd+Option+V
Reindent lines	Ctrl+I	Cmd+I
(Un)Comment lines	**Ctrl+Shift+C**	**Cmd+Shift+C**
Reflow Comment	Ctrl+Shift+/	Cmd+Shift+/
Reformat Selection	Ctrl+Shift+A	Cmd+Shift+A
Select within braces	Ctrl+Shift+E	Ctrl+Shift+E
Show Diagnostics	Ctrl+Shift+Alt+P	Cmd+Shift+Opt+P
Transpose Letters		Ctrl+T
Move Lines Up/Down	Alt+↑/↓	Option+↑/↓
Copy Lines Up/Down	Shift+Alt+↑/↓	Cmd+Option+↑/↓
Add New Cursor Above	Ctrl+Alt+Up	Ctrl+Option+Up
Add New Cursor Below	Ctrl+Alt+Down	Ctrl+Option+Down
Move Active Cursor Up	Ctrl+Alt+Shift+Up	Ctrl+Option+Shift+Up
Move Active Cursor Down	Ctrl+Alt+Shift+Down	Ctrl+Opt+Shift+Down
Find and Replace	Ctrl+F	Cmd+F
Use Selection for Find	Ctrl+F3	Cmd+E
Replace and Find	Ctrl+Shift+J	Cmd+Shift+J

WHY RSTUDIO SERVER PRO?

RSP extends the the open source server with a commercial license, support, and more:

- open and run multiple R sessions at once
- tune your resources to improve performance
- edit the same project at the same time as others
- see what you and others are doing on your server
- switch easily from one version of R to a different version
- integrate with your authentication, authorization, and audit practices

Download a free 45 day evaluation at
www.rstudio.com/products/rstudio-server-pro/

5 DEBUG CODE

	Windows/Linux	Mac
Toggle Breakpoint	Shift+F9	Shift+F9
Execute Next Line	F10	F10
Step Into Function	Shift+F4	Shift+F4
Finish Function/Loop	Shift+F6	Shift+F6
Continue	Shift+F5	Shift+F5
Stop Debugging	Shift+F8	Shift+F8

6 VERSION CONTROL

	Windows/Linux	Mac
Show diff	Ctrl+Alt+D	Ctrl+Option+D
Commit changes	Ctrl+Alt+M	Ctrl+Option+M
Scroll diff view	Ctrl+↑/↓	Ctrl+↑/↓
Stage/Unstage (Git)	Spacebar	Spacebar
Stage/Unstage and move to next	Enter	Enter

7 MAKE PACKAGES

	Windows/Linux	Mac
Build and Reload	Ctrl+Shift+B	Cmd+Shift+B
Load All (devtools)	**Ctrl+Shift+L**	**Cmd+Shift+L**
Test Package (Desktop)	**Ctrl+Shift+T**	**Cmd+Shift+T**
Test Package (Web)	Ctrl+Alt+F7	Cmd+Opt+F7
Check Package	Ctrl+Shift+E	Cmd+Shift+E
Document Package	**Ctrl+Shift+D**	**Cmd+Shift+D**

8 DOCUMENTS AND APPS

	Windows/Linux	Mac
Preview HTML (Markdown, etc.)	Ctrl+Shift+K	Cmd+Shift+K
Knit Document (knitr)	**Ctrl+Shift+K**	**Cmd+Shift+K**
Compile Notebook	Ctrl+Shift+K	Cmd+Shift+K
Compile PDF (TeX and Sweave)	Ctrl+Shift+K	Cmd+Shift+K
Insert chunk (Sweave and Knitr)	Ctrl+Alt+I	Cmd+Option+I
Insert code section	Ctrl+Shift+R	Cmd+Shift+R
Re-run previous region	Ctrl+Shift+P	Cmd+Shift+P
Run current document	Ctrl+Alt+R	Cmd+Option+R
Run from start to current line	**Ctrl+Alt+B**	**Cmd+Option+B**
Run the current code section	**Ctrl+Alt+T**	**Cmd+Option+T**
Run previous Sweave/Rmd code	Ctrl+Alt+P	Cmd+Option+P
Run the current chunk	Ctrl+Alt+C	Cmd+Option+C
Run the next chunk	Ctrl+Alt+N	Cmd+Option+N
Sync Editor & PDF Preview	Ctrl+F8	Cmd+F8
Previous plot	Ctrl+Alt+F11	Cmd+Option+F11
Next plot	Ctrl+Alt+F12	Cmd+Option+F12
Show Keyboard Shortcuts	**Alt+Shift+K**	**Option+Shift+K**

Your Turn

Open 02-Data-basics.Rmd