What is digital trace data and how do we collect it?

Game plan

Tuesday

- Collecting data: API
- Collecting data: Web Scraping

Wednesday

- Collecting: Browser Automation
- Cleaning: String Operations
- Cleaning: Regular Expressions

Game plan

- Alternating between lecture-style introduction and exercises
- Starting real basic!
- But tasks are not easy.

Problem solving

Doctors: Googling stuff online doesn´t make you a doctor
Programmers:
imgflip.com

But!

But!

Why quarto®?

[Click!]

What is digital trace data?

https://commons.wikimedia.org/wiki/File:Duchamp_Fountaine.jpg
https://commons.wikimedia.org/wiki/File:%27David%27_by_Michelangelo_JBU0001.JPG

What is digital trace data?

For our purposes:

Data that is not created for the purpose of being analyzed by social science researchers, but is a byproduct of everyday online activity.

Benefits and issues of digital trace data

- Big (enables analysis of small differences/prevalence)
- Always on (enable capturing of rare and surprising events)
- Non-Reactive
- Captures Social Relationships

- Big (difficult to handle)
- Non-Representative
- Biases depending on platform
- Drifting
- Algorithmic Confounding
- Unstructured and noisy
- Sensitive
- Incomplete (e.g., demographic info)
- Fake?

Accessibility

Collecting digital trace data

- Our focus: Textual data
- Two main ways of collecting text data online:

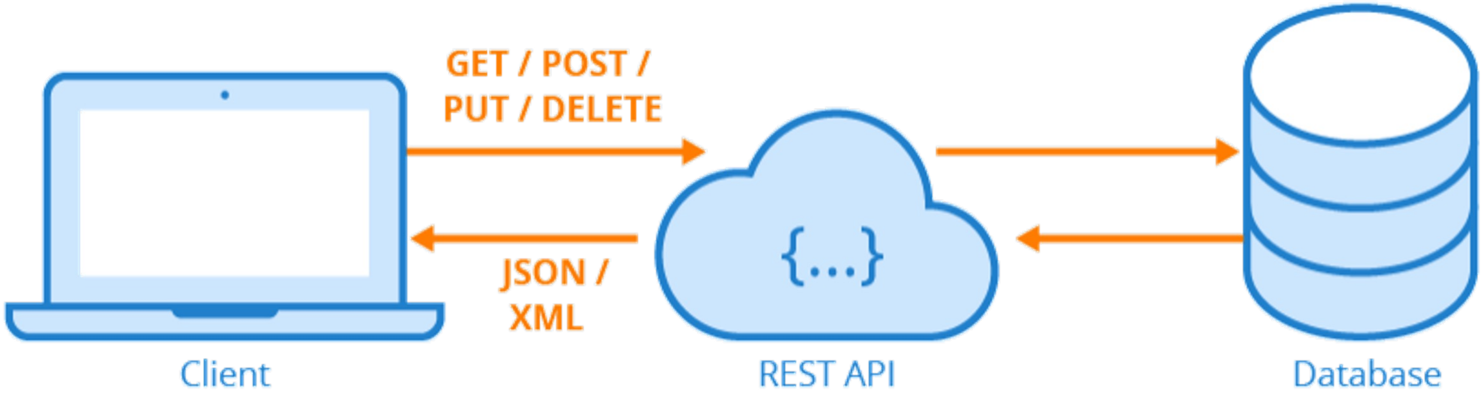
API

Web scraping

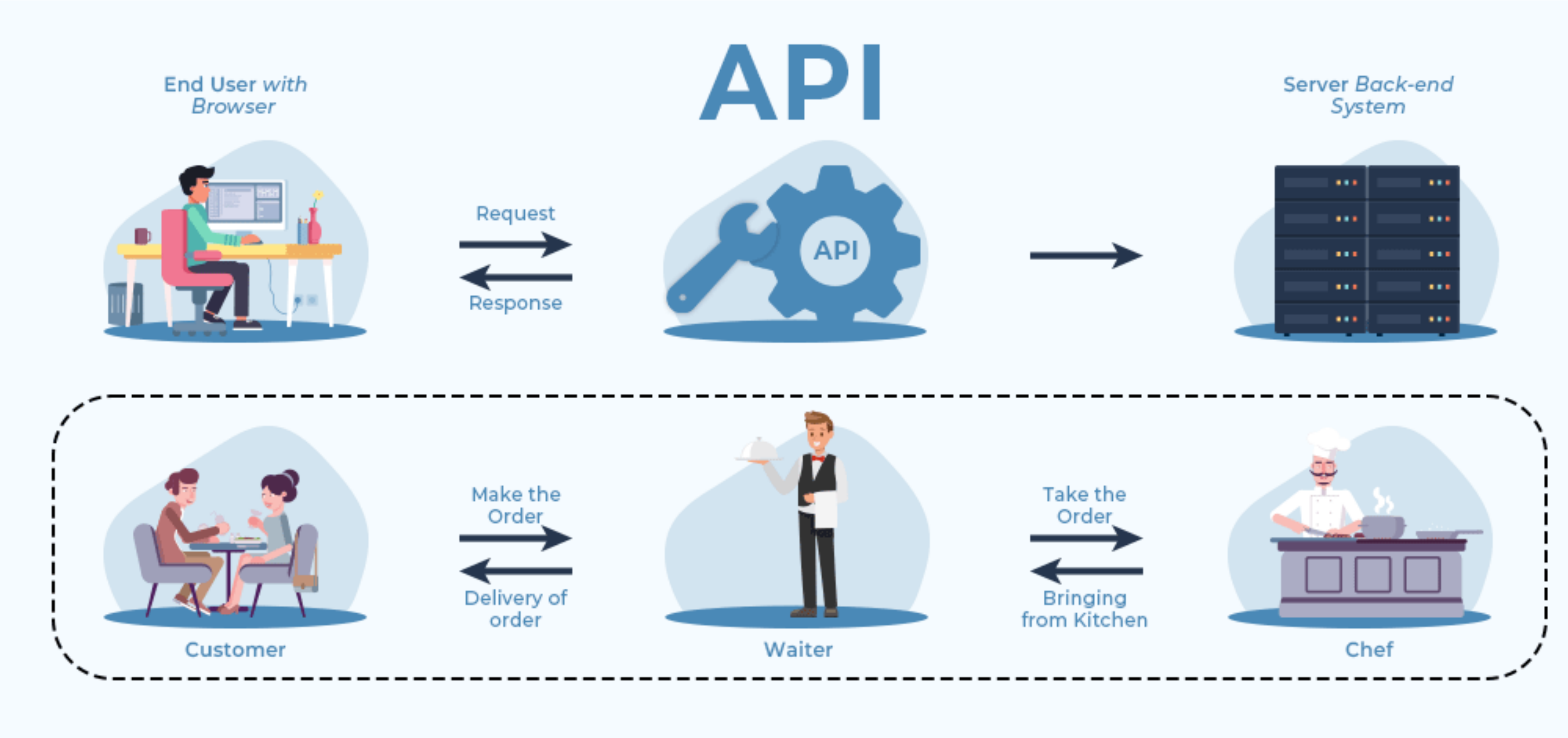
API Intro

What is an API

- Aplication Programming Interface
- An interface provided by the data base owner which enables you to access data on their server conveniently



What is an API



How do we make an order?

https://api.genderize.io?name=anna&country_id=DE

What is an URL

APIs are always accessed via an URL, therefore it is important to know how an URL is actually structured.

Protocol/ scheme Domain Path Query

https://www.website.com/api/cheese/cheesecake?color=yellow&form=circular

1 n

What is an URL

Protocol Domain Query

https://api.genderize.io?name=anna&country_id=DE

How do we know which queries to use?

Documentation!

Example: https://api.congress.gov/

API Authentification

Different forms of authentication.

- None
- API key (fully open, registration)
- Client key + secret key (mostly for sensitive or paid data)
- OAuth2 (most secure, involves separate authentication server)

API Authentification

Do not save your key directly in your script.
Instead you can use environment variables:

Run: `savefile.edit("~/.Renviron")`
Write: `"key = [your key]“`
Save and restart R
Run: `viaSys.getenv("key")`

API call, example with `httr`

https://www.website.com/api/cheese/cheesecake?color=yellow&form=circular

```
httr_rec <- GET(
  "https://www.website.com/",
  path = "api/cheese/cheesecake",
  query = list(color = yellow,
               form = circular,
               api_key = viaSys.getenv("key")))
```

Note, that in some cases you may want to also supply a header (mostly for authentication), see `?httr::add_headers`

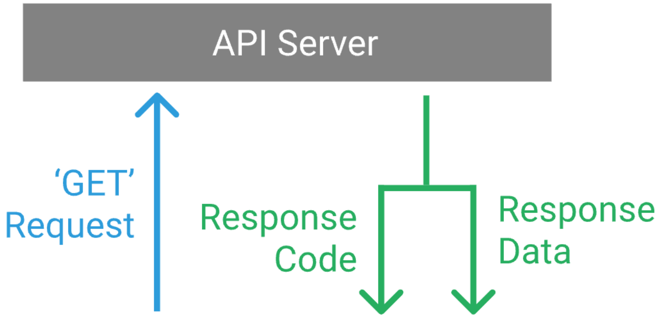
API call, example with `httr`

https://www.website.com/api/cheese/cheesecake?color=yellow&form=circular

```
httr_rec <- GET(
  "https://www.website.com/api/cheese/cheesecake",
  query = list(color = yellow,
               form = circular,
               api_key = viaSys.getenv("key")))
```

Note, that in some cases you may want to also supply a header (mostly for authentication), see `?httr::add_headers`

API Response



- An HTTP status code (200 is what you want)
- Headers
- A body typically consisting of XML, JSON, plain text, HTML, or some kind of binary representation.

Extract body using `content()` from the `httr` package

API – Example

Example script

https://api.congress.gov/

API-KEY: https://api.congress.gov/sign-up/

API – Tasks

/sessions/day2_webdata/

PLEASE ALWAYS COPY THE EXERCISES TO YOUR OWN FOLDER BEFORE OPENING AND CHANGING THEM!

Open **1_2_api_exercise.qmd** and/or **1_2_api_exercise.html**

Copy liberally from **1_1_api_example.R**, but try to understand what you are doing!

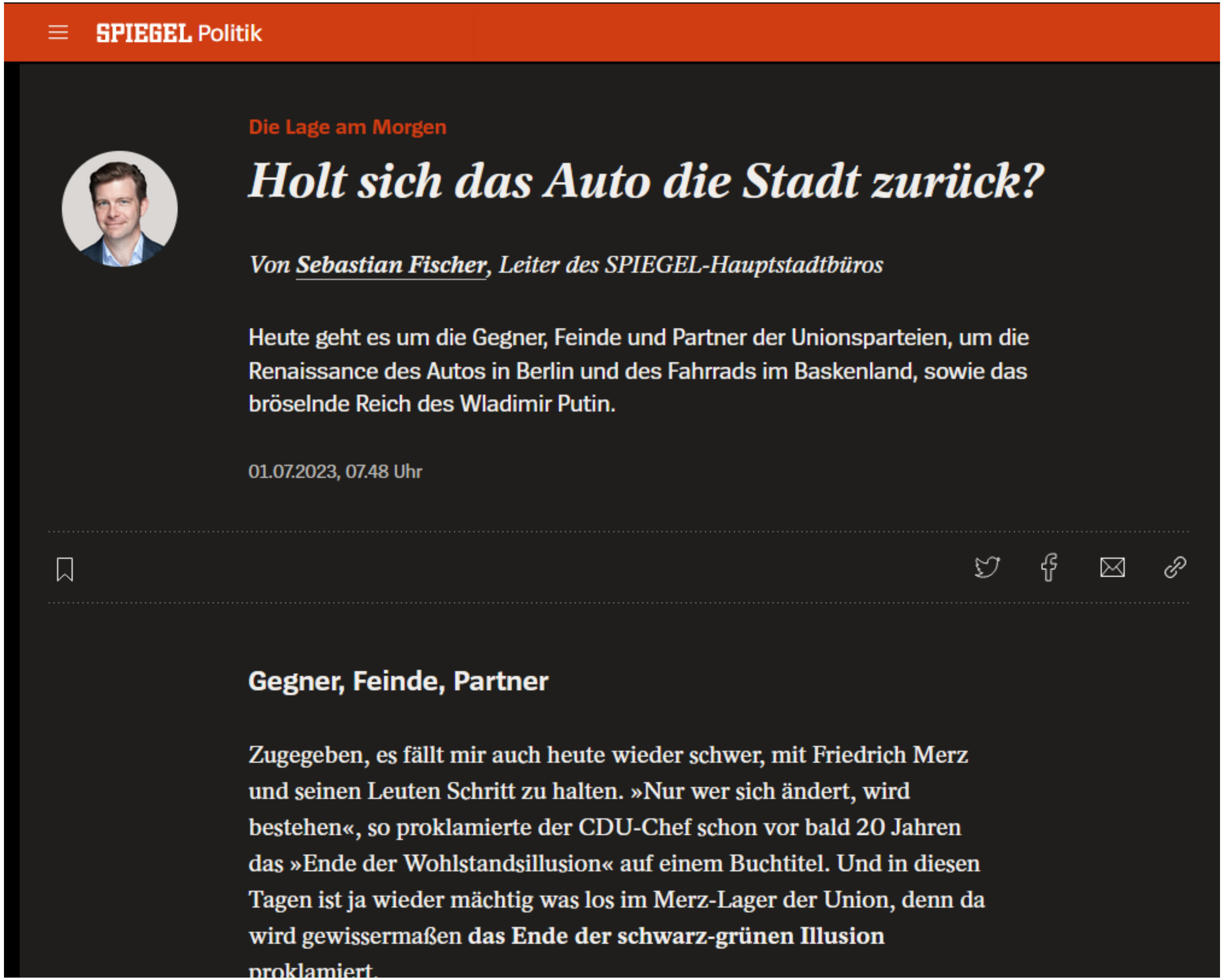
Web Scraping Intro

What is web scraping

- Automated extraction of data from websites
- Data can be text, images, links, and more
- Alternative to manually copying information from websites

  → Enabling the extraction of large amounts of up to date data

But how?



SPIEGEL Politik

Die Lage am Morgen

Holt sich das Auto die Stadt zurück?

Von Sebastian Fischer, Leiter des SPIEGEL-Hauptstadtbüros

Heute geht es um die Gegner, Feinde und Partner der Unionsparteien, um die Renaissance des Autos in Berlin und des Fahrrads im Baskenland, sowie das böselnde Reich des Wladimir Putin.

01.07.2023, 07.48 Uhr

Gegner, Feinde, Partner

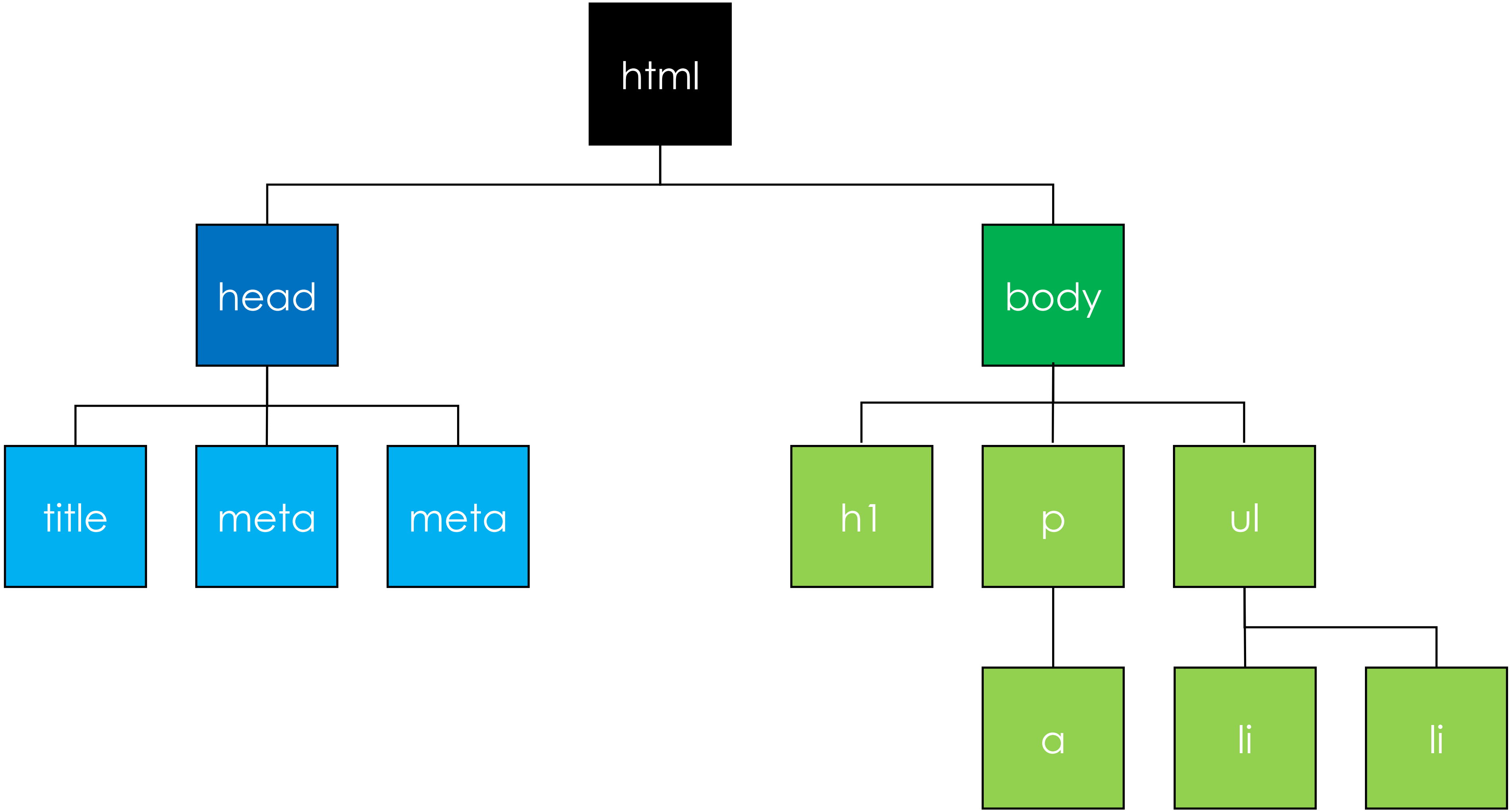
Zugegeben, es fällt mir auch heute wieder schwer, mit Friedrich Merz und seinen Leuten Schritt zu halten. »Nur wer sich ändert, wird bestehen«, so proklamierte der CDU-Chef schon vor bald 20 Jahren das »Ende der Wohlstandsillusion« auf einem Buchtitel. Und in diesen Tagen ist ja wieder mächtig was los im Merz-Lager der Union, denn da wird gewissermaßen **das Ende der schwarz-grünen Illusion** proklamiert.

Background Info: HTML

```
<!doctype html><html x-data lang="de" :class="{ 'audio-player-open': !!$store.WebAudio.clip }"><head><title>News: CDU-Chef Friedrich Merz und die Grünen, Berliner
Friedrichstraße, Wladimir Putin - DER SPIEGEL</title><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1,user-
scalable=no"><meta name="MSSmartTagsPreventParsing" content="true"><meta http-equiv="imagetoolbar" content="no"><meta name="apple-itunes-app"
content="app-id=424881832"><link rel="manifest" href="https://www.spiegel.de/public/spon/json/manifest.json"><meta name="theme-color" content="#e64415"
media="(prefers-color-scheme: light)"><="2023-07-01T07:48:02+02:00"><meta name="locale" content="de_DE"><meta name="description" content="Die
Unionsparteien ringen mit Gegnern, Feinden und Partnern. In Berlin gewinnt das Auto heute 500 Meter zurück. Und Putins Reich bröselt. Das ist die Lage am
Samstag."><meta name="news_keywords" content="Politik, Deutschland, Die Lage am Morgen"><meta name="twitter:card" content="summary_large_image"><meta
name="twitter:site" content="@derspiegel"><meta name="twitter:title" content="Die Lage am Morgen - CDU-Chef Friedrich Merz und die Grünen, Berliner
Friedrichstraße, Putins Reich"><meta name="twitter:creator" content="@sefi99"><meta name="twitter:image"
content="https://cdn.prod.www.spiegel.de/images/dd17189a-707c-4b57-af32-bf59855d3d25_w1195_r1.77_fpx28.09_fpy49.93.png"><meta property="og:title"
content="Die Lage am Morgen - CDU-Chef Friedrich Merz und die Grünen, Berliner Friedrichstraße, Putins Reich"><meta property="og:type" content="article"><meta
property="og:url" content="https://www.spiegel.de/politik/deutschland/news-cdu-chef-friedrich-merz-und-die-gruenen-berliner-friedrichstrasse-putins-reich-a-
aff057e5-4db6-4055-8d12-85cf8bc1fe2c"><meta property="og:image" content="https://cdn.prod.www.spiegel.de/images/dd17189a-707c-4b57-af32-
bf59855d3d25_w1195_r1.77_fpx28.09_fpy49.93.png"><meta property="og:description" content="Die Unionsparteien ringen mit Gegnern, Feinden und Partnern. In
Berlin gewinnt das Auto heute 500 Meter zurück. Und Putins Reich bröselt. Das ist die Lage am Samstag."><script
type="application/ld+json">[{"@context":"http://schema.org","@type":"NewsArticle","articleSection":"Politik","author":{"@type":"Person","name":"Sebastian
Fischer"},"dateCreated":"2023-07-01T05:27:01+02:00","dateModified":"2023-07-01T07:48:02+02:00","datePublished":"2023-07-01T05:27:01+02:00","headline":"Die
Lage am Morgen: Holt sich das Auto die Stadt […] <span class="bg-gradient-to-r from-white dark:from-dm-shade-darkest w-24 lg:h-56 md:h-56 sm:h-
40"></span></div></div></div><nav role="navigation" class="polygon-swiper flex items-center grow relative overflow-hidden h-full bottom-negative"><ul
class="polygon-swiper-wrapper flex items-center lg:h-56 md:h-56 sm:h-40 relative bottom-px:focus:border-white hover:border-shade-light border-transparent inline-
flex items-center text-black dark:text-shade-lightest text-s h-full px-4"><span class="border-b whitespace-nowrap border-inherit">Politik</span><span class="leading-
none ml-8"><svg width="16" height="16"><use xlink:href="#spon-chevron-right-m"/></svg></span></a></li><li class="polygon-swiper-slide flex items-center h-full
shrink-0"><a href="https://www.spiegel.de/politik/deutschland/" target="_self" title="Deutschland" class="focus:border-black dark:focus:border-white hover:border-
shade-light border-transparent inline-flex items-center text-black dark:text-shade-lightest text-s h-full px-4"><span class="border-b whitespace-nowrap border-
inherit">Deutschland</span><span class="leading-none ml-8"><svg width="16" height="16"><use xlink:href="#spon-chevron-right-m"/></svg></span></a></li><li
class="polygon-swiper-slide flex items-center h-full shrink-0"><a href="https://www.spiegel.de/thema/morningbriefing/" target="_self" title="Die Lage am Morgen"
class="focus:border-black dark:focus:border-white hover:border-shade-light border-transparent inline-flex items-center text-black dark:text-shade-lightest text-s h-full
px-4"><span class="border-b whitespace-nowrap border-inherit">Die Lage am Morgen</span><span class="leading-none ml-8"><svg width="16" height="16"><use
xlink:href="#spon-chevron-right-m"/></svg></span></a></li>
```

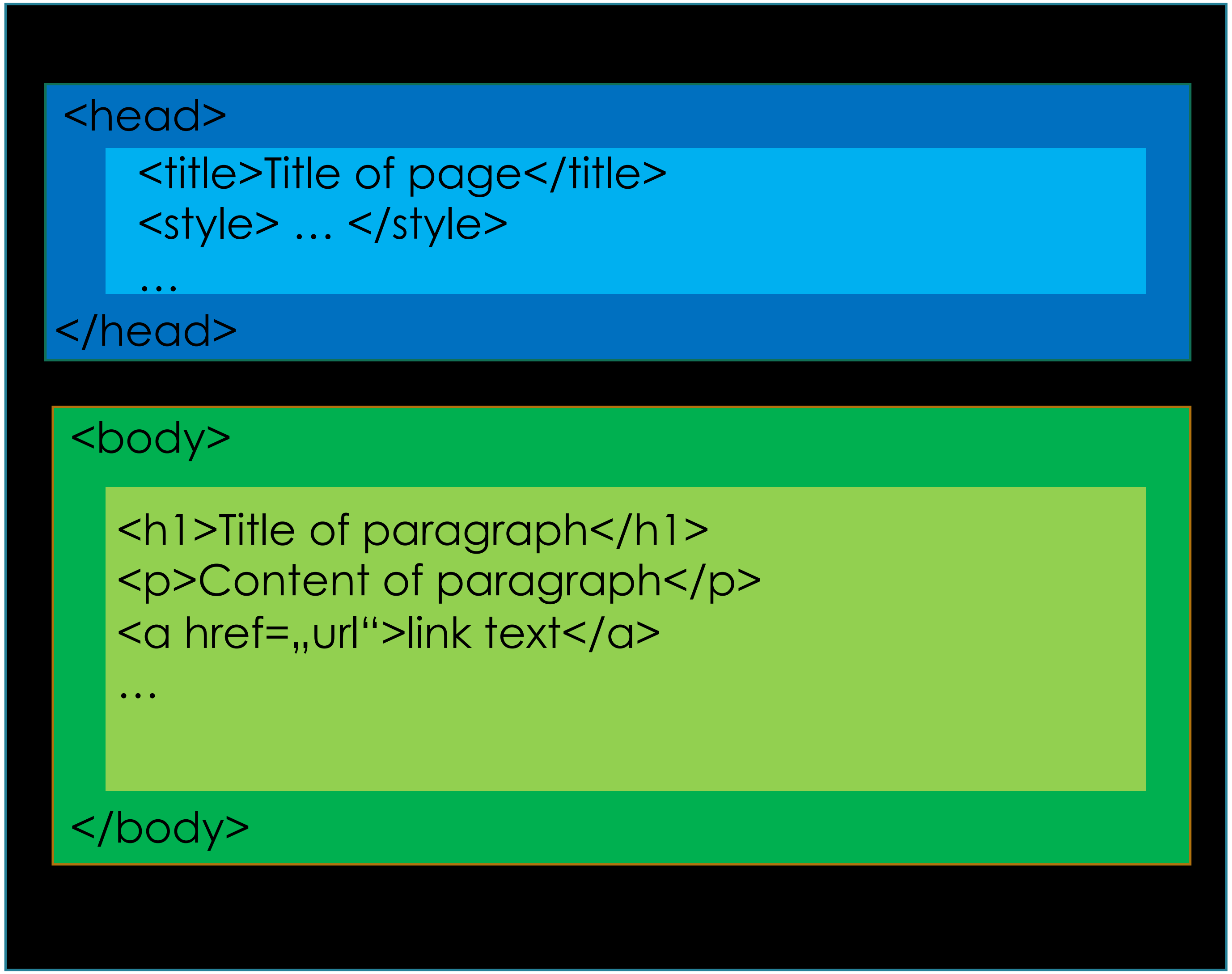
+ a few hundred lines of html

Background Info: HTML



Background Info: HTML

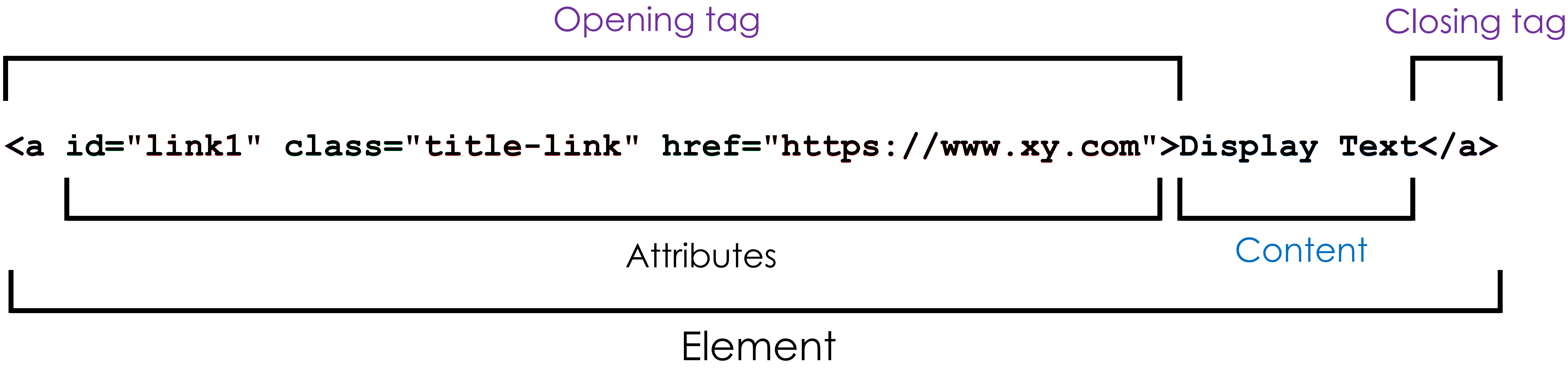
Meta data of website



```
<head>
    <title>Title of page</title>
    <style> … </style>
    …
</head>

<body>

   <h1>Title of paragraph</h1>
   <p>Content of paragraph</p>
   <a href=„url“>link text</a>
   …

</body>
```

Content of website

Background Info: HTML



Opening tag

Closing tag

```
<a id="link1" class="title-link" href="https://www.xy.com">Display Text</a>
```

Attributes

Content

Element

See https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML/Getting_started for more info

Background Info: HTML

`id="link1" class="title-link"`

`id` and `class` can both point to a specific style

- `id` can only appear once per document
- `class` can be used multiple times

- Targeting `class` is especially useful if you want to scrape
  a. multiple similar elements on the same page or
  b. across the entire website
- Targeting `id` is the easiest way to get one specific element

But how?

- Inspect in browser
- Add-on like SelectorGadget

https://www.spiegel.de

Targeting HTML Elements (with `rvest`)

```
<a id="link1" class="title-link" href="https://www.xy.com">Display Text</a>
```

*Let's imagine that this link has the underlying CSS class ".title-link"

You could target this using

`html_node("a")`	as it is an <a> element
`html_node(".title-link")`	as it has the CSS class .title-link
`html_node(xpath = "//a[@id='link1']")`	as it is an <a> element and has the id "link1"

`html_node(xpath =`
`"//*[contains(text(),'Display Text')]")`
	as the text shown on the webpage is "Display Text". Here we just target any element with that text.

`html_node(xpath =`
`"//*[contains(text(),'Display Text')`
`and not(contains(text(),'Cheesecake')]")`
	as the text shown on the webpage is "Display Text" and not "Cheesecake". Here we just target any element with that text.

There are many more options
See (for example) https://jtr13.github.io/cc19/web-scraping-using-rvest.html for a more detailed list of

Extracting HTML Elements (with `rvest`)

`html_text()`	Extract the displayed text
`html_table()`	Extract a table
`html_attr()`	Extract by attribute
`html_attr("href")`	Extract by attribute, in this case a link (i.e., most likely an URL)

One last note on limits

Both web scraping and APIs have some limits that you need to be aware of.

APIs have explicit limits (how much you are allowed to download and how fast) and **you should always adhere to them.**

Web scraping has implicit limits on the number and speed of requests you can make to their server. Implicit, because they don't tell you and will just block you if you exceed the limit. **Always make use of waiting periods (~1 sec) between requests!**

Web Scraping – Tasks

PLEASE ALWAYS COPY THE EXERCISES TO YOUR OWN FOLDER BEFORE OPENING AND CHANGING THEM!

Open **2_2_scraping_exercise.qmd** and/or **2_2_scraping_exercise.html**

Copy liberally from **2_1_scraping_example.R**, but try to understand what you are doing!