

Intro to R

RStudio

Help! Office hours

Poll: What times are best for you for office hours?

Today is different

Ava will hold office hours from 5:00pm - 6:00pm EST.

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

Working with R – RStudio

RStudio is an Integrated Development Environment (IDE) for R

- It helps the user effectively use R
- Makes things easier
- Is NOT a dropdown statistical tool (such as Stata)
 - See [Rcmdr](#) or [Radiant](#)
- All R Studio snapshots are taken from <http://ayeimanol-r.net/2013/04/21/289/>



[[source](#)]

RStudio

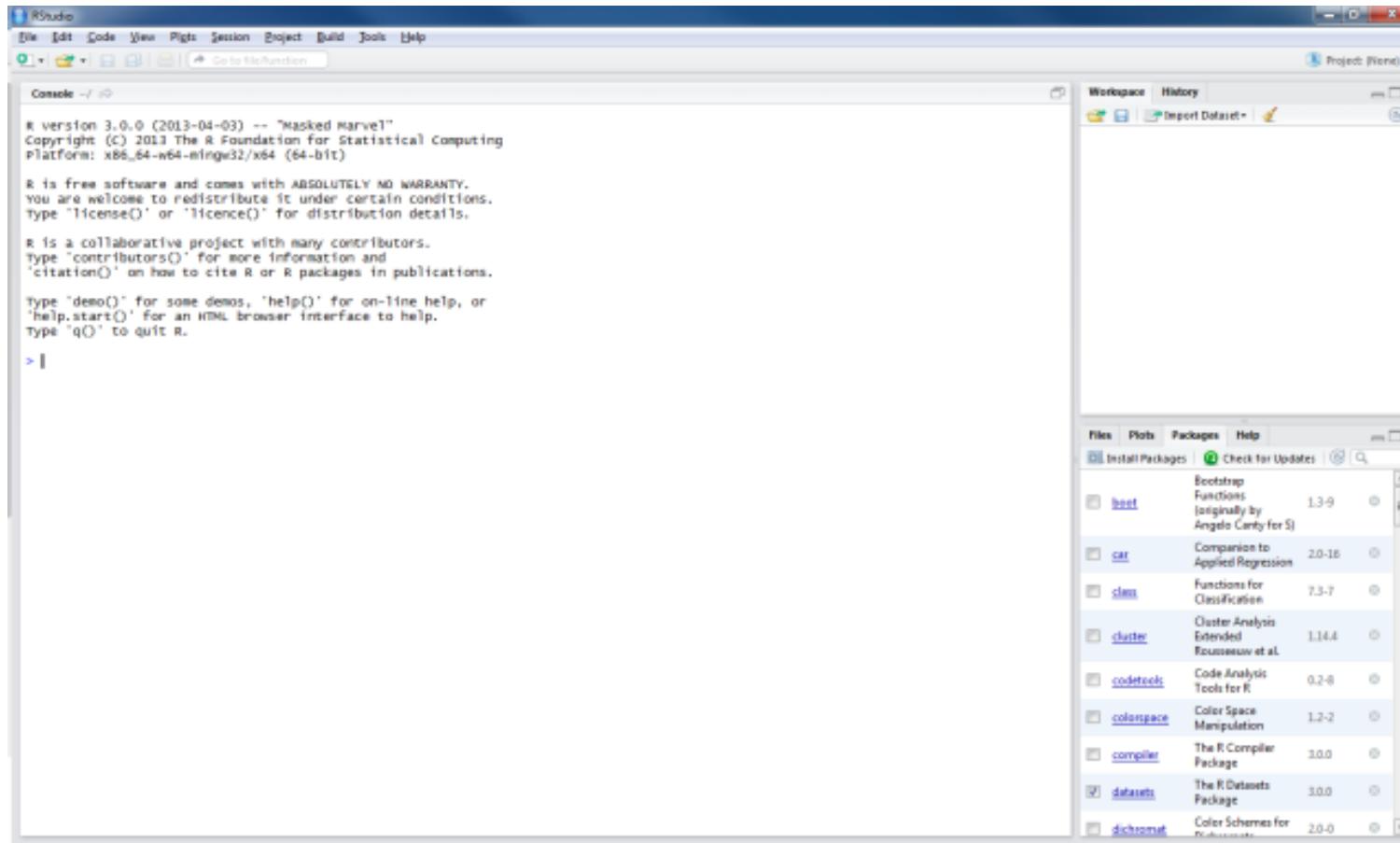
Easier working with R

- 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
- Searchable command history

RStudio



Getting the editor

R version 4.0.3 (2020-10-10) -- "Bunny-Munnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin17.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

Only console

```
> |
```

R Script **Open** ps

R Notebook
Create a new R
R Markdown
Shiny Web App...
Plumber API...

Text File
C++ File
Python Script
SQL Script
Stan File
D3 Script
R Sweave
R HTML
R Presentation
R Documentation...

R-10) -- "Bunny-Munnies Freak Out"
R Foundation for Statistical Computing
darwin17.0 (64-bit)

comes with ABSOLUTELY NO WARRANTY.
distribute it under certain conditions.
.license() for distribution details.

port but running in an English locale

oject with many contributors.
or more information and
cite R or R packages in publications.

demos, 'help()' for on-line help, or
HTML browser interface to help.

```
> |
```

Untitled1

```
1 |
```

Editor

1:1 (Top Level) R Script

Console Terminal Jobs

~/Documents/GitHub/Teaching/intro_to_r/

Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

Console

```
> |
```

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

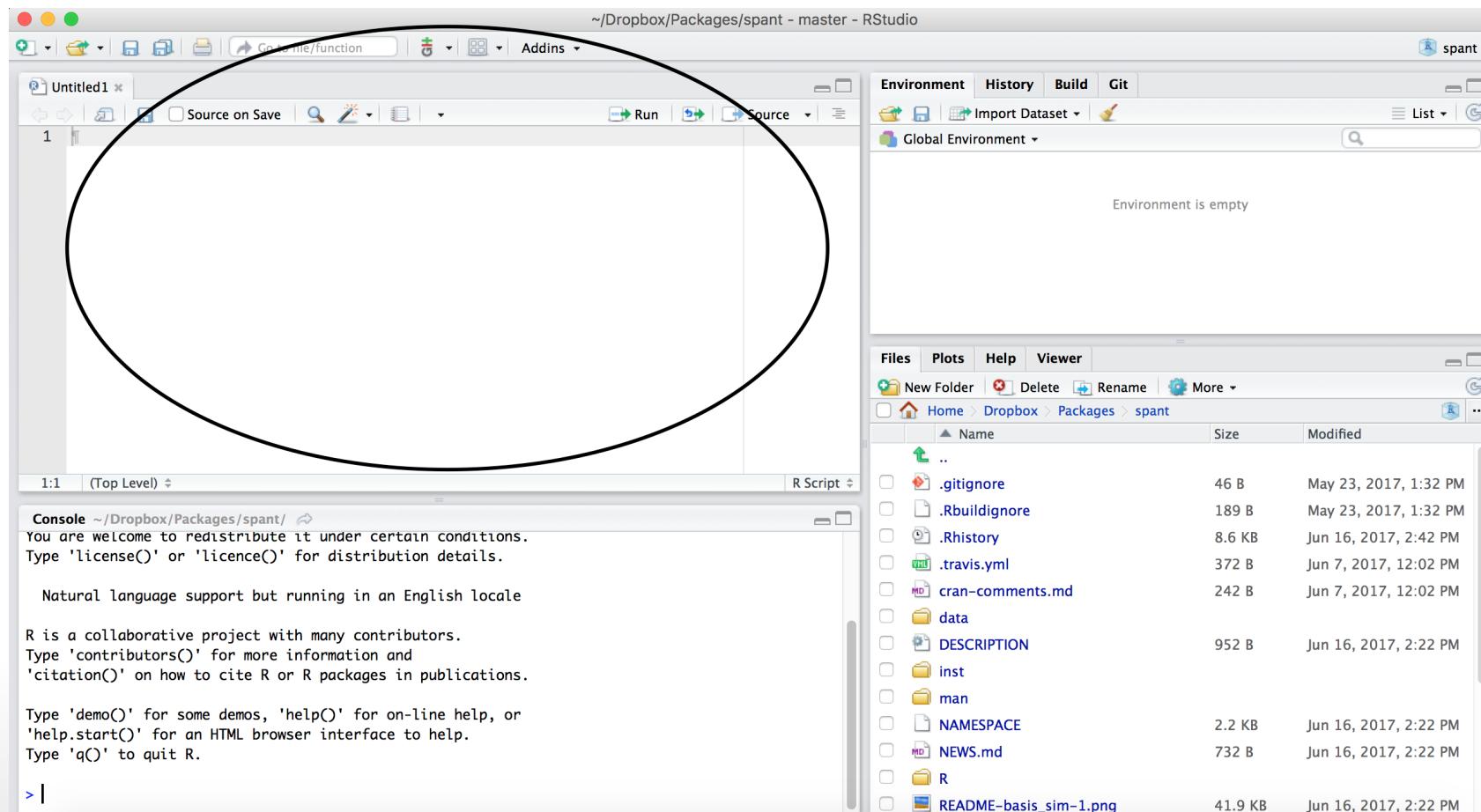
2. The **R Console**: "interprets" whatever you type

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

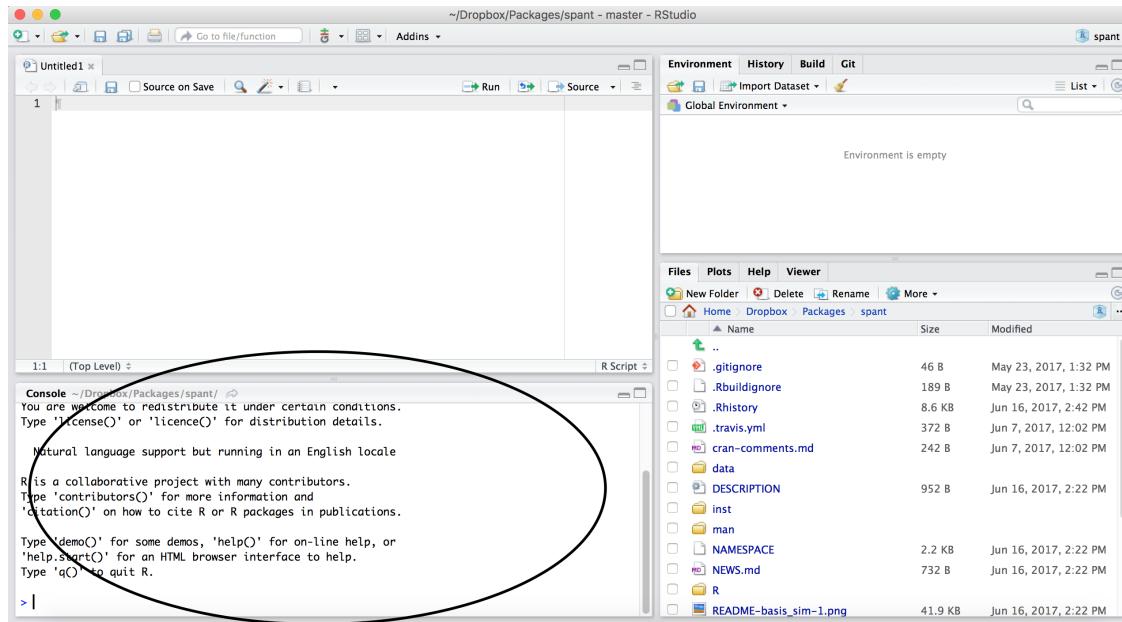
Source / Editor

- 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

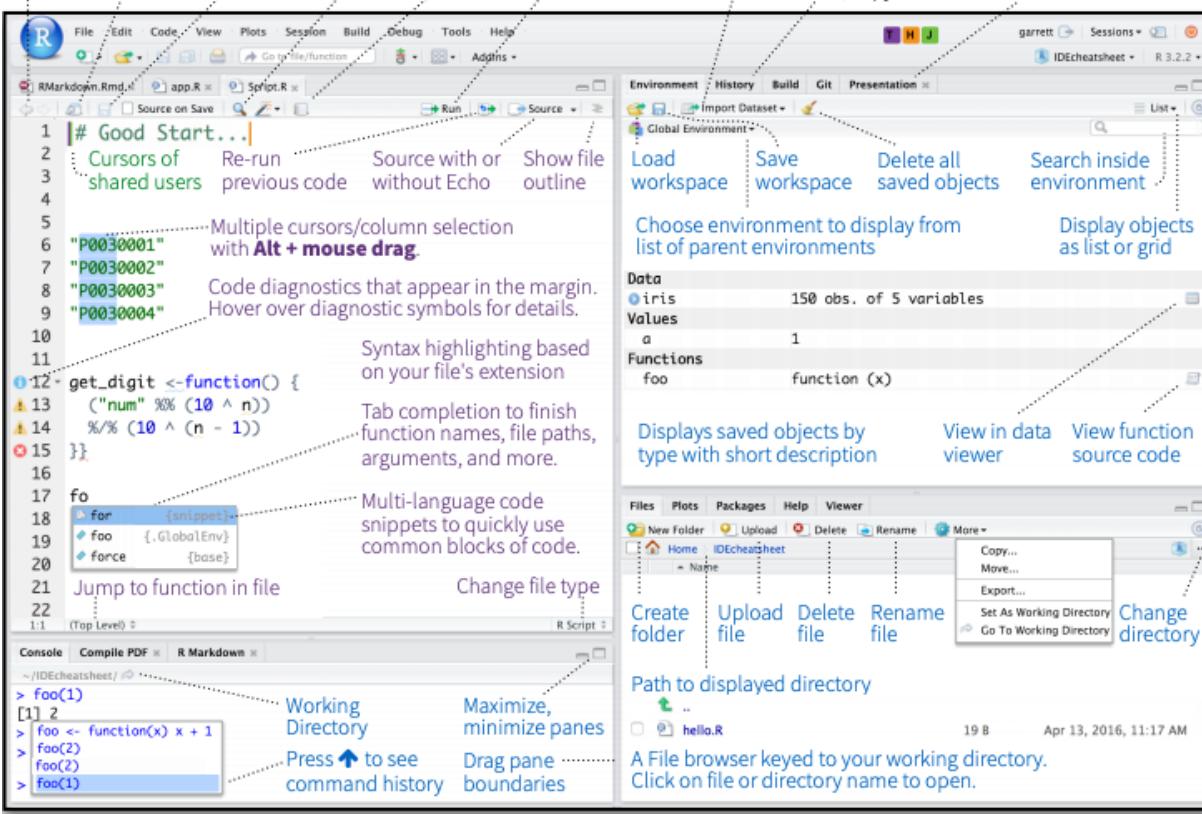
RStudio

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

RStudio layout

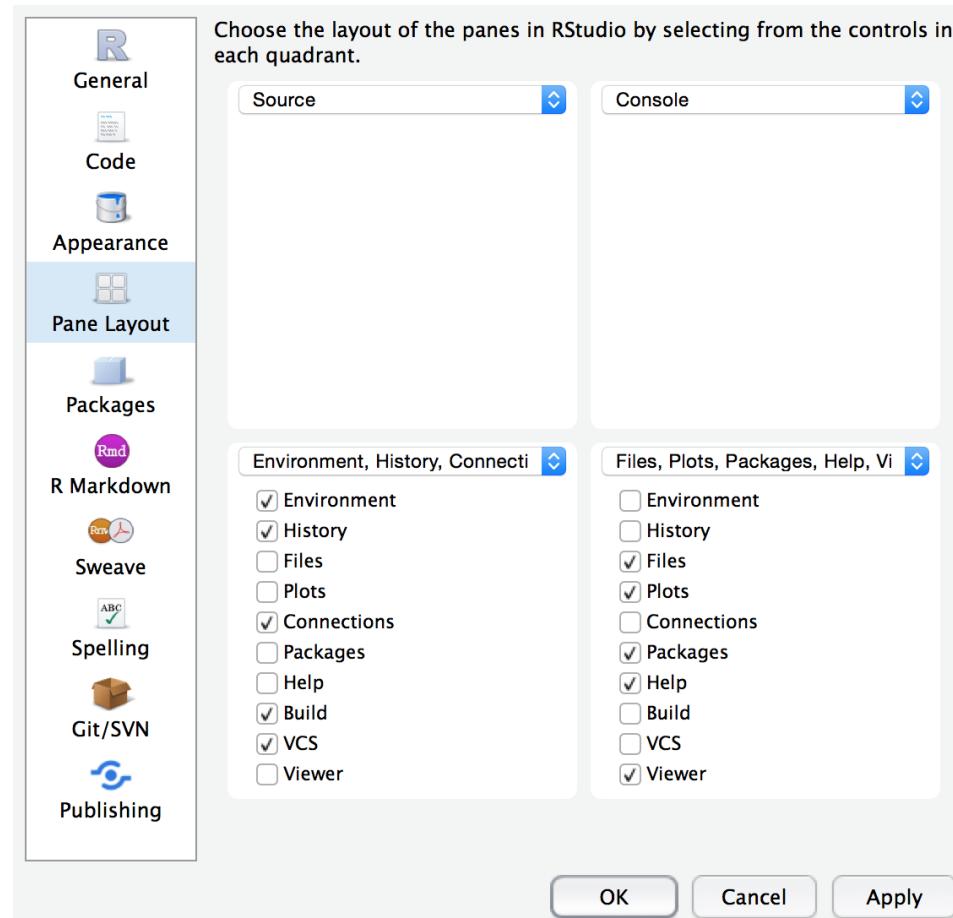
The screenshot displays the RStudio interface with the following components:

- Code Editor (Left Panel):** Shows the R Markdown file "Untitled1.Rmd". The code includes setup code for knitr, followed by a section titled "# R Markdown" which describes R Markdown syntax and provides a link to <http://rmarkdown.rstudio.com>. It also shows an example of embedding an R code chunk.
- Environment (Top Right Panel):** Shows the Global Environment pane, which is currently empty.
- File Browser (Bottom Right Panel):** Shows the file structure for the "intro_to_r" directory. The contents include .gitignore, .Rbuildignore, .Rhistory, .travis.yml, all_functions.xlsx, all_the_functions.csv, all_the_packages.txt, Arrays_Split, Basic_R, Best_Model_Coefficients.csv, Best_Model_Coefficients.xlsx, bibliography.bib, black_and_white_theme.pdf, and bloomberg_logo_small_horizontal.png.
- Console (Bottom Left Panel):** Displays the R console output, including the R welcome message, license information, natural language support, and R's collaborative nature. It also shows the command to run demos and how to quit R.

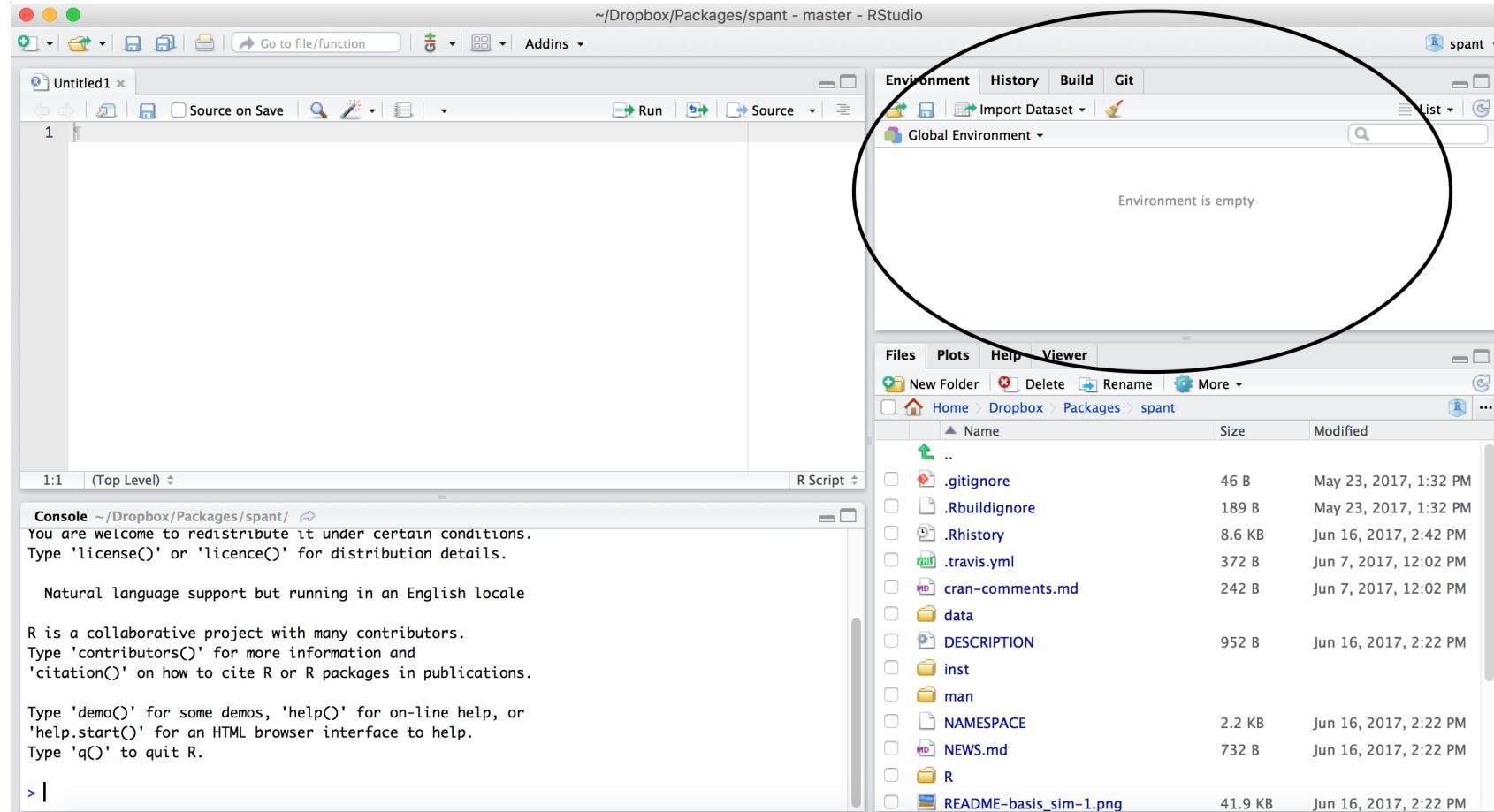
RStudio Layout

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

RStudio → Preferences → Pane Layout



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:

1. Download the file at the link above by clicking on the link or typing in:
https://jhudatascience.org/intro_to_r/modules/RStudio/lab/RStudio_Lab.Rmd

(Also on the [website](#) schedule page - Lab for day 1) 2) Find the downloaded file on your computer 3) Open the file in RStudio

This may require finding your downloads on your computer.

Recall that these videos can help:

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

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

R Markdown file

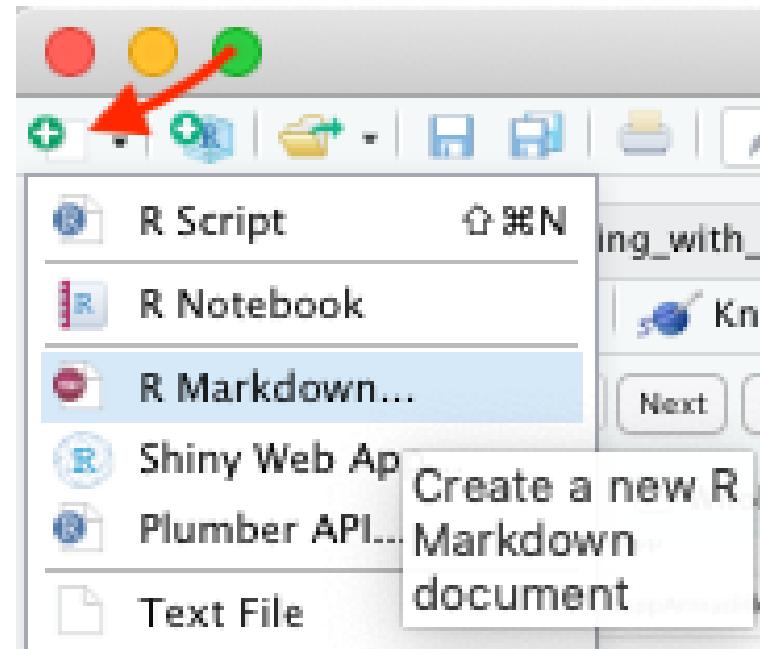
R Markdown files (.Rmd) help generate reports that include your code and output. Think of them as fancier scripts.

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

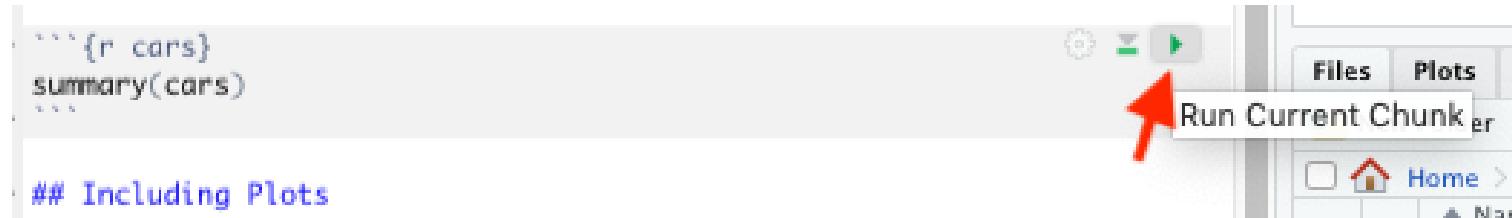
Call your file “first_markdown”



Code chunks

Within R Markdown files are code “chunks”

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



A screenshot of the RStudio interface. On the left, there is a code editor window containing R code:

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

## Including Plots
```

On the right, the RStudio toolbar is visible, featuring several icons. A red arrow points to the icon for "Run Current Chunk", which is a green square with a white play button symbol.

Create Chunks

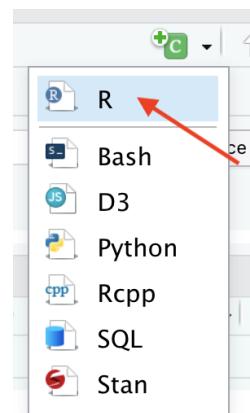
To create a new R code chunk:

Copy paste an existing chunk in the R Markdown file and replace the code **OR**

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

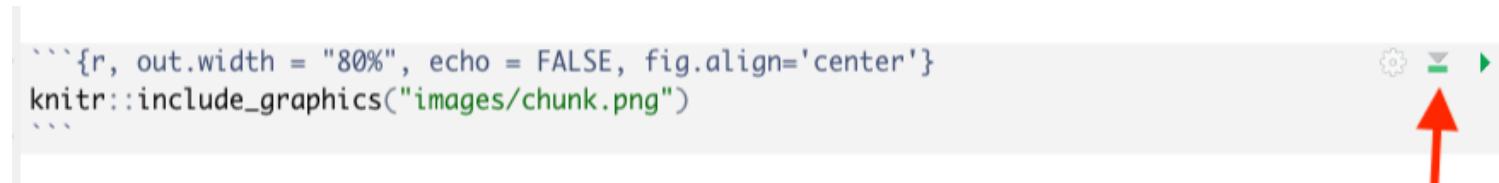


1. Select R (default) as the language:

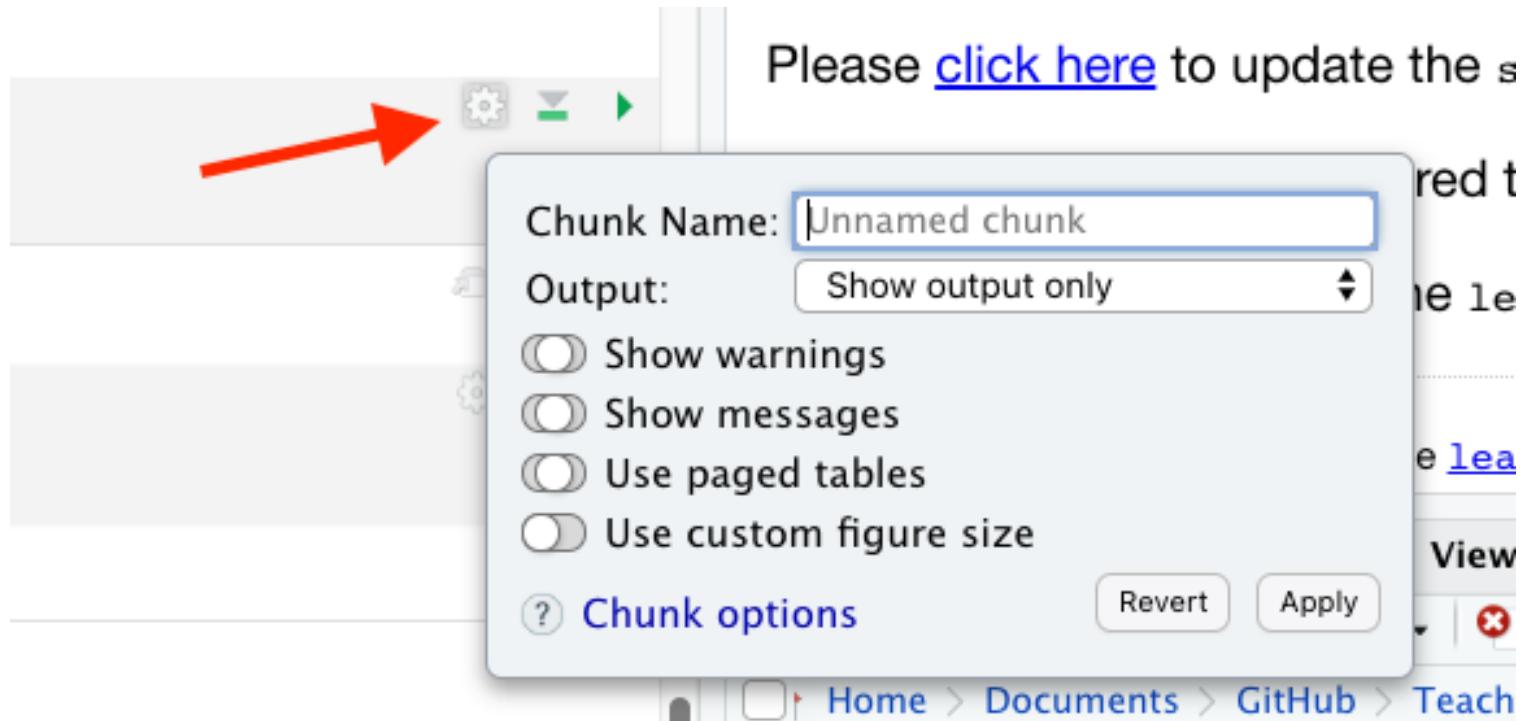


Run previous chunks button

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

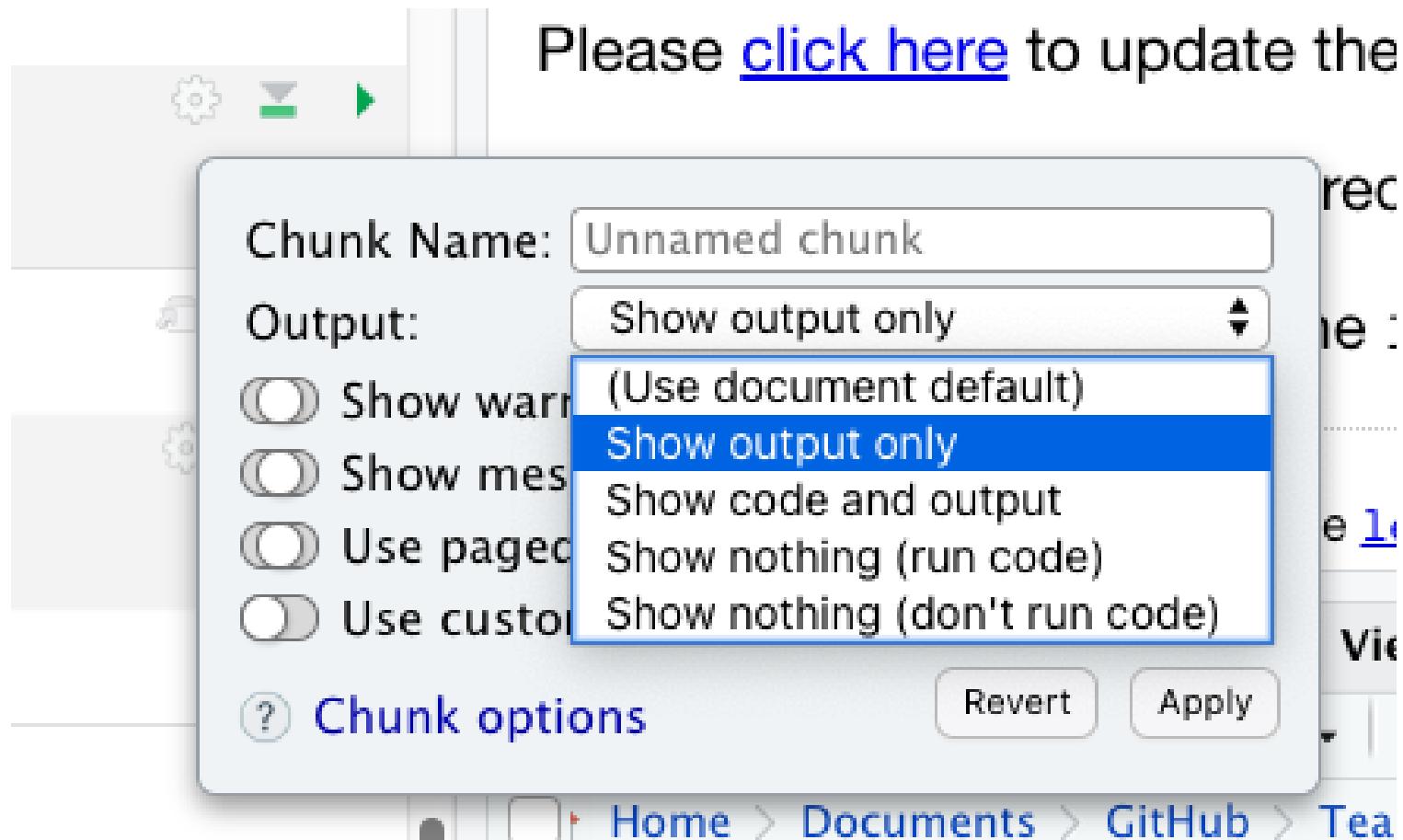


Chunk settings



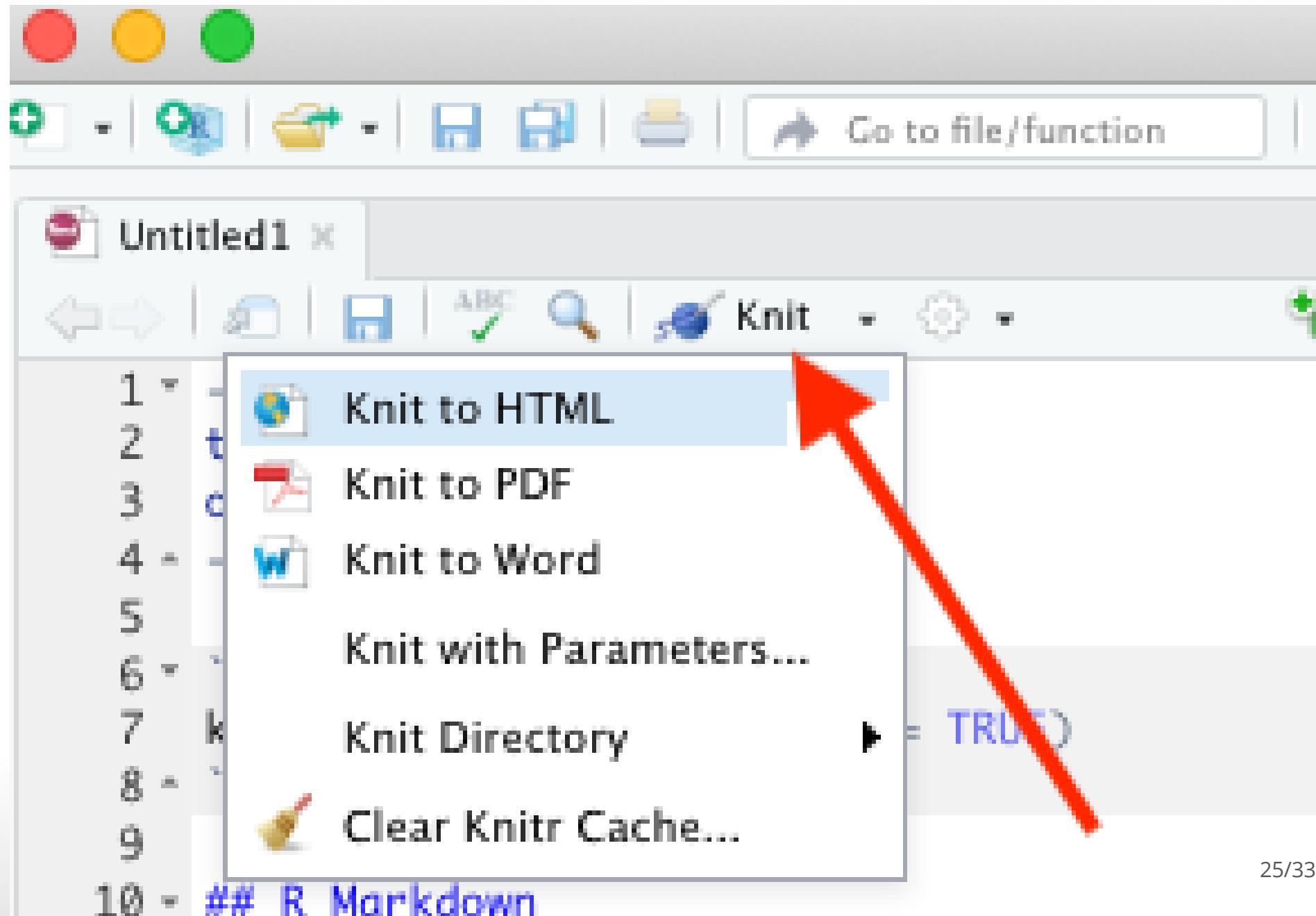
Chunk settings

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



Knit file to html

This will create a report from the R Markdown document!



Useful R Studio Shortcuts

- **Ctrl + Enter** 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** takes you to the script page
- **Ctrl+2** takes you to the console
- http://www.rstudio.com/ide/docs/using/keyboard_shortcuts

If you get annoyed by inline 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 has tabs at the top: Basic (selected), Advanced, Visual, and Citations. Under the 'R Markdown' section, there are several configuration options:

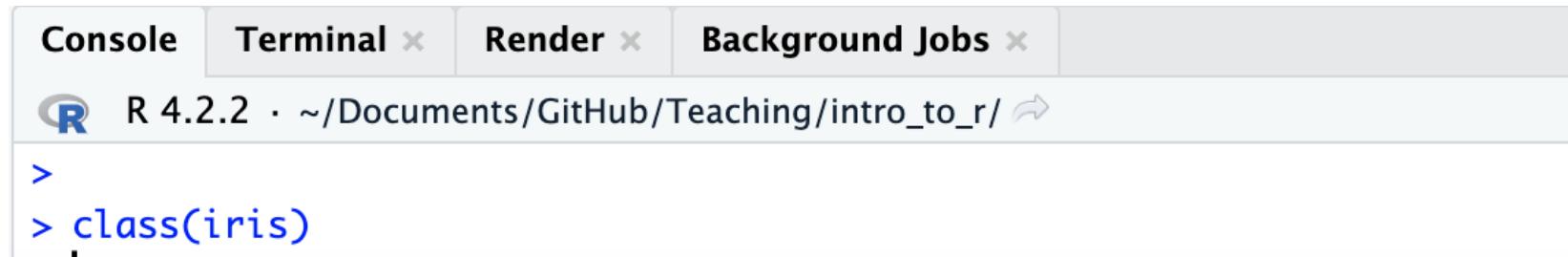
- Show document outline by default
- Soft-wrap R Markdown files
- Show in document outline:
- Show output preview in:
- Show output inline for all R Markdown documents (this option is highlighted with a red border)
- Show equation and image previews:
- Evaluate chunks in directory:

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

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

Recap of where code goes

- you can test code in the console



R 4.2.2 · ~/Documents/GitHub/Teaching/intro_to_r/ ↗

```
>
> class(iris)
.
```

- 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.
Press F1 for additional help

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(),
 ...)
Press F1 for additional help

Get help with the help pane



The screenshot shows the RStudio interface with the 'Help' tab selected in the top navigation bar. Below the navigation 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 text '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`.

The screenshot shows the RStudio interface with two main panes. The left pane is the Console, displaying the command `?class` entered by the user. The right pane is the Help viewer, showing the documentation for the `Object Classes` in R. The title bar of the Help viewer includes tabs for Files, Plots, Packages, Help, Git, Viewer, and Presentation. The main content area displays the `class` function under the heading `Object Classes`. The `Description` section states: "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." The `Usage` section lists the following functions: `class(x)`, `class(x) <- value`, `unclass(x)`, `inherits(x, what, which = FALSE)`, and `isa(x, what)`.

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 R console window displays the following session history:

```
>  
>  
>  
>  
>  
>  
> ?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  
> |
```

The right side of the interface is the help viewer for the `tidyverse` package. The title bar says "R: tidyverse: Easily Install and Load the 'Tidyverse'". The main content area shows the package documentation for `tidyverse-package`. It includes a section titled "tidyverse: Easily Install and Load the 'Tidyverse'" with a brief description of what the tidyverse is and how it works. Below that is a "Description" section. To the right of the text is a logo for "tidyverse" featuring a dark hexagon with numerous small colored dots. At the bottom of the help viewer, it lists the maintainer as Hadley Wickham (`hadley@rstudio.com`) and other contributors.

Summary

- RStudio makes working in R easier
- the Editor is for static code like scripts or R Markdown documents
- The console is for testing code
- 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](#)