

# Intro to R

RStudio

## Help! Office hours

Office hours will always be held at the *same Zoom link*.

# Working with R – RStudio

RStudio is an Integrated Development Environment (IDE) for R

- Helps you write code - makes suggestions
- Helps you view the output of your code
- Helps you find errors
- Is NOT a dropdown statistical tool (such as Stata)
  - See [Rcmdr](#) or [Radiant](#)



[[source](#)]

RStudio used to be the name of a company that is now called [Posit](#).

# RStudio

## Easier working with R

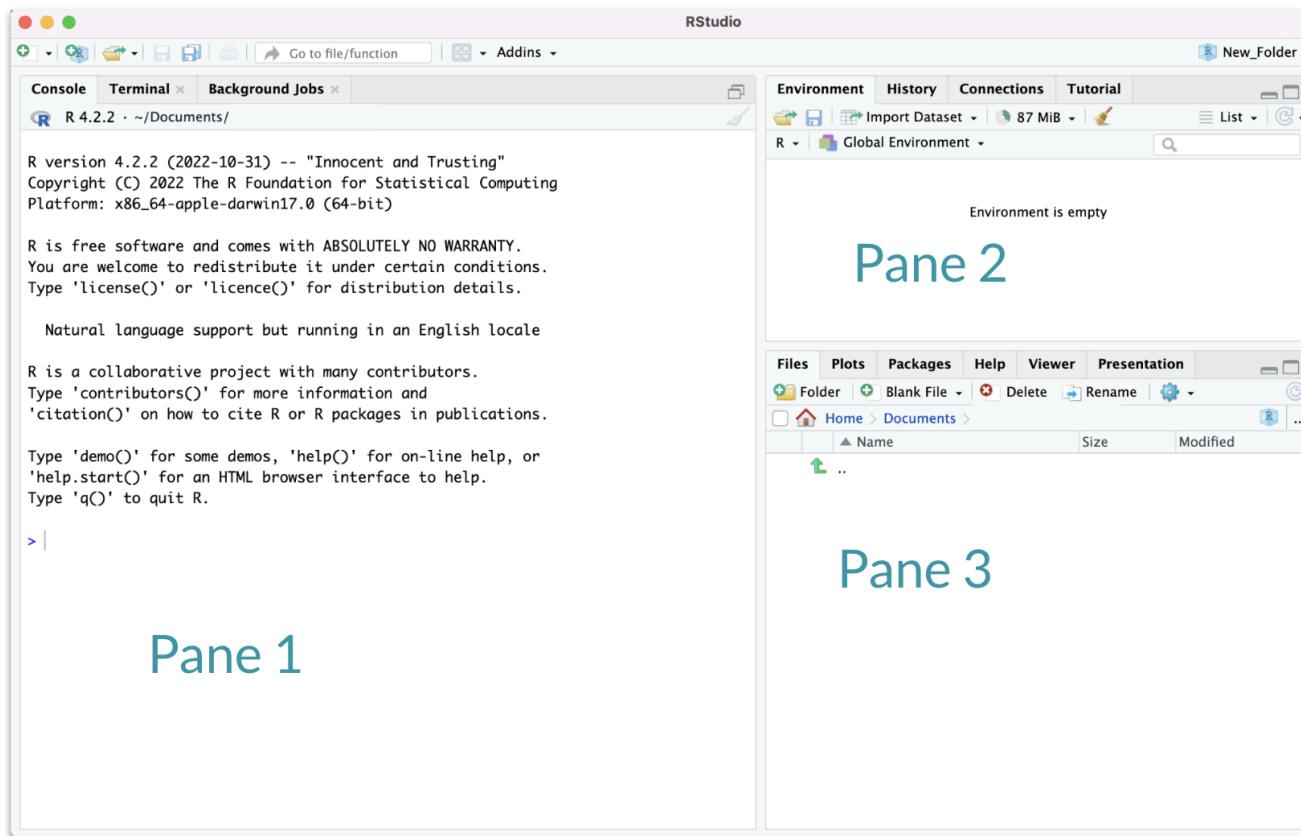
- Syntax highlighting, code completion, and smart indentation
- Easily manage multiple working directories and projects

## More information

- Workspace browser and data viewer
- Plot history, zooming, and flexible image and file export
- Integrated R help and documentation

# RStudio

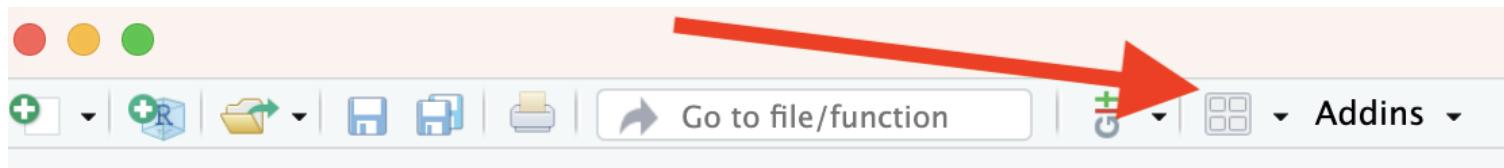
First it is important to be familiar with the layout. When you first open RStudio, you will see 3 panes.



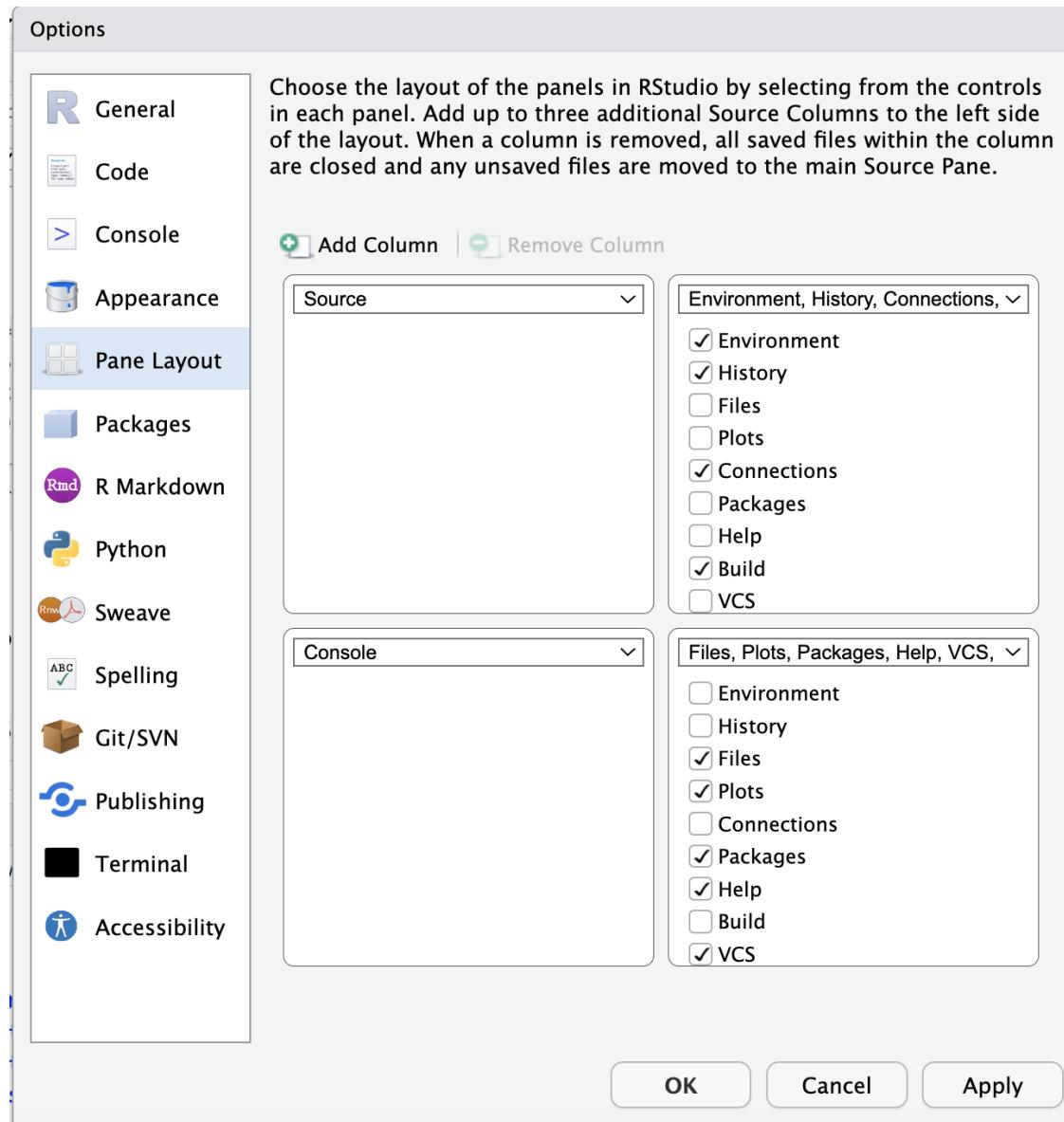
## RStudio Layout

If RStudio doesn't look the way you want (or like our RStudio), then:

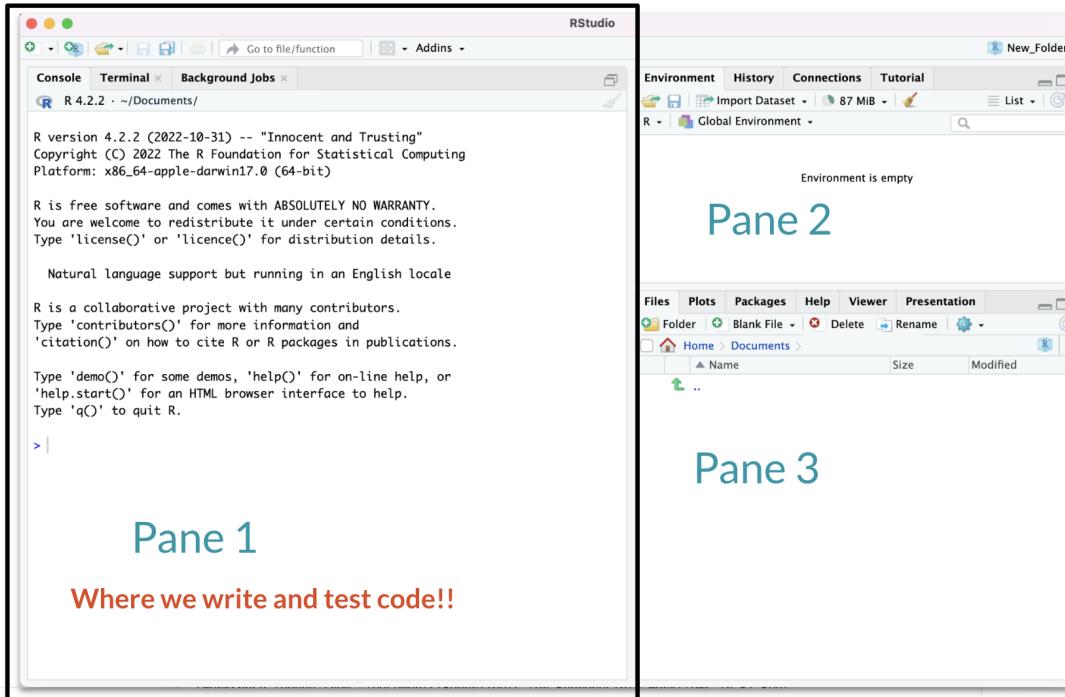
Click on the pane button, which looks like a waffle with 4 indentations. Scroll down to "Pane Layout".



# Default Layout

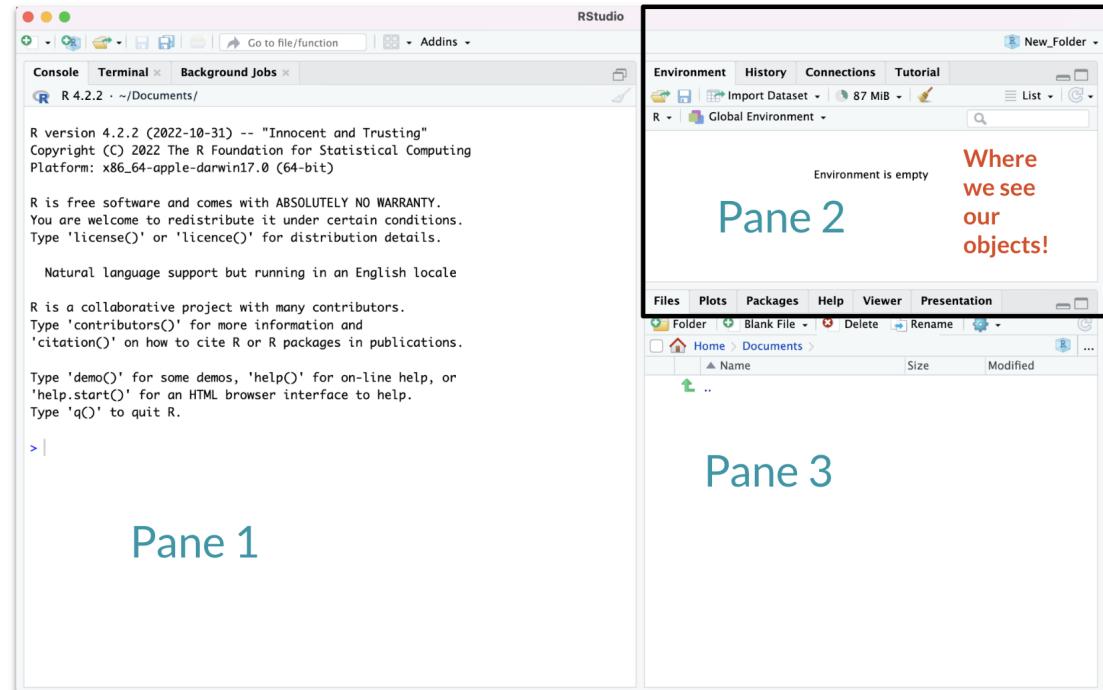


# Pane 1 (Left side) for writing code

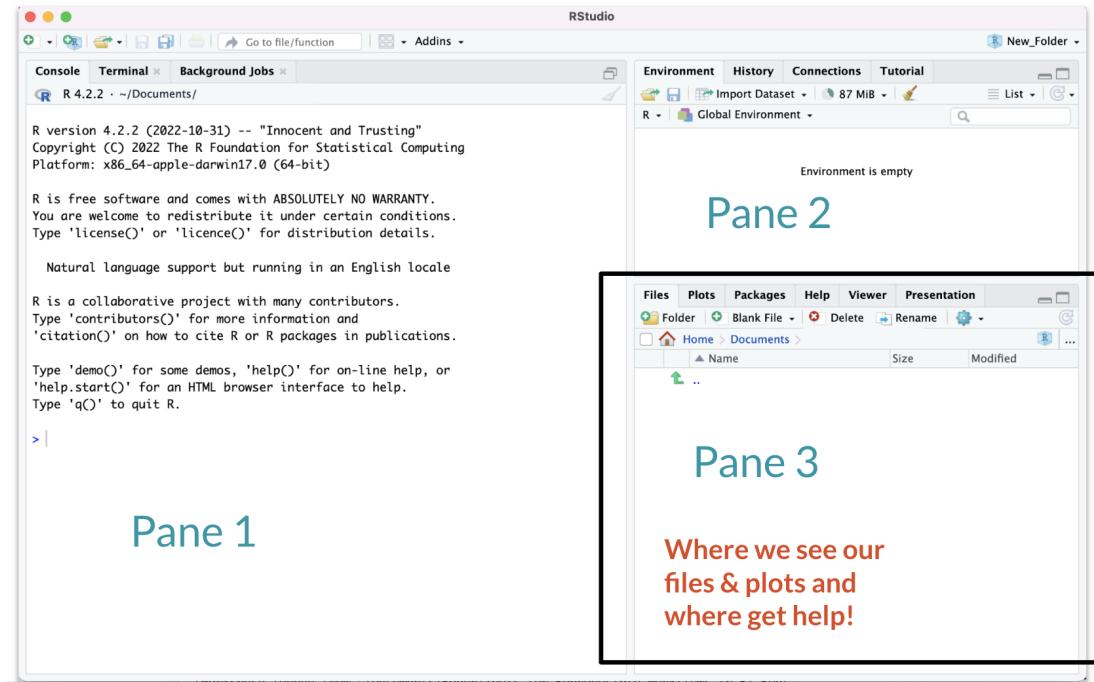


# Pane 2 - where objects will be

We will start seeing this tomorrow!

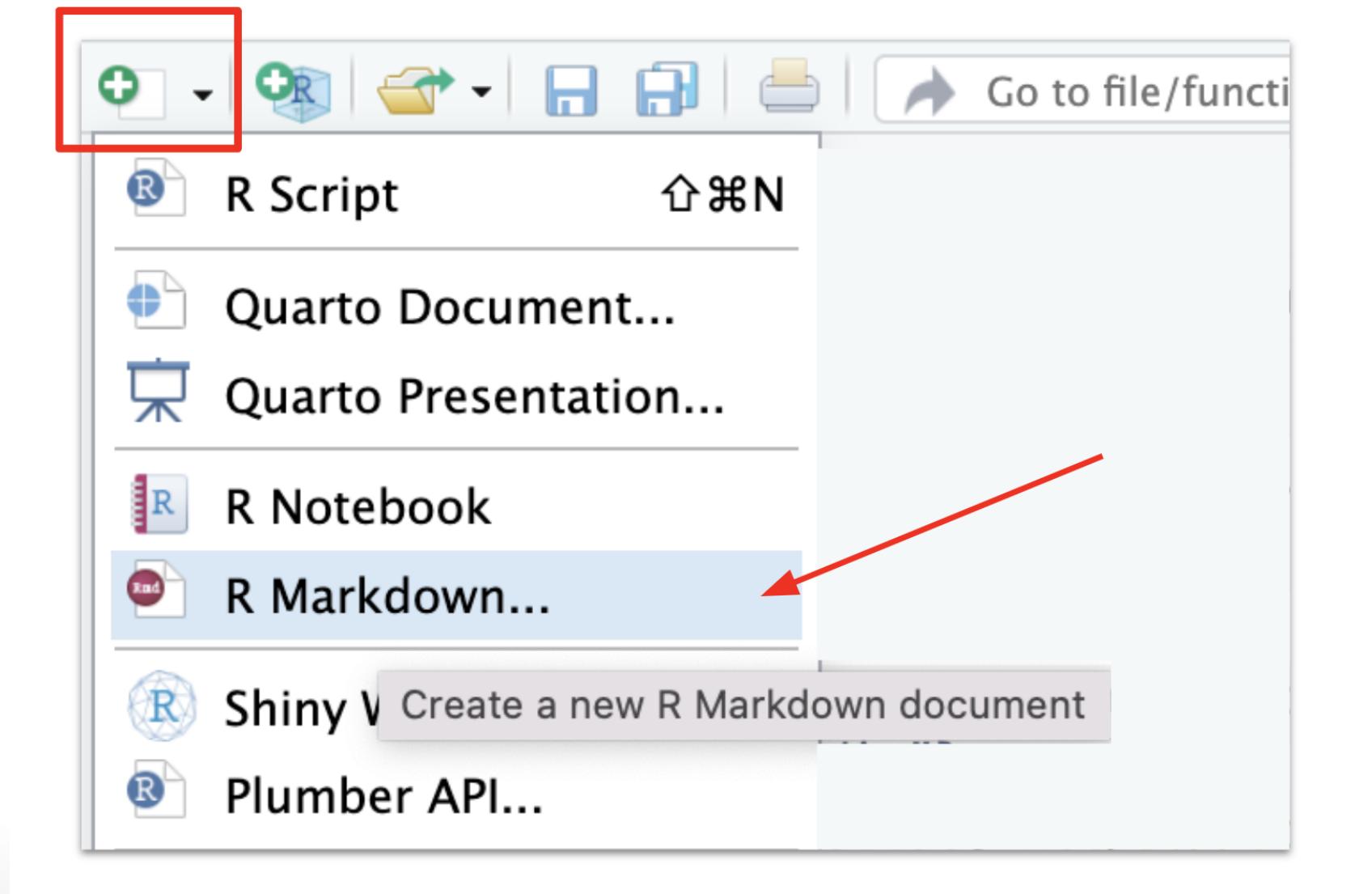


# Pane 3 - where we get help and see plots



## Hidden Pane

To save a copy of your code. You must open a file first - this will open a 4th pane. These files include Scripts or what are called R Markdown files.



## Hidden Pane

You will see a popup that you can just say “OK” to for now.

New R Markdown

Document       Presentation       Shiny       From Template

**Title:**

**Author:**

**Date:**

Use current date when rendering document

**Default Output Format:**

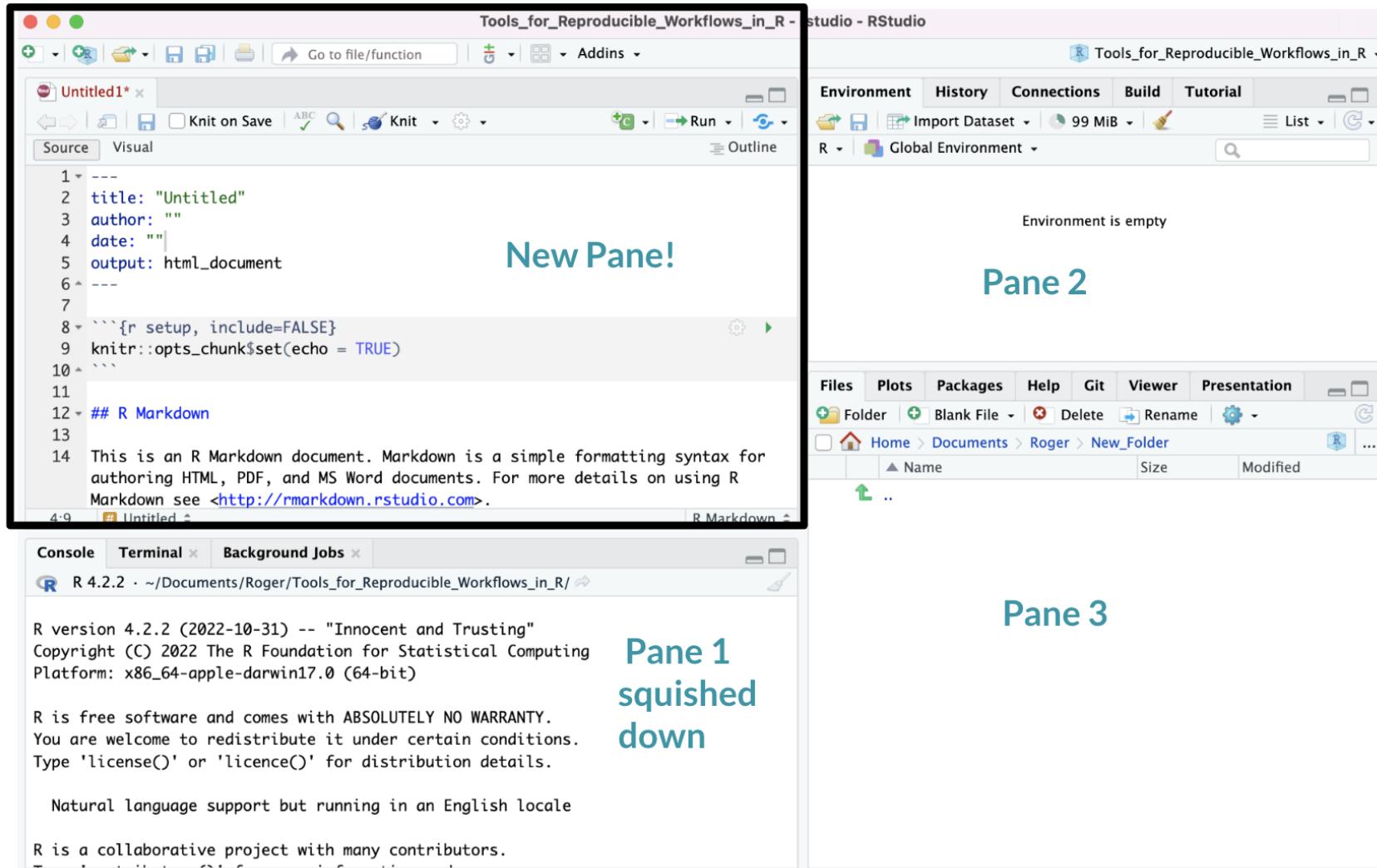
**HTML**  
Recommended format for authoring (you can switch to PDF or Word output anytime).

**PDF**  
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

**Word**  
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

# Hidden Pane

Nice! now we have a place to save code! This is where we will mostly be working.



## Working with R in R Studio - 2 major panes:

1. The **Source/Editor**: "Analysis" Script + Interactive Exploration

- Static copy of what you did (reproducibility)
- Top by default

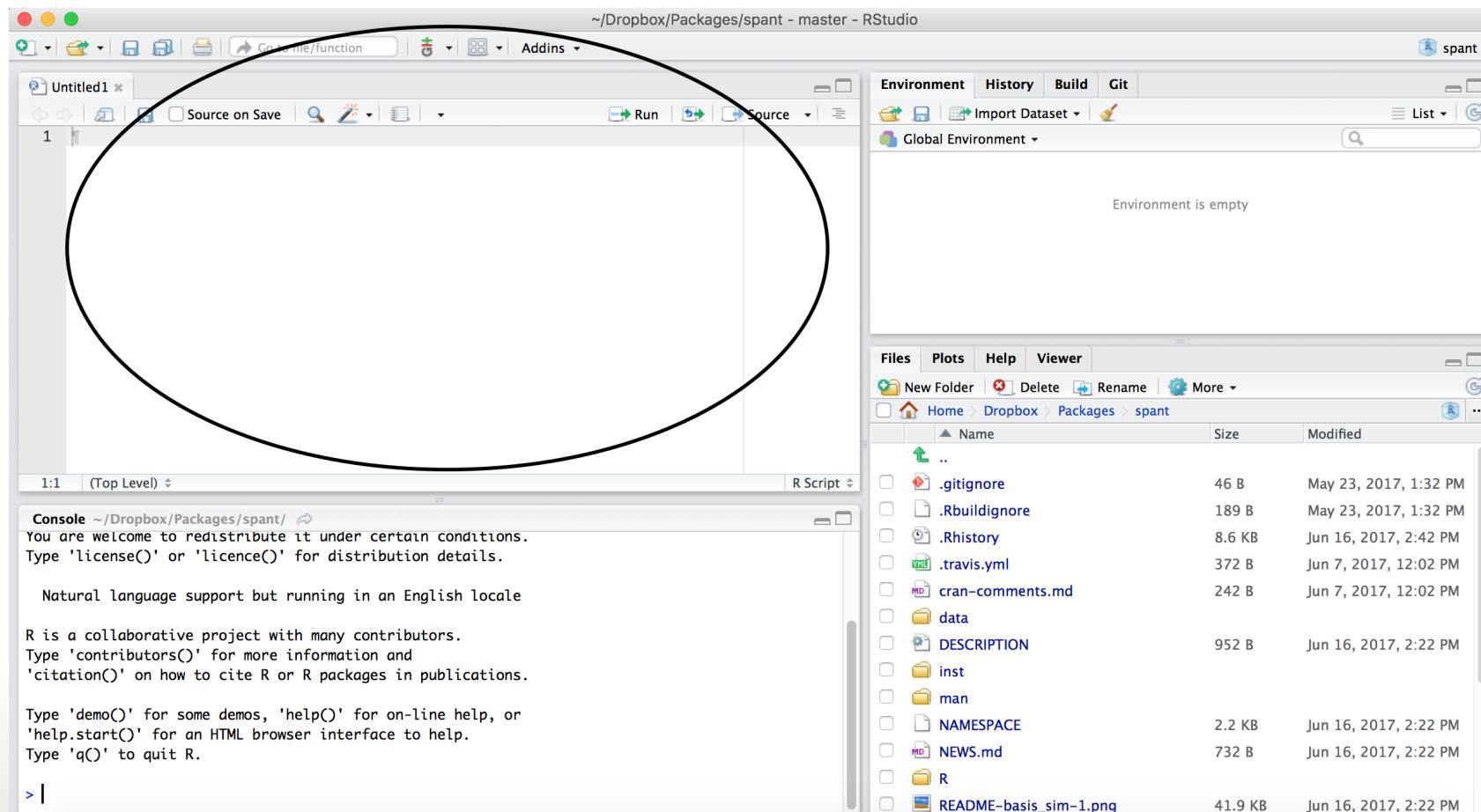
2. The **R Console**: "interprets" whatever you type

- Calculator
- Try things out interactively, then add to your editor
- Bottom by default

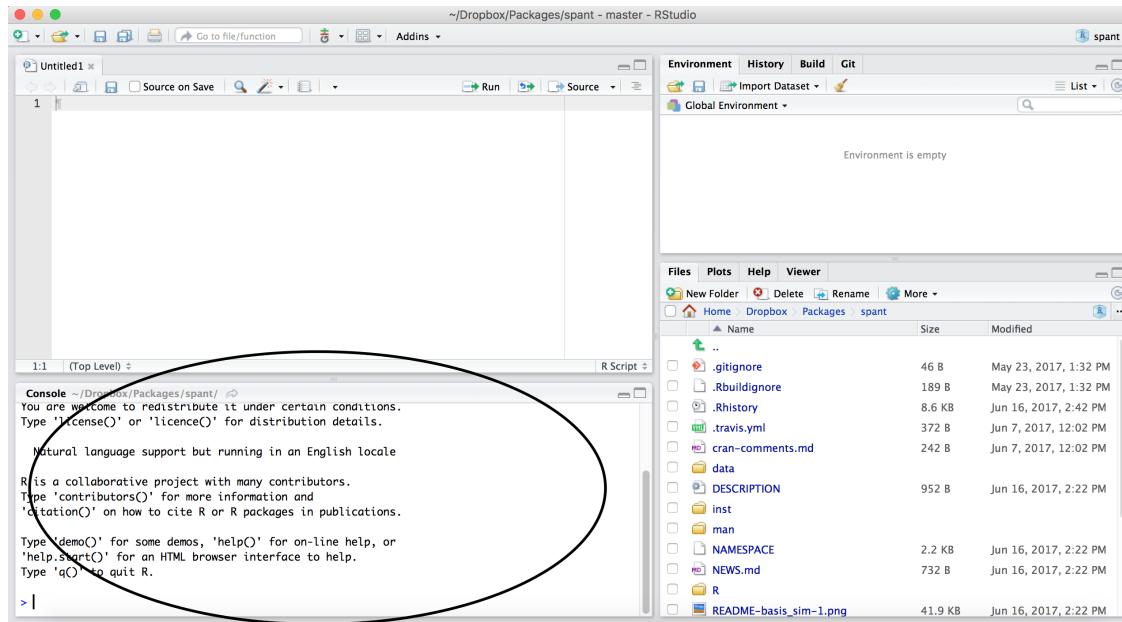
# Source / Editor

- Where files open to
- Have R code and comments in them
- Can highlight and press (CMD+Enter (Mac) or Ctrl+Enter (Windows)) to run the code

In a .R file (we call a script), code is saved on your disk



# R Console



- Where code is executed (where things happen)
- You can type here for things interactively to test code
- Code is **not saved** on your disk

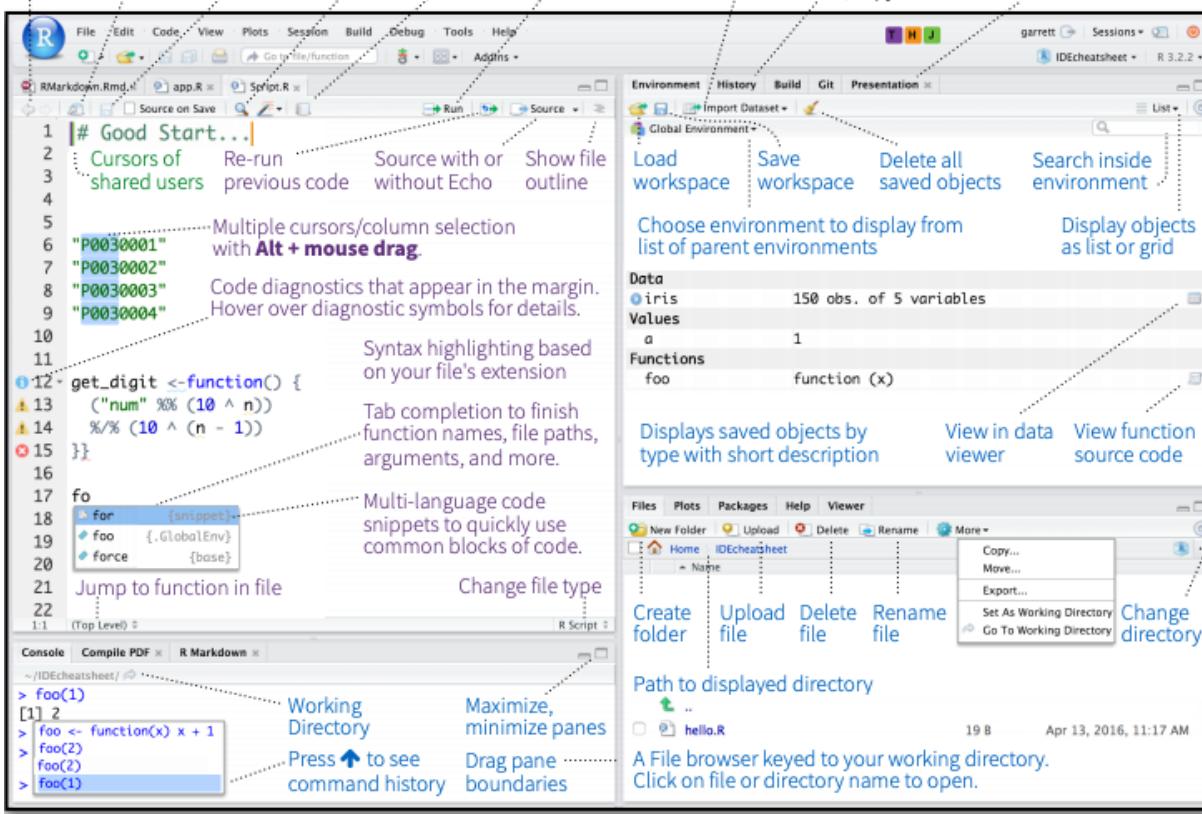
# RStudio

Super useful “cheat sheet”:

<https://github.com/rstudio/cheatsheets/raw/master/rstudio-ide.pdf>

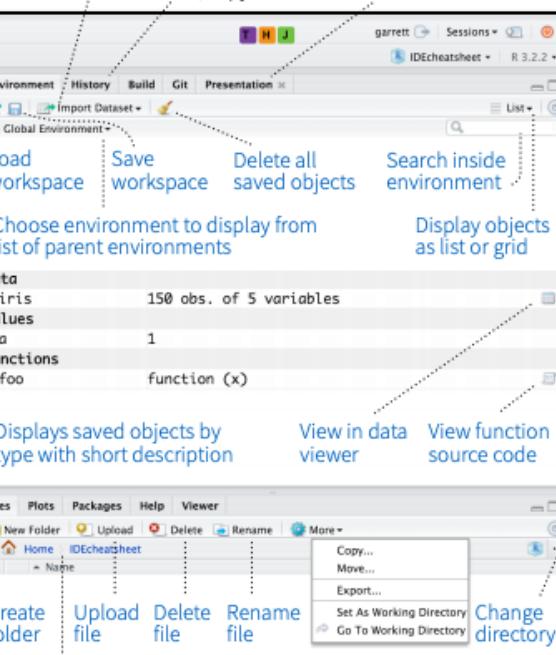
## Write Code

Navigate tabs  
Open in new window  
Save  
Find and replace  
Compile as notebook  
Run selected code



## R Support

Import data with wizard  
History of past commands to run/copy  
Display .RPres slideshows  
**File > New File > R Presentation**



# R Markdown files look different from scripts

It will look like this with text in it, unlike a script.

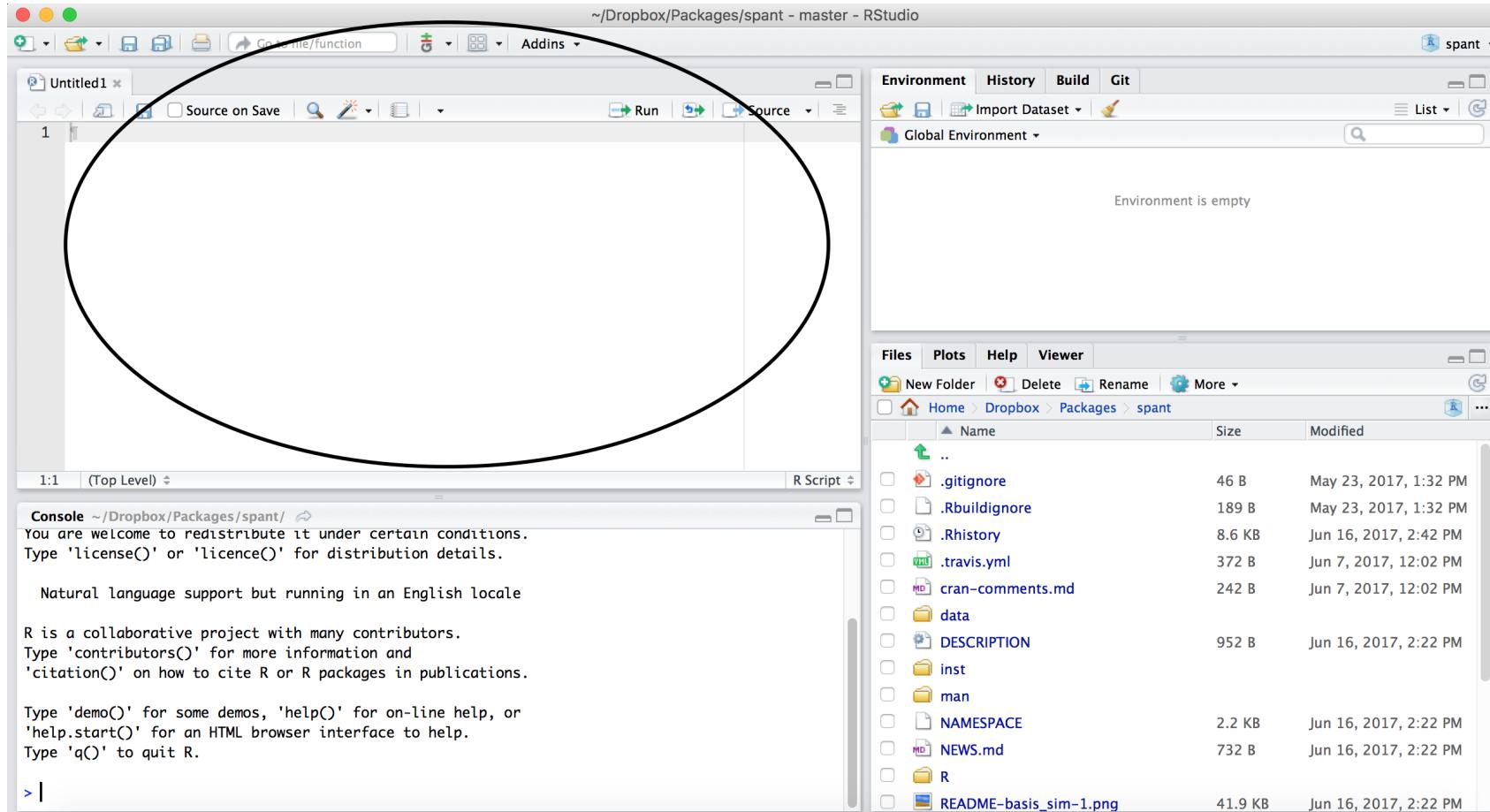
The screenshot shows the RStudio interface with a red box highlighting the code editor area. The code editor contains the following R Markdown code:

```
1 ---  
2 title: "first_markdown"  
3 output: html_document  
4 ---  
5  
6 ```{r setup, include=FALSE}  
7 knitr::opts_chunk$set(echo = TRUE)  
8 ```  
9  
10 ## R Markdown  
11  
12 This is an R Markdown document. Markdown is a simple formatting syntax for  
authoring HTML, PDF, and MS Word documents. For more details on using R  
Markdown see <http://rmarkdown.rstudio.com>.  
13  
14 When you click the **Knit** button a document will be generated that includes  
both content as well as the output of any embedded R code chunks within the  
document. You can embed an R code chunk like this:  
15  
16 ```{r cars}  
2:23 # first_markdown
```

The R Markdown code includes a title, output type, and a code chunk setup. The code chunk setup includes the `knitr` package and sets the echo option to `TRUE`. The main content of the document discusses R Markdown and provides an example of an R code chunk. The code editor also shows the current file path: `/Documents/GitHub/Teaching/intro\_to\_r/` and the R version information: `Natural language support but running in an English locale`.

The RStudio interface also includes the Environment, History, Connections, Build, Git, and Tutorial tabs. The Global Environment panel shows "Environment is empty". The Files panel lists the contents of the `intro\_to\_r` directory, including files like `.gitignore`, `Rbuildignore`, `Rhistory`, and various CSV and XLSX files.

# Recall that a script was just empty

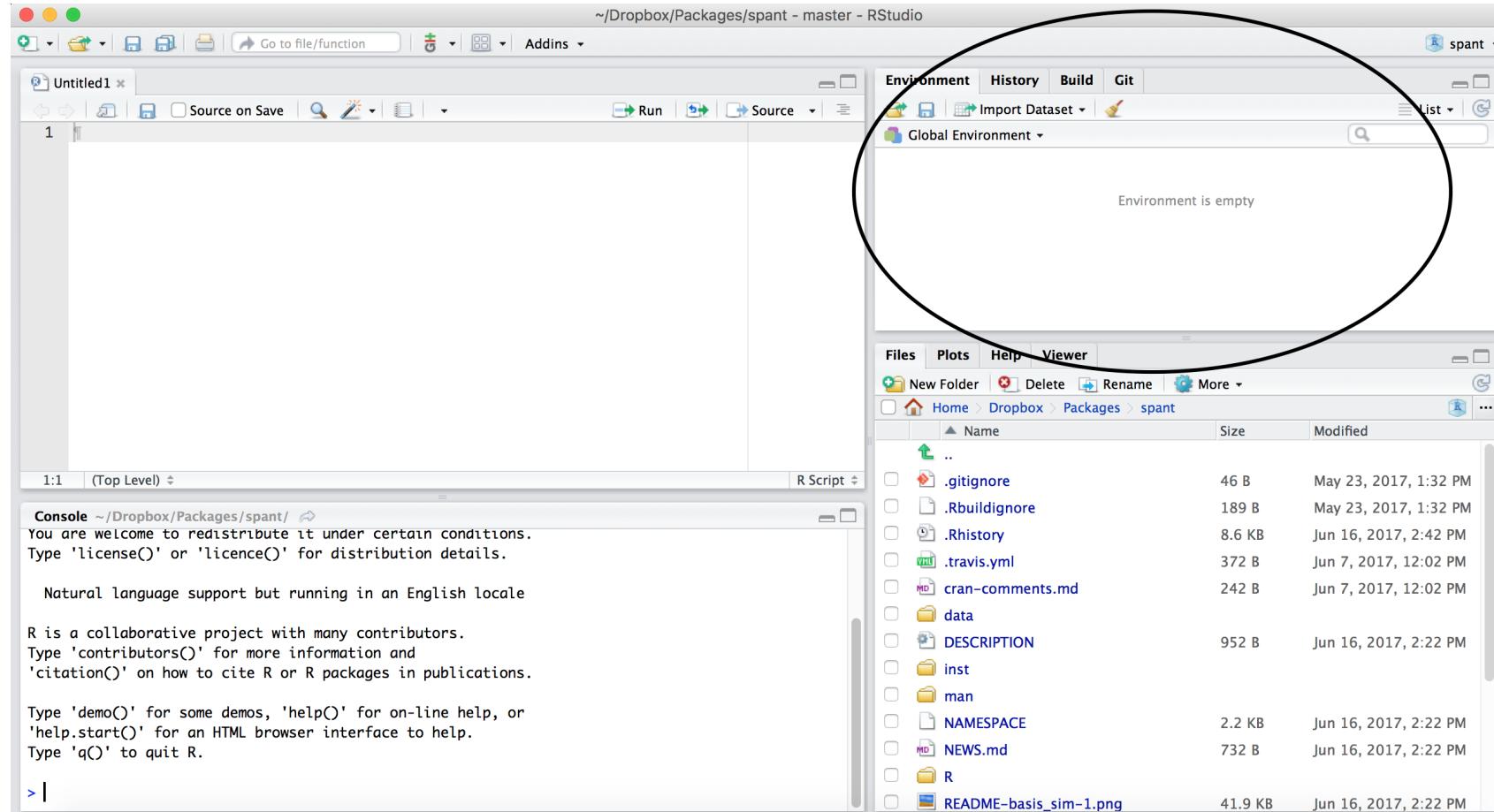


## Scripts and R Markdown

Although people will use scripts often, and they are good for more programmatic purposes, we generally don't recommend them for Public Health Researchers.

For data analyses, R Markdown files are generally superior because they allow you to check your code and write more info about your code.

# Workspace/Environment



## Workspace/Environment

- Tells you what **objects** are in R
- What exists in memory/what is loaded?/what did I read in?

## History

- Shows previous commands. Good to look at for debugging, but **don't rely** on it.  
Instead use RMarkdown!
- Also type the “up” key in the Console to scroll through previous commands

## Other Panes

- **Files** - shows the files on your computer or the directory you are working in
- **Viewer** - can view data or R objects
- **Help** - shows help of R commands
- **Plots** - pictures and figures
- **Packages** - list of R packages that are loaded in memory

Let's take a look at R Studio  
ourselves!

# Lab: Starting with R and RMarkdown

## RStudio Lab

To do this lab we need to:

- Download the file at the link above by clicking on it or go to the [website](#) schedule page
- Find the downloaded file on your computer
- Open the file in RStudio (double clicking the file name typically works)

These videos can help if you aren't sure where your downloads are:

If you have a PC: <https://youtu.be/we6vwB7DsNU>

If you have a Mac: <https://www.youtube.com/watch?v=Ao9e0cDzMrE>

You can find these on the resource page of the class website.

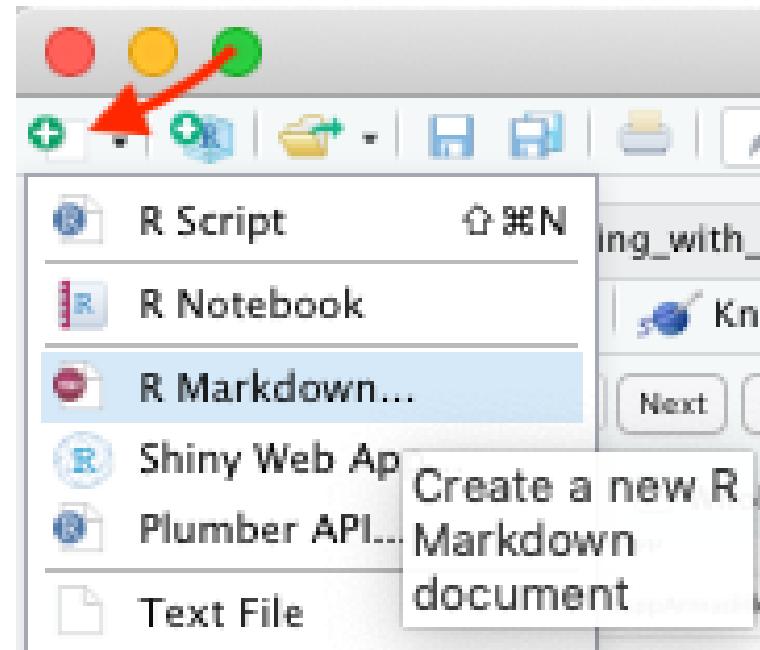
## R Markdown file

R Markdown files (.Rmd) help generate reports that include your code and output. Think of them as fancier scripts.

1. Helps you describe your code
2. Allows you to check the output
3. Can create many different file types

# Create an R Markdown file

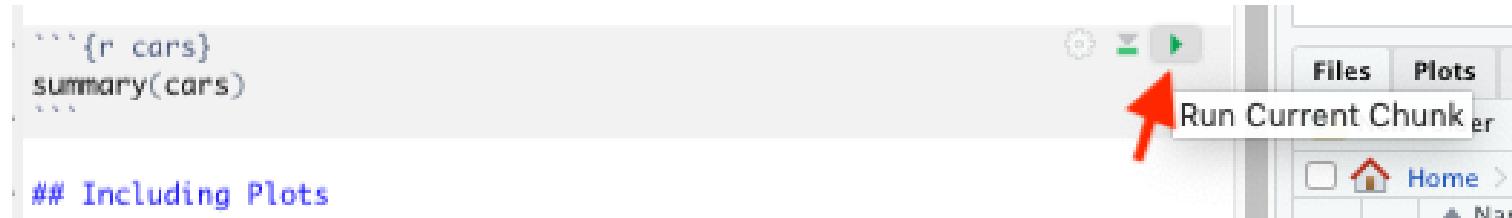
Go to File → New File → R Markdown or click the green add file button.



## Code chunks

Within R Markdown files are code “chunks”.

This is where you can type R code and run it!



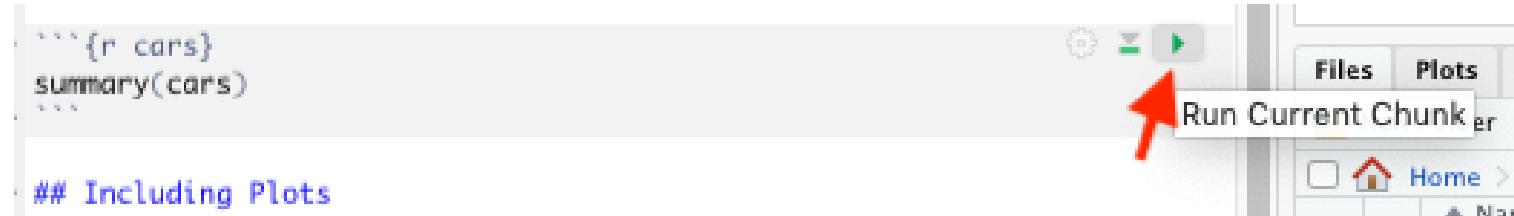
The image shows a screenshot of the RStudio IDE. On the left, there is a code editor window containing R code. The code includes a code chunk indicator (three backticks) followed by `r cars`, then `summary(cars)`, and another three backticks. Below this, the text `## Including Plots` is visible. On the right side of the interface, there is a toolbar with several icons. A red arrow points to the icon for running the current code chunk, which is a green square with a white play symbol. The menu bar at the top has options like "File", "Plots", and "Edit".

```
```{r cars}
summary(cars)
```

## Including Plots
```

## Run code in a chunk

Clicking the run (play) button runs the code in the chunk.



Ctrl + Enter on Windows or Command + Enter on Mac in your script evaluates that line of code

# Running a chunk executes the code

- generally see a preview of the output of the code just below the chunk
- see the code in the console

The screenshot shows the RStudio interface. At the top, there are two tabs: "Untitled2" and "RStudio.Rmd". Below the tabs is a toolbar with icons for back, forward, knit, run, and other functions. The main area is divided into two panes: "Source" (top) and "Visual" (bottom). In the Source pane, the R Markdown code is visible:

```
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS  
Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.  
15  
16 When you click the **Knit** button a document will be generated that includes both content as well as  
the output of any embedded R code chunks within the document. You can embed an R code chunk like this:  
17  
18 ```{r cars}  
19 summary(cars)  
20 ```
```

In the Visual pane, the output of the `summary(cars)` chunk is displayed as a table:

|         | speed | dist           |
|---------|-------|----------------|
| Min.    | : 4.0 | Min. : 2.00    |
| 1st Qu. | :12.0 | 1st Qu.: 26.00 |
| Median  | :15.0 | Median : 36.00 |
| Mean    | :15.4 | Mean : 42.98   |
| 3rd Qu. | :19.0 | 3rd Qu.: 56.00 |
| Max.    | :25.0 | Max. :120.00   |

At the bottom of the RStudio window, the "Console" tab is active, showing the R session history:

```
>  
>  
>  
> summary(cars)  
    speed          dist  
Min. : 4.0   Min. : 2.00  
1st Qu.:12.0  1st Qu.: 26.00  
Median :15.0  Median : 36.00  
Mean   :15.4  Mean   : 42.98  
3rd Qu.:19.0  3rd Qu.: 56.00
```

# If you get annoyed by code previews in Markdown files:

In RStudio Click the Edit tab → scroll down to Preferences... → R Markdown

Uncheck the following:

The screenshot shows the 'Options' dialog in RStudio. On the left, a sidebar lists various sections: General, Code, Console, Appearance, Pane Layout, Packages, R Markdown (which is selected and highlighted in blue), Python, Sweave, and Spelling. The main area contains tabs for Basic, Advanced, Visual, and Citations, with 'Basic' selected. Under the 'R Markdown' section, there are several configuration options:

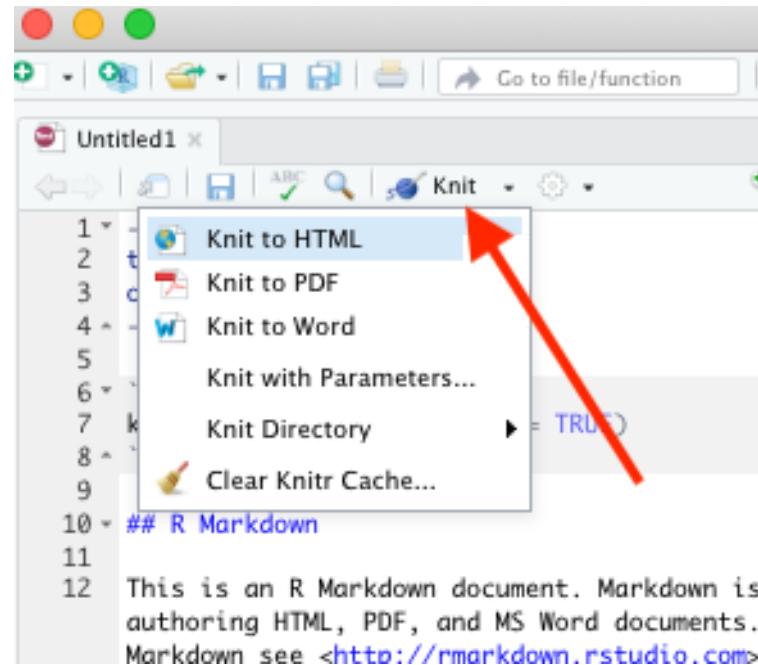
- Show document outline by default
- Soft-wrap R Markdown files
- Show in document outline:
- Show output preview in:
- Show output inline for all R Markdown documents (this option is highlighted with a red border)
- Show equation and image previews:
- Evaluate chunks in directory:

Below this, under 'R Notebooks', are two more options:

- Execute setup chunk automatically in notebooks
- Hide console automatically when executing notebook chunks

## Knit file to html

Running all chunks - this will create a report from the R Markdown document!



# Nice report!

This generates a nice report that you can share with others who can open in any browser.

The screenshot shows a window titled 'Untitled.html' with the URL ' ~/Documents/Roger/New\_Folder/Untitled.html'. The window includes standard OS X window controls (red, yellow, green) and a toolbar with 'Untitled.html', 'Open in Browser', 'Find', and a 'Publish' button. The main content area displays an R Markdown document:

# Untitled

Your Name  
2023-03-29

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed      dist
## Min.   : 4.0   Min.   : 2.00
## 1st Qu.:12.0   1st Qu.: 26.00
## Median :15.0   Median : 36.00
## Mean   :15.4   Mean   : 42.98
## 3rd Qu.:19.0   3rd Qu.: 56.00
## Max.   :25.0   Max.   :120.00
```

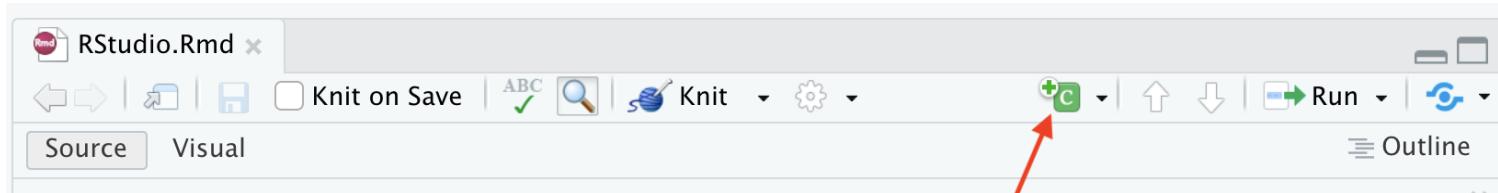
## Including Plots

You can also embed plots, for example:

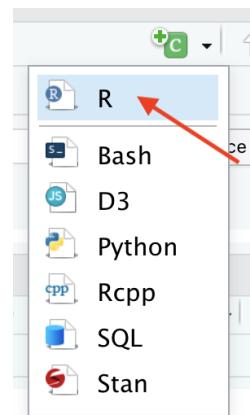
# Create Chunks

To create a new R code chunk:

- Use the insert code chunk button at the top of RStudio.



- Select R (default) as the language:



## Create Chunks

If you like keyboard shortcuts:

- Windows & Linux use Ctrl+Alt+I
- Mac use Command+Option+I

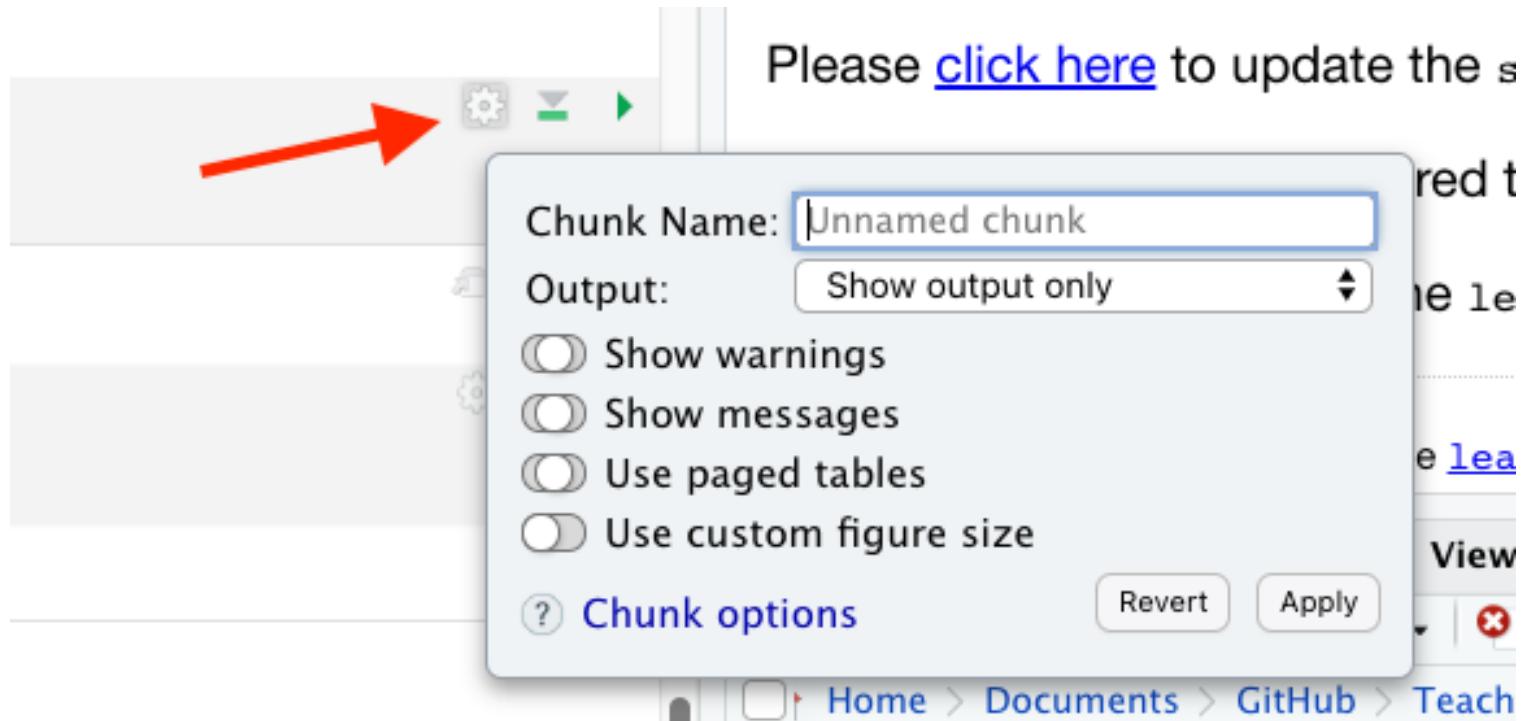
I is for insert.

## Run previous chunks button

You can run all chunks above a specific chunk using this button:

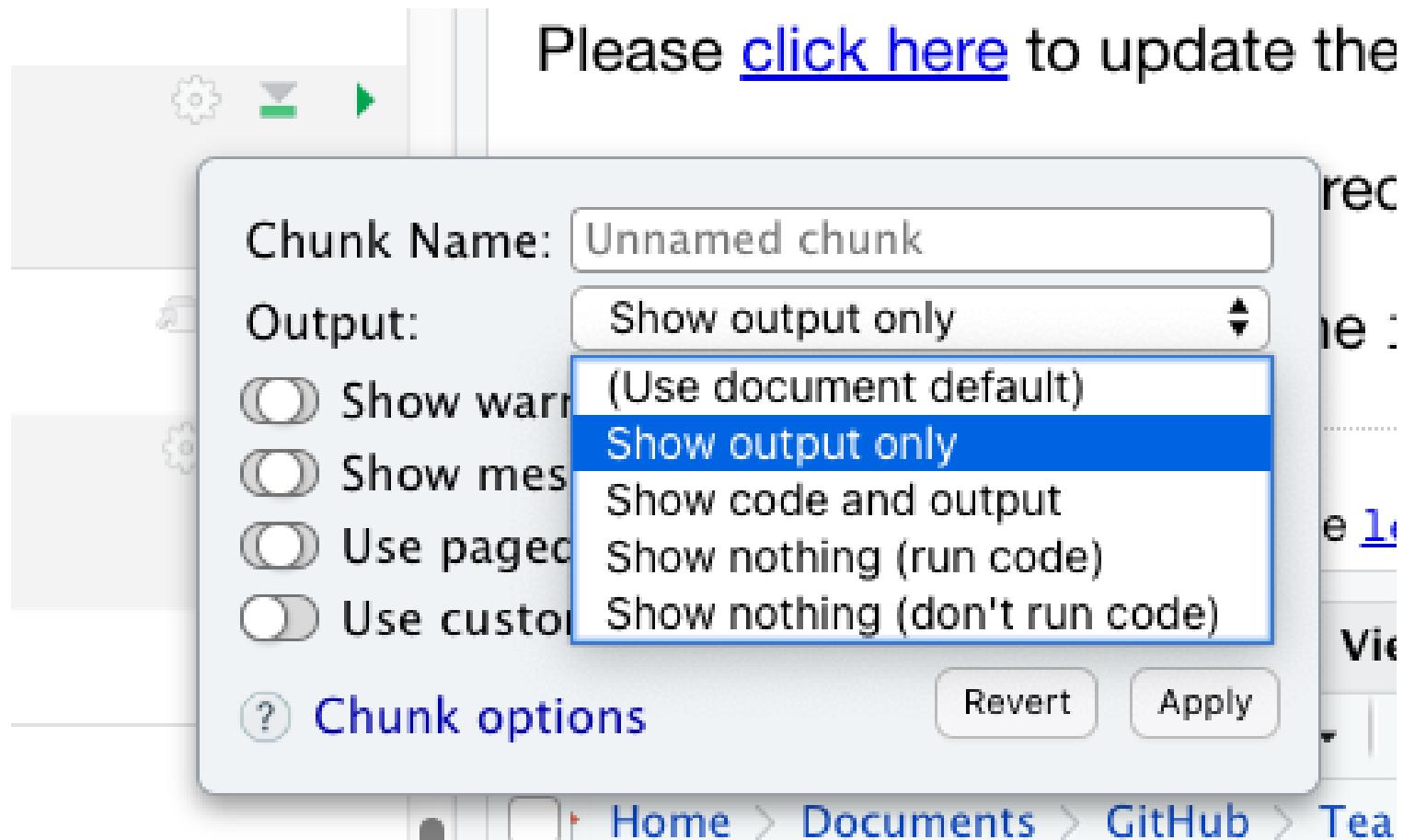


## Chunk settings



## Chunk settings

You can specify if a chunk will be seen in the report or not.



## Errors

R studio can help you find issues in your code. Note that sometimes the error occurs earlier than RStudio thinks.



A screenshot of the RStudio interface showing a code editor with the following content:

```
305 print(x, ...)  
306 - {r}  
✖ 307 print(x))  
308 ...  
3 unexpected token ')'  
3 unexpected end of document
```

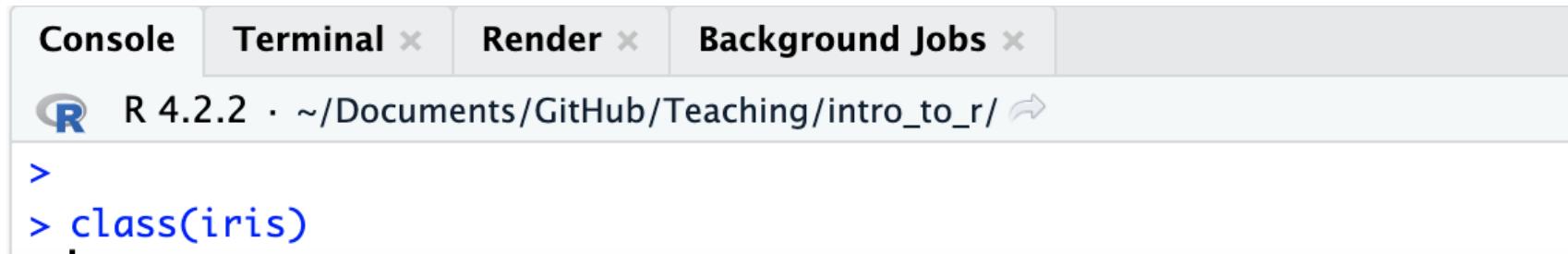
The line "print(x))" is highlighted with a red error indicator, and a tooltip box is open over it displaying the error message: "unexpected token ')'". The RStudio toolbar is visible at the top right, showing icons for settings, file, and run.

## Useful R Studio Shortcuts

- **Ctrl + Enter** on Windows or **Command + Enter** on Mac in your script evaluates that line of code
  - It's like copying and pasting the code into the console for it to run.
- **Ctrl+1** on Windows or **Command + 1** on Mac takes you to the script page
- **Ctrl+2** on Windows or **Command + 2** on Mac takes you to the console
- [http://www.rstudio.com/ide/docs/using/keyboard\\_shortcuts](http://www.rstudio.com/ide/docs/using/keyboard_shortcuts)

## Recap of where code goes

- you can test code in the console



The screenshot shows the RStudio interface with the 'Console' tab selected. The top bar displays 'Console', 'Terminal x', 'Render x', and 'Background Jobs x'. Below the bar, the R logo and 'R 4.2.2 · ~/Documents/GitHub/Teaching/intro\_to\_r/' are visible, along with a refresh icon. The console window contains the following text:  
>  
> `class(iris)`  
[1] "

- you can save code in a chunk in the editor (Markdown file)

```
## R Markdown
```

Code does not go here and instead goes within the grey chunks like this:

```
```{r}
summary(cars)
```
```



# Getting help from the preview

When you type in a function name, a pop up will preview documentation to help you. It also helps you remember the name of the function if you don't remember all of it!

The screenshot shows two examples of RStudio's documentation preview feature.

**Top Example:** The user has typed "`> class`". A tooltip appears over the first result in the completion dropdown, which is "class(x)". The tooltip contains the following text:

```
class(x)
Object Classes
R possesses a simple generic function mechanism which can be used for an object-oriented style of programming. Method dispatch takes place based on the class of the first argument to the generic function.
```

**Bottom Example:** The user has typed "`> read_`". A tooltip appears over the first result in the completion dropdown, which is "read\_csv". The tooltip contains the following text:

```
read_csv(file, col_names = TRUE, col_types = NULL,
        col_select = NULL, id = NULL, locale =
        default_locale(), na = c("", "NA"), quoted_na =
        TRUE, quote = "\"", comment = "", trim_ws = TRUE,
        skip = 0, n_max = Inf, guess_max = min(1000,
        n_max), name_repair = "unique", num_threads =
        readr_threads(), progress = show_progress(),
        show_col_types = should_show_types(),
```

Both examples include a "Press F1 for additional help" link at the bottom of the tooltip.

# Get help with the help pane



The screenshot shows the R help pane interface. At the top, there is a menu bar with tabs: Files, Plots, Packages, Help, Git, Viewer, and Presentation. Below the menu bar are several icons: a left arrow, a right arrow, a house icon, and a refresh/circular arrow icon. To the right of these icons is a search bar containing the word "class". Below the search bar is a dropdown menu labeled "R: Object Classes" with a "Find in Topic" button next to it. The main content area displays the title "class {base}" and the text "R Documentation".

## Object Classes

### Description

R possesses a simple generic function mechanism which can be used for an object-oriented style of programming. Method dispatch takes place based on the class of the first argument to the generic function.

### Usage

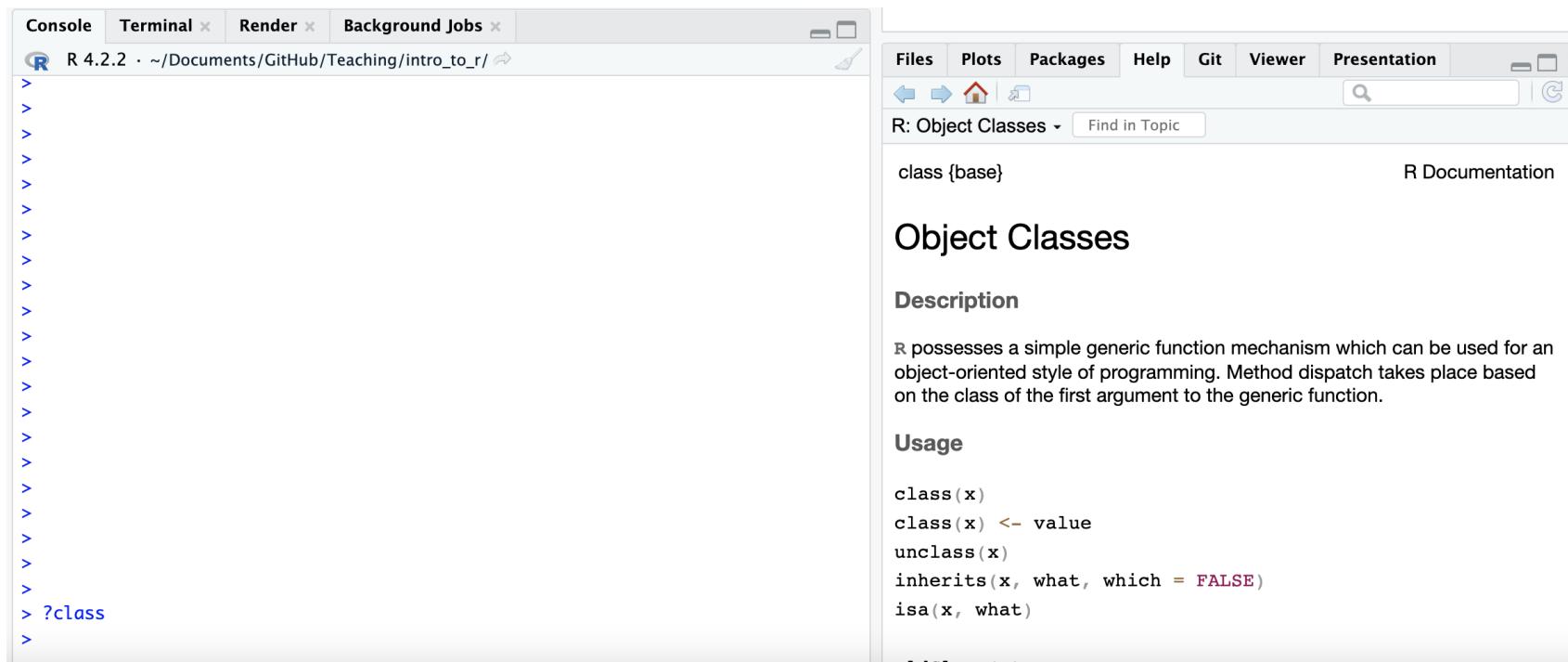
```
class(x)
class(x) <- value
unclass(x)
inherits(x, what, which = FALSE)
isa(x, what)
oldClass(x)
```

# Getting Help with ?

If you know the name of a package or function:

Type `?package_name` or `?function_name` in the console to get information about packages and functions.

For example: ?readr or ?read\_csv.



# Double Question Mark

If you haven't loaded a package yet into R than you may get a response that there is no documentation.

Typing in `??package_name` can show you packages that you haven't loaded yet.

The screenshot shows the RStudio interface. On the left, the Console tab is active, displaying R code and its output. The user has run several commands related to the tidyverse package, including `?class`, `?tidyverse`, and `??tidyverse`. The output for `??tidyverse` shows that no documentation was found in the specified packages and libraries, but suggests trying `??tidyverse`. The user then loads the tidyverse package using `library(tidyverse)`. The output shows the packages attached and conflicts resolved. On the right, the Help tab is active, showing the documentation for the tidyverse package. The title is "tidyverse: Easily Install and Load the 'Tidyverse'". The description explains that the tidyverse is a set of packages designed to work in harmony, sharing common data representations and API design. It includes a link to the tidyverse website (<https://www.tidyverse.org>). The maintainer is listed as Hadley Wickham ([hadley@rstudio.com](mailto:hadley@rstudio.com)). A "tidyverse" logo, which is a dark hexagon with numerous small colored dots, is displayed. Other contributors are also mentioned.

```
>
>
>
>
>
> ?class
> ?tidyverse
No documentation for 'tidyverse' in specified packages and libraries:
you could try '??tidyverse'
> ??tidyverse
> library(tidyverse)
— Attaching packages ——————— tidyverse 1.3.2 —
✓ ggplot2 3.4.0   ✓ dplyr  1.0.10
✓ tibble  3.1.8   ✓ stringr 1.5.0
✓ tidyr   1.2.0   ✓forcats 0.5.1
✓ purrr   1.0.0
— Conflicts ——————— tidyverse_conflicts() —
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
> ?tidyverse
> |
```

R: tidyverse: Easily Install and Load the 'Tidyverse'

tidyverse-package {tidyverse} R Documentation

## tidyverse: Easily Install and Load the 'Tidyverse'

### Description

The 'tidyverse' is a set of packages that work in harmony because they share common data representations and 'API' design. This package is designed to make it easy to install and load multiple 'tidyverse' packages in a single step. Learn more about the 'tidyverse' at <https://www.tidyverse.org>.

### Author(s)

Maintainer: Hadley Wickham [hadley@rstudio.com](mailto:hadley@rstudio.com)

Other contributors:



## Summary

- RStudio makes working in R easier
- the Editor (top) is for static code like scripts or R Markdown documents
- The console is for testing code (bottom) - best to save your code though!
- R markdown documents are really helpful for lots of reasons!
- R code goes within what is called a chunk (the gray box with a green play button)
- Code chunks can be modified so that they show differently in reports

[Class Website](#)

[Lab](#)



Image by [Gerd Altmann](#) from [Pixabay](#)