Web Scraping Process:

(A) No "Pipe" (%>%)

- step 0 → what is the url(s) of interest?

in plain english, know your webpage

- single page
- multiple pages

Q: what are the URLs?

- manually create a vector of all URLs
- attempt to automate the generation of URLS to save time

single page

↳ step 0 = "_____"

multiple pages → increment the URL @ the end (e.g. you will paste0 year or some counter @ the end)

- Step1 ⟹ get the html into your software (backend of the html & Not how your browser is showing it)

step1 = read_html(x = url)

↓

rvest pkg

For your code to work:

(*) a static webpage

(*) does the website allow you to scrape the data?

if its working

step1 will return a list

↳ head

↳ body

- step2 "often repeat that to get diff elements/parts of webpage"

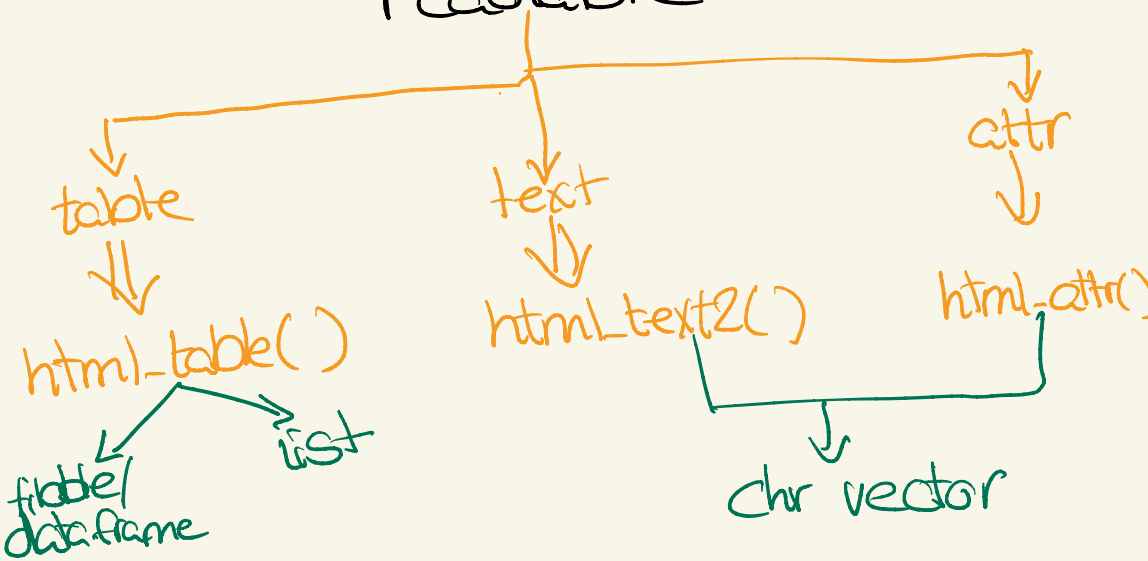
inputs:
* read webpage stored in step1
* css for the elements of int
  "typically divided by appearance on page"

"iterative"

when to know that your result is correct
↳ starting, ending & total # results seems to be correct

incorrect #s
↳ 1 result
  ↳ see if you have a nth-child( ) in your selector
  ↳ html_element( ) instead of html_elements( )
↳ too many results ⇒ add tags
↳ too few ⇒ remove classes / tags

- Step3 → make your results clean/ readable



input: only step2

output: readable value similar to step2

- step4 → merge diff text and put in a tibble (not always needed)

to make it work: Length of the diff objs has to be the same

if you pipe, you do not need to store intermediate results.

(you do not have to pipe if you don't like it)

native pipe: |>

same as the magrittr/dplyr pipe %>%