Statistical Programming and Open Science Methods

Tidy data scraping

Joachim Gassen
Humboldt-Universität zu Berlin

September 02, 2022

SFB/Transregio 266
ACCOUNTING FOR TRANSPARENCY

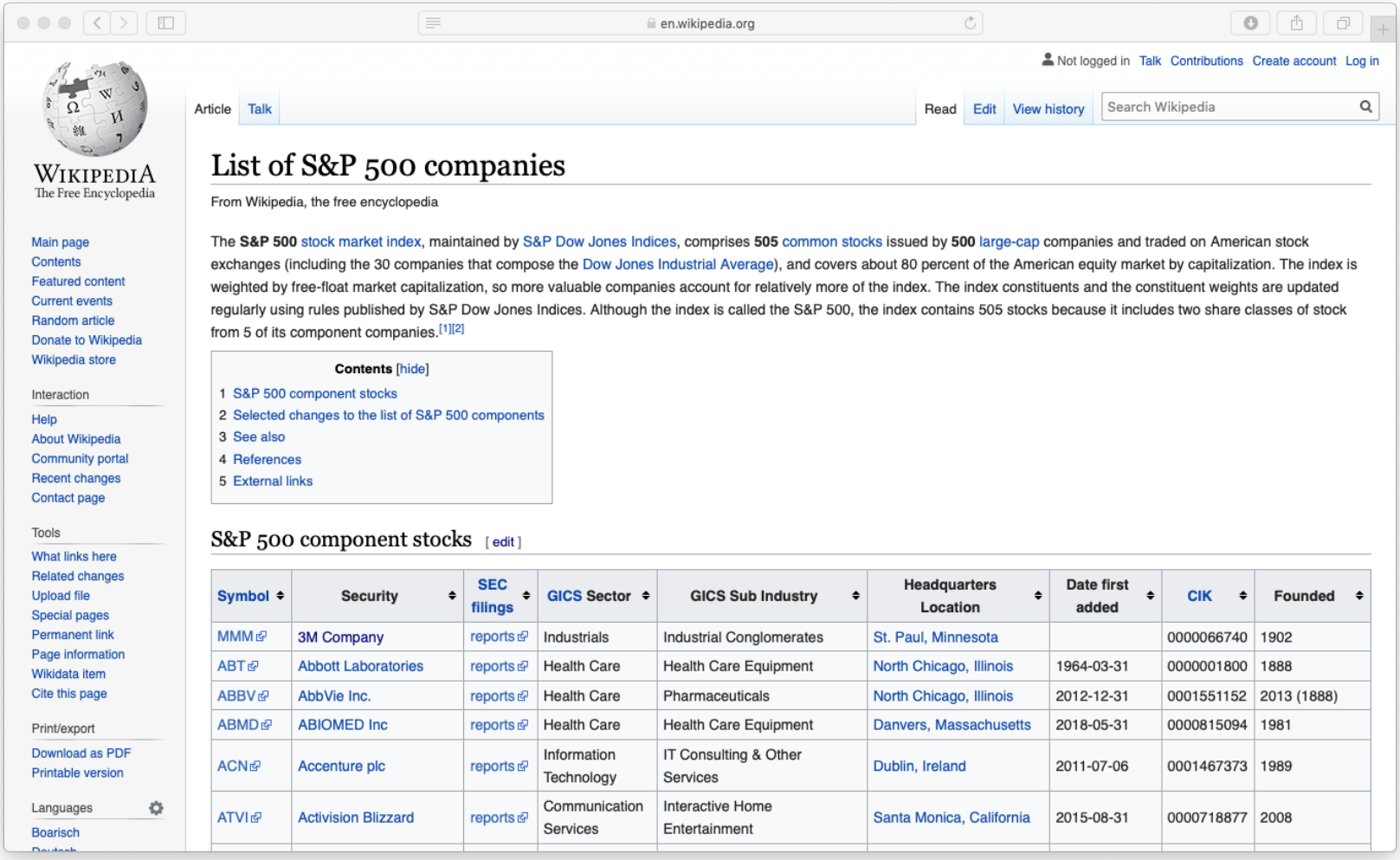
Time table Monday, February 17

When?	What?
10:00	Welcome and Coffee
10:30	Tidy data scraping
12:00	Lunch
13:30	Code along: Unit testing in function development
15:00	Coffee
15:30	Group work presentations
16:30	End of Day

Time table Tuesday, February 18

When?	What?
09:00	Explore your researcher degrees of freedom
10:30	Coffee
11:00	Providing data access via RESTful APIs
12:30	Lunch
13:30	Group Work Presentations
15:00	Coffee and Wrap Up
15:30	End of Event

Parsing an HTML table I



en.wikipedia.org

Not logged in Talk Contributions Create account Log in

Article Talk | Read Edit View history | Search Wikipedia

WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Download as PDF
Printable version

Languages
Boarisch

List of S&P 500 companies

From Wikipedia, the free encyclopedia

The **S&P 500** stock market index, maintained by S&P Dow Jones Indices, comprises **505** common stocks issued by **500** large-cap companies and traded on American stock exchanges (including the 30 companies that compose the Dow Jones Industrial Average), and covers about 80 percent of the American equity market by capitalization. The index is weighted by free-float market capitalization, so more valuable companies account for relatively more of the index. The index constituents and the constituent weights are updated regularly using rules published by S&P Dow Jones Indices. Although the index is called the S&P 500, the index contains 505 stocks because it includes two share classes of stock from 5 of its component companies.[1][2]

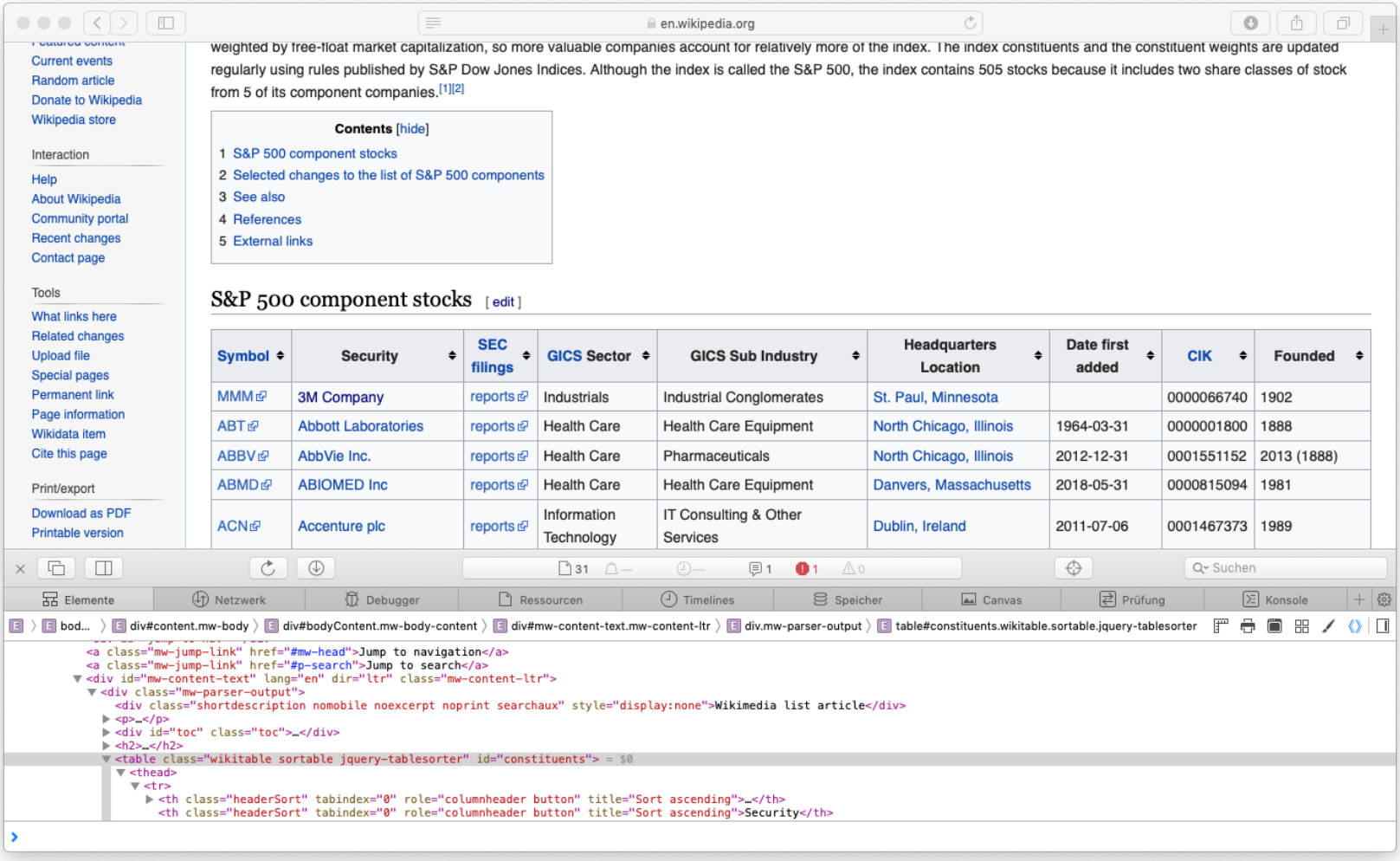
Contents [hide]

1 S&P 500 component stocks
2 Selected changes to the list of S&P 500 components
3 See also
4 References
5 External links

S&P 500 component stocks [edit]

Symbol	Security	SEC filings	GICS Sector	GICS Sub Industry	Headquarters Location	Date first added	CIK	Founded
MMM	3M Company	reports	Industrials	Industrial Conglomerates	St. Paul, Minnesota		0000066740	1902
ABT	Abbott Laboratories	reports	Health Care	Health Care Equipment	North Chicago, Illinois	1964-03-31	0000001800	1888
ABBV	AbbVie Inc.	reports	Health Care	Pharmaceuticals	North Chicago, Illinois	2012-12-31	0001551152	2013 (1888)
ABMD	ABIOMED Inc	reports	Health Care	Health Care Equipment	Danvers, Massachusetts	2018-05-31	0000815094	1981
ACN	Accenture plc	reports	Information Technology	IT Consulting & Other Services	Dublin, Ireland	2011-07-06	0001467373	1989
ATVI	Activision Blizzard	reports	Communication Services	Interactive Home Entertainment	Santa Monica, California	2015-08-31	0000718877	2008

Parsing an HTML table II



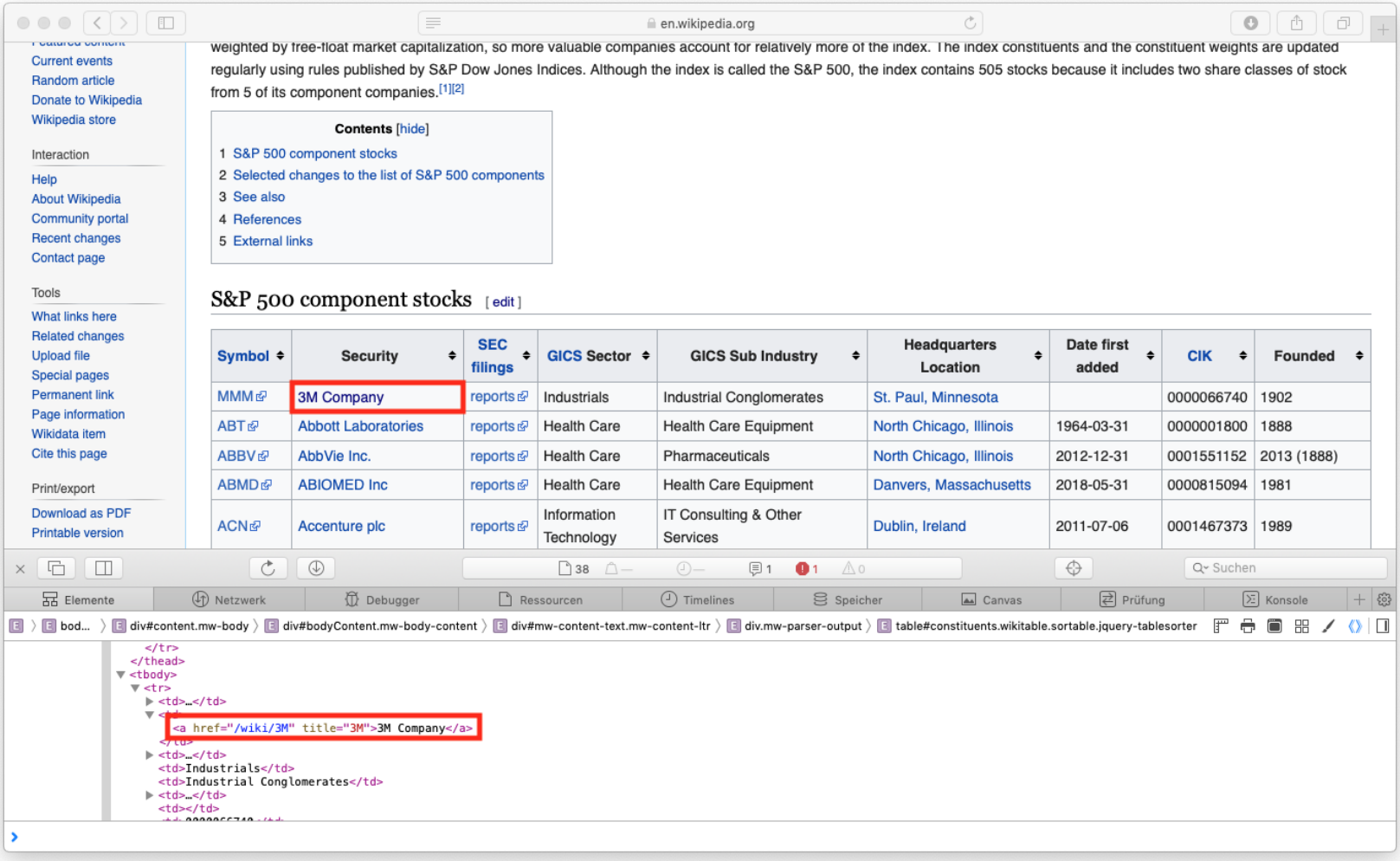
Parsing an HTML table III

```
library(tidyverse)
library(rvest)

url_sp500_const <- paste0(
  "https://en.wikipedia.org/wiki/",
  "List_of_S%26P_500_companies"
)

url_sp500_const %>%
  read_html() %>%
  html_node(xpath = '//*[@id="constituents"]') %>%
  html_table() -> sp500_constituents_raw
```

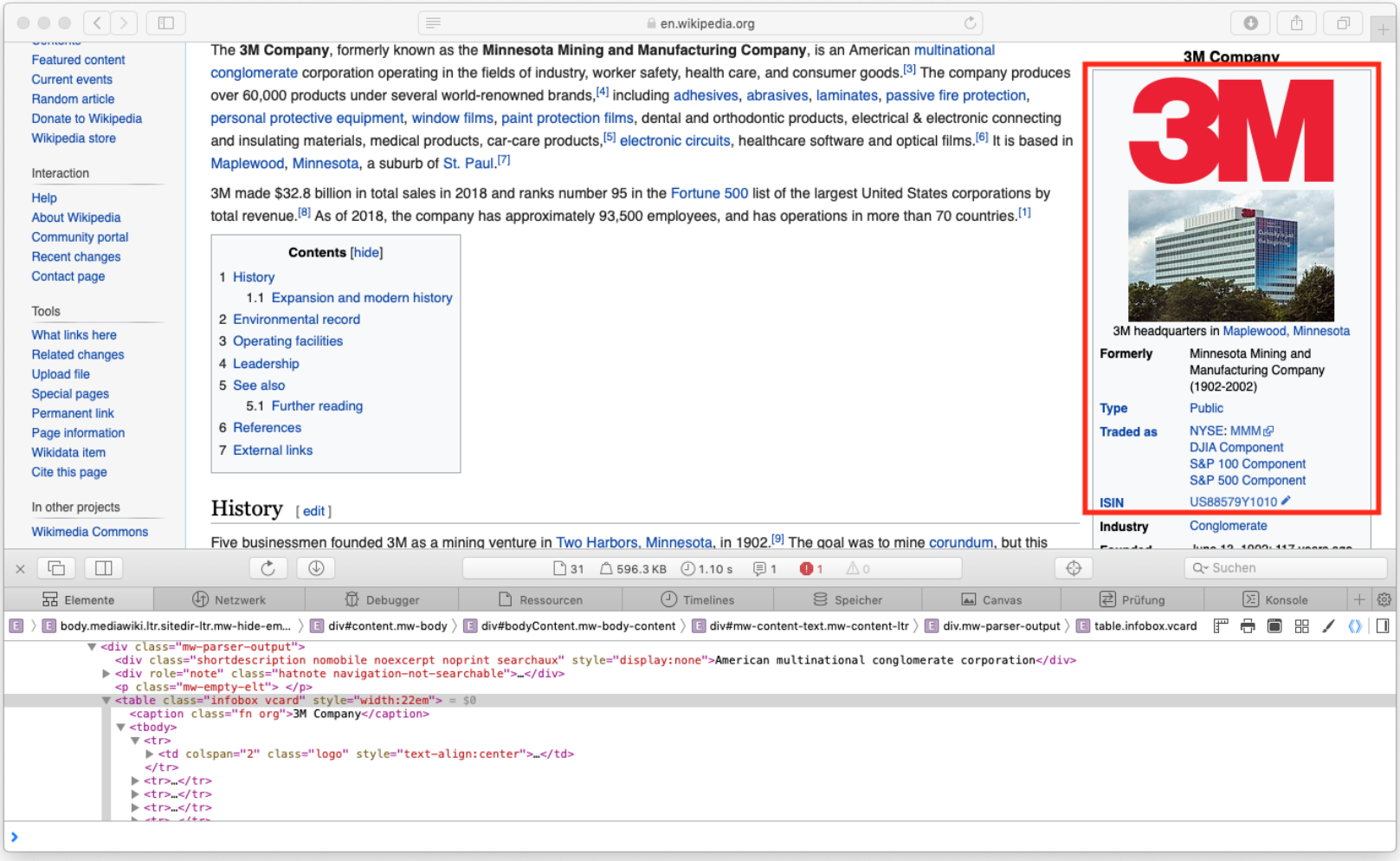
Retrieving local URLs from within table I



Retrieving local URLs from within table II

```
url_sp500_const %>%
  read_html() %>%
  html_node(xpath = '//*[@id="constituents"]') %>%
  html_nodes("td:nth-child(2) a") %>%
  html_attr("href")-> links
```

Scraping ill-structured tables I

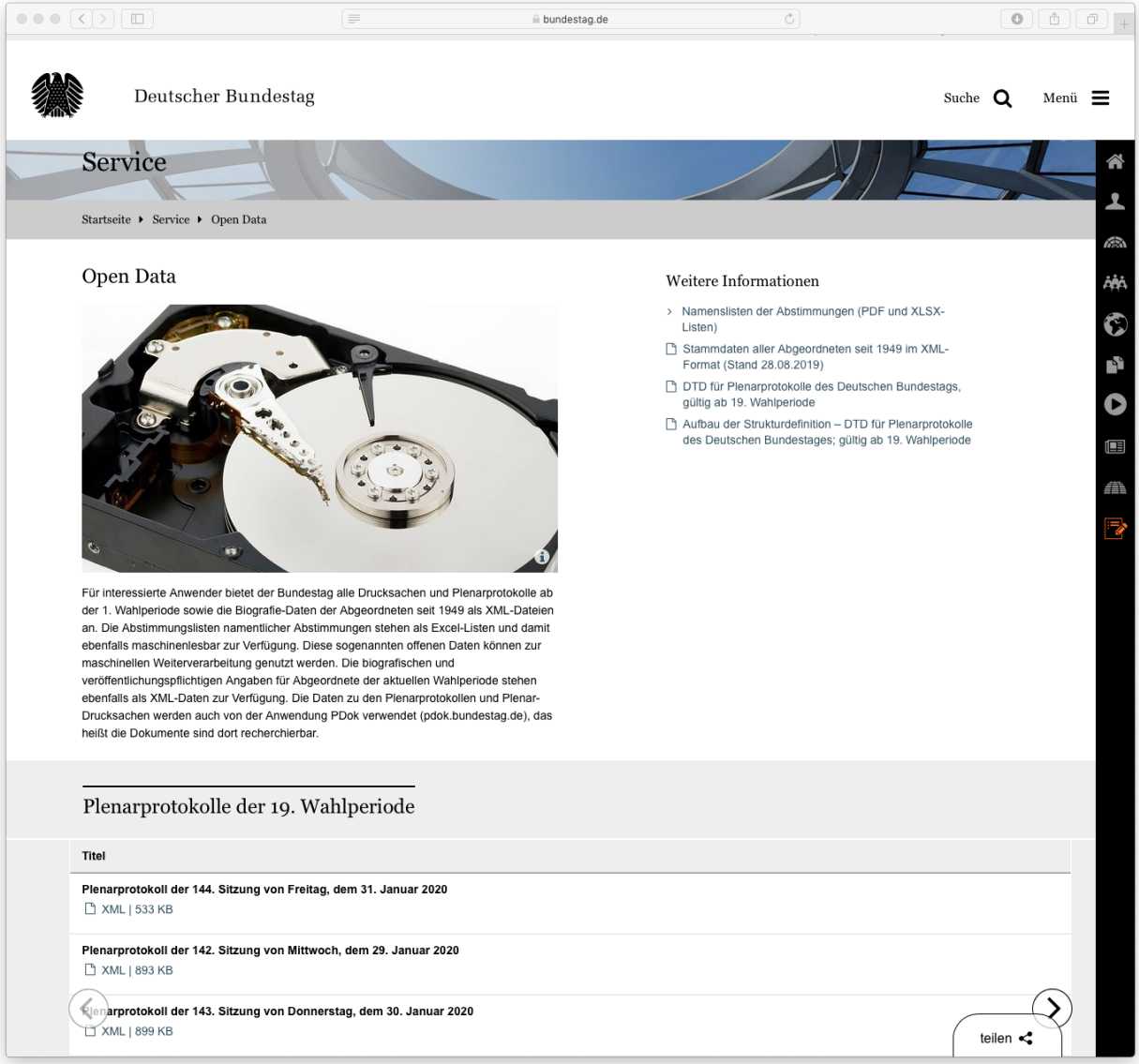


Scraping ill-structured tables II

```
xml_data <- read_html(url) %>%
  html_node('#mw-content-text div table.infobox.vcard')

xml_data  %>%
  html_table(fill = TRUE) %>%
  rename(tag = X1, content = X2) %>%
  filter(tag != "")
```

Another approach to scraping



This won't work

```
library(rvest)

url_bt_open_data <- "https://www.bundestag.de/services/opendata"
url_bt_open_data %>%
  read_html() %>%
  html_node(
    xpath = '//*[@id="bt-collapse-543410"]/div[1]/div/div/div[1]/table'
  ) %>% html_table() -> pp_table
```

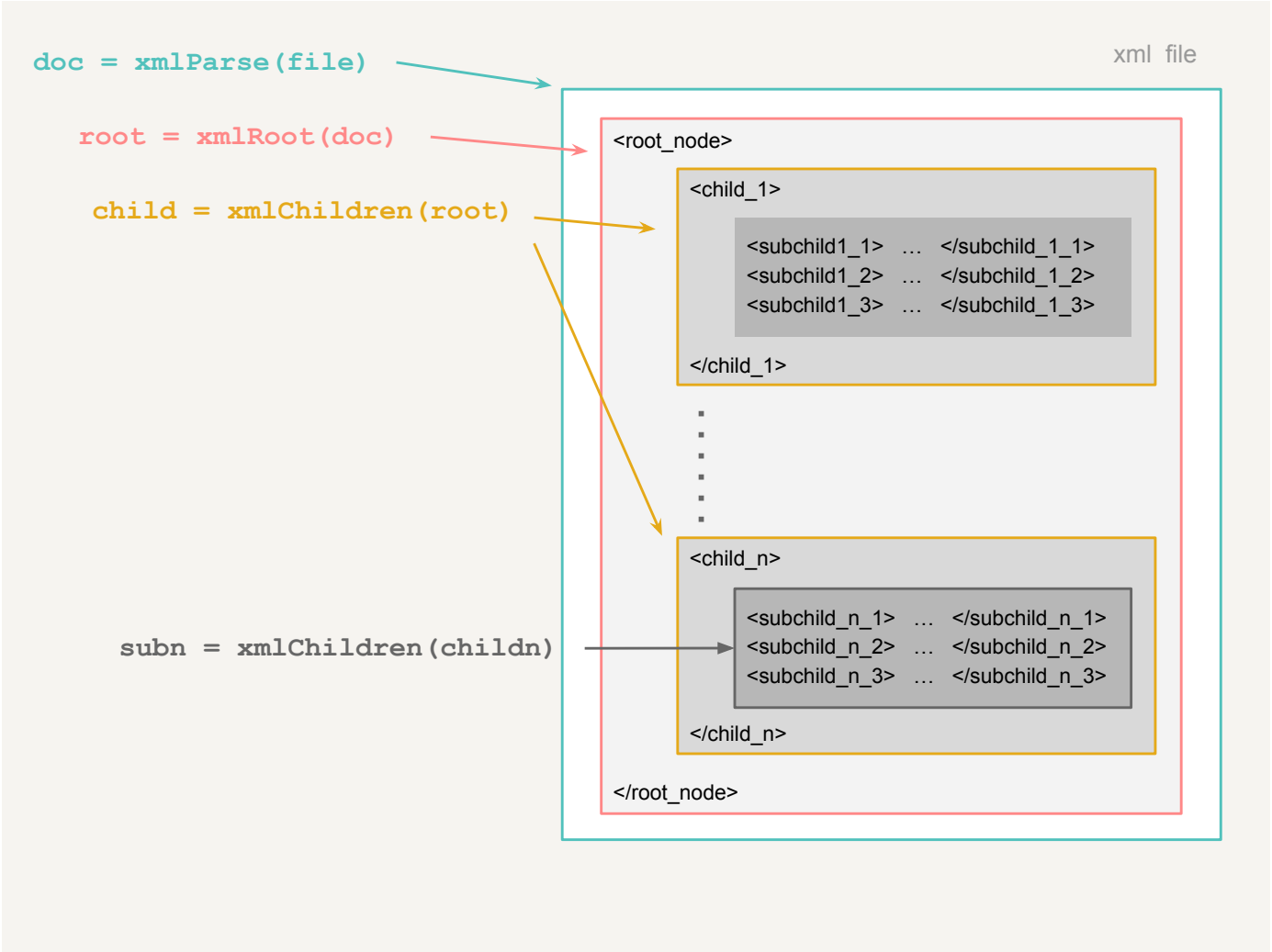
```
## Error in UseMethod("html_table"): no applicable method for 'html_tab
```

Reason: The web page is being created dynamically by JavaScript (or similar)

The idea of headless browsing: meet Selenium

- Selenium offers a way to script a web browser so that data can be scraped using the navigation that a web page provides
- Allows for various web browser and all sorts of user web browser interaction
- Uses a docker container holding the actual browser environment
- See `code/btag_open_data_scrape_data.R` for a demonstration on how to use it

Parsing XML Data



My task ...

- Develop a function that parses the XML files of the Plenarprotokolle of the 19th Wahlperiode to extract all speaches into a tidy data structure
- Implement some basic test routines verifying that the code works
- See `code/btag_open_data_scrape_data.R` and `code/test/test_btag_open_data_scrape_data.R`