

# Web and Cloud Computing – Introduction II

UBCO Master of Data Science – DATA 534



# Lecture Learning Goals

---

- Identify the HTML tags that are used to structure data on a web page and describe how they do this.
- Explain the difference between static and dynamic web data and how these concepts relate to the the first three learning goals and how they impact web scraping.
- Explain what a web scraper is, why it is useful and how it works.
- Be able to use a web-scraping tool to collect static and dynamic data from web pages.

# Internet and Web Basics

---

## Difference between internet and web

- *the Internet* is infrastructure while *the Web* is served on top of that infrastructure.
- Use different protocols:
  - Internet: TCP, UDP, etc.
  - Web: HTTP

TCP/UDP defines how to establish and maintain a network connection through which data is then exchanged.

HTTP is a request-response protocol that allows users to communicate data on the WWW and transfer hypertext.

# Internet and Web Basics

---

A typical HTTP requests contains:

- HTTP version type
- URL
- HTTP method
  - GET, POST
- HTTP requests headers
- HTTP body (optional)

## URLs

- Protocol
- Domain name (host)
- Path
- Parameter values

<https://courses.students.ubc.ca/cs/courseschedule?pname=subjarea&dept=DATA>



# Example

Point Chrome to <https://people.ok.ubc.ca/ylucet/> then right click and choose Inspect

The screenshot shows a web browser displaying the website of Yves Lucet. The website content includes a header with the name "YVES LUCET", a bio stating "I am currently a professor in computer science at the University of British Columbia (Okanagan campus).", and sections for "Research Interests", "Keywords", and "Student Opportunities". The "Research Interests" section lists topics like "designing new algorithms in Computer-Aided Convex Analysis" and "modeling and applications of optimization". The "Keywords" section lists "Computational Science and Engineering, computational mathematics, convex analysis, numerical optimization, modelling, and applications." The "Student Opportunities" section mentions looking for graduate students and provides a link to "BSc/PhD in Computer Science".

Overlaid on the right side of the browser window is the Chrome DevTools Network tab. It shows a list of requests, with the first request selected: "ylucet/". The details for this request are expanded, showing the "General" tab. The "Request URL" is "https://people.ok.ubc.ca/ylucet/", the "Request Method" is "GET", and the "Status Code" is "200 OK". The "Response Headers" section shows "HTTP/1.1 200 OK", "Date: Fri, 08 Jan 2021 22:45:01 GMT", "Server: Apache", "Content-Length: 4997", "Connection: close", and "Content-Type: text/html; charset=UTF-8". The "Request Headers" section shows "Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.9", "Accept-Encoding: gzip, deflate, br", "Accept-Language: en-US,en;q=0.9,fr-FR;q=0.8,fr;q=0.7", and "Cache-Control: max-age=0".

HTTP request from Chrome  
GET content of webpage

Server response  
This is an Apache web server

# HTTP headers

## General

- **Request URL:** <https://cmps-people.ok.ubc.ca/ylucet/>
- **Request Method:** GET
- **Status Code:** 200 OK
- **Remote Address:** 206.87.25.212:443
- **Referrer Policy:** strict-origin-when-cross-origin

## Response Headers

[View source](#)

- **Connection:** Keep-Alive
- **Content-Type:** text/html; charset=UTF-8
- **Date:** Fri, 07 Jan 2022 18:40:10 GMT
- **Keep-Alive:** timeout=5, max=100
- **Server:** Apache/2.4.37 (Red Hat Enterprise Linux) OpenSSL/1.1.1k
- **Strict-Transport-Security:** max-age=63072000; includeSubDomains; preload
- **Transfer-Encoding:** chunked
- **X-Content-Type-Options:**
- nosniff
- **X-Powered-By:**
- PHP/7.4.19

# Example

## Performance

people.ok.ubc.ca/yjucet/

**YVES LUCET**

I am currently a professor in computer science at the University of British Columbia (Okanagan campus).

**Research Interests**

My research is at the boundary between Mathematics and Computer Science and is roughly split into 2 directions:

- designing new algorithms in Computer-Aided Convex Analysis to compute operators arising in convex analysis. The focus is on building tools to manipulate fundamental convex analysis objects thereby giving more insight into the structure of optimization problems. The field is conveniently summarized as Computational Convex Analysis and involves investigating hybrid symbolic-numerical algorithms.
- I have been developing the [CCA toolbox](#) to compute fundamental transforms of convex analysis under Scilab. Computational geometry algorithms form the core of the techniques. The scope of the library has been slowly expanded beyond convex analysis to cover monotone operators, and nonsmooth (nonconvex) analysis. Only basic programming skills are needed to contribute to that area with a very wide field of applications (image processing, network communication, PDE, etc.).
- Visualizing operators is an ongoing challenge considering the interesting operators are set-valued and require 4 dimensions to visualize. Virtual Reality is a technology I am exploring to provide intuitive interactive visualization.
- modeling and applications of optimization. My main project focuses on designing a road at minimal cost while still satisfying regulatory and safety constraints. It is a collaboration with colleagues in mathematics, statistics, and engineering; and in partnership with a private company.

The road design problem usually split into the horizontal, vertical, and earth-moving subproblems. It is a large-scale global optimization problem. Some programming experience is really helpful to contribute to that project (and producing good well documented code). Numerical experiments will be performed on multi-core workstations or clusters using state of the art optimization software.

All the above project are funded by [NSERC](#).

Keywords: Computational Science and Engineering, computational mathematics, convex analysis, numerical optimization, modelling, and applications.

**Student Opportunities**

I am looking for graduate students who are interested in these topics. If you are interested, please [contact me](#) and consider applying to our graduate programs: [MSc/PhD in Computer Science](#). (There are ongoing graduate funding opportunities; apply [here](#).)

Join the [Centre for Optimization Convex Analysis and Nonsmooth Analysis](#), an active, fun, small, but dedicated group of researchers and students with interest in Optimization and Analysis. Enjoy campus life in the [beautiful city of Kelowna](#), at the shores of Lake Okanagan, and nestled right between the Rocky Mountains and Vancouver.

2 requests | 5.2 kB transferred

Console | What's New

Highlights from the Chrome 87 update

- New CSS Grid debugging tools  
Debug and inspect CSS Grid with the new CSS Grid debugging tools.
- New WebAuthn tab  
Emulate authenticators and debug the Web Authentication API with the new WebAuthn tab.
- Move tools between top and bottom panel  
Move tools in DevTools between the top and bottom panel.
- Elements panel updates  
View the Computed sidebar pane in the Styles pane, and more.

new 87

Look for the IP address  
Found in the cache

Create connection

Time to First Byte

# Example

## Performance

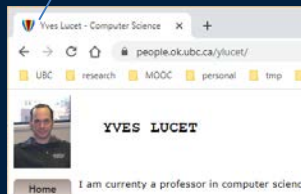
The screenshot shows a web browser window with the URL `people.ok.ubc.ca/ylucet/`. The page content includes a navigation menu with links like Home, Research, Teaching, COSC, COCANA, CCA, WCOM, Links, LinkedIn, and Contact. Below the menu, there is a section titled 'YVES LUCET' with a bio and research interests. The browser's developer tools are open, showing the 'Network' tab. The 'Network' tab displays a list of requests, including 'yves\_lucet.jpg', 'yves\_lucet.jpg', 'mystyle.css', and 'favicon.png'. The 'yves\_lucet.jpg' request is highlighted, showing its details and a timeline. The 'Index.php' request is also highlighted, showing its details and a timeline. The 'Index.php' request is also highlighted, showing its details and a timeline. The 'Index.php' request is also highlighted, showing its details and a timeline.

If no file is indicated, default to index.html (or index.php)

Index.php requires image

Index.php also requires a css stylesheet

Browser requests the icon from the website to display on the tab





# Example

## Files included

The screenshot shows a web browser displaying a personal website for Yves Lucet. The browser's developer tools are open, showing the file structure of the website. The file structure includes: top, people.okubc.ca, ylvect, index, mystyle.css, and ylvect.jpg. The browser's console shows a message about the new CSS Grid debugging tools. The website content includes a header with the name 'YVES LUCET', a bio, research interests, and student opportunities.

**YVES LUCET**

I am currently a professor in computer science at the University of British Columbia (Okanagan campus).

**Research Interests**

My research is at the boundary between Mathematics and Computer Science and is roughly split into 2 directions:

- designing new algorithms in Computer-Aided Convex Analysis to compute operators arising in convex analysis. The focus is on building tools to manipulate fundamental convex analysis objects thereby giving more insight into the structure of optimization problems. The field is conveniently summarized as Computational Convex Analysis and involves investigating hybrid symbolic-numerical algorithms.

I have been developing the [CCA toolbox](#) to compute fundamental transforms of convex analysis under Scilab. Computational geometry algorithms form the core of the techniques. The scope of the library has been slowly expanded beyond convex analysis to cover monotone operators, and nonsmooth (nonconvex) analysis. Only basic programming skills are needed to contribute to that area with a very wide field of applications (image processing, network communication, PDE, etc.).

Visualizing operators is an ongoing challenge considering the interesting operators are set-valued and require 4 dimensions to visualize. Virtual Reality is a technology I am exploring to provide intuitive interactive visualization.

- modeling and applications of optimization. My main project focuses on designing a road at minimal cost while still satisfying regulatory and safety constraints. It is a collaboration with colleagues in mathematics, statistics, and engineering; and in partnership with a private company.

The road design problem usually splits into the horizontal, vertical, and earth-moving subproblems. It is a large-scale global optimization problem. Some programming experience is really helpful to contribute to that project (and producing good well documented code). Numerical experiments will be performed on multi-core workstations or clusters using state of the art optimization software.

All the above project are funded by [NSERC](#).

**Keywords:** Computational Science and Engineering, computational mathematics, convex analysis, numerical optimization, modelling, and applications.

**Student Opportunities**

I am looking for graduate students who are interested in these topics. If you are interested, please [contact me](#) and consider applying to our graduate programs: [MSc/PhD in Computer Science](#). (There are ongoing graduate funding opportunities; apply [here](#).)

Join the [Centre for Optimization Convex Analysis and Nonsmooth Analysis](#), an active, fun, small, but dedicated group of researchers and students with interest in Optimization and Analysis. Enjoy campus life in the [beautiful city of Kelowna](#), at the shores of Lake Okanagan, and nestled right between the Rocky Mountains and Vancouver.

Default is index.html

CSS file

Image file

# HTML the Language of Web Pages

```
<!DOCTYPE html>
<html lang="en-US">
  <head>
  </head>
  <body>
  </body>
</html>
```

## XML syntax

- hierarchical collection of elements
- element consists of start tag, content and end tag, and attributes
  - tags have names and are delimited with “<” and “>”; end tag with “</”
  - attributes added inside of
- HTML defines an XML namespace and schema
  - specific names that can be used as tags, their meaning and their format

## HTML document

- consists of a single HTML element with children for head and body
- and a doctype declaration

# HTML Head

```
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
  <title>UBC Student Services - Courses</title>
  <!-- Stylesheets -->
  <link href="/static/ubcccl/7.0.2/css/ubc-clf-full.min.css" rel="stylesheet">
  <script src="/static/shared/scripts/jquery-1.8.1.min.js"></script>
</head>
```

The head contains meta information for document

- document title
- names of other files that are part of the document (e.g., css, js, etc)
- other meta information such as author and search engine guidance

# HTML Body

```
<body>
  <div class="collapse expand" id="ubc7-global-menu">
    ...
  </div>
  <table class="sortable table table-striped" id="mainTable">
    ...
  </table>
</body>
```

Body contains a list of elements

- This is the webpage
- Each element has children
- Forms a tree

Body elements

- Structure the page, assign classes, contain text, allow interaction

# Document Object Model (DOM)

---

## HTML

- a text based description of a tree of *objects*

## Object

- a collection of named property values and operations

## DOM

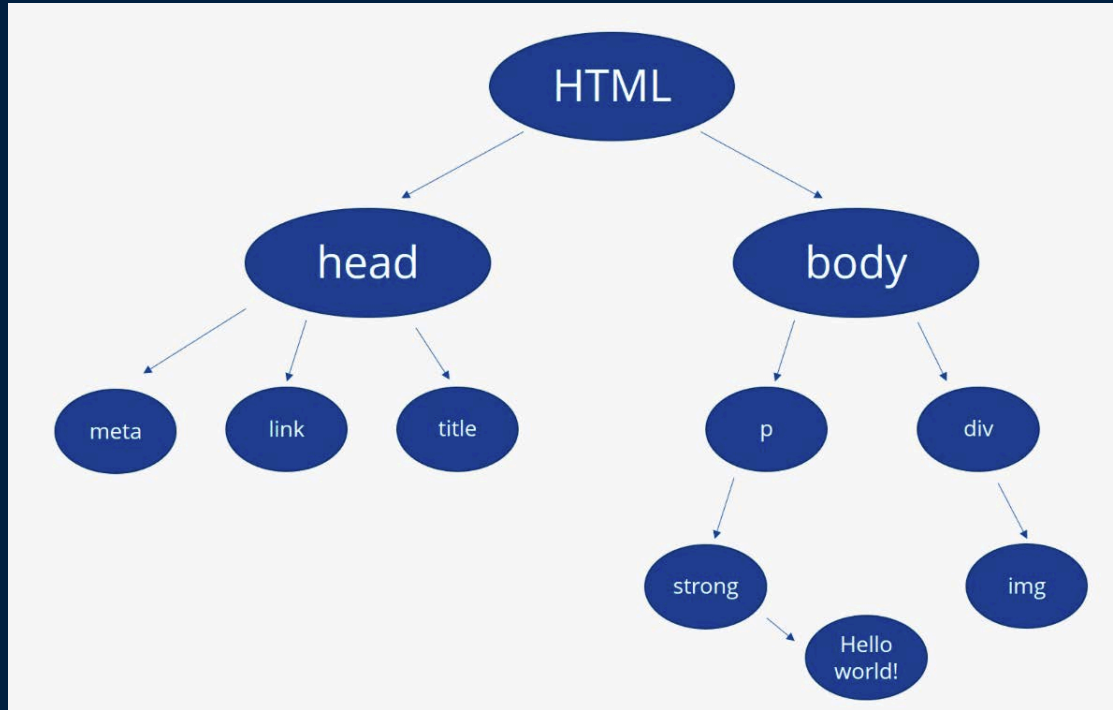
- specification objects that correspond to HTML elements

## DOM tree

- a tree of objects that make up a web document (a.k.a., page)



# DOM Tree



# Basic Structure

---

`<div>`

- groups blocks of other elements
- typically to assign them a common class or common style (css)

`<p>`

- groups blocks of text

`<span>`

- groups in-line runs of elements or text

`<table>`

- row / column list of data

`<ul>`, `<ol>`, `<dl>`

- linear list of data either unordered or ordered ... elements are `<li>` `<dt>` `<dd>`

`<form>`

- group of input-widget elements that allows user to enter data and send it back to the server

# Tables

```
<table class="sortable table table-striped" id="mainTable">
  <thead>
    <tr>
      <th>Course</th><th>Title</th>
    </tr>
  </thead>
  <tbody>
    <tr class="6#39;section16#39;">
      <td>DSCI 511</td><td>Programming for Data Science</td>
    </tr>
  </tbody>
</table>
```

<thead>, <tfoot>, <tbody>

- header, footer and body of the table

<tr>

- table row

<th>, <td>

- a column of either the header or body/footer.

# User Interaction with Hyperlinks

```
<a href="/cs/main;jsessionid=MMzbPz-yJYAt7il-6Vh-ntQF?
pname=subjarea&tname=subjareas&req=3&dept=DSCI&course=511">DSCI 511</a>
```

`<a href=link>Hyper link text</a>`

- displays “Hyper link text”
- if users clicks their mouse on this text, then the browser goes to link

link can be

- different position on current page
- URL for which the browser will issue a new HTML GET request
- javascript code (e.g., procedure call) to execute

# User Interaction with Forms

```
<form class="form-search" method="get" action="http://www.ubc.ca/search/refine/" role="search">
  <input type="search" name="q" placeholder="Search Course Schedule" class="input-xlarge search-query">
  <button type="submit" class="btn">Search</button>
</form>
```

## Forms group inputs and related elements

- an input allows user to enter text, click button, etc
- each input has a field name
- types of inputs include: <input>, <select>, <button>



# Naming Elements

```
<div class="collapse expand" id="ubc7-global-menu">
```

tag name

- html element's name
- div

id

- unique id assigned with id attribute
- #ubc7-global-menu

class

- list of class names assigned with class attribute
- .collapse or .expand or .collapse.expand

pseudo names

- ::after, ::text, :nth-child(n), :not(selector), ...

# CSS Element Selectors

---

## selector

- one selector or several concatenated together to be more selective; \* selects all
- div or div.collapse

## alternative

- separate with commas to select all that match
- div.expand, div.collapse

## child

- separate with > to select specified child
- div.expand > span

## descendant

- separate with space to select specified descendant (does not have to be child)
- div.expand span

# What has all of that to do with Data Science?

Internet was created to share information

Web hosts huge amount of data

# Web Pages are for Humans, but ...

---

There's lots of valuable data embedding in web pages

- course listings
- bank records
- blogs

What if we want to collect this data for analysis

- we need a program that acts like a web browser
- but collect web document data rather than displaying it
- collecting data this way is called ***web scraping***

# Web Scrapers

---

## A Web Scraper

- acts like a web browser (i.e., sends HTTP GET requests to web server)
- but allows you to process the data that comes back

## curl

- utility and library for accessing web servers
- delivers web data as text

## Beautiful Soup

- python library that can parse HTML

## scrapy

- python utility to organize crawling of web tree
- collect subsets of web data
- process in python



# Beautiful Soup

---

Parse a web page

- `s = BeautifulSoup(request.get(url).text, 'html5lib')`

Use CSS selectors to find things

- `dls = s.select('dl.double')`
- `dls` is array of matching dom objects

Digging into a dom object

- `o.children` is an iterator over its children

```

from bs4 import BeautifulSoup

import requests

r = requests.get('http://www.calendar.ubc.ca/okanagan/courses.cfm?code=DATA')

s = BeautifulSoup(r.text, 'html5lib')

# we are interested in the dl list with class '.double'

dls = s.select ('dl.double')    # array of all dom objects that match 'dl.double'

dl = dls[0]                      # there is just one in this case, get it

# now select again to get all of the dts that are in the subtree routed by dl

dts = dl.select ('dt')

# take a look at what one of the looks like

print(dts[0])                    # notice that the course name is the second child

# now get the course name from each of them

courses = []

for dt in dts:

    courses.append ([c for c in dt.children][1]) # add the course name to the list

for course in courses:

    print(course)

```

```
<dl class="double">
```

```

    <dt><a name="101"></a>DATA 101 (3) <b>Making
Predictions with Data</b></dt>

```

```

    <dd>Introduction to the techniques and software
for handling real-world data. Topics include data cleaning,
visualization, simulation, basic modelling, and prediction making. [3-
1-0]<br>

```

```
</dd>
```

```

    <dt><a name="301"></a>DATA 301 (3)
<b>Introduction to Data Analytics</b></dt>

```

```

    <dd>Techniques for computation, analysis, and
visualization of data using software. Manipulation of small and large
data sets. Automation using scripting. Real-world applications from
life sciences, physical sciences, economics, engineering, or
psychology. No prior computing background is required. Credit will
be granted for only one of COSC 301, DATA 301 or DATA 501. [3-2-
0]<br> <i>Prerequisite:</i> Either (a) third-year standing, or (b) one
of COSC 111 or COSC 122<br> <i>Equivalency:</i> COSC 301.

```

```
</dd>
```

```
.....
```

```
</dl>
```

```

<dt><a name="101"></a>DATA 101 (3) <b>Making Predictions with Data</b></dt>
DATA 101 (3)
DATA 301 (3)

```

# Scrapy

Install Anaconda; then launch Anaconda powershell and run

- `conda install scrapy`
- `conda intall protego`
- Go to the folder you wish to create your project

Create a project (from Anaconda powershell)

- `scrapy startproject myproject`
  - *`cd myproject\`; it contains `myproject.cfg` and `myproject\`*
  - *In `myproject\myproject\spiders` copy the file `C2Scrapy.py` (see next slide)*

*Run the spider (from Anaconda powershell in `myproject\`)*

- *run `scrapy crawl quotes`*

# Scrapy

---

## Create a spider

- a class that extends scrapy.Spider
- start\_request method yields a request for every URL, specifying a callback method

## Spider callback method

- parses web page and yields results as *json*
- possibly follow other links to generate additional requests

```
import scrapy
```

```
# a class is a spider if it extends scrapy.Spider
```

```
class QuoteSpider (scrapy.Spider):
```

```
    # the name used to run spider (e.g., from project root type: scrapy crawl quotes)
```

```
    name = 'quotes'
```

```
    # you need this procedure ... this is where crawling starts
```

```
    def start_requests (self):
```

```
        url = 'http://quotes.toscrape.com/page/1/'
```

```
    # yield one or more results from scrapy.Request
```

```
    # each request fetches a url, parses the resulting page, and calls the specified callback
```

```
    function
```

```
    # in this case self.parse is called for this URL
```

```
        yield scrapy.Request (url = url, callback = self.parse)
```

```
    # called by scrapy.Request to parse a single web page
```

```
    def parse(self, response):
```

```
        # get the text of every quote as a collection
```

```
        quotes = response.css('.quote > .text::text').getall()
```

```
        # iterate yielding a json object for every quote
```

```
        # the resulting collection of objects is placed in specified output file
```

```
        # you can change, augment this by changing items.py
```

```
        for quote in quotes:
```

```
            yield {'quote': quote}
```

```
        # locate URL associated with the page's NEXT button
```

```
        nextU = response.css('.pager .next a::attr(href)').get()
```

```
        # if there is a next button then yield result of calling parse on that page
```

```
        if nextU is not None:
```

```
            yield response.follow(nextU, callback=self.parse)
```

Spider name

url to scrap

```
{'quote': '"The world as we have created it is a process of our thinking. It cannot be changed without changing our thinking.'"}  
{'quote': '"It is our choices, Harry, that show what we truly are, far more than our abilities."'}
```



# JavaScript Object Notation (json)

---

Textual format for structured data

- `[a,b,c]` for arrays
- `{'x': m, 'y': n, 'z': o}` for objects

More popular alternative to XML

For example:

```
[  
  {'title': '1984', 'author': 'Orwell'},  
  {'title': 'Moby Dick', 'author': 'Melville'}  
]
```

# Next Lecture

---

- APIs



THE UNIVERSITY OF BRITISH COLUMBIA

