

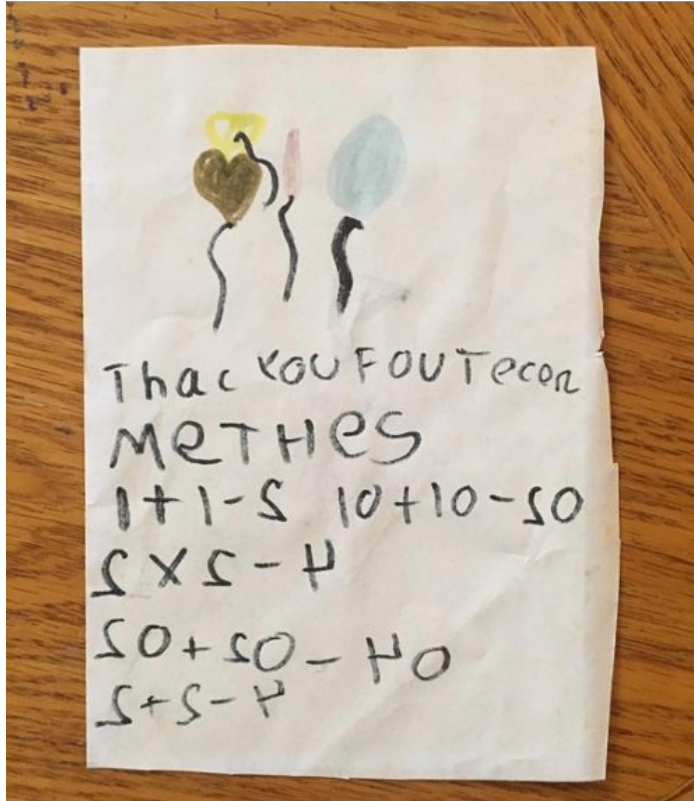
# Data Science Basics in R

Day 1: Introduction to Statistical Programming

# Introductions

- Your name
- What you do for school, work, and/or fun
- Why you signed up for this course
- Have you ever used R before?

# Introductions



# Housekeeping notes

- Take a break whenever you need one, we will also have a few structured breaks as a larger group
- Outlet locations
- Trash cans
- Try to come with a charged laptop
- If you have questions, you can find me at *sde31@georgetown.edu*

## Housekeeping notes

All course materials are available on github, and we'll talk more about github in general later in this course.

**<https://github.com/seaneff/data-science-basics-2024>**

## What to expect: Learning R

- Learning R is fun! And also frustrating.
- You won't be an expert by the end of this week.
- But over time and as you practice, it gets easier!

# What to expect: The next week

- We'll balance slides/demos with hands-on exercises.
- You decide how you learn best...
  - listening with your computer away
  - laptop out and typing along
  - taking notes with paper/pen
- All of these materials are publicly accessible on github.

## Workshop goals

This workshop will build literacy and basic proficiency in statistical programming, with a focus on the skills needed to conduct data analyses in professional healthcare and public health workplaces.

We will cover the basics of data management, data cleaning, data visualization, and basic statistical calculations in R, and version control in github. Participants will leave with a small portfolio of relevant data visualizations and analyses completed using a real-world public health dataset.



# Workshop goals

- Learning to program (in R) can be fun and creative, and doesn't have to be overwhelming or intimidating.
- Anyone can learn to write code.

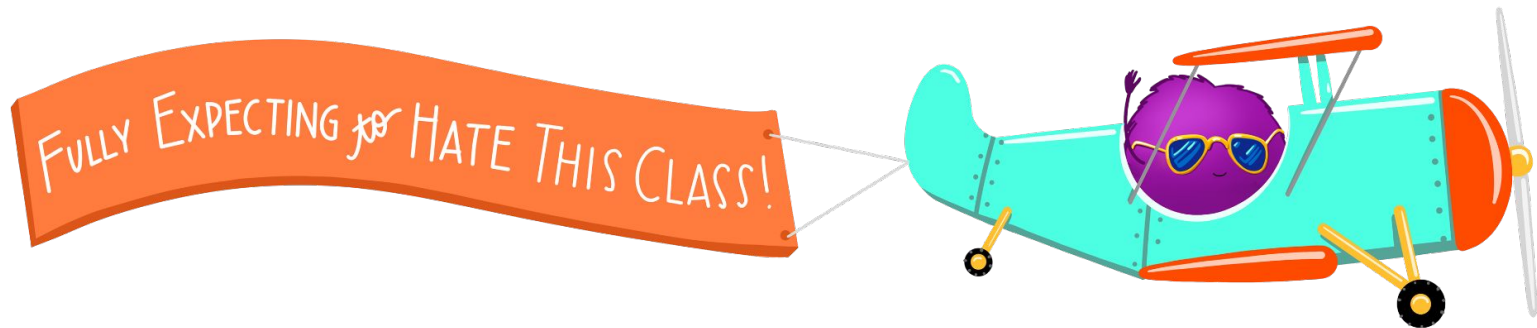


# Learning by doing

For some people, it's easier to learn by doing, typing, and making mistakes. Others prefer to listen, think, and work through problems later on their own.

In this workshop, we'll pause to do worked examples. Sometimes these will be confusing. This is the point! We will learn together by trial and error.

If you are more comfortable following along for now, feel free to just watch and try at home. But I really encourage you to try, the best way to learn R is to repeatedly do stuff wrong and then figure out the errors.



@allison\_horst

Artwork by Allison Horst

# Goals for today

- Understand what statistical programming is
- Get acquainted with Rstudio
- Write your very first R code (at least, of this workshop)
  - vectors
  - functions
  - accessing documentation
- Explore github to access course materials

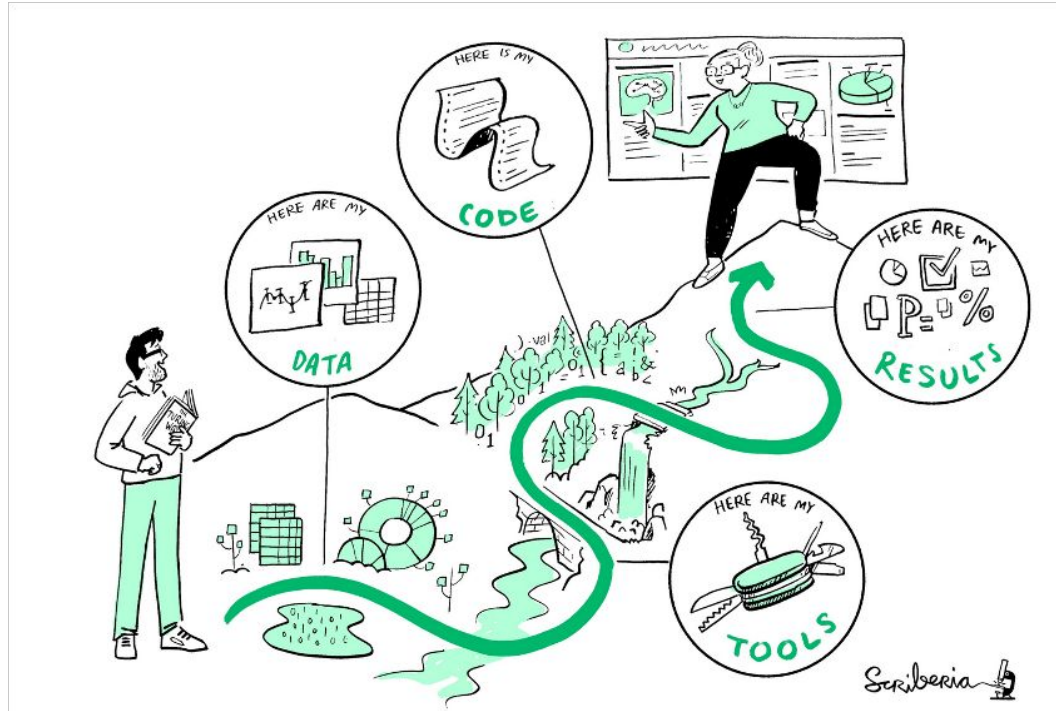
# Goals for today

- Understand what statistical programming is
- Get acquainted with Rstudio
- Write your very first R code (at least, of this workshop)
  - vectors
  - functions
  - accessing documentation
- Explore github to access course materials

**What is statistical  
programming, and why  
should I care?**

# What is statistical programming?

Statistical programming is using code to clean, analyze, visualize, and interpret data.



# What is R?

- R is a programming language for statistical computing
- Created by Ross Ihaka and Robert Gentleman in 1996
- R is open-source and free
- Many people use R in different ways and for different purposes, but it's defined specifically for data analysis and visualization (unlike other open-source languages like python)





# What is RStudio?

- R Studio is, put simply, a place to write and run R code
- It's an IDE (integrated development environment) and supports both R and python
- It's also free (with enterprise upgrades)



# Why learn R?

- Learning R helps you understand your data and understand how analysis works, whether you're a researcher, a data scientist, or someone who collaborates with folks who do analysis
- Coding helps you think rigorously about your questions
- It's free (vs. other more expensive tools like SAS or SPSS)
- Sharable, reproducible code and research
- Lots of academics/companies/agencies use it
- It's fun (honestly)

# Goals for today

- Understand what statistical programming is
- Get acquainted with Rstudio
- Write your very first R code (at least, of this workshop)
  - vectors
  - functions
  - accessing documentation
- Explore github to access course materials

# How do I use RStudio?

# R Studio console

The screenshot displays the RStudio environment with the following components:

- Script Editor:** Contains R code for reading and summarizing data. The code includes comments and a `summary()` call.
- Environment:** Shows the `Global Environment` with a data frame `countries` containing 194 observations and 9 variables.
- Console:** Displays the output of the script execution, showing the data structure and the results of the `summary()` function.
- Documentation Panel:** Shows the R documentation for the `duplicated()` function, including a description and usage examples.

```
## NAPHIS country data - one row per WHO member state
## Data as of
countries <- read.delim("countries.tsv")

#####
## Summarize data: #####
## Printed summary #####
#####

## print summary info, globally
countries %>%
  summarize(total_member_states = sum(who_member_state == TRUE),
            completed_jees = sum(completed_jees == TRUE),
            completed_naphs = sum(completed_naphs == TRUE),
            published_naphs = sum(published_naphs == TRUE, na.rm = TRUE),
            published_naphs_data = sum(naphs_includes_line_item_costs == TRUE, na.rm = TRUE),
            machine_readable_data = sum(naphs_data_machine_readable == TRUE, na.rm = TRUE))

#####
## Global funnel: Manuscript barplot #####
#####
## end of script - end of script - end of script
```

```
R 4.1.2 ~/Documents/work/CT/NAPHIS-data/global-summary/
> object 'line_items' not found
> ## NAPHIS country data - one row per WHO member state
> ## Data as of
> countries <- read.delim("countries.tsv")
> ## print summary info, globally
> countries %>%
+   summarize(total_member_states = sum(who_member_state == TRUE),
+             completed_jees = sum(completed_jees == TRUE),
+             completed_naphs = sum(completed_naphs == TRUE),
+             published_naphs = sum(published_naphs == TRUE, na.rm = TRUE),
+             published_naphs_data = sum(naphs_includes_line_item_costs == TRUE, na.rm = TRUE),
+             machine_readable_data = sum(naphs_data_machine_readable == TRUE, na.rm = TRUE))
+ total_member_states completed_jees completed_naphs published_naphs published_naphs_data machine_readable_data
1          194          103           77           14           9           0
```

**Determine Duplicate Elements**

**Description**

`duplicated()` determines which elements of a vector or data frame are duplicates of elements with smaller subscripts, and returns a logical vector indicating which elements (rows) are duplicates.

`anyDuplicated()` is a "generalized" more efficient version `any(duplicated())`, returning positive integer indices instead of just `TRUE`.

**Usage**

```
duplicated(x, incomparables = FALSE, ...)
```

**Default S3 method:**

```
duplicated(x, incomparables = FALSE,
           fromLast = FALSE, nmax = NA, ...)
```

**S3 method for class 'array'**

```
duplicated(x, incomparables = FALSE, MARGIN = 1,
           fromLast = FALSE, ...)
```

# R Studio console

Write R scripts to re-use (.R files)

```
#####  
22  
23  
24 ## NAPHS country data - one row per WHO member state  
25 ## Data as  
26 countries <- read.delim("countries.tsv")  
27  
28 #####  
29  
30  
31  
32  
33 #####  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

Environment History Connections Tutorial  
R - Global Environment -  
Data  
countries 194 obs. of 9 variables

Files Plots Packages Help Viewer Presentation  
R: Determine Duplicate Elements - Find in Topic  
duplicated (base) R Documentation  
Determine Duplicate Elements  
Description  
duplicated() determines which elements of a vector or data frame are duplicates of elements with smaller subscripts, and returns a logical vector indicating which elements (rows) are duplicates.  
anyDuplicated() is a "generalized" more efficient version any(duplicated(.)), returning positive integer indices instead of just TRUE.  
Usage  
duplicated(x, incomparables = FALSE, ...)  
## Default S3 method:  
duplicated(x, incomparables = FALSE, fromLast = FALSE, nmax = NA, ...)  
## S3 method for class 'array'  
duplicated(x, incomparables = FALSE, MARGIN = 1, fromLast = FALSE, ...)

# R Studio console

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for loading and summarizing data. The code includes comments and function calls like `read.delim()` and `summarize()`.
- Run Button:** A pink circle highlights the 'Run' button in the top toolbar, with the text "This button runs code" next to it.
- Console:** A pink circle highlights the console window, with the text "Run one-off code here, or see the results of code you ran from above" next to it. The console shows the output of the code, including the number of rows and columns for each variable.
- Environment:** Shows the 'Global Environment' with a data frame 'countries' containing 194 observations and 9 variables.
- Help Window:** Displays the documentation for the `deduplicated()` function, including a description and usage examples.

```
## NAPHS country data - one row per WHO member state
## Data as of
countries <- read.delim("countries.tsv")

#####
## Summarize data: #####
## Printed summary #####
#####

## print summary info, globally
countries %>%
  summarize(total_member_states = sum(who_member_state == TRUE),
            completed_jee = sum(completed_jee == TRUE),
            completed_naphs = sum(completed_naphs == TRUE),
            published_naphs = sum(published_naphs == TRUE, na.rm = TRUE),
            published_naphs_data = sum(naphs_includes_line_item_costs == TRUE, na.rm = TRUE),
            machine_readable_data = sum(naphs_data_machine_readable == TRUE, na.rm = TRUE))

#####
## Global funnel: Manuscript barplot #####
#####
## end of NAPHS data summary #####
```

Run one-off code here, or see the results of code you ran from above

Environment: Global Environment (226 MB)

Data: countries (194 obs. of 9 variables)

Files Plots Packages Help Viewer Presentation

R: Determine Duplicate Elements - Find in Topic

deduplicated (base) R Documentation

### Determine Duplicate Elements

**Description**

`deduplicated()` determines which elements of a vector or data frame are duplicates of elements with smaller subscripts, and returns a logical vector indicating which elements (rows) are duplicates.

`anyDuplicated()` is a "generalized" more efficient version `any(duplicated())`, returning positive integer indices instead of just TRUE.

**Usage**

```
deduplicated(x, incomparables = FALSE, ...)
```

**Default S3 method:**

```
deduplicated(x, incomparables = FALSE,
             fromLast = FALSE, nmax = NA, ...)
```

**S3 method for class 'array'**

```
deduplicated(x, incomparables = FALSE, MARGIN = 1,
             fromLast = FALSE, ...)
```

# R Studio console

The screenshot shows the RStudio interface with the following components:

- Script Editor:** Contains R code for reading and summarizing data. The code includes comments and function calls like `read.delim()` and `summarize()`.
- Console:** Shows the output of the R code, including the execution of `countries <- read.delim("countries.tsv")` and the resulting summary statistics.
- Environment Pane:** Located on the right, it shows the objects in the current environment. A pink oval highlights this pane with the text "Stuff you have in memory". It displays the object `countries` with 194 observations and 9 variables.
- Help Pane:** At the bottom right, it shows the documentation for the `duplicate()` function, including its description and usage.

```
## NAPHS country data - one row per WHO member state
## Data as of
countries <- read.delim("countries.tsv")

## Summarize data: #####
## Printed summary #####
#####

## print summary info, globally
countries %>%
  summarize(total_member_states = sum(who_member_state == TRUE),
            completed_jee = sum(completed_jee == TRUE),
            completed_naphs = sum(completed_naphs == TRUE),
            published_naphs = sum(published_naphs == TRUE, na.rm = TRUE),
            published_naphs_data = sum(naphs_includes_line_item_costs == TRUE, na.rm = TRUE),
            machine_readable_data = sum(naphs_data_machine_readable == TRUE, na.rm = TRUE))

## Global funnel: Manuscript barplot #####
## end of script #####
```

Environment

countries 194 obs. of 9 variables

Stuff you have in memory

R: Determine Duplicate Elements - Find in Topic

Determine Duplicate Elements

Description

`duplicate()` determines which elements of a vector or data frame are duplicates of elements with smaller subscripts, and returns a logical vector indicating which elements (rows) are duplicates.

`anyDuplicated()` is a "generalized" more efficient version `any(duplicate())`, returning positive integer indices instead of just TRUE.

Usage

```
duplicate(x, incomparables = FALSE, ...)
```

## Default S3 method:

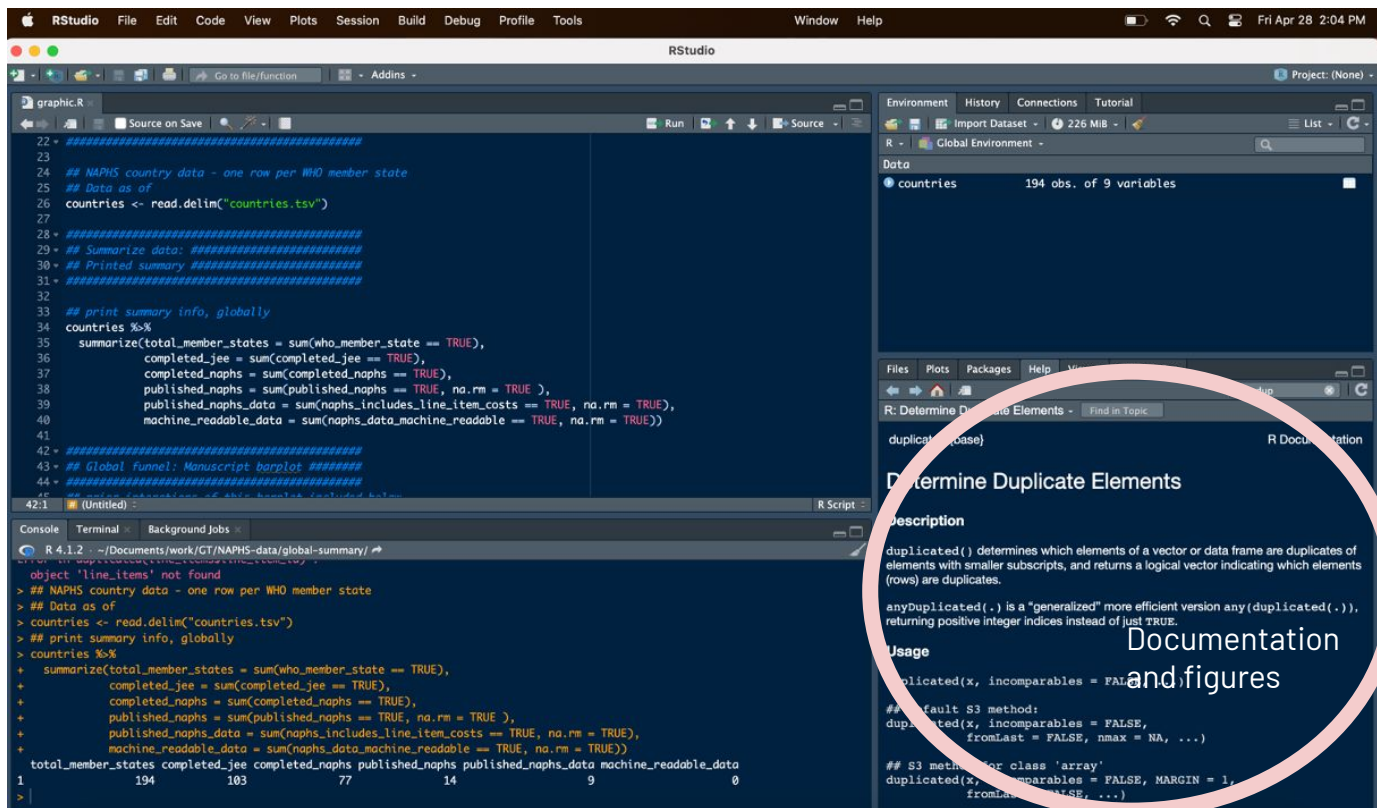
```
duplicate(x, incomparables = FALSE,
          fromLast = FALSE, nmax = NA, ...)
```

## S3 method for class 'array'

```
duplicate(x, incomparables = FALSE, MARGIN = 1,
          fromLast = FALSE, ...)
```



# R Studio console



The screenshot displays the RStudio environment with the following components:

- Script Editor:** Contains R code for reading and summarizing data from a file named "countries.tsv". The code includes comments and a summary function that calculates various statistics.
- Console:** Shows the output of the R script, including the message "object 'line\_items' not found" and the results of the summary function.
- Environment Pane:** Displays the current environment, showing a variable named "countries" with 194 observations and 9 variables.
- Documentation Pane:** Shows the documentation for the `duplicate()` function, including a description and usage examples. A pink circle highlights the text "Documentation and figures" in the documentation pane.

```
## NAPHIS country data - one row per WHO member state
## Data as of
countries <- read.delim("countries.tsv")

## print summary info, globally
countries %>%
  summarize(total_member_states = sum(who_member_state == TRUE),
            completed_jees = sum(completed_jees == TRUE),
            completed_naphs = sum(completed_naphs == TRUE),
            published_naphs = sum(published_naphs == TRUE, na.rm = TRUE),
            published_naphs_data = sum(naphs_includes_line_item_costs == TRUE, na.rm = TRUE),
            machine_readable_data = sum(naphs_data_machine_readable == TRUE, na.rm = TRUE))

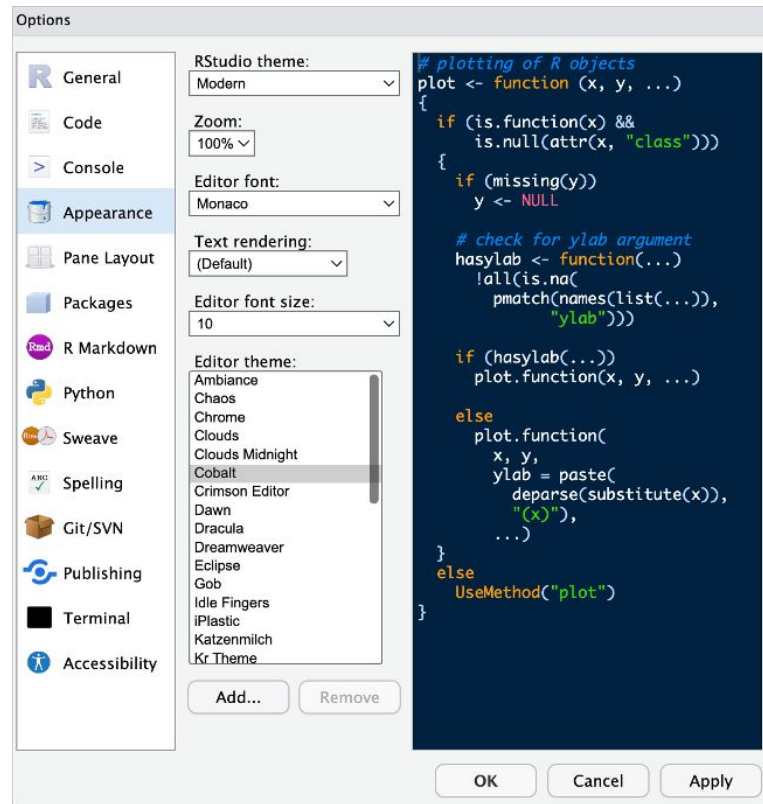
## Global funnel: Manuscript barplot
## Data as of
countries %>%
  summarize(total_member_states = sum(who_member_state == TRUE),
            completed_jees = sum(completed_jees == TRUE),
            completed_naphs = sum(completed_naphs == TRUE),
            published_naphs = sum(published_naphs == TRUE, na.rm = TRUE),
            published_naphs_data = sum(naphs_includes_line_item_costs == TRUE, na.rm = TRUE),
            machine_readable_data = sum(naphs_data_machine_readable == TRUE, na.rm = TRUE))

total_member_states completed_jees completed_naphs published_naphs published_naphs_data machine_readable_data
1 194 103 77 14 9 0
```

**Documentation and figures**

# Changing RStudio's appearance

- Navigate from the top bar
- -> Tools
- -> Global Options
- -> Appearance
- (then click "apply")



## **15 minute break (and a note on worked examples)**

- If you haven't already, try to download R and Rstudio before tomorrow's class. I'll be around by Zoom or email if you have any questions, and can help troubleshoot.
- Today, we'll do some worked examples sharing my laptop. If you already have R and Rstudio installed on your laptop, feel free to follow along there.

# Goals for today

- Understand what statistical programming is
- Get acquainted with Rstudio
- Write your very first R code (at least, of this workshop)
  - vectors
  - functions
  - accessing documentation
- Explore github to access course materials

# R Basics

# R as a calculator

- R can do everything a basic calculator can do
- Using R as a calculator is a great first step



Comic by Jessica Wang. Accessed online: <https://i.redd.it/dmayt2tc3e551.jpg>

# Using R as a calculator

```
1 + 1
```

```
## [1] 2
```

```
8 - 10
```

```
## [1] -2
```

```
(32198 + 8943289)/12
```

```
## [1] 747957.2
```

# Using R as a calculator

---

```
log(1)
```

```
## [1] 0
```

```
sqrt(64)
```

```
## [1] 8
```

```
abs(-14)
```

```
## [1] 14
```

---



# Using R as a calculator

## Symbols and syntax

- **Addition** ( $1+1$ )
- **Subtraction** ( $2-1$ )
- **Multiplication** ( $3*4$ )
- **Division** ( $7.2/9$ )
- **Exponents** ( $2^7$ )
- **Square root** (`sqrt(9)`)
- **Order of operations** ( $7/(3*2)$ )

## Now you try!

- Use R to do some basic math
  - Add two numbers together
  - Multiply three or more numbers
  - Take the square root of a number

# Objects

- An object is something you save to R's working memory
- It can be almost anything
  - A string (e.g., your name)
  - A number (e.g., 3.14)
  - A dataset (e.g., that file you have in Excel)
- We assign objects using a little arrow with the syntax (`<-`)
- When doing data analysis, the most common object you'll probably save is a dataframe, like an Excel or .csv file that you can access from within R (more on this later)

# Objects

```
my_first_object <- 3.14
```

```
my_first_object
```

```
## [1] 3.14
```

```
my_second_object <- 1+1
```

```
my_second_object
```

```
## [1] 2
```

# Objects

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for a presentation slide. Line 35 sets `my_second_object <- 1+1`. Line 39 shows the output `[1] 2`.
- Console:** Shows the execution of the code. It displays `> my_first_object <- 3.14`, `> my_first_object` returning `[1] 3.14`, `> my_second_object <- 1+1`, and `> my_second_object` returning `[1] 2`.
- Environment:** A panel on the right showing the current environment. It lists two objects: `my_first_object` with value `3.14` and `my_second_object` with value `2`. This panel is circled in pink.
- Files:** A panel at the bottom right showing the file explorer. It lists files like `.gitignore`, `.Rhistory`, `before_course`, `data-science-basics`, `day1`, `day2`, `day3`, `day4`, `README.md`, and `reference_data`.

Our new objects are shown in this corner

## Using (numeric) objects to do math

- Just like we did when we used R as a calculator, you can also use numeric objects to do math
- When you do this, the objects themselves don't change unless you explicitly re-assign them to new variables

# Using (numeric) objects to do math

```
my_first_object * my_second_object
```

```
## [1] 6.28
```

```
my_first_object/my_second_object
```

```
## [1] 1.57
```

```
my_first_object^my_second_object
```

```
## [1] 9.8596
```

# Now you try!

- Pick your favorite number, and save it as an object
- Pick another number, and save it as another object
- Do one basic calculation (e.g., addition) with your objects
- You may run into issues. That's okay! We'll talk them through.



Source: XKCD



# Data types in R

- **numeric:** a number (e.g., -1, 0, 893243.343)
- **logical:** TRUE or FALSE (no quotations)
- **character:** letters and words (tricky: or a number stored as letter!)
- The function `is()` helps us figure out what type of data we have

```
is(-1)
```

```
## [1] "numeric" "vector"
```

```
is(TRUE)
```

```
## [1] "logical" "vector"
```

```
is("What is this?")
```

```
## [1] "character"
```

```
"vector"
```

```
"data.frameRowLabels"
```

```
## [4] "SuperClassMethod"
```

# Numeric data

```
my_first_object <- 3.14
```

```
my_first_object
```

```
## [1] 3.14
```

```
my_second_object <- 1+1
```

```
my_second_object
```

```
## [1] 2
```

---

# Character data

---

```
policy <- "International Health Regulations (IHR)"
```

```
organization <- "UNAIDS"
```

# Logical data

```
logical_example <- TRUE
```

```
second_logical_example <- FALSE
```

# Check your understanding!

```
is(-1)
```

```
is(TRUE)
```

```
is("What is this?")
```

---

# Check your understanding!

```
is(-1)
```

```
## [1] "numeric" "vector"
```

```
is(TRUE)
```

```
## [1] "logical" "vector"
```

```
is("What is this?")
```

```
## [1] "character"          "vector"              "data.frameRowLabels"
```

```
## [4] "SuperClassMethod"
```

---



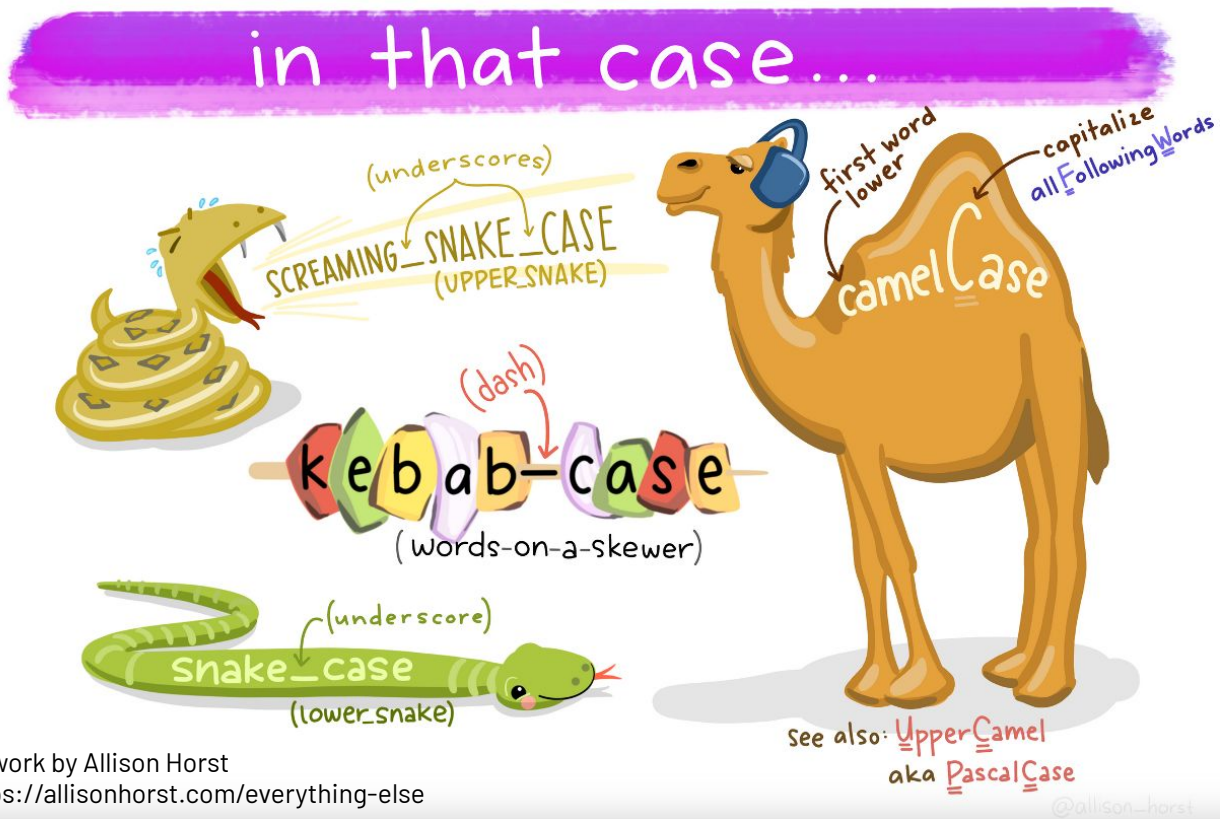
Artwork by Allison Horst  
<https://allisonhorst.com/everything-else>

# Rules for naming objects

- General naming requirement: a variable name can't start with a number or a dot (.)
- R is case sensitive ('A' is different than 'a')
- General rules of thumb: aim for consistency
  - snake\_case
  - camelCase
  - whatever.this.is
- Chose a name you'll understand when you open your code the next day, or when someone else reviews it

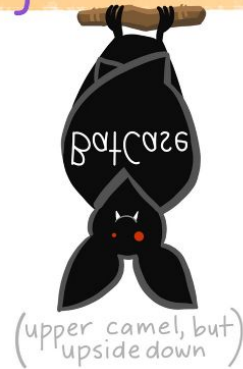


# Rules for naming objects



# Rules for naming objects

## failed programming cases



## Now you try!

- Create three new objects, with any allowable names you want. Try to use a consistent naming style.
  - Numeric (we already did this, but practice is good)
  - Character
  - Logical

# Vectors

- Vectors are grouped data elements in a specific order
- For example, data in a specific column in Excel
- When you've thought previously about data analysis, you probably think about vectors, even if you didn't use that name

name	iso_3166	stanag_code	internet_code	who_member_state
Afghanistan	AFG	AF   AFG   004	AFG	TRUE
Albania	ALB	AL   ALB   008	ALB	TRUE
Algeria	DZA	DZ   DZA   012	DZA	TRUE
Andorra	AND	AD   AND   020	AND	TRUE
Angola	AGO	AO   AGO   024	AGO	TRUE
Antigua and Barbuda	ATG	AG   ATG   028	ATG	TRUE
Argentina	ARG	AR   ARG   032	ARG	TRUE
Armenia	ARM	AM   ARM   051	ARM	TRUE
Australia	AUS	AU   AUS   036	AUS	TRUE

Each column is a vector

# Vectors

```
c("HIV", "malaria", "TB")
```

```
## [1] "HIV"      "malaria" "TB"
```

```
c(1419, 4832, 10342)
```

```
## [1] 1419 4832 10342
```

# Vectors

The `c()` stands for “concatenate”

```
c("HIV", "malaria", "TB")
```

```
## [1] "HIV"      "malaria" "TB"
```

```
c(1419, 4832, 10342)
```

```
## [1] 1419 4832 10342
```

# Vectors

```
c("HIV", "malaria", "TB")
```

Vectors can contain strings

```
## [1] "HIV"      "malaria" "TB"
```

```
c(1419, 4832, 10342)
```

Or numbers

```
## [1] 1419 4832 10342
```

# Vectors

```
c("HIV", "malaria", "TB")
```

Vectors can contain strings

```
## [1] "HIV"      "malaria" "TB"
```

```
c(1419, 4832, 10342)
```

Or numbers

```
## [1] 1419 4832 10342
```

```
c("HIV", "malaria", 10342)
```

... but not both

***What happened here?***

```
## [1] "HIV"      "malaria" "10342"
```



## Now you try!

- Make two vectors in R and assign them to objects.
  - Numeric
  - String

# Vectorized calculations

```
c(1,2,3,4,5) + 1
```

```
## [1] 2 3 4 5 6
```

```
c(1,2,3,4,5) * 2
```

```
## [1] 2 4 6 8 10
```

```
c(1,2,3,4,5) + c(8,0,0,0,0)
```

```
## [1] 9 2 3 4 5
```

# Vectorized calculations

```
c(1,2,3,4,5) + c(8,0)
```

```
## Warning in c(1, 2, 3, 4, 5) + c(8, 0): longer object length is not a mult  
## of shorter object length
```

```
## [1]  9  2 11  4 13
```

# Functions

- Functions are instructions to perform a task
  - They are *algorithms*, or consistent set of rules
- R has built-in functions for many basic things
- Functions generally look like this: *function(object)*
- We can also “add on” extra functions by loading new libraries (we’ll get to this later), or we can write our own functions to do whatever we want

# Functions

- Most functions in R are vectorized
  - This means they act on all items in a vector
- Why does this matter?
  - If you misunderstand it, your math will be wrong
  - It's useful for basic calculations and analysis:
    - divide all numbers by 100 to calculate a %
    - multiply per-capita rates by total population

# Functions

```
mean(c(1,2,3,4,5))
```

```
## [1] 3
```

```
sd(c(1,2,3,4,5))
```

```
## [1] 1.581139
```

```
summary(c(1,2,3,4,5))
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   
##         1         2         3         3         4         5
```

# Functions

```
mean(c(1, 2, 3, 4, 5))
```

```
## [1] 3
```

```
sd(c(1, 2, 3, 4, 5))
```

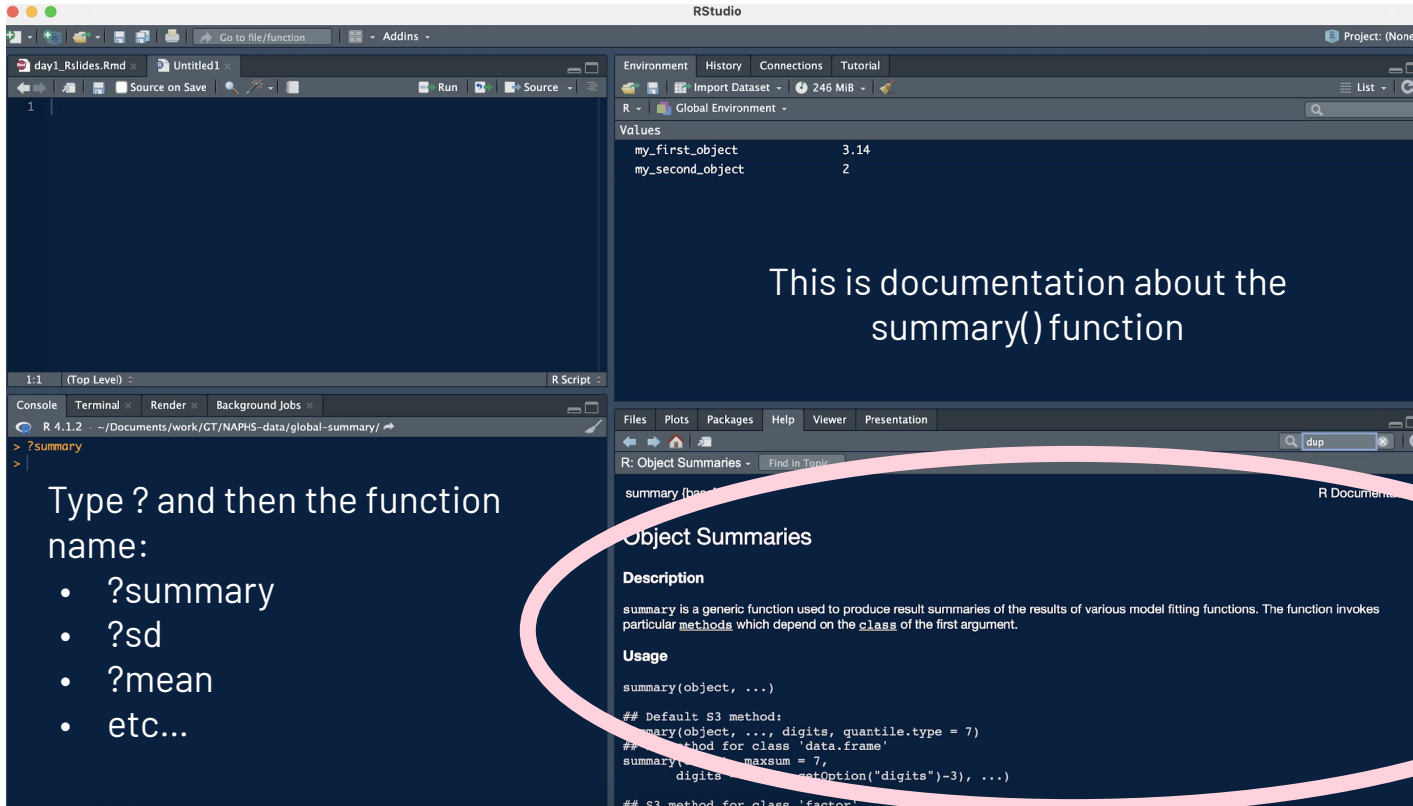
```
## [1] 1.581139
```

```
summary(c(1, 2, 3, 4, 5))
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   
##         1         2         3         3         4         5
```

# Learn more about functions

## ?function or help(function)



The screenshot shows the RStudio interface. The top-left pane contains a script editor with a single line of code: `?summary`. The bottom-left pane shows the console output: `> ?summary`. The right-hand side of the interface displays the help documentation for the `summary` function. A pink oval highlights the 'Object Summaries' section, which includes the 'Description' and 'Usage' sections. The 'Description' section states: 'summary is a generic function used to produce result summaries of the results of various model fitting functions. The function invokes particular methods which depend on the class of the first argument.' The 'Usage' section shows the function signature: `summary(object, ...)` and provides examples for the default S3 method and the method for class 'data.frame'.

This is documentation about the `summary()` function

Type ? and then the function name:

- `?summary`
- `?sd`
- `?mean`
- etc...

**Object Summaries**

**Description**

`summary` is a generic function used to produce result summaries of the results of various model fitting functions. The function invokes particular methods which depend on the `class` of the first argument.

**Usage**

```
summary(object, ...)
```

**## Default S3 method:**

```
summary(object, ..., digits, quantile.type = 7)
```

**## Method for class 'data.frame'**

```
summary(object, maxsum = 7,
  digits = getOption("digits")-3, ...)
```

**## S3 method for class 'factor'**



## Now you try!

- Take the average of three or more numbers
- Use "?" to learn more about the function `sd()`

# Goals for today

- Understand what statistical programming is
- Get acquainted with Rstudio
- Write your very first R code (at least, of this workshop)
  - vectors
  - functions
  - accessing documentation
- Explore github to access course materials

# What is github?

# What is github?

- Have you ever saved a bunch of versions of a paper on your computer with different file names at different dates or times of day?
- Backups are useful to save progress, understand what we've done before, and look into problems/bugs
- Github is a tool do help do this with code

# What is github?

We'll talk more about github later in this workshop. For now, I'd like you to be able to use it to access course materials any time you'd like to.

**<https://github.com/seaneff/data-science-basics-2024>**

 main ▾ 1 branch 0 tags

Go to file

Add file ▾

&lt;&gt; Code ▾



seaneff download R during workshop or after first day

6e107b6 1 minute ago ⌚ 23 commits

📁 day1	download R during workshop or after first day	1 minute ago
📁 day2	course updates	last week
📁 day3	course updates	last week
📁 day4	course updates	last week
📁 download_R	download R during workshop or after first day	1 minute ago
📁 reference_data	end of day commit, still figuring out dataset	2 months ago
📄 .gitignore	fix gitignore update	2 months ago
📄 README.md	download R during workshop or after first day	1 minute ago

main 1 branch 0 tags

Go to file

Add file

[Code](#)

seaneff download R during workshop or after first day

6e107b6 1 minute ago 23 commits



day2

download R during workshop or after first day  
course updates

1 minute ago



day3

course updates

last week



day4

course updates

last week



download\_R

download R during workshop or after first day

1 minute ago



reference\_data

end of day commit, still figuring out dataset

2 months ago



.gitignore

fix gitignore update

2 months ago



README.md

download R during workshop or after first day

1 minute ago

Day 2 course materials are in this folder

**Now you try!**

Open the course github and click around for a few minutes

**<https://github.com/seaneff/data-science-basics-2024>**



**Thank you!**

**See you tomorrow.**

***Please come with a fully charged laptop.***