Introduction to Webscraping

Theresa Gessler

last updated: 2020-01-15

Plan of the day

- if you have not, **install packages!!!**

Plan of the day

- if you have not, **install packages!!!**
- A introduction to the class
  - what is scraping
  - why do we scrape?
  - current debates about scraping
  - HTML
- B Scraping tables and static pages
- C selecting parts of pages
- D automation and more...

Plan of the day

- if you have not, **install packages!!!**
- A introduction to the class
  - what is scraping
  - why do we scrape?
  - current debates about scraping
  - HTML
- B Scraping tables and static pages
- C selecting parts of pages
- D automation and more...

How this course works

- **learning by doing**
  - slides with 'lecture'
  - doing exercises together and alone

Introduction

About me: Dr. Theresa Gessler

- Who am I?
  - postdoc at Digital Democracy Lab, University of Zurich
  - co-organizer of the Zurich Summer School for Women in Political Methodology

About me: Dr. Theresa Gessler

- Who am I?
  - postdoc at Digital Democracy Lab, University of Zurich
  - co-organizer of the Zurich Summer School for Women in Political Methodology

- my research
  - (digital) democracy
  - immigration
  - political parties

About me: Dr. Theresa Gessler

- Who am I?
  - postdoc at Digital Democracy Lab, University of Zurich
  - co-organizer of the Zurich Summer School for Women in Political Methodology

- my research
  - (digital) democracy
  - immigration
  - political parties

- teaching
  - webscraping, text analysis, data journalism
  - (& substantive courses)

About me: Dr. Theresa Gessler

- Who am I?
  - postdoc at Digital Democracy Lab, University of Zurich
  - co-organizer of the Zurich Summer School for Women in Political Methodology

- my research
  - (digital) democracy
  - immigration
  - political parties

- teaching
  - webscraping, text analysis, data journalism
  - (& substantive courses)

- contact
  - gessler@ipz.uzh.ch / www.theresagessler.github.io / @th_ges

Your turn

- name (infront of you?)
- research interests
- experience with
  - R
  - HTML
  - webscraping
- why are you taking this course?
  - any plans that include webscraping?

What is webscraping?

What is webscraping?

- extracting data from webpages
  - anything from university webpage to social media
  - lots of different techniques

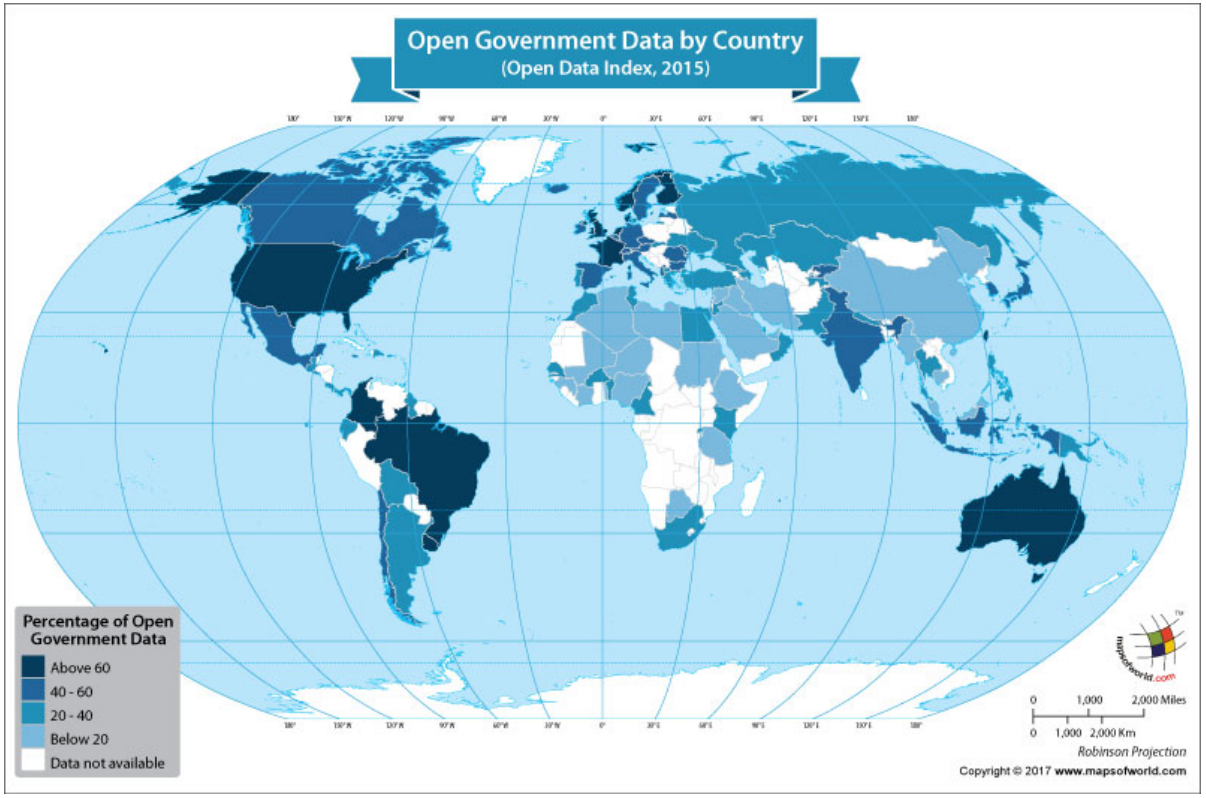
What is webscraping?

- extracting data from webpages
  - anything from university webpage to social media
  - lots of different techniques

- types of scraping
  - gathering as diverse information as possible from different pages vs. very specific scrapers
  - fully automated scrapers to half-automated scripts
  - single-use scraping vs. regular data collection

Why online data?

- increasing amount of public data online ('open government')

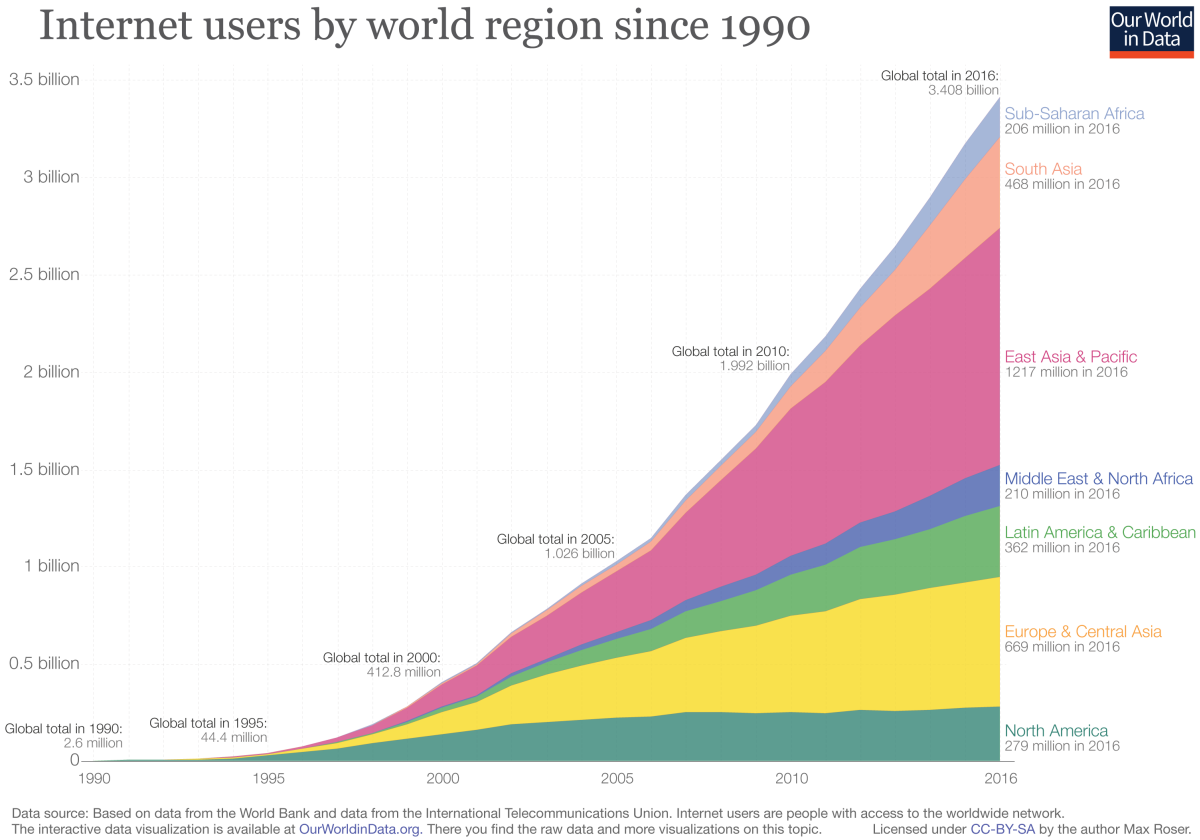


Why online data?

- increasing amount of politics happens online

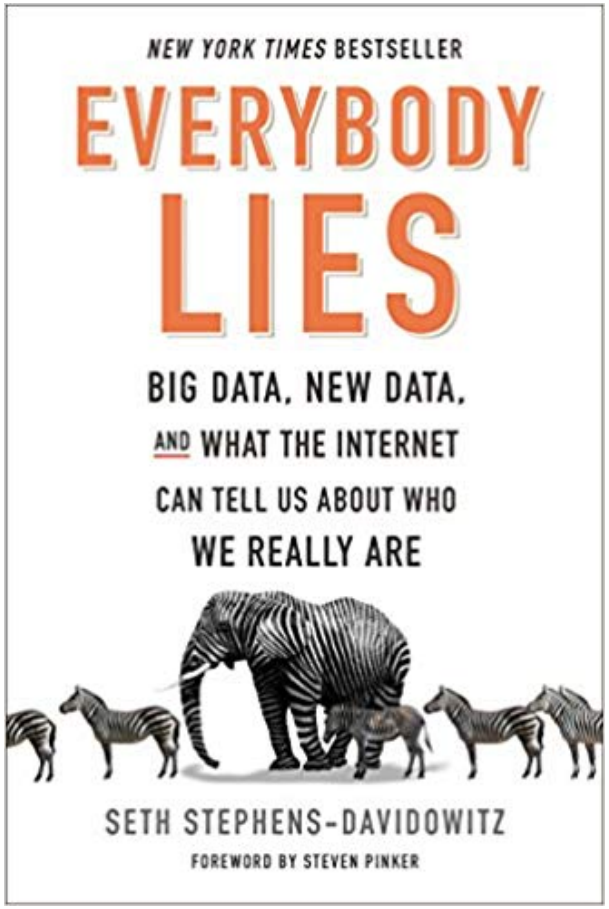
Why online data?

- increasing amount of people use the internet



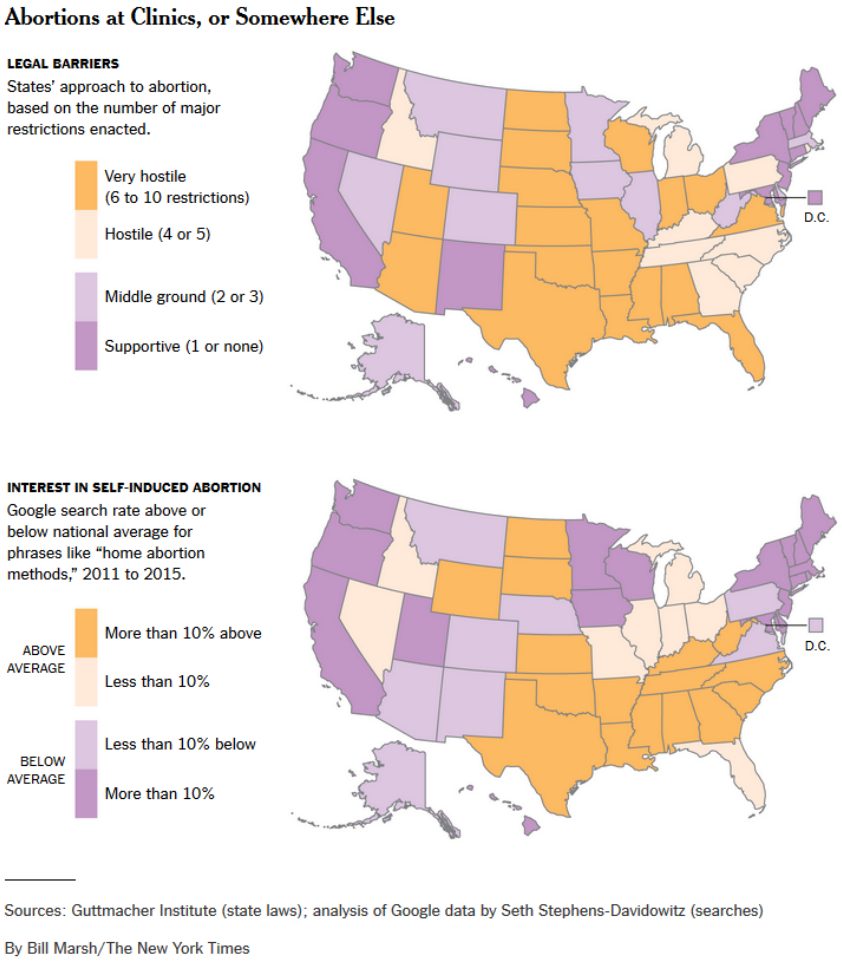
Why online data?

- we share everything online



Why online data?

- that makes real world phenomena more visible online



Why should you scrape?

- masses of data
- reproducible and renewable data collection
- once you learned it: simpler than manual data collection

Why should you scrape?

- masses of data
- reproducible and renewable data collection
- once you learned it: simpler than manual data collection

The role of webscraping in a research project

- something you often need to do at first
  - collecting first data points
  - seeing viability of project
  - one-off data collection

Why should you scrape?

- masses of data
- reproducible and renewable data collection
- once you learned it: simpler than manual data collection

The role of webscraping in a research project

- something you often need to do at first
  - collecting first data points
  - seeing viability of project
  - one-off data collection

- something you should not continuously be doing
  - less sustainable for large-scale projects
  - changes to webpages over time

Current debates about scraping

Is scraping legal? Is scraping ethical?

- complicated legal situation
  - 'terms of service'
  - freely accessible content vs. commercial content
  - measures that aim to prevent 'screen scraping'
  - data protection regulations like GDPR (and research exceptions)
  - e.g. recent 9th circuit ruling

Current debates about scraping

Is scraping legal? Is scraping ethical?

- complicated legal situation
  - 'terms of service'
  - freely accessible content vs. commercial content
  - measures that aim to prevent 'screen scraping'
  - data protection regulations like GDPR (and research exceptions)
  - e.g. recent 9th circuit ruling
- clear ethical boundaries
  - data protection: data means traces of individuals
  - right to be forgotten

Current debates about scraping

Is scraping legal? Is scraping ethical?

- complicated legal situation
  - 'terms of service'
  - freely accessible content vs. commercial content
  - measures that aim to prevent 'screen scraping'
  - data protection regulations like GDPR (and research exceptions)
  - e.g. recent 9th circuit ruling
- clear ethical boundaries
  - data protection: data means traces of individuals
  - right to be forgotten
- good practices
  - reading terms of services and considering non-intrusive ways to gather data
  - economic considerations: reducing traffic
  - secure storage vs. deletion of data
  - anonymization of users

Current debates about scraping

Do we need to scrape?

- 'post-API age' / APIcalypse
  - companies restrict data access and inhibit independent research

Current debates about scraping

Do we need to scrape?

- 'post-API age' / APIcalypse
  - companies restrict data access and inhibit independent research

→ researchers as watchdogs

Current debates about scraping

Do we need to scrape?

- 'post-API age' / APIcalypse
  - companies restrict data access and inhibit independent research

→ researchers as watchdogs

Three pieces of advice from Freelon (2018)

- use authorized methods whenever possible
- do not confuse terms of service compliance with data protection
- understand the risks of violating terms of service

HTML

Browsing vs. scraping

- browsing
  - you click on something
  - browser sends request to server that hosts webpage
  - server returns resource (e.g. HTML document)
  - browser interprets HTML and renders it in a nice fashion

Browsing vs. scraping

- browsing
  - you click on something
  - browser sends request to server that hosts webpage
  - server returns resource (e.g. HTML document)
  - browser interprets HTML and renders it in a nice fashion

- scraping with R
  - you manually specify a resource
  - R sends request to server that hosts website
  - server returns resource
  - R parses HTML (i.e., interprets the structure), but does not render it in a nice fashion
  - you tell R which parts of the structure to focus on and what to extract

Browsing vs. scraping

- browsing
  - you click on something
  - browser sends request to server that hosts webpage
  - server returns resource (e.g. HTML document)
  - browser interprets HTML and renders it in a nice fashion

- scraping with R
  - you manually specify a resource
  - R sends request to server that hosts website
  - server returns resource
  - R parses HTML (i.e., interprets the structure), but does not render it in a nice fashion
  - you tell R which parts of the structure to focus on and what to extract

→ First step is to learn to understand some HTML

HTML: The basics

- **H**yper **T**ext **M**arkup **L**anguage
  - *markup*: additional description of formatting beyond the content of the text
- language consists of **HTML tags** to specify character / behaviour of text
- HTML tags typically consist of a starting and an end tag (exceptions: images, line breaks etc.)
- they surround the text they are formatting

Example:

```
<tagname>Content goes here...</tagname>
```

- example page we will use: http://quotes.toscrape.com/

Example: In the browser

Quotes to Scrape

Login

"The world as we have created it is a process of our thinking. It cannot be changed without changing our thinking."

by **Albert Einstein** (about)

Tags: change deep-thoughts thinking world

"It is our choices, Harry, that show what we truly are, far more than our abilities."

by **J.K. Rowling** (about)

Tags: abilities choices

"There are only two ways to live your life. One is as though nothing is a miracle. The other is as though everything is a miracle."

by **Albert Einstein** (about)

Tags: inspirational life live miracle miracles

Top Ten tags

love
inspirational
life
humor
books
reading
friendship
friends
truth
smile

Example: HTML Code

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Quotes to Scrape</title>
    <link rel="stylesheet" href="/static/bootstrap.min.css">
    <link rel="stylesheet" href="/static/main.css">
</head>
<body>
    <div class="container">
        <div class="row header-box">
            <div class="col-md-8">
                <h1>
                    <a href="/" style="text-decoration: none">Quotes to Scrape</a>
                </h1>
            </div>
            <div class="col-md-4">
                <p>

                    <a href="/login">Login</a>

                </p>
            </div>
        </div>


<div class="row">
    <div class="col-md-8">

    <div class="quote" itemscope itemtype="http://schema.org/CreativeWork">
        <span class="text" itemprop="text">“The world as we have created it is a process of our thinking. It cannot be changed without
changing our thinking.”</span>
        <span>by <small class="author" itemprop="author">Albert Einstein</small>
        <a href="/author/Albert-Einstein">(about)</a>
        </span>
        <div class="tags">
            Tags:
            <meta class="keywords" itemprop="keywords" content="change,deep-thoughts,thinking,world" /    >

            <a class="tag" href="/tag/change/page/1/">change</a>

            <a class="tag" href="/tag/deep-thoughts/page/1/">deep-thoughts</a>

            <a class="tag" href="/tag/thinking/page/1/">thinking</a>

            <a class="tag" href="/tag/world/page/1/">world</a>

        </div>
    </div>
```

Basic HTML tags

```
<html>
    <head>
        <title>Title of your web page</title>
    </head>
    <body>
    HTML web page content
    </body>
</html>
```

Basic HTML tags

```
<html>
    <head>
        <title>Title of your web page</title>
    </head>
    <body>
    HTML web page content
    </body>
</html>
```

- we are mostly interested in what is inside the **body**, that is, the content of a webpage
- **head** gives meta information, often used by search engines
- tags can be **nested**

Basic HTML Tags: Headings

Headings are defined by numbered h tags. Examples (with code and outcome):

```
<h1> your heading</h1>
```

your heading

```
<h2> a smaller heading</h2>
```

a smaller heading

```
<h3> an even smaller heading</h3>
```

an even smaller heading

```
<h4> an even smaller heading</h4>
```

Basic HTML Tags: Paragraphs

Paragraphs are defined by div or p tags.

Examples:

```
<p>this is a paragraph.</p><p>and this is the next.</p>
```

this is a paragraph.

and this is the next.

Basic HTML Tags: Paragraphs

Paragraphs are defined by div or p tags.

Examples:

```
<p>this is a paragraph.</p><p>and this is the next.</p>
```

this is a paragraph.

and this is the next.

```
<div>this is a paragraph.</div><div>and this is the next.
</div>
```

this is a paragraph.
and this is the next.

Basic HTML Tags: Attributes

- All HTML elements can have attributes
- Attributes provide additional information about an element
  - they are included inside the tag

Basic HTML Tags: Attributes

- All HTML elements can have attributes
- Attributes provide additional information about an element
  - they are included inside the tag

Usage

- they are always specified in the starting tag
  - e.g. `<title attribute="x"> Title </title>`
- Attributes usually come in name and value pairs
  - e.g. attributename="attributevalue"

Basic HTML Tags: Attributes - Links

- Most common case of attributes: **links**
  - text or images turned into a link by surrounding `<a>` tag (*anchor*)
  - link address specified as href attribute (*hyperreference*)

Example:

```
This is text <a href="http://quotes.toscrape.com/">with a
link</a>.
```

This is text with a link.

Basic HTML Tags: Attributes

- other examples of attributes
  - alt: descriptions, e.g. for images
    - when image is missing, they will be written out
    - descriptions for users with visual impairments
  - styles: formatting

Examples:

```
<img src="no_smiley.jpg" alt="Image that does not exist.">
```

```
<p style="color:red">This is a paragraph.</p>
```

This is a paragraph.

Basic HTML Tags: Classes

- **Classes** are another special case of attributes that is used for formatting
  - usage within tags:

```
<div class="container"> This is the text</div>
```

This is the text

Basic HTML Tags: Classes

- **Classes** are another special case of attributes that is used for formatting
  - usage within tags:

```
<div class="container"> This is the text</div>
```

This is the text

Styling with Classes

WEbpages like blogs often define **Styles** and apply them to classes across the whole webpage. This use of classes is very common because it reduces the risk of accidentally formatting one instance of a repeated element differently.

```
<style>
p.error {
  color: red;   border: 1px solid red;
}
</style>
<p class="error">Red highlight</p>
```

Red highlight

Example: Quotes to scrape Webpage

Have another look at the webpage - do you understand more now?

Scraping tables and static pages

rvest: The Swiss army knife of scraping

- r package for scraping
- strengths
  - covers most frequent use cases
  - integration with other packages, e.g. tidyverse
- weaknesses
  - relatively simple: no dynamic webpages

rvest: The Swiss army knife of scraping

- r package for scraping
- strengths
  - covers most frequent use cases
  - integration with other packages, e.g. tidyverse
- weaknesses
  - relatively simple: no dynamic webpages

Main uses

- Tables
- Texts
- extracting links
- (downloading files)

Overview of rvest commands

Limited set of commands:

- `read_html()`
- `html_nodes()` (`html_node()`)
- `html_text()`
- `html_table()`
- `html_attrs()` (`html_attr()`)

'Parsing' HTML

'Parsing' HTML

- scraping with R
  - you manually specify a resource
  - R sends request to server that hosts website
  - server returns resource
  - R parses HTML (i.e., interprets the structure), but does not render it in a nice fashion
  - you tell R which parts of the structure to focus on and what to extract

'Parsing' HTML

- scraping with R
  - you manually specify a resource
  - R sends request to server that hosts website
  - server returns resource
  - R parses HTML (i.e., interprets the structure), but does not render it in a nice fashion
  - you tell R which parts of the structure to focus on and what to extract
- Example: http://quotes.toscrape.com/

'Parsing' HTML

- scraping with R
  - you manually specify a resource
  - R sends request to server that hosts website
  - server returns resource
  - R parses HTML (i.e., interprets the structure), but does not render it in a nice fashion
  - you tell R which parts of the structure to focus on and what to extract

- Example: http://quotes.toscrape.com/

```
library(rvest)
url <- "http://quotes.toscrape.com/"
parsed <- read_html(url)
```

'Parsing' HTML

- scraping with R
  - you manually specify a resource
  - R sends request to server that hosts website
  - server returns resource
  - R parses HTML (i.e., interprets the structure), but does not render it in a nice fashion
  - you tell R which parts of the structure to focus on and what to extract

- Example: http://quotes.toscrape.com/

```
library(rvest)
url <- "http://quotes.toscrape.com/"
parsed <- read_html(url)
```

We practice this together in R

Tables

Tables in HTML

- syntax for tables in HTML highly standardized
- out of the box, `rvest` converts tables into data frames when calling `html_table()`

Tables in HTML

- syntax for tables in HTML highly standardized
- out of the box, `rvest` converts tables into data frames when calling `html_table()`

Why should we still learn about the structure of Tables?

- some content within tables gets lost with `html_table()` command
  - images
  - hyperlinks
- sometimes webpages do not conform with code standards (e.g. incoherent number of cells)

HTML Table Basics

- environment defined by `<table>` tags
  - `<th>` contains *table headers*
  - `<tr>` generates new *table row*
  - `<td>` contains *table data* (i.e. each column within row)
- styles may be used for individual cells, rows or entire table

HTML Table Basics

- environment defined by `<table>` tags
  - `<th>` contains *table headers*
  - `<tr>` generates new *table row*
  - `<td>` contains *table data* (i.e. each column within row)
- styles may be used for individual cells, rows or entire table

Table Skeleton

```
<table>

    <tr> <th> </th> <th> </th> <th> </th> </tr>
    <tr> <td> </td> <td> </td> <td> </td> </tr>
    <tr> <td> </td> <td> </td> <td> </td> </tr>

</table>
```

An HTML Table

```
 <table>
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

An HTML Table

Can you create your own?

An HTML Table

Can you create your own?

Firstname	Lastname	Age
Jill	Smith	50
Eve	Jackson	94

Another HTML Table

```
<table>
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
    <td>currently travelling</td>
  </tr>
</table>
```

Another HTML Table

Firstname	Lastname	Age	
Jill	Smith	50	
Eve	Jackson	94	currently travelling

Another HTML Table

Firstname	Lastname	Age	
Jill	Smith	50	
Eve	Jackson	94	currently travelling

- we practice this in R!
- using `html_table()` command
  - specify `fill=TRUE` to deal with irregular rows

CSS Selectors

CSS Selectors

- we use the *appearance* / *style* of text to select specific parts
- based on any of the elements we learned about
  - tags
  - classes
  - attributes
- CSS selectors provide a *language* in which we can describe what we select at a more abstract level

The process of scraping

```
url <- "http://www.ceu.edu"
page <- read_html(url)
```

Specify URL

‚parse' URL

The process of scraping

```
url <- "http://www.ceu.edu"
```

Specify URL

```
page <- read_html(url)
```

‚parse' URL

Keep whole page

Select parts

```
sel_nodes <- html_nodes(page)
```

Basic selectors

element	Type selector	Matches an element
.class	Class selector	Matches the value of a class attribute
#id	ID selector	Matches the value of an id attribute
*	universal selector	Matches everything.
[attribute]	attribute selector	Matches elements containing a given attribute
[attribute=value]	attribute selector	Matches elements containing a given attribute with a given value

More complex attribute selectors

[attribute*=value]	Matches elements with an attribute that contains a given value	a[href*="pressrelease"]
[attribute^="value"]	Matches elements with an attribute that starts with a given value	a[href*="/press/"]
[attribute$="value"]	Matches elements with an attribute that ends with a given value	[href$=".pdf"]

Combining CSS Selectors

There are several ways to combine CSS Selectors:

element,element	Selects all elements and all <p> elements	div, p
element element	Selects all <p> elements inside elements	div p
element>element	Selects all <p> elements where the parent is a <>div> element	div > p
element+element	Selects all <p> elements that are placed immediately after elements	div + p
element1~element2	Selects every element that are preceded by a <p> element	p ~ ul

Combining CSS Selectors

There are several ways to combine CSS Selectors:

element,element	Selects all elements and all <p> elements	div, p
element element	Selects all <p> elements inside elements	div p
element>element	Selects all <p> elements where the parent is a <>div> element	div > p
element+element	Selects all <p> elements that are placed immediately after elements	div + p
element1~element2	Selects every element that are preceded by a <p> element	p ~ ul

Dine at the CSS Diner

Extracting links from webpages

Extracting links from webpages

- unlike a book that we read cover to cover, webpages distribute information over multiple pages
- *hyperlinks* connect one page to the others → we follow them by clicking
  - we need to deal with this differently when scraping

The process of scraping

```
url <- "http://www.ceu.edu"
page <- read_html(url)
```

Specify URL

‚parse' URL

The process of scraping

```
url <- "http://www.ceu.edu"
```

Specify URL

```
page <- read_html(url)
```

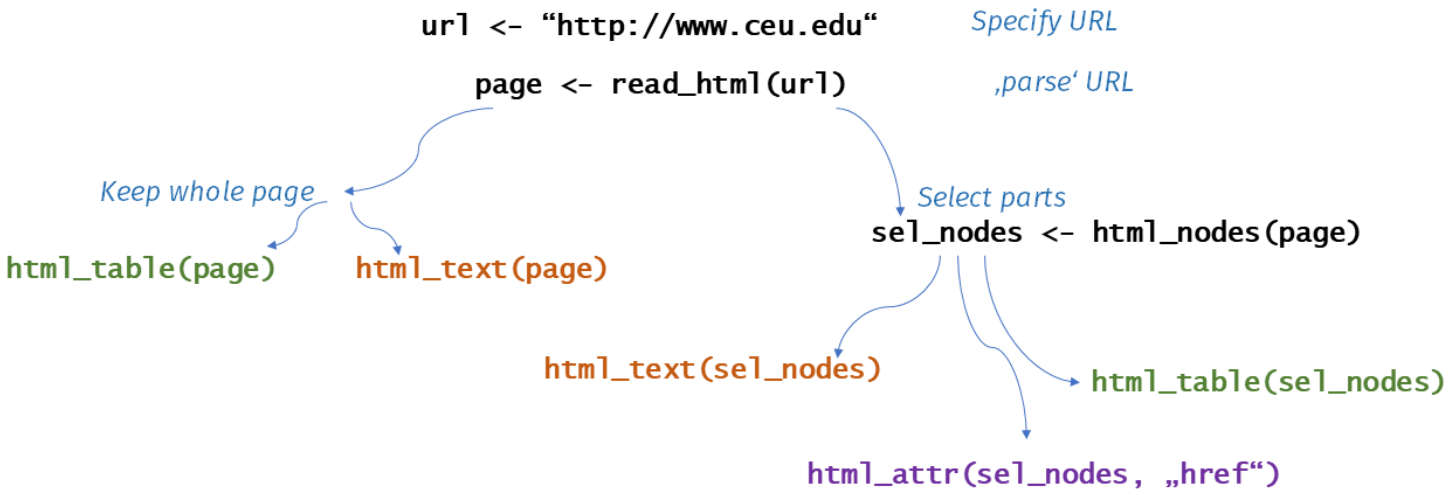
‚parse' URL

Keep whole page

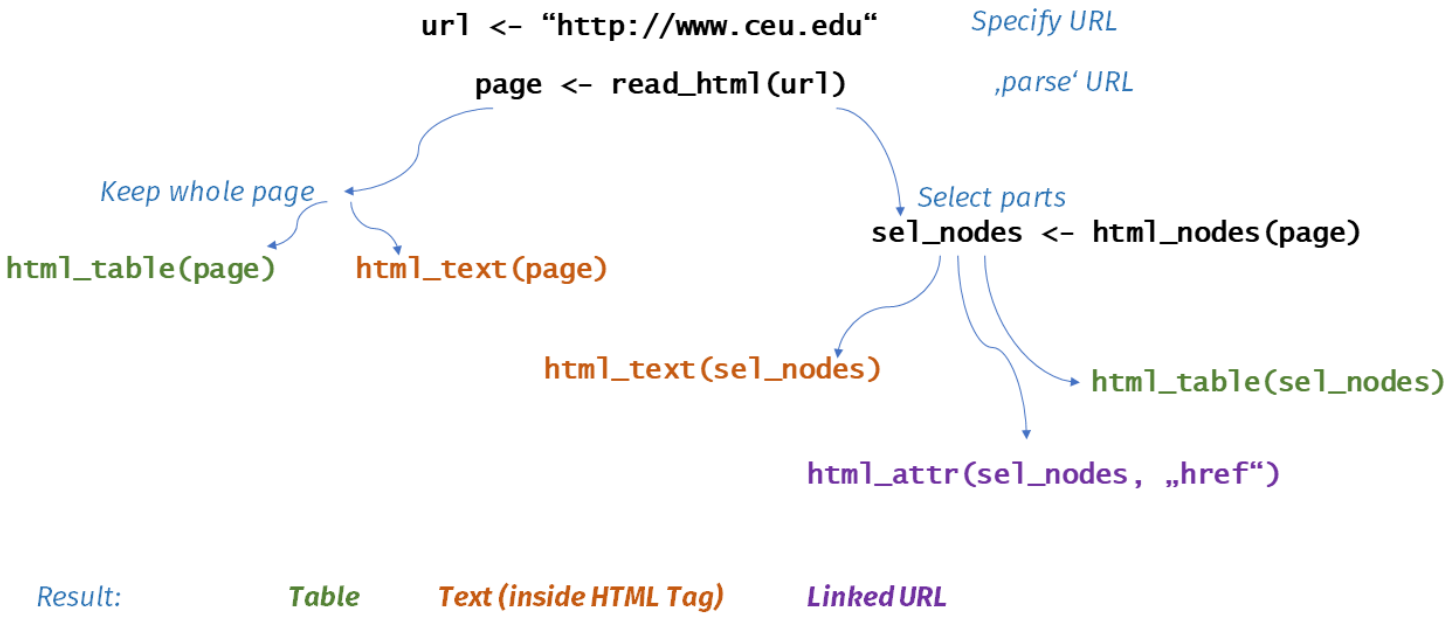
Select parts

```
sel_nodes <- html_nodes(page)
```

The process of scraping



The process of scraping



Links in HTML

- We discussed links as a common case of attributes
  - content turned into a link with `<a>` tag (*anchor*)
  - link address specified as href attribute (*hyperreference*)

--

This is text with a link.

```
This is text <a href="http://quotes.toscrape.com/">with a
link</a>.
```

Extracting links with rvest

- extracting the text of the link
  - `html_nodes(parsed_page,"a") %>% html_text()`
- extracting the attribute of the link (the hyperreference)
  - `html_nodes(parsed_page,"a") %>% html_attr("href")`

Extracting links with rvest

- extracting the text of the link
  - `html_nodes(parsed_page,"a") %>% html_text()`
- extracting the attribute of the link (the hyperreference)
  - `html_nodes(parsed_page,"a") %>% html_attr("href")`

Caution: Link is attribute of `<a>`-Tag!

```
html_attr(parsed_page,"href")
```

vs.

```
html_nodes(parsed_page,"a") %>% html_attr("href")
```

Automation

Extracting links with rvest

Budapest (/ˈbuːdəpɛst/, Hungarian pronunciation: [ˈbudɒpɛʃt]) is the capital and the most populous city of Hungary, and the tenth-largest city in the European Union by population within city limits.[9][10][11] The city had an estimated population of 1,752,286 in 2019 distributed over a land area of about 525 square kilometres (203 square miles).[12] Budapest is both a city and county, and forms the centre of the Budapest metropolitan area, which has an area of 7,626 square kilometres (2,944 square miles) and a population of 3,303,786, comprising 33 percent of the population of Hungary.[13][14]

https://en.wikipedia.org/wiki/Help:IPA/English
https://en.wikipedia.org/wiki/Help:IPA/Hungarian
https://en.wikipedia.org/wiki/Capital_city
https://en.wikipedia.org/wiki/List_of_cities_and_towns_of_Hungary
https://en.wikipedia.org/wiki/Hungary
https://en.wikipedia.org/wiki/Largest_cities_of_the_European_Union_by_population_within_city_limits
https://en.wikipedia.org/wiki/European_Union
https://en.wikipedia.org/wiki/Budapest#cite_note-TIME2-9
https://en.wikipedia.org/wiki/Budapest#cite_note-10
https://en.wikipedia.org/wiki/Budapest#cite_note-11
https://en.wikipedia.org/wiki/Budapest#cite_note-Encarta-12
https://en.wikipedia.org/wiki/List_of_cities_and_towns_of_Hungary
https://en.wikipedia.org/wiki/Counties_of_Hungary
https://en.wikipedia.org/wiki/Budapest_metropolitan_area
https://en.wikipedia.org/wiki/Budapest#cite_note-13
https://en.wikipedia.org/wiki/Budapest#cite_note-14

Extracting links with rvest

Budapest (/ˈbuːdəpɛst/, Hungarian pronunciation: [ˈbudɒpɛʃt]) is the capital and the most populous city of Hungary, and the tenth-largest city in the European Union by population within city limits.[9][10][11] The city had an estimated population of 1,752,286 in 2019 distributed over a land area of about 525 square kilometres (203 square miles).[12] Budapest is both a city and county, and forms the centre of the Budapest metropolitan area, which has an area of 7,626 square kilometres (2,944 square miles) and a population of 3,303,786, comprising 33 percent of the population of Hungary.[13][14]

!!! Too many links !!!

https://en.wikipedia.org/wiki/Help:IPA/English
https://en.wikipedia.org/wiki/Help:IPA/Hungarian
https://en.wikipedia.org/wiki/Capital_city
https://en.wikipedia.org/wiki/List_of_cities_and_towns_of_Hungary
https://en.wikipedia.org/wiki/Hungary
https://en.wikipedia.org/wiki/Largest_cities_of_the_European_Union_by_population_within_city_limits
https://en.wikipedia.org/wiki/European_Union
https://en.wikipedia.org/wiki/Budapest#cite_note-TIME2-9
https://en.wikipedia.org/wiki/Budapest#cite_note-10
https://en.wikipedia.org/wiki/Budapest#cite_note-11
https://en.wikipedia.org/wiki/Budapest#cite_note-Encarta-12
https://en.wikipedia.org/wiki/List_of_cities_and_towns_of_Hungary
https://en.wikipedia.org/wiki/Counties_of_Hungary
https://en.wikipedia.org/wiki/Budapest_metropolitan_area
https://en.wikipedia.org/wiki/Budapest#cite_note-13
https://en.wikipedia.org/wiki/Budapest#cite_note-14

Extracting links with rvest

Budapest (/ˈbuːdəpɛst/, Hungarian pronunciation: [ˈbudɒpɛʃt]) is the capital and the most populous city of Hungary, and the tenth-largest city in the European Union by population within city limits.[9][10][11] The city had an estimated population of 1,752,286 in 2019 distributed over a land area of about 525 square kilometres (203 square miles).[12] Budapest is both a city and county, and forms the centre of the Budapest metropolitan area, which has an area of 7,626 square kilometres (2,944 square miles) and a population of 3,303,786, comprising 33 percent of the population of Hungary.[13][14]

!!! Too many links !!!

https://en.wikipedia.org/wiki/Help:IPA/English
https://en.wikipedia.org/wiki/Help:IPA/Hungarian
https://en.wikipedia.org/wiki/Capital_city
https://en.wikipedia.org/wiki/List_of_cities_and_towns_of_Hungary
https://en.wikipedia.org/wiki/Hungary
https://en.wikipedia.org/wiki/Largest_cities_of_the_European_Union_by_population_within_city_limits
https://en.wikipedia.org/wiki/European_Union
https://en.wikipedia.org/wiki/Budapest#cite_note-TIME2-9
https://en.wikipedia.org/wiki/Budapest#cite_note-10
https://en.wikipedia.org/wiki/Budapest#cite_note-11
https://en.wikipedia.org/wiki/Budapest#cite_note-Encarta-12
https://en.wikipedia.org/wiki/List_of_cities_and_towns_of_Hungary
https://en.wikipedia.org/wiki/Counties_of_Hungary
https://en.wikipedia.org/wiki/Budapest_metropolitan_area
https://en.wikipedia.org/wiki/Budapest#cite_note-13
https://en.wikipedia.org/wiki/Budapest#cite_note-14

→ How do we get from one page to multiple?

Automation

- repetition of code across different units

Automation

- repetition of code across different units

→ `for`-loops

Automation

- repetition of code across different units

→ `for`-loops

→ `apply()` with functions

for-loops

```
for (i in vector) {
    code with i
}
```

For-Loops

Example

for-loops

Your pick: index that varies

variation of index

Your code, generalized to work with the arguments

```
for (i in vector) {
    code with i
}
```

```
for (i in VECTOR){ do something with i }
```

`for`-loops

Advantages

- easy to write
- do not require full translation of code into functions
- easy to interrupt and continue for prolonged scraping

`for`-loops

Advantages

- easy to write
- do not require full translation of code into functions
- easy to interrupt and continue for prolonged scraping

Disadvantages

- become inefficient for high numbers of iterations
- no swag: stackoverflow: Are For loops evil in R?

`for`-loops

Advantages

- easy to write
- do not require full translation of code into functions
- easy to interrupt and continue for prolonged scraping

Disadvantages

- become inefficient for high numbers of iterations
- no swag: stackoverflow: Are For loops evil in R?

Good to know

- loops with `for` are just the most well-known type of loop
  - `while` loops
  - `repeat` loops
  - `break` and `next` clauses

Apply functions

- family of commands in base R
  - `apply()`, `sapply()`, `lapply()`, `mapply()`, `rapply()`, `tapply()`, `vapply()`
- apply a function to each element of an object (e.g. a list)
  - need to define a function
- like for-loops, useful for repeated tasks
- we focus on `sapply()` that returns as simple output as possible

sapply()

```
sapply(object, function,
    simplify=TRUE,USE.NAMES=TRUE,…)
```

sapply()

vector, e.g. all pages to scrape from

Function that has your code

```
sapply(object, function,
    simplify=TRUE,USE.NAMES=TRUE,…)
```

Place where you add if needed

Apply functions

Apply functions

Advantages

- memory & time efficient
- encourages you to write functions

Disadvantages

- challenging to understand what is output
- output often a list
- more difficult to trouble-shoot
- need to define a function

Good to know

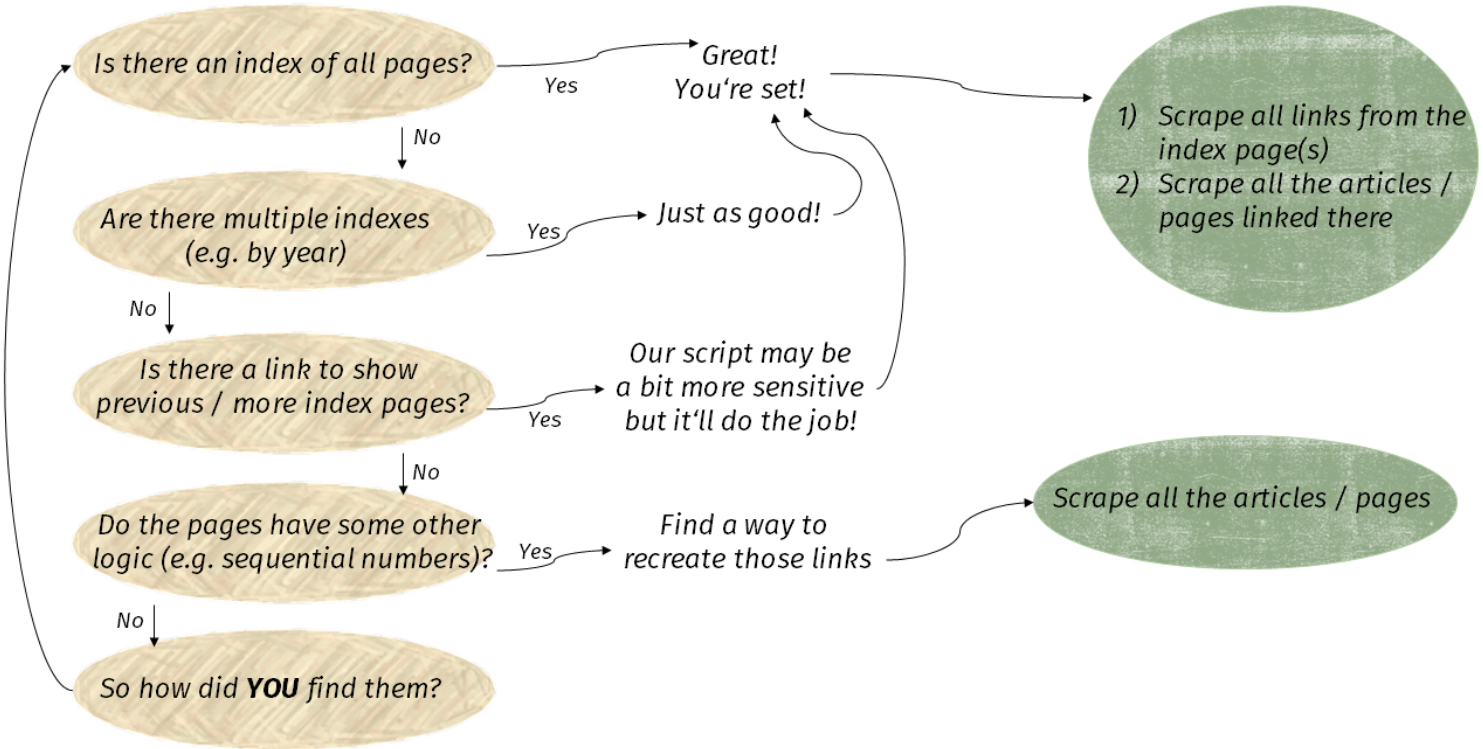
- lots of tutorials for the different apply-functions exist online - just google

The logic of scraping multiple pages

For scraping multiple pages, we need to find an **index page**. These pages

- come in many shapes and forms
- we will need to scrape at least twice:
  - index page
  - article pages
- both scraping procedures will have different code
- when index page consists of multiple pages, we may even add a third round

The logic of scraping multiple pages



Index pages: wikipedia

List of members of the Swiss Council of States (2007–11)

From Wikipedia, the free encyclopedia

This is a list of members of the Swiss Council of States of the 48th legislature (2007-2011). Most members were elected in the 2007 Swiss federal election.

Contents [hide]

Current members [edit]

C.	Councillor	Party	W.	Notes
AG	Christine Egerszegi	FDP.The Liberals	[1]	
AG	Maximilian Reimann	Swiss People's Party	[2]	
AI	Ivo Bischofberger	Christian Democratic People's Party		Reelected
AR	Hans Altherr	FDP.The Liberals		Reelected
BE	Simonetta Sommaruga	Social Democratic Party		Reelected
BE	Werner Luginbühl	Conservative Democratic Party		Elected for the Swiss People's Party
BL	Claude Janiak	Social Democratic Party		Reelected
BS	Anita Fetz	Social Democratic Party		Reelected
FR	Urs Schwaller	Christian Democratic People's Party		Reelected
FR	Alain Berset	Social Democratic Party		Reelected; Vice-President 2007/2008
GE	Liliane Maury Pasquier	Social Democratic Party		
GE	Robert Cramer	Green Party		
GL	This Jenny	Swiss People's Party		
GL	Pankraz Freitag	FDP.The Liberals		Elected in February 2008 to succeeded to Fritz Schiesser
GR	Theo Maissen	Christian Democratic People's Party		Reelected
GR	Christoffel Brändli	Swiss People's Party		Reelected; President 2007/2008
JU	Claude Hêche	Social Democratic Party		
JU	Anne Seydoux-Christe	Christian Democratic People's Party		
LU	Helen Leumann-Würsch	FDP.The Liberals		Reelected

- table with links to wikipedia pages
- Source: wikipedia

Index pages: wikipedia

List of members of the Swiss Council of States (2007–11)

From Wikipedia, the free encyclopedia

This is a list of members of the Swiss Council of States of the 48th legislature (2007-2011). Most members were elected in the 2007 Swiss federal election.

Contents [hide]

Current members [edit]

C.	Councillor	Party	W.	Notes
AG	Christine Egerszegi	FDP.The Liberals	[1]	
AG	Maximilian Reimann	Swiss People's Party	[2]	
AI	Ivo Bischofberger	Christian Democratic People's Party		Reelected
AR	Hans Altherr	FDP.The Liberals		Reelected
BE	Simonetta Sommaruga	Social Democratic Party		Reelected
BE	Werner Luginbühl	Conservative Democratic Party		Elected for the Swiss People's Party
BL	Claude Janiak	Social Democratic Party		Reelected
BS	Anita Fetz	Social Democratic Party		Reelected
FR	Urs Schwaller	Christian Democratic People's Party		Reelected
FR	Alain Berset	Social Democratic Party		Reelected; Vice-President 2007/2008
GE	Liliane Maury Pasquier	Social Democratic Party		
GE	Robert Cramer	Green Party		
GL	This Jenny	Swiss People's Party		
GL	Pankraz Freitag	FDP.The Liberals		Elected in February 2008 to succeeded to Fritz Schiesser
GR	Theo Maissen	Christian Democratic People's Party		Reelected
GR	Christoffel Brändli	Swiss People's Party		Reelected; President 2007/2008
JU	Claude Hêche	Social Democratic Party		
JU	Anne Seydoux-Christe	Christian Democratic People's Party		
LU	Helen Leumann-Würsch	FDP.The Liberals		Reelected

- table with links to wikipedia pages
- Source: wikipedia
- single index page

Index pages: White House Briefings

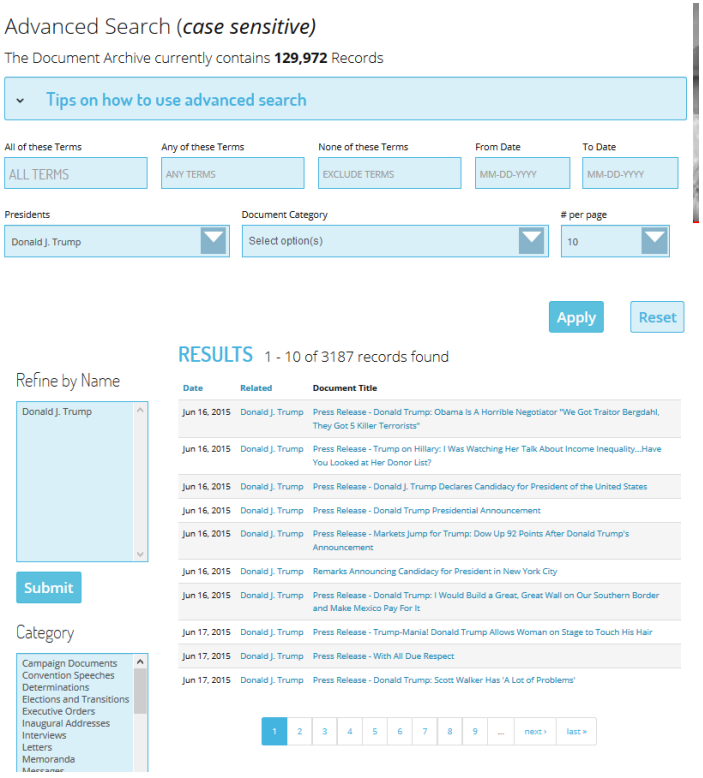
- simple list published in multiple pages, links to statements
- Source: White House Briefing Statements

Index pages: White House Briefings

- simple list published in multiple pages, links to statements
- Source: White House Briefing Statements

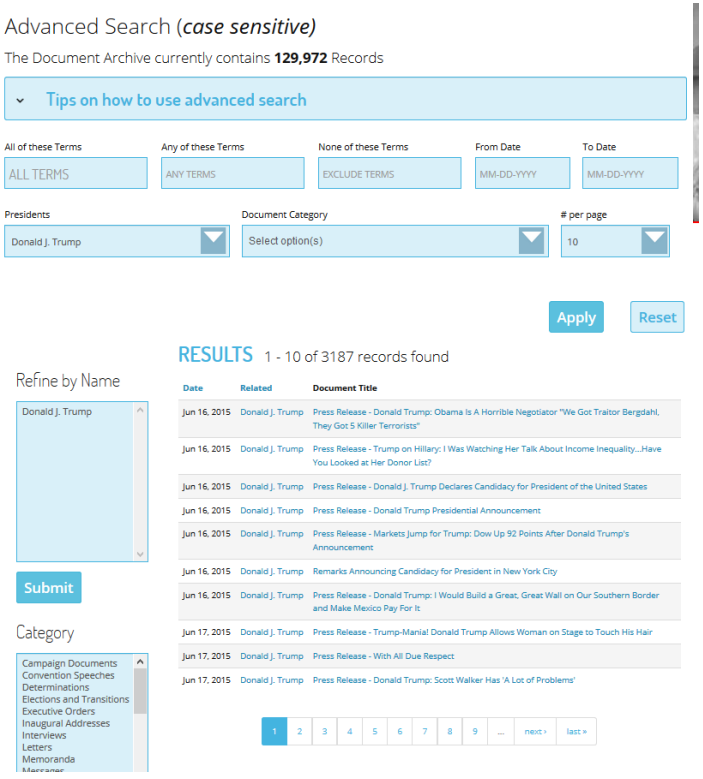
- multiple indexes

Index pages: Presidential Speeches



- search result pages with links to speeches
- Source: presidency.ucsb.edu

Index pages: Presidential Speeches



- search result pages with links to speeches
- Source: presidency.ucsb.edu

- clicking on 'next'

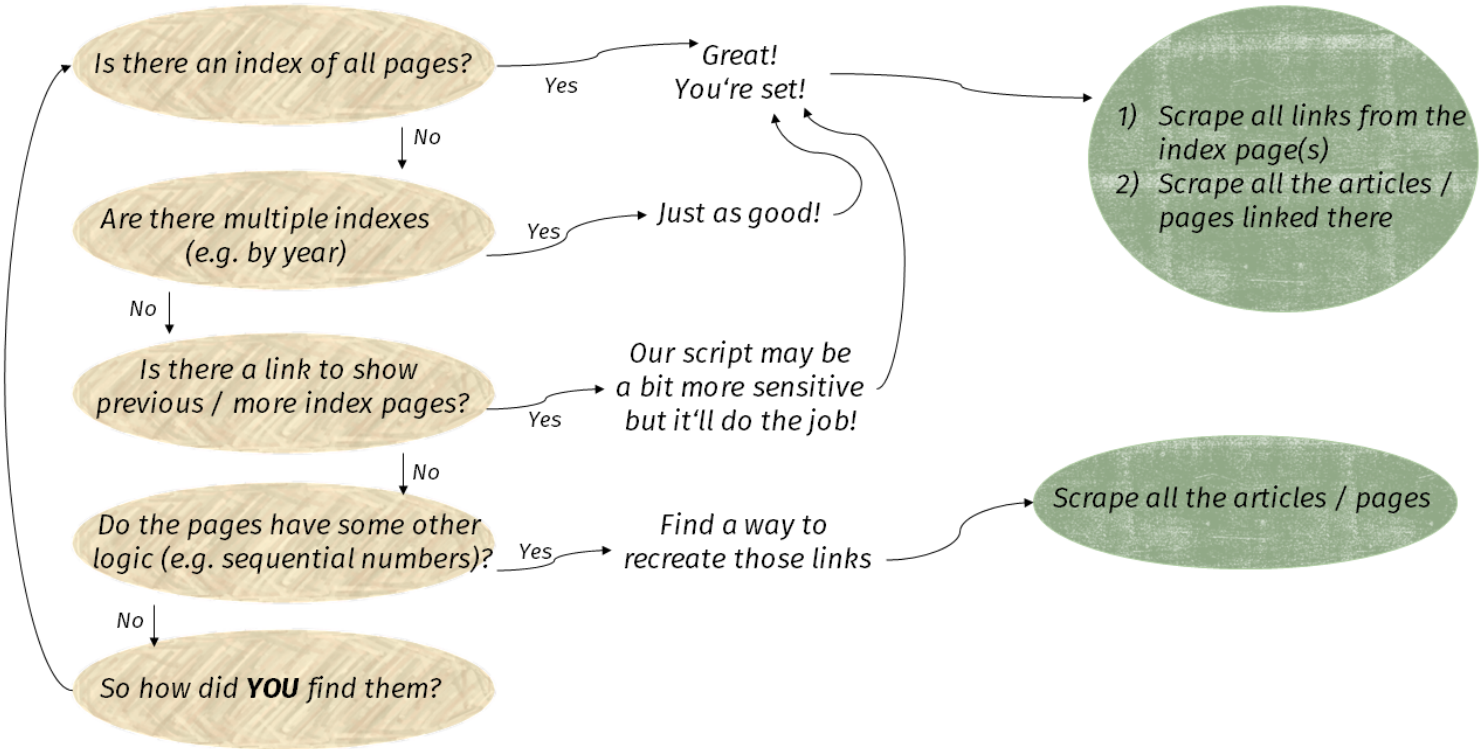
Index pages: Cinque Stelle Blog

- list of entries that follows numeric logic: year & month
- Source: Cinque Stelle Party

Index pages: Cinque Stelle Blog

- list of entries that follows numeric logic: year & month
- Source: Cinque Stelle Party
- sequential numbering (very frequent on wordpress)

The logic of scraping multiple pages



...and more?

Remember? What is webscraping?

- extracting data from webpages
  - anything from university webpage to social media
  - lots of different techniques

Remember? What is webscraping?

- extracting data from webpages
  - anything from university webpage to social media
  - lots of different techniques

- types of scraping
  - gathering as diverse information as possible from different pages vs. very specific scrapers
  - fully automated scrapers to half-automated scripts
  - single-use scraping vs. regular data collection

Other techniques

APIs

dynamic pages and javascript

web crawling

Other techniques

APIs

- sending 'requests' for data to webpage with `httr` package
- frequently used by pages that aim to provide data, e.g. governments, organizations, (social media) companies, ...
    - special case: social media → often more efficient to use ready-made packages that are frequently updated to changing APIs

Other techniques

scraping dynamic pages and javascript

- automation of interaction with webpages through commands sent to browser
  - e.g. using search engines, logging in, scrolling...
- `RSelenium` package
  - helps with getting to the desired webpage
  - extraction of text can often be done with rvest

Other techniques

web crawling / spiders

- following of all links included in a page → generalization rather than webpage-adapted scripts
- messy but massive collection of data
  - often involves automated parsing with heuristics, e.g. with `boilerpipeR` package

Thanks - and enjoy scraping!