UNIVERSIDADE CATÓLICA PORTUGUESA

BRAGA

Behavior Analysis Technologies

Session 9

Data Collection and Scraping

Applied Data Science

2024/2025

The First Step of Data Science

- The first step in data science…

  … is to get some data!

- You will typically get data in one of four ways:
  1. Directly download a data file(s) manually!
  2. Query data from a database.
  3. Query an API (Application Programming Interface).
  4. Scrap data from a webpage.

Data Formats

- The three most common formats:
  - CSV - Comma Separate Values
  - JSON - JavaScript Object Notation
  - HTML/XML - HyperText Markup Language / eXtensible Markup Language

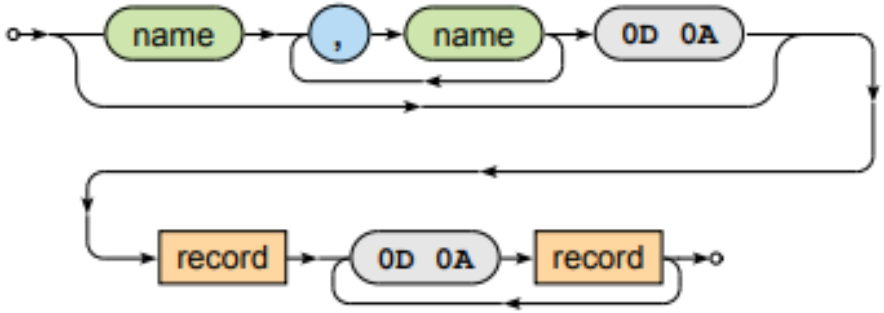
CSV Files

- A CSV file is a file where information is separated by commas.
  - Well, not always by commas!
- CSVs are plain text files
- Data can be saved in tabular format (meaning a table of rows and columns)

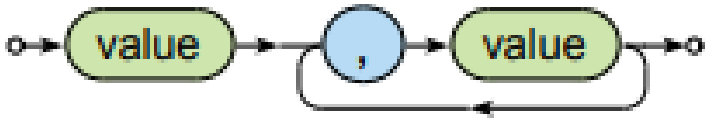
```
firstname,lastname,year
Ivan,Trojan,1964
Jiří,Macháček,1966
Jitka,Schneiderová,1973
Zdeněk,Svěrák,1936
Anna,Geislerová,1976
```

CSV Structure

- Document
  - Optional **header** + list of **records**



- Record
  - Comma separated list of **fields**



JSON Files

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

- JSON originated as a way of encapsulating Javascript objects

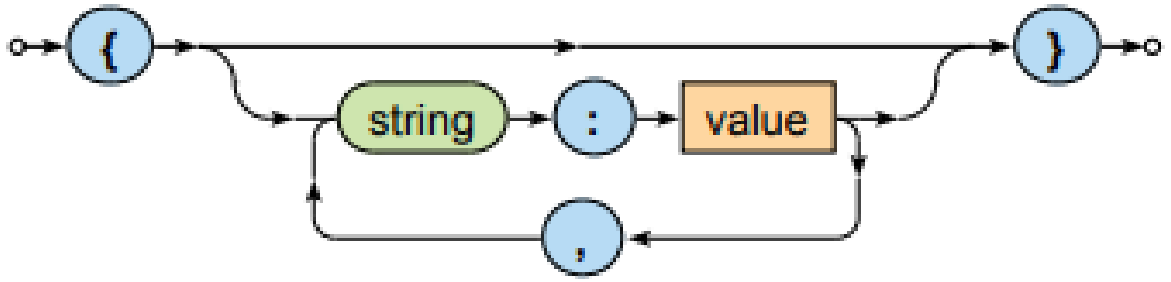
- JSON files are often used for transmitting data in web applications (e.g. sending some data from the server to the client)

```
{
  "title" : "Medvídek",
  "year" : 2007,
  "actors" : [
    {
      "firstname" : "Jiří",
      "lastname" : "Macháček"
    },
    {
      "firstname" : "Ivan",
      "lastname" : "Trojan"
    }
  ],
  "director" : {
    "firstname" : "Jan",
    "lastname" : "Hřebejk"
  }
}
```

JSON Structure

- Unordered collection of name-value pairs (properties)
  - Correspond to structures such as objects, records, structs, dictionaries, hash tables, keyed lists, associative arrays, …

- Values can be of different types, names should be unique



- Examples:

```
{ "name" : "Ivan Trojan", "year" : 1964 }
{ }
```

JSON Data Types

- There are 6 data types in JSON
  - Strings `"Hello World"`
  - Numbers `10  1.5  30  1.2e10`
  - Booleans `true  false`
  - Null `null`
  - Array `[ 1, 2, 3]  [ "Hello", "World" ]`
  - Object `{ "name" : "John" }  { "age" : 21 }`

JSON Example ... Again


```json
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 27,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ],
  "children": [
      "Catherine",
      "Thomas",
      "Trevor"
  ],
  "spouse": null
}
```

HTML/XML Files

- The main format for the web (though XML seems to be loosing a bit of popularity to JSON for use in APIs / file formats)
- XML files contain hiearchical content delineated by tags

```
<tag attribute="value">
    <subtag>
        Some content for the subtag
    </subtag>
    <openclosetag attribute="value2”/>
</tag>
```

- HTML is syntactically like XML but horrible (e.g., open tags are not always closed), more fundamentally, HTML is mean to describe appearance

JSON vs XML

- Which is better?
  - JSON has largely won over XML
  - You can occasionally still find hierarchical data as XML, but it usually is JSON.
  - XML is still relevant in data science for one reason: it is a generalization of HTML, the language used to specify webpages.

Web Scraping

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

- Web scraping is the process of **automatically extracting data** from websites.

- Why is Web Scaping Useful?
  - Collect data for research, analysis, and insights.
  - Track prices, news, and trends.
  - Gather large datasets for machine learning or statistical models.

- Common use cases:
  - Price comparison websites.
  - Sentiment analysis on social media or reviews.
  - Market research by extracting competitor data.
  - Academic research in data mining or behavior analysis.

Web Scraping – Before Getting Started

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

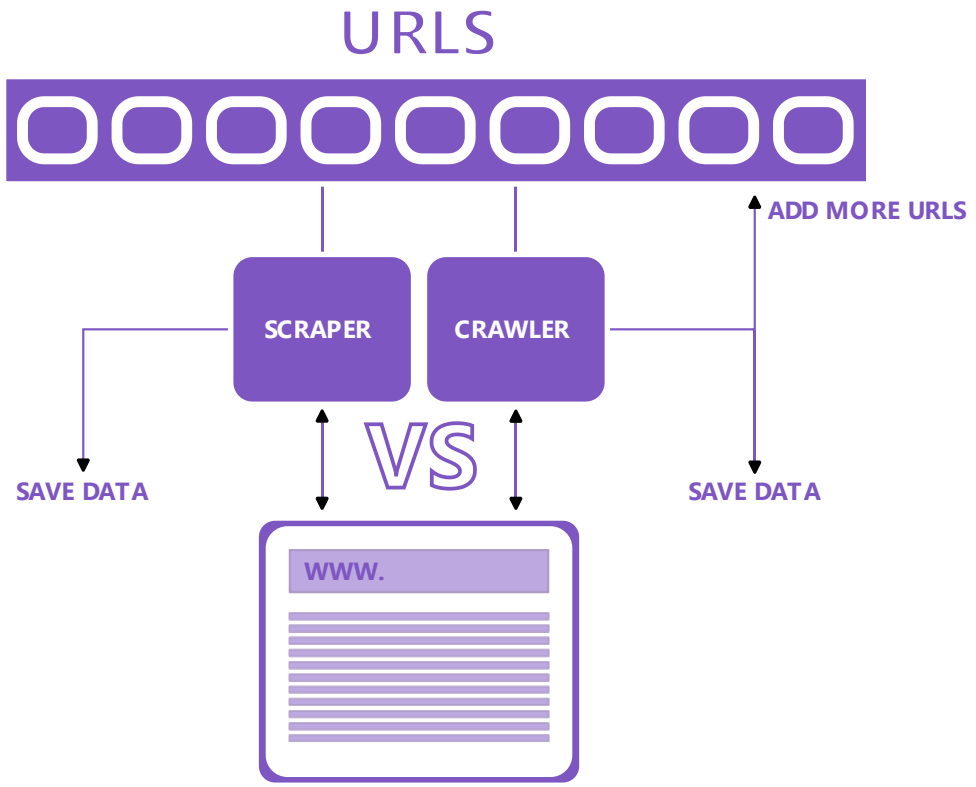
- Warnings and Ethical Considerations:
  - Respect `robots.txt` and Terms of Service
    - Always check if the website allows scraping by reviewing the `robots.txt` file and terms of service. - add /robots.txt to the end of the URL
  - Avoid Overloading Servers
    - Be considerate: Use rate limiting to prevent sending too many requests too quickly.
  - Data Privacy
    - Avoid scraping personal or sensitive data that is not publicly available or intended for scraping.
  - Legal Risks
    - Some sites explicitly prohibit scraping. Ignoring this can lead to legal consequences (e.g., bans, lawsuits).
  - Consider using APIs
    - Many websites offer public APIs that provide structured data legally and efficiently.

Web Scraping vs Web Crawling

- Web scraping and crawling are often used interchangeably but refer to different techniques.

- **Web scraping**: Any automated process that collects data from websites.

- **Web crawling:** A specific form of web scraping that involves systematically exploring websites, often for indexing purposes (e.g., search engines).



Devtools and Reverse Engineering

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

- The first step in web scraping is understanding how the target website works, often referred to as **reverse engineering**.

- **DevTools**: A development suite built into modern browsers used for debugging and inspecting web pages.
  - DevTools is essential for reverse engineering and is crucial in web scraping projects.

Devtools and Reverse Engineering

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

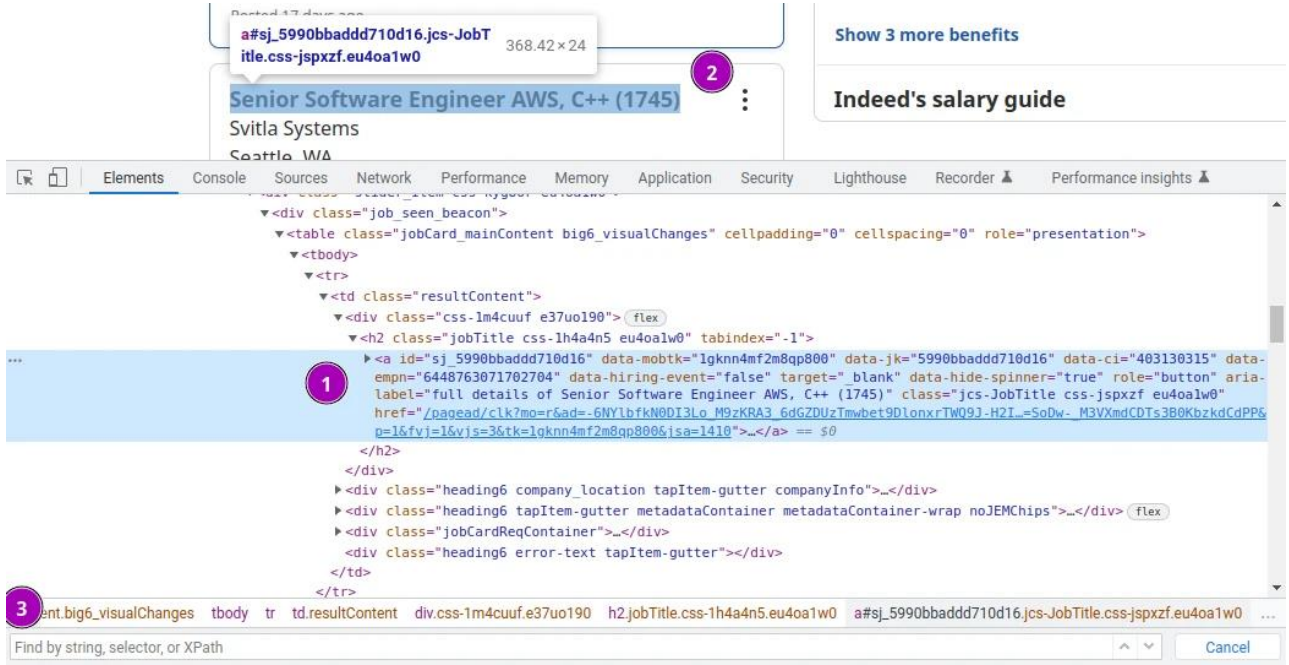
- The **DevTools console** can typically be opened using the **F12 key** (in Chrome and Firefox) or by right-clicking on the page and selecting **"Inspect"**.

- The tool suite is divided into multiple tabs, each offering different functionalities.

- For web scraping, the most relevant tab is the **Elements tab**, which allows you to inspect and analyze the structure of the webpage's HTML.

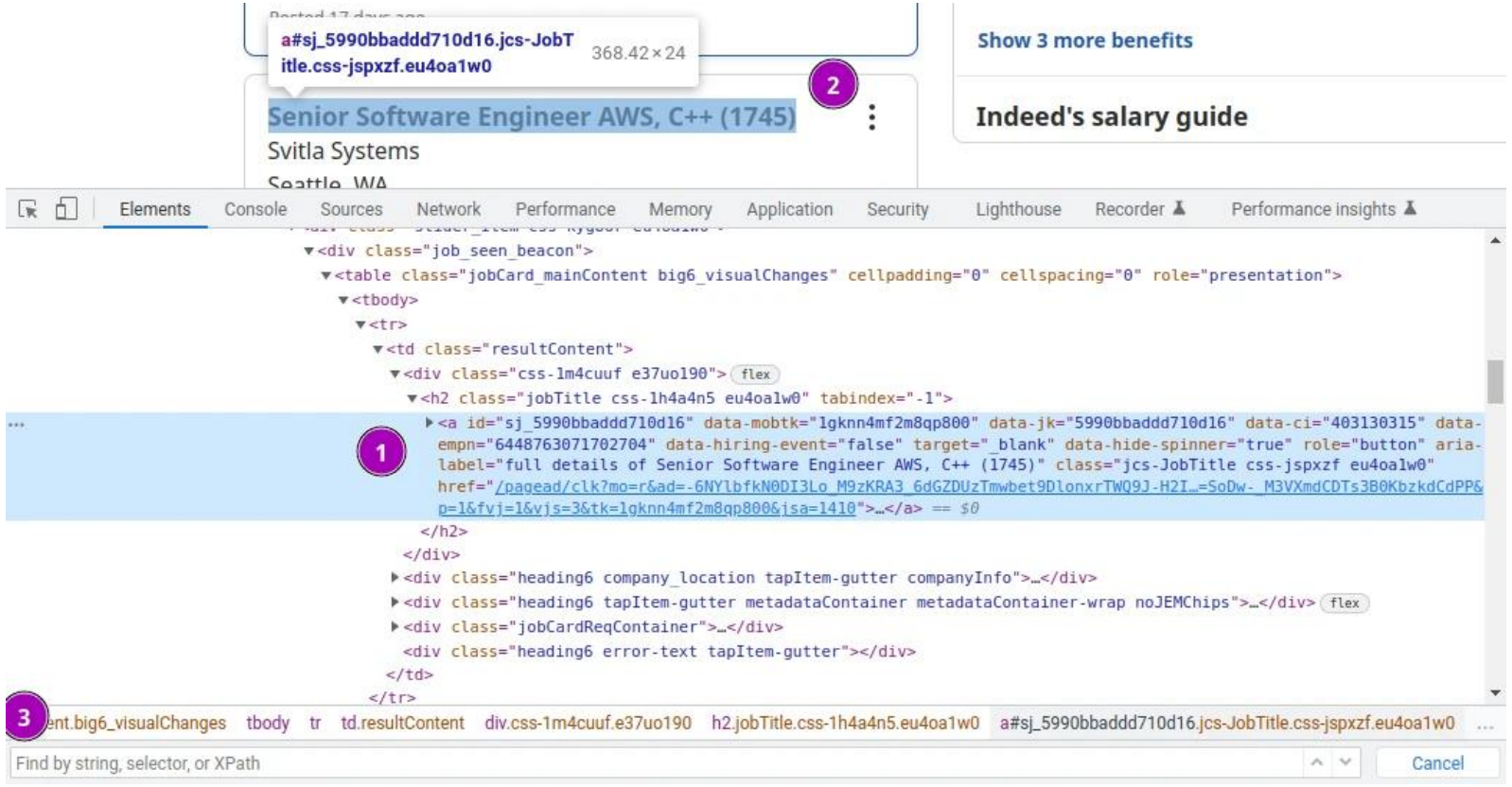
Elements Tab

- In web scraping, the **Elements** tab in DevTools helps in understanding the website's structure.
- You can use this tool to **build parsing logic** for your scraper by inspecting the HTML elements and identifying patterns.



Elements Tab

1. The navigation tree where we can see and interact with all of the HTML elements.
2. The selected elements will be highlighted on the page.
3. We can search for elements by CSS selectors, XPath or just text.



HTTP

- **HyperText Transfer Protocol (HTTP)** is the foundation of the web and essential for web scraping.
- Understanding HTTP is key to writing effective web scrapers, as it governs how we retrieve web pages.
- In Python, you can use HTTP to retrieve a webpage with a script like this one:

```
import httpx
response = httpx.get("http://httpbin.org/html")
print(response.text)
```

- The script above is sending a **request** to URL http://httpbin.org/html. In return we get a **response** from the server with the web page data.

HTTP

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

- The goal of a web scraper is to send a **valid HTTP request** to receive response data from the server.

- To ensure the request is valid, it must **match the server's expectations**.

- This means the requests should appear as if they are coming from a **real user** using a web browser.

URL

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

- A **Universal Resource Locator (URL)** indicates the address of a web resource.

Protocol	Host	Path	Query	Anchor
https	example.com	/path/to/resource	arg=value&arg2=value2	#anchor

- It consists of several key parts, each with a role in web scraping:
  - **Protocol**: Specifies how data is transferred (e.g., HTTP or HTTPS).
  - **Domain**: Identifies the website (e.g., www.example.com).
  - **Path**: Points to a specific page or resource on the website.
  - **Query Parameters**: Provide additional information or filters for the request (e.g., ?id=123).
  - **Anchor/Fragment**: Refers to a specific section of a page (e.g., #section1).

Request Types

- There are various types of HTTP requests, but in web scraping, we primarily use:
  - **GET**: Requests data from a specified resource (commonly used to retrieve web pages).
  - **POST**: Sends data to a server to create or update a resource (e.g., submitting forms).
  - **HEAD**: Similar to GET, but only retrieves metadata, not the actual content.

Response Status Code

- After sending a request, we will receive either a **success**, **failure**, or **timeout** response (servers can also ignore the request).
- Each response includes a **status code** indicating the outcome:
  - **200 range**: **Success!**
    - However, for websites with anti-scraping protection, a 200 response may be misleading—HTML content might indicate blocking.
  - **300 range**: **Redirection.**
    - The page location has changed, but most HTTP clients handle redirects automatically.
  - **400 range**: **Client-side error or blocking.**
    - This can indicate the server is blocking the scraper or that there's an issue with the request (e.g., missing headers, cookies, or a bad URL).
  - **500 range**: **Server-side error or blocking.**
    - This typically means the server is unable to process the request due to internal issues or blocking the client.

HTML

- Hypertext Markup Language: the language that provides a template for web pages
- See HTML when inspecting the source of a webpage
- Made up of tags that represent different elements (links, text, photos, etc)

HTML Tags

- <html>, indicates the start of an html page (end tag </html>)
- <body>, contains the items on the actual webpage (text, links, images, etc)
- <p>, the paragraph tag. Can contain text and links
- <a>, the link tag. Contains a link url, and possibly a description of the link
- <input>, a form input tag. Used for text boxes, and other user input
- <form>, a form start tag, to indicate the start of a form
- <img>, an image tag containing the link to an image
- Many other tags! Check it out: https://html.com/tags/

Web Scrapping

- Let's build our first web scraper!

- Follow the instructions in the notebook "**web_scraping_exercises.ipynb**".