

# Data I/O

Data Wrangling in R

# R Basics

## Explaining output on slides

In slides, a command (we'll also call them code or a code chunk) will look like this

```
head(mtcars)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

And then directly after it, will be the output of the code.

These slides were made in R using `knitr` and R Markdown (covered later today when we discuss reproducible research)

# R variables

A few reminders: \* You can create variables from within the R environment and from files on your computer \* Use "<-" to assign values to a variable name \* Variable names are case-sensitive, i.e. X and x are different

```
x <- 2  
x
```

```
[1] 2
```

```
x * 4
```

```
[1] 8
```

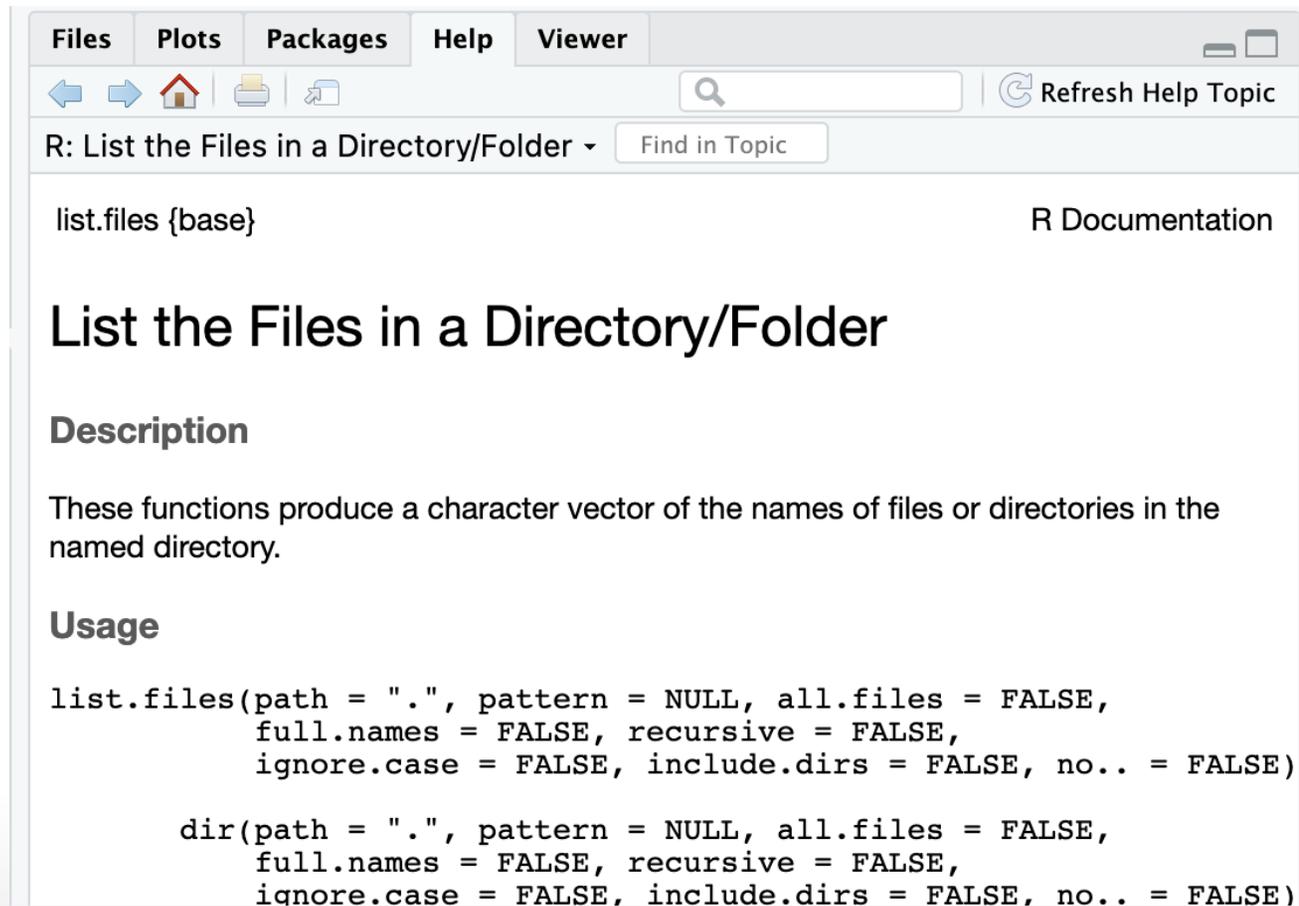
```
x + 2
```

```
[1] 4
```

# Help

For any function, you can write `?FUNCTION_NAME`, or `help("FUNCTION_NAME")` to look at the help file:

```
?dir  
help("dir")
```



The screenshot shows the R help viewer window. The title bar includes tabs for Files, Plots, Packages, Help, and Viewer. Below the title bar is a navigation bar with back, forward, home, print, and refresh icons, a search box, and a 'Refresh Help Topic' button. The main content area displays the help page for 'list.files {base}'. The page title is 'R: List the Files in a Directory/Folder'. The main heading is 'List the Files in a Directory/Folder'. Below the heading is the 'Description' section, which states: 'These functions produce a character vector of the names of files or directories in the named directory.' The 'Usage' section shows the function signatures for 'list.files' and 'dir' with their default arguments.

```
list.files {base} R Documentation
```

## List the Files in a Directory/Folder

### Description

These functions produce a character vector of the names of files or directories in the named directory.

### Usage

```
list.files(path = ".", pattern = NULL, all.files = FALSE,  
           full.names = FALSE, recursive = FALSE,  
           ignore.case = FALSE, include.dirs = FALSE, no.. = FALSE)  
  
dir(path = ".", pattern = NULL, all.files = FALSE,  
    full.names = FALSE, recursive = FALSE,  
    ignore.case = FALSE, include.dirs = FALSE, no.. = FALSE)
```

# Packages

Not all packages are available by default.

```
install.packages("tidyverse")  
library(tidyverse)
```

```
install.packages("light")
```



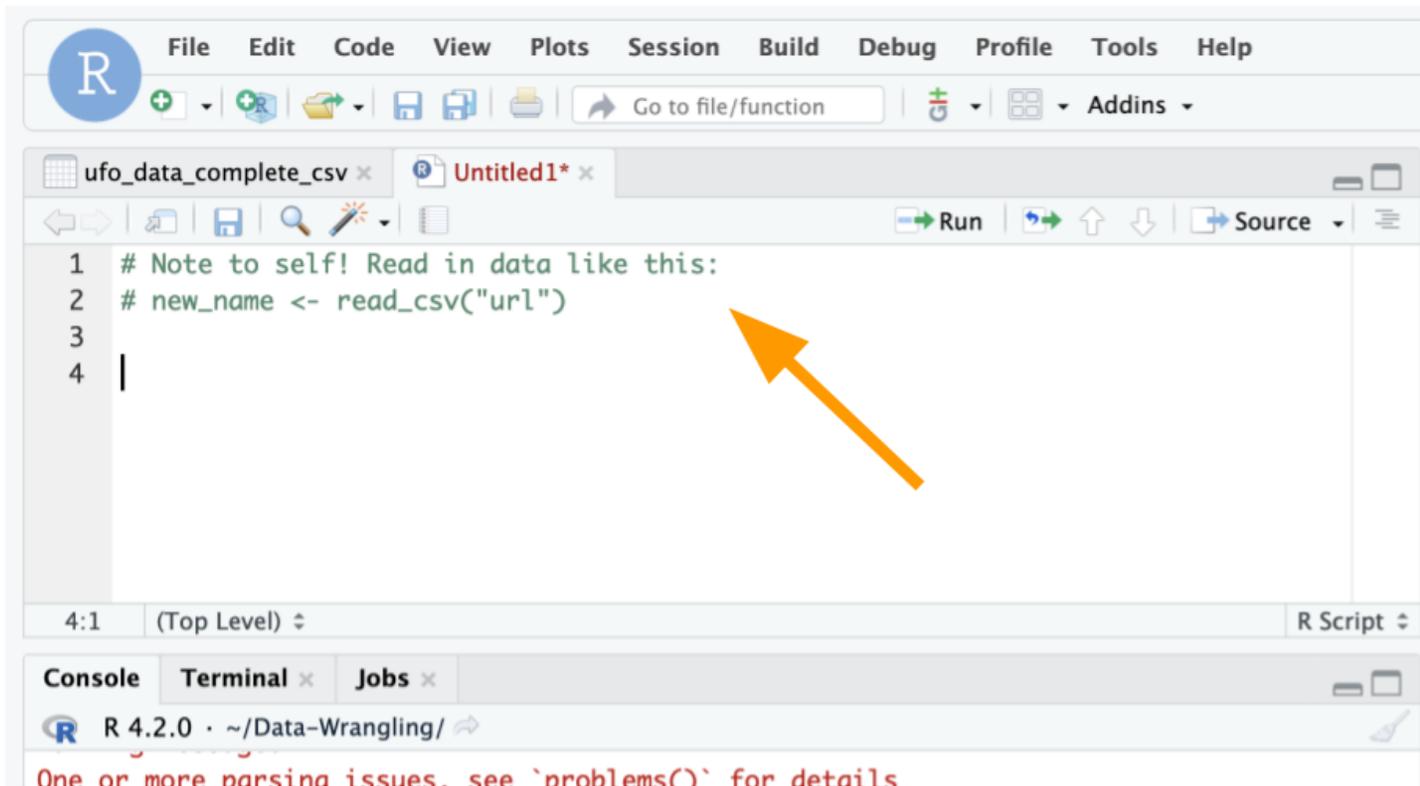
```
library("light")
```



Images sourced from <https://www.wikihow.com/Change-a-Light-Bulb>

# Commenting in Scripts

Commenting in code is super important. You should be able to go back to your code years after writing it and figure out exactly what the script is doing. Commenting helps you do this. Also handy for notes!



```
1 # Note to self! Read in data like this:
2 # new_name <- read_csv("url")
3
4 |
```

The screenshot shows the RStudio interface. The menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The toolbar contains icons for file operations and a search bar. The editor window shows a script with four lines of code. Line 1 is a comment: `# Note to self! Read in data like this:`. Line 2 is another comment: `# new_name <- read_csv("url")`. Lines 3 and 4 are blank. An orange arrow points to the comment on line 2. The status bar at the bottom shows the current cursor position as 4:1 (Top Level) and the file type as R Script. The console window at the bottom shows the R version 4.2.0 and a warning message: `One or more parsing issues. see `problems()` for details`.

# Commenting in Scripts



avahoffman Add code to save discarded outliers in a csv

1 contributor

127 lines (108 sloc) | 4.16 KB

```
1 # Search for outliers among biomass subplots in preparation for the rest of the analysis
2 #####
3 library(dplyr)
4 library(ggplot2)
5 library(cowplot)
6
7 # Useful information here: http://r-statistics.co/Outlier-Treatment-With-R.html
8 #####
9
10 make_outlier_plot <-
11   function(d) {
12     # This function will test for chi-square scores that are outside the
13     # percentile cutoff, and color them blue.
14     # For best results, use only on a specific site-category-treatment subset
15     # Probably best for viz only!!
16     ggplot() +
17       geom_point(aes(
18         x = as.numeric(rownames(d)),
19         y = d$biomass
```

**Data Input**

# Outline

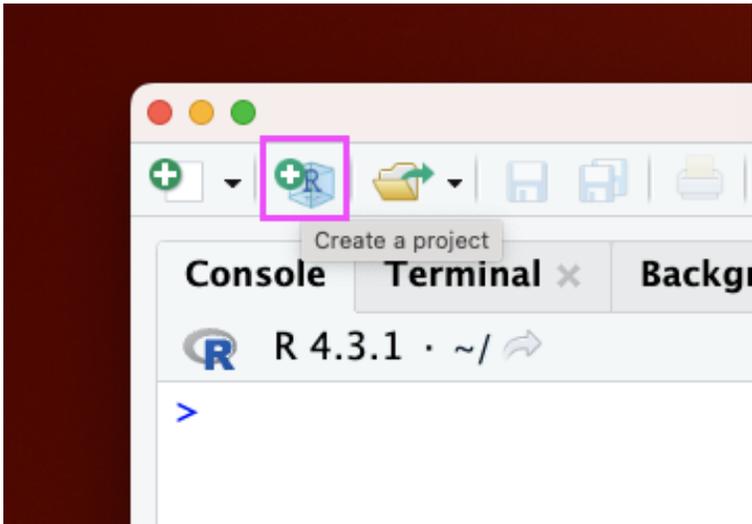
- Part 0: A little bit of set up!
- Part 1: reading in manually (point and click)
- Part 2: reading in directly & working directories
- Part 3: checking data & multiple file formats

# Part 0: Setup - R Project

# New R Project

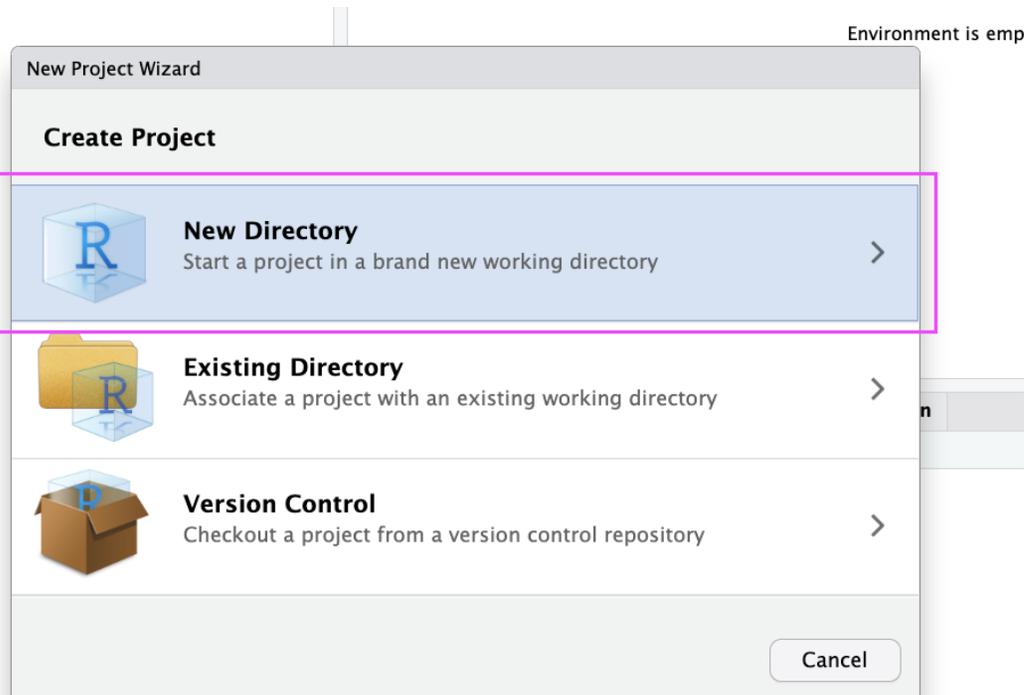
Let's make an R Project so we can stay organized in the next steps.

Click the new R Project button at the top left of RStudio:



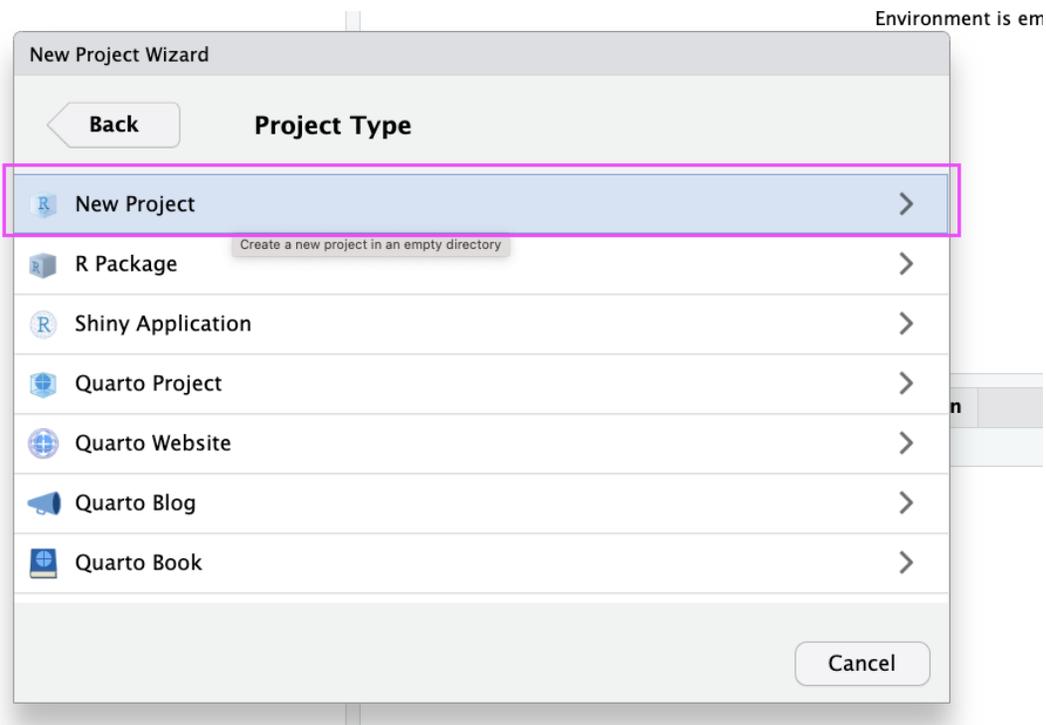
# New R Project

In the New Project Wizard, click “New Directory”:



# New R Project

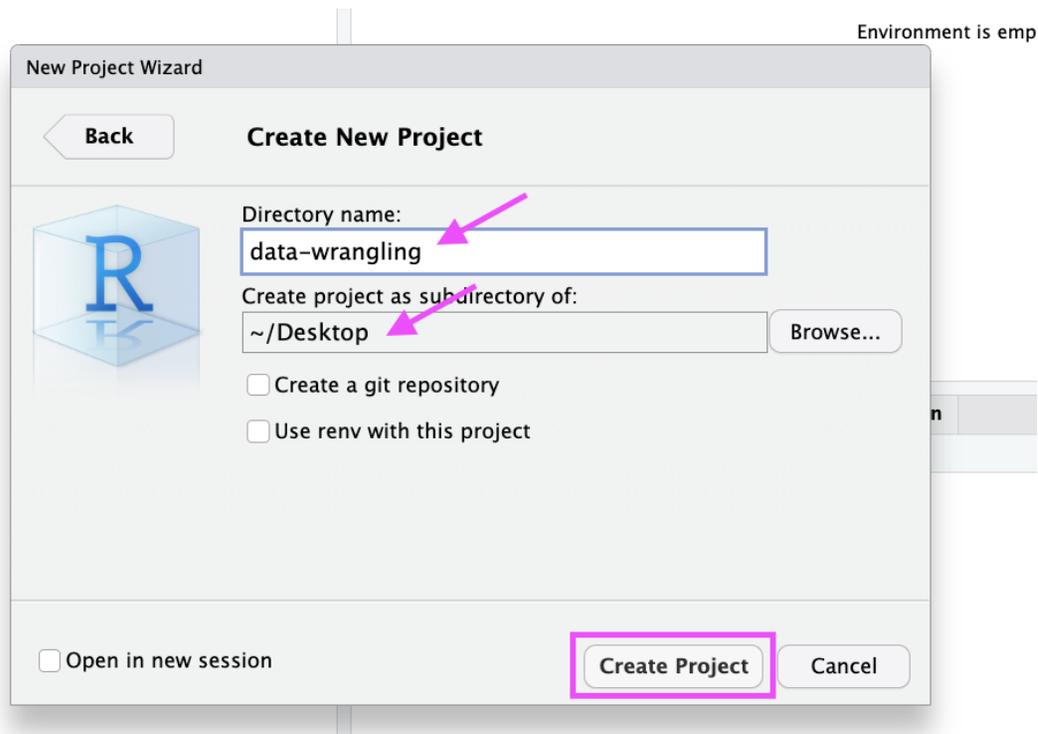
Click “New Project”:



# New R Project

Type in a name for your new folder.

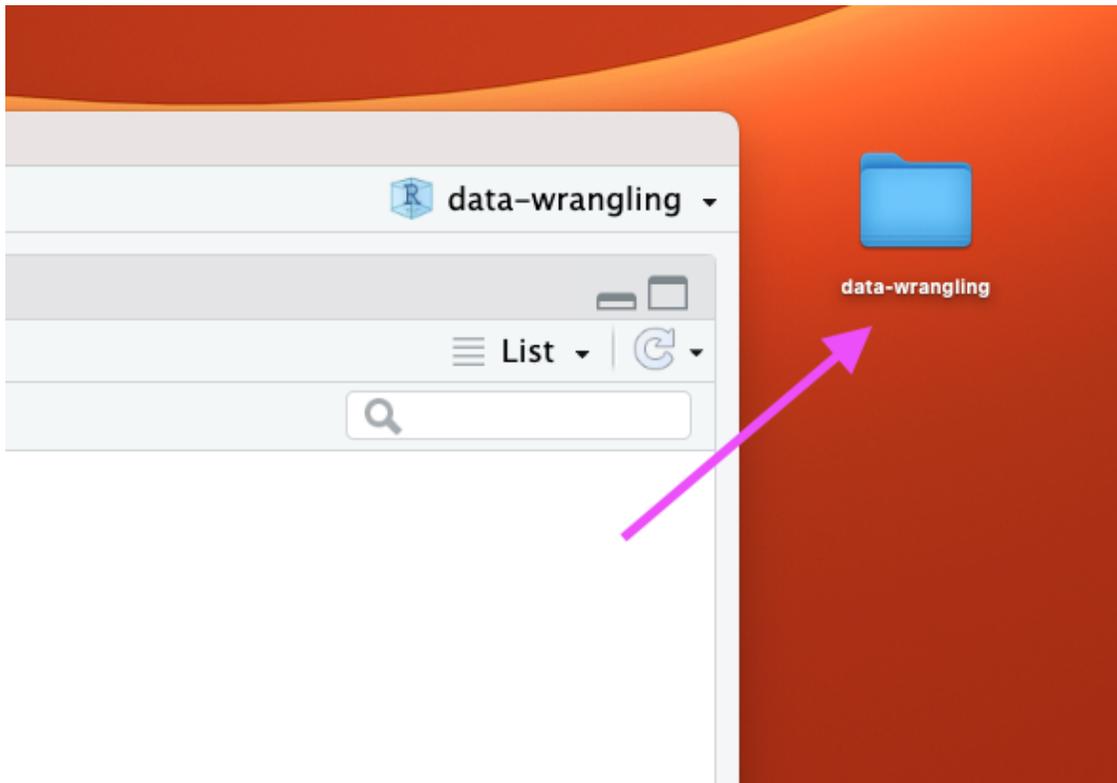
Store it somewhere easy to find, such as your Desktop:



# New R Project

You now have a new R Project folder on your Desktop!

Make sure you add any scripts or data files to this folder as we go through today's lesson. This will make sure R is able to "find" your files.



# Part 1: Getting data into R (manual/point and click)

# Data Input

- 'Reading in' data is the first step of any real project/analysis
- R can read almost any file format, especially via add-on packages
- We are going to focus on simple delimited files first
  - comma separated (e.g. '.csv')
  - tab delimited (e.g. '.txt')
  - Microsoft Excel (e.g. '.xlsx')

# Data Input

Youth Tobacco Survey (YTS) dataset:

“The YTS was developed to provide states with comprehensive data on both middle school and high school students regarding tobacco use, exposure to environmental tobacco smoke, smoking cessation, school curriculum, minors’ ability to purchase or otherwise obtain tobacco products, knowledge and attitudes about tobacco, and familiarity with pro-tobacco and anti-tobacco media messages.”

- Check out the data at: <https://catalog.data.gov/dataset/youth-tobacco-survey-yts-data>

## Data Input: Dataset Location

Dataset is located at [https://sisbid.github.io/Data-Wrangling/data/Youth\\_Tobacco\\_Survey\\_YTS\\_Data.csv](https://sisbid.github.io/Data-Wrangling/data/Youth_Tobacco_Survey_YTS_Data.csv)

- Download data by clicking the above link
  - Safari - if a file loads in your browser, choose File → Save As, select, Format “Page Source” and save

# Import Dataset

- > File
- > Import Dataset
- > From Text (readr)
- > paste the url ([https://sisbid.github.io/Data-Wrangling/data/Youth\\_Tobacco\\_Survey\\_YTS\\_Data.csv](https://sisbid.github.io/Data-Wrangling/data/Youth_Tobacco_Survey_YTS_Data.csv))
- > click "Update" and "Import"

## What Just Happened?

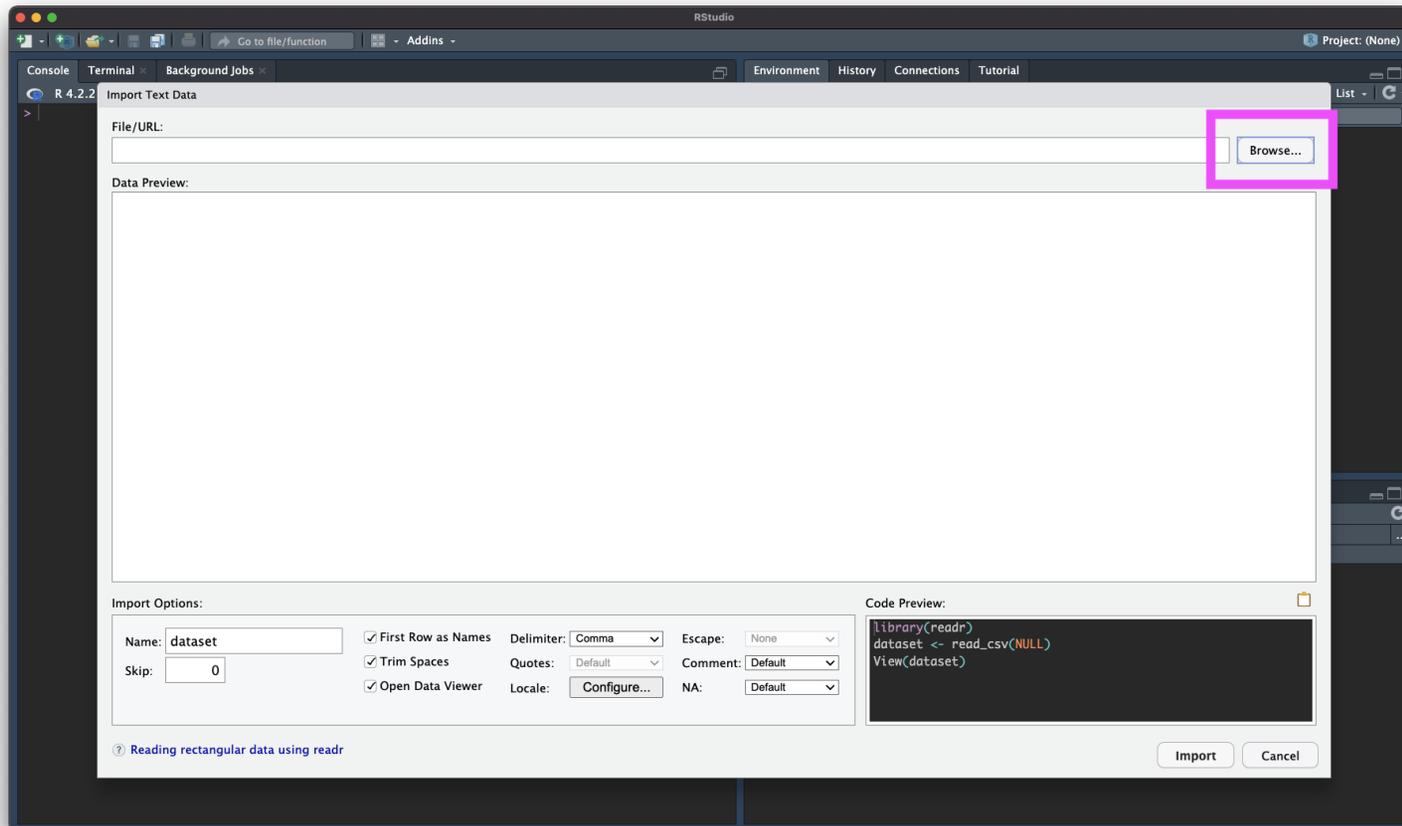
- You see a preview of the data on the top left pane.
- You see a new object called `Youth_Tobacco_Survey_YTS_Data` in your environment pane (top right). The table button opens the data for you to view.
- R ran some code in the console (bottom left).

# Import Dataset (recap)

The screenshot shows the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Window, and Help. The main workspace is divided into several panes:

- Console:** Shows the R prompt `>` and the R version `R 4.1.2`.
- Environment:** Shows the current environment is empty, with a memory usage of 549 MiB.
- Viewer:** Displays R documentation for the `read_delim` function. The title is "Read a delimited file (including csv & tsv) into a tibble". The description states: "read\_csv() and read\_tsv() are special cases of the general read\_delim(). They're useful for reading the most common types of flat file data, comma separated values and tab separated values, respectively. read\_csv2() uses ; for the field separator and , for the decimal point. This is common in some European countries."

# Browsing for Data on Your Machine



## Example 2: Delimiters

- > File
- > Import Dataset
- > From Text (`readr`)
- > paste the url (<https://sisbid.github.io/Data-Wrangling/data/dropouts.txt>)
- > select delimiter
- > click “Update” and “Import”

## Example 3: Excel

```
library(readxl)
```

- > File
- > Import Dataset
- > From Excel
- > paste the url (<https://sisbid.github.io/Data-Wrangling/data/asthma.xlsx>)
- > click "Update" and "Import"

# Manual Import: Pros and Cons

Pros: easy!!

Cons: obscures some of what's happening, others will have difficulty running your code

## Summary & Lab

Review the process: <https://youtu.be/LEkNfJgpunQ>

- > File
- > Import Dataset
- > From Text (`readr`) / From Excel
- > paste the url / browse
- > click "Update" and "Import"

<https://sisbid.github.io/Data-Wrangling/labs/data-io-lab-part1.Rmd>