

Web scraping

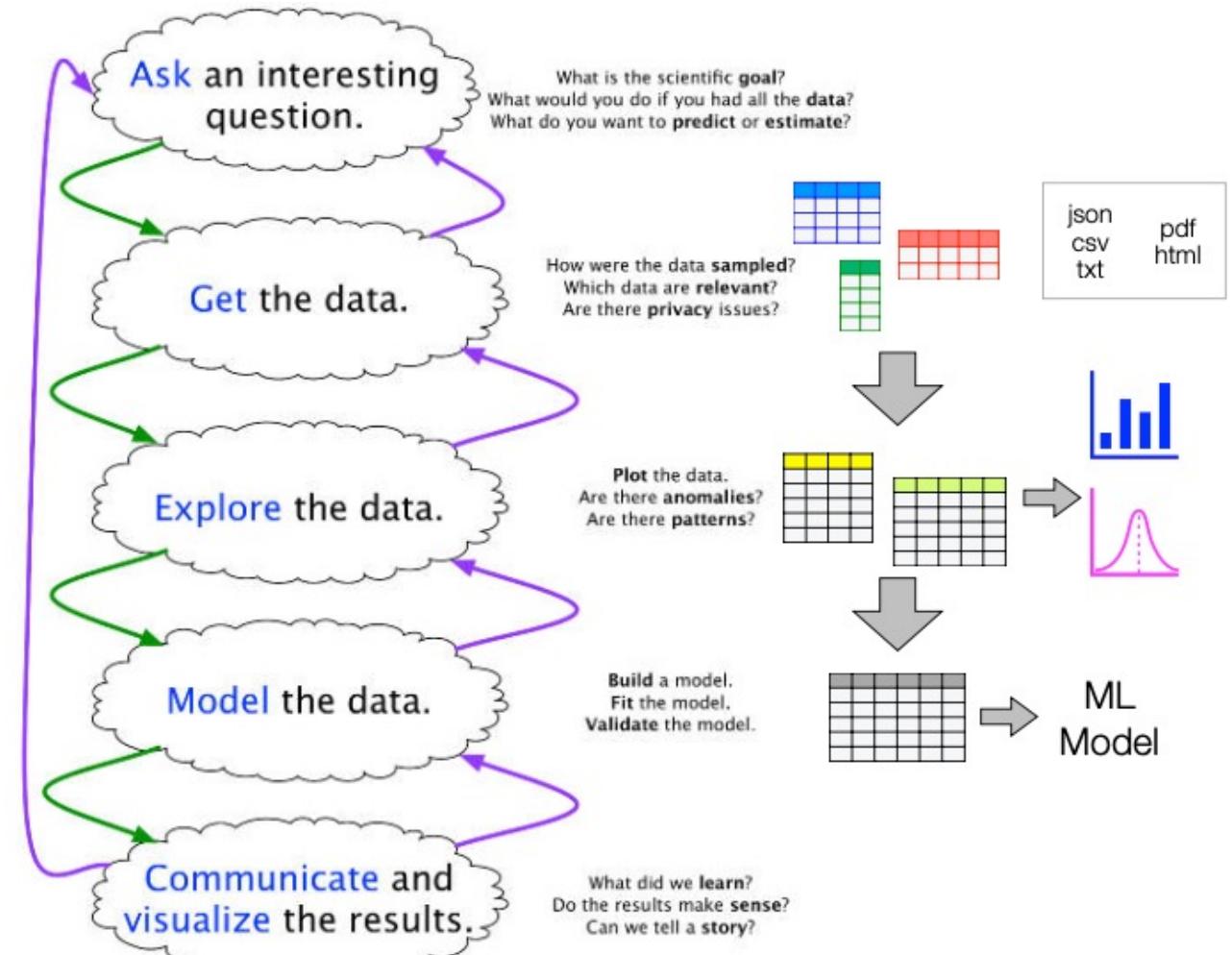
Based on Prof.Natawut's slide

Outline

- Introduction & data extraction steps
- Web scraping
 - Understand web page mechanism
 - HTML
 - CSS & CSS Selector
 - Web page mechanism (DOM & CSSOM)
 - MyWebsite example & DOM tree node structure
 - Parsing a simple webpage: BeautifulSoup
 - Lab1: Basic Web Scraping
 - Lab2: Extract a wiki page
 - Lab3: REST API extraction from Settrade
 - Lab4: Data extraction with Selenium (Google)

Data science process

How to get the data?

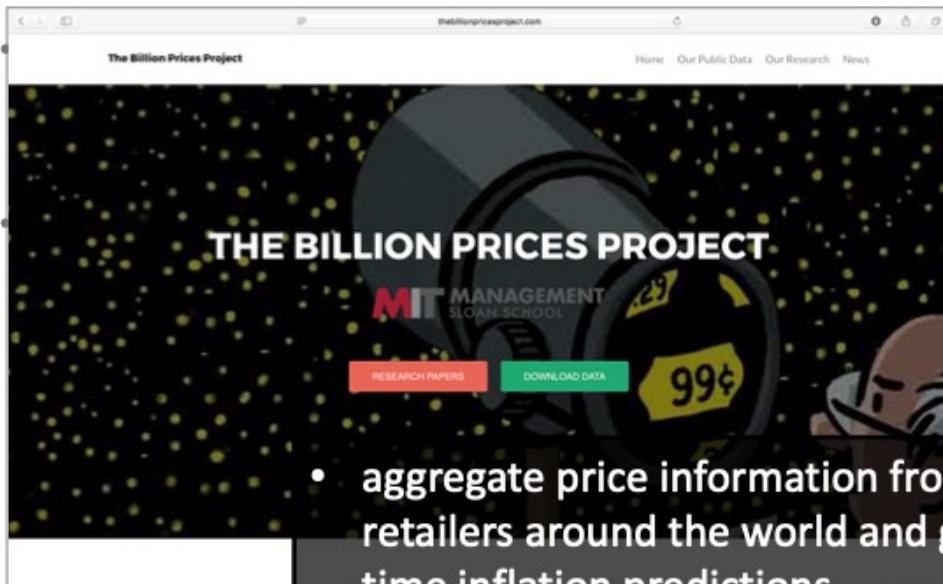


Joe Blitzstein and Hanspeter Pfister, created for the Harvard data science course <http://cs109.org/>.

Internet data sources

- Internet provides lots of data sources
 - Financial
 - Weather
 - Sports
 - News
- Google creates business by utilizing Internet information
- Most data sources are human-readable, it is also possible to extract information systematically with program. → Web scraping

MIT's Billion Prices Project



The screenshot shows the homepage of the Billion Prices Project. The header features the project's name and the MIT Management School logo. Below the header is a large image of a speedometer with a yellow needle pointing to '99¢'. Overlaid on the speedometer are the words 'THE BILLION PRICES PROJECT' and 'MANAGEMENT SLOAN SCHOOL'. At the bottom of the page, there are two buttons: 'RESEARCH PAPERS' (red) and 'DOWNLOAD DATA' (green). A dark grey sidebar on the right contains the following text:

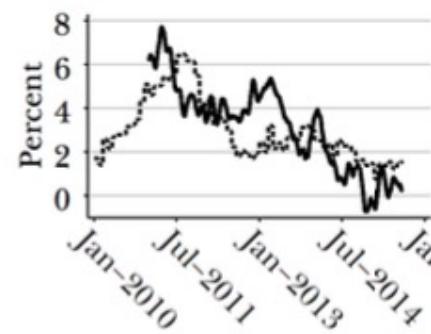
- aggregate price information from multitude of online retailers around the world and gives real time inflation predictions
- Well integrated with many applications to monitor daily price fluctuations of ~5 million items sold by ~300 online retailers in more than 70 countries

data structures and
data with

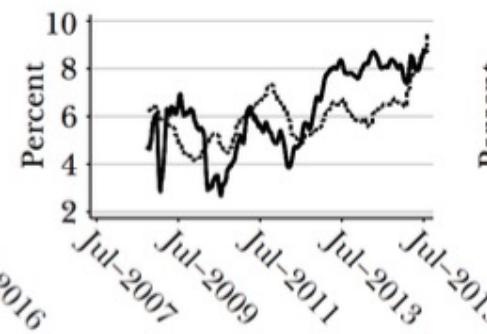
Cavallo, A., & Rigobon, R. (2016). The billion prices project: Using online prices for measurement and research. *The Journal of Economic Perspectives*, 30(2), 151-178.

Annual inflation rate: online vs. CPI

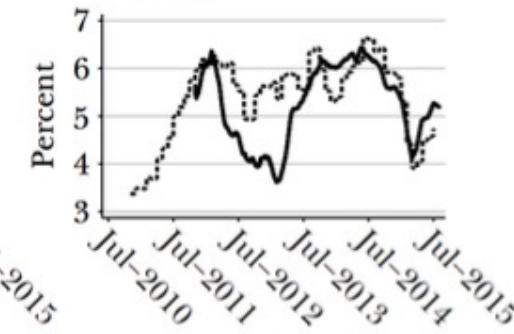
A: China



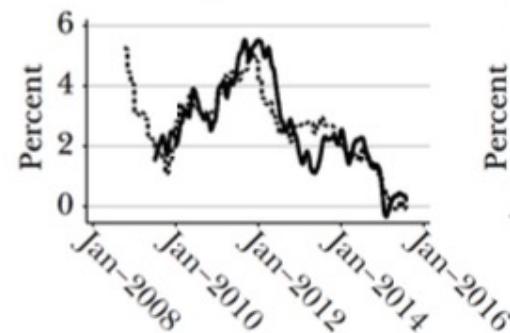
B: Brazil



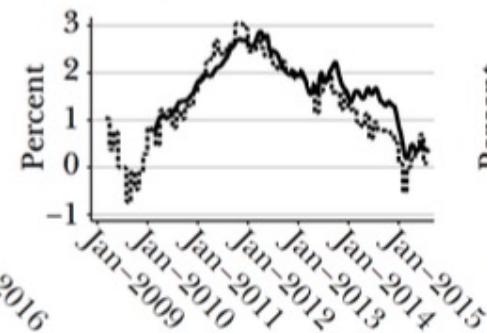
C: South Africa



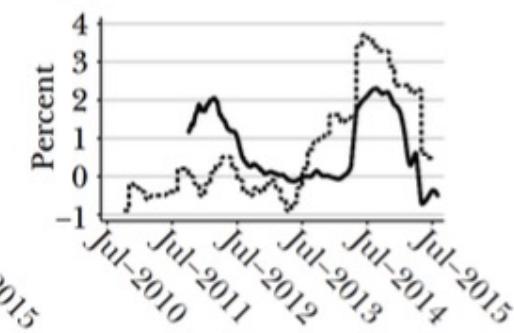
D: United Kingdom



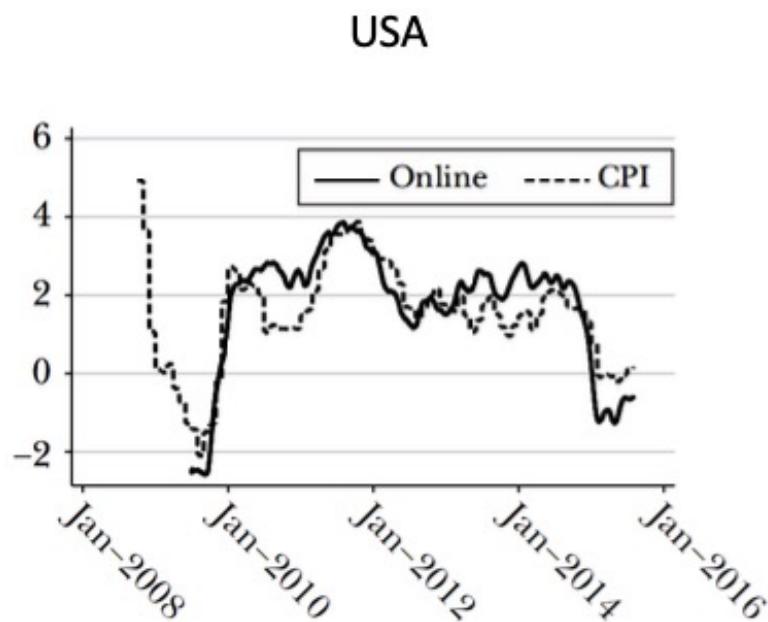
E: Germany

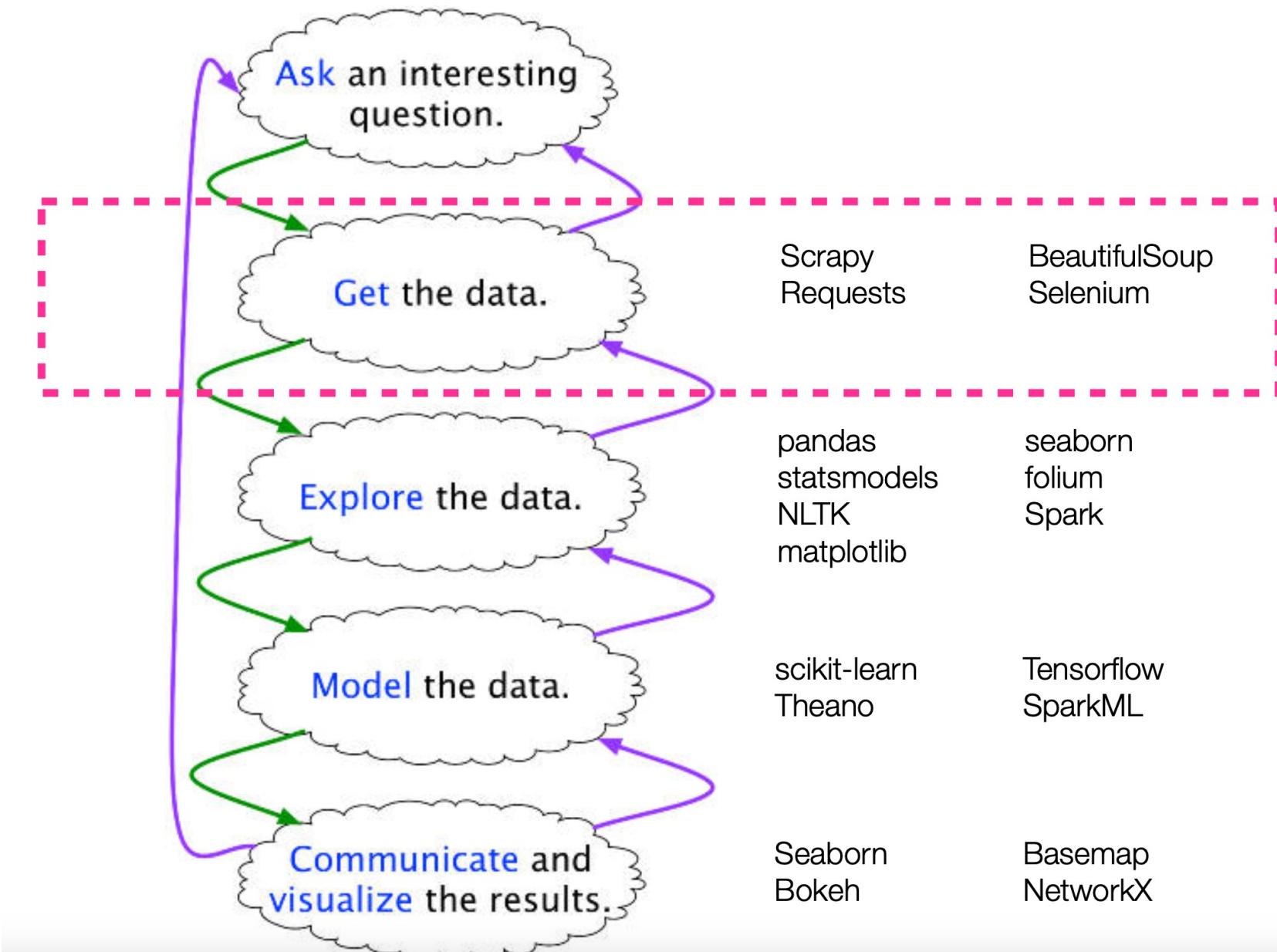


F: Japan



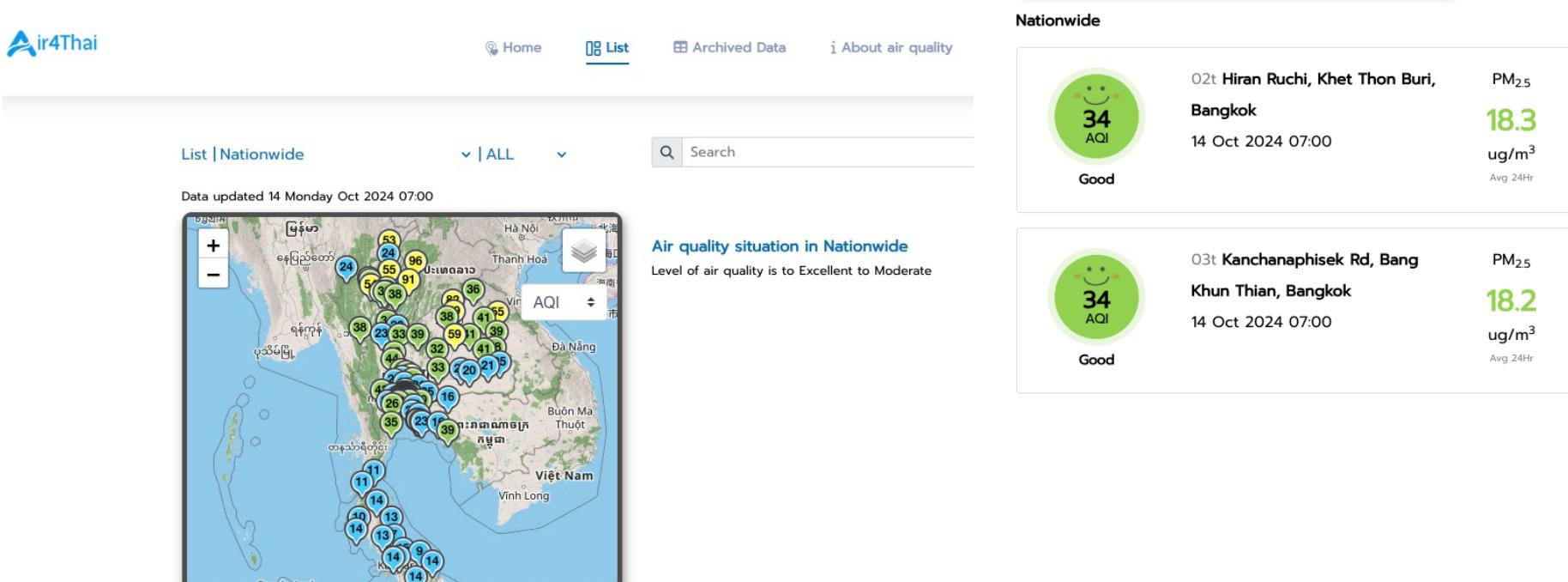
Annual inflation rate: online vs. CPI (cont.)





Example data sources: Air4Thai

- <http://air4thai.pcd.go.th/webV3/#/Home>



Data extraction steps

- 1) Data Accessing
- 2) Data Parsing
- 3) Data Loading

Data extraction steps (cont.)

- 1) Data Accessing
 - Reading a file [\[Lab1\]](#)
 - Issue a HTTP request [\[Lab2\]](#)
 - Call REST API using Request [\[Lab3\]](#)
 - Call Third-Party API using compatible library
 - Collect complicated web page with Selenium [\[Lab4\]](#)

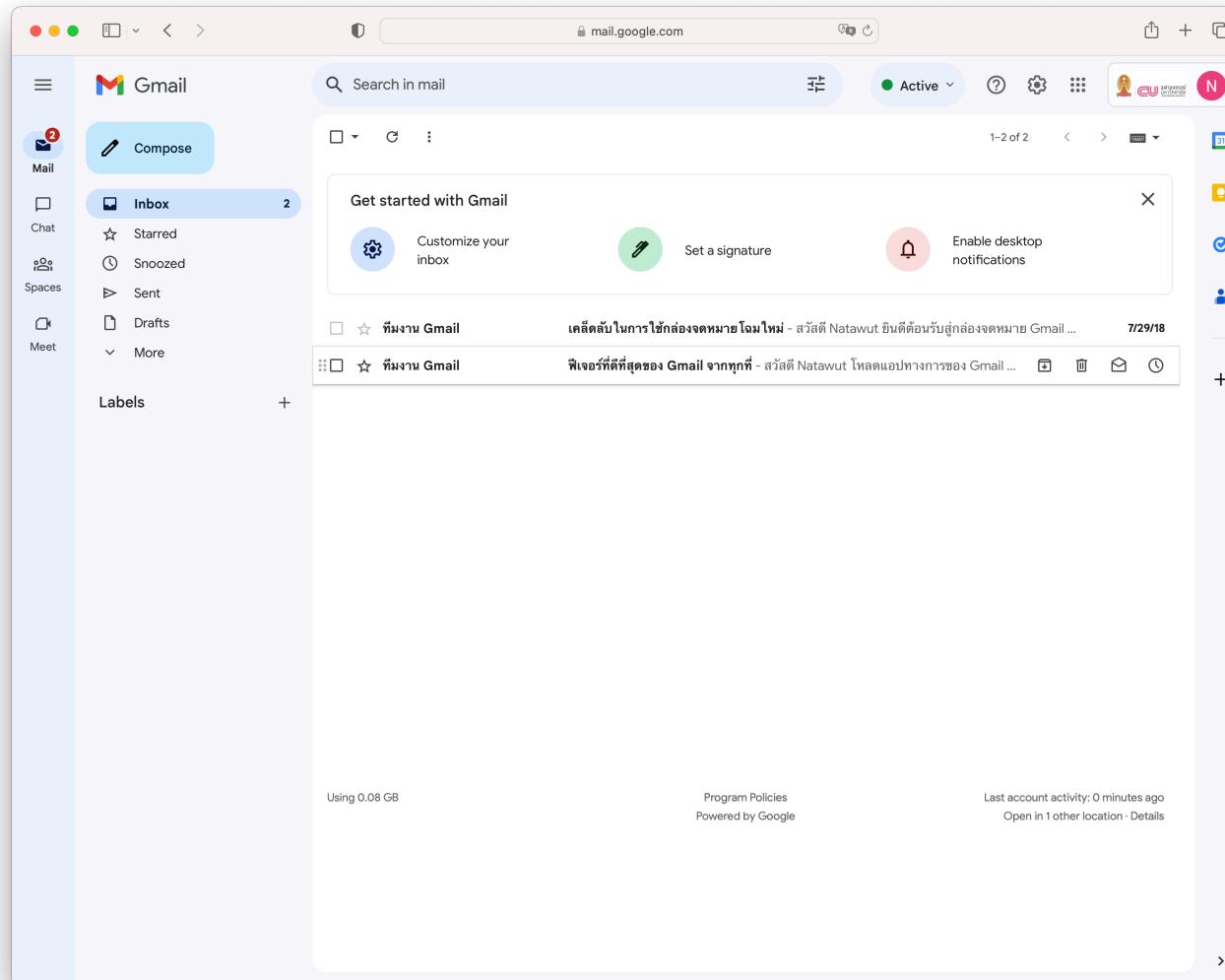
Data extraction steps (cont.)

- 2) Data Parsing
 - Parsing HTML page using BeautifulSoup
 - **Parsing HTML page using GenAI (Simple but Expensive)**
 - Parsing text using regular expressions
 - Parsing docx using python-docx
 - Parsing Excel using Pandas
 - Parsing JSON string using JSON

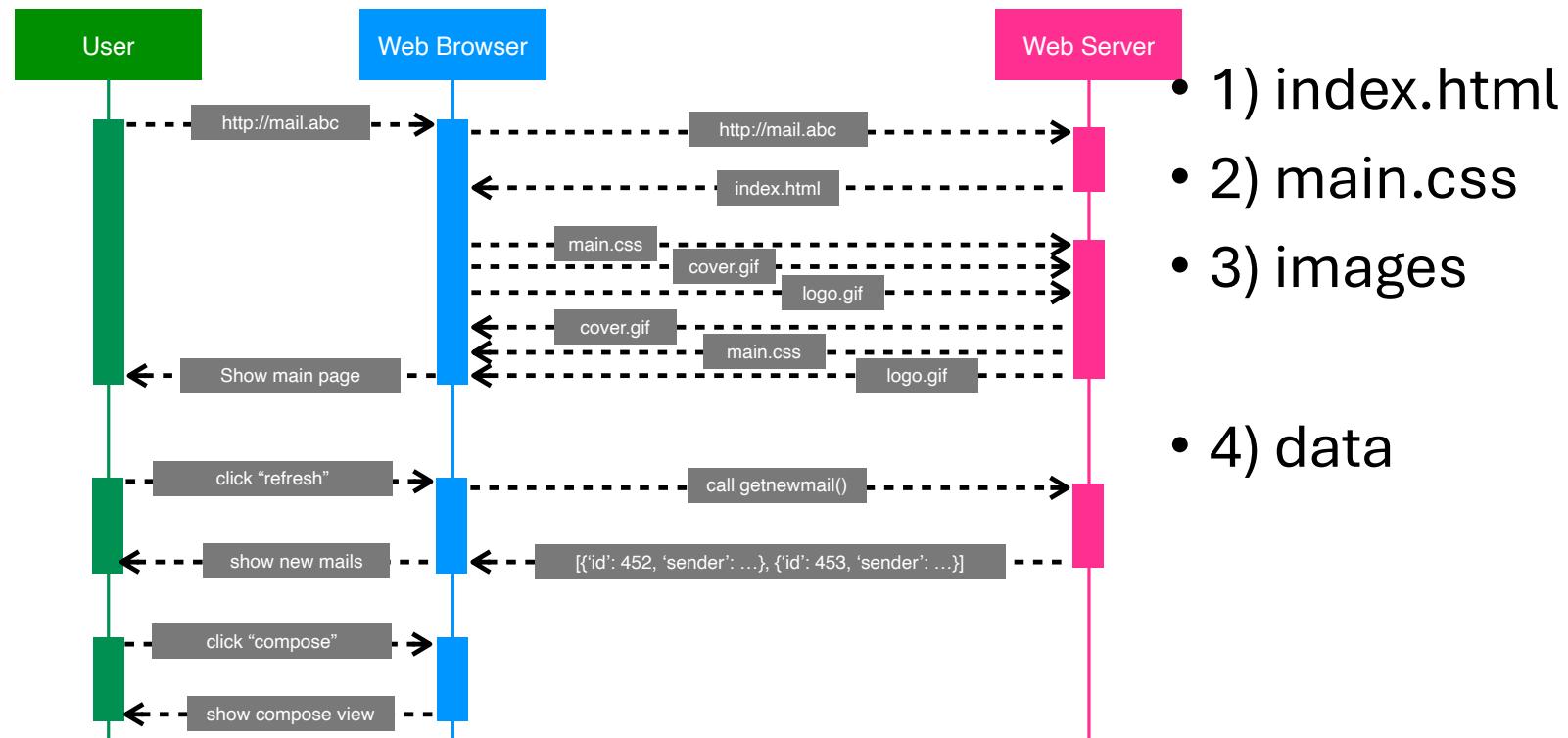
Data extraction steps (cont.)

- 3) Data Loading
 - Save to database
 - Save to data file
 - Save to dataframe

Scraping a web page



Understand Web Page Mechanism



HTML: tag

- HTML is a markup language that defines the structure of a web page using series of elements to wrap different parts of the content to make it appear a certain way, or act a certain way
- For example, `<p>` is a tag to indicate a paragraph

```
<p>Hello, my name is Natawut Nupairoj. Nice to meet you!</p>
```

- Furthermore, HTML element can be nested

```
<p>Hello, my name is <b>Natawut Nupairoj</b>. Nice to meet you!</p>
```

- A web browser will show this snippet as followed

Hello, my name is **Natawut Nupairoj**. Nice to meet you!

More HTML element: class & attrs

- HTML element can have an attribute to provide extra information regarding to this element

```
<p class="greeting">Hello, my name is <b>Natawut Nupairoj</b>. Nice to meet you!</p>
```

Note that “class” is an attribute that can be used in conjunction with CSS

- Some elements have no content and are called void elements

```

```

Anatomy of HTML Page (my_page.html)

```
<!DOCTYPE html>
<html lang="en-US">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <title>My test page</title>
  </head>
  <body>
    <p class="greeting">Hello, my name is <b>Natawut Nupairoj</b>. Nice to meet you!
  </p>
    
  </body>
</html>
```

Greeting

Hello, my name is **Natawut Nupairoj**. Nice to meet you!



CSS

- CSS (Cascading Style Sheets) is a declarative language that controls how webpages look in the browser
 - Font, size, color, background color, spacing
 - Alignment, number of columns
- The browser applies CSS style declarations to selected elements to display them properly
- A style declaration contains the properties and their values, which determine how a webpage looks

CSS Rules and Selectors

- A CSS rule is a set of **properties** associated with a **selector**
- Here is an example that makes every HTML paragraph yellow against a black background:

```
/* The selector "p" indicates that all paragraphs in the document will be
affected by that rule */
p {
    /* The "color" property defines the text color, in this case yellow. */
    color: yellow;

    /* The "background-color" property defines the background color, in this case
black. */
    background-color: black;
}
```

- "Cascading" refers to the rules that govern how selectors are prioritized to change a page's appearance

```
<!DOCTYPE html>
<html lang="en-US">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <style>
      p {
        color: yellow;
        background-color: black;
      }
    </style>
    <title>My test page</title>
  </head>
  <body>
    <p class="greeting">Hello, my name is <b>Natawut Nupairoj</b>. Nice to meet you!</p>
    
  </body>
</html>
```

<style>...</style>

Note that CSS can be embedded in an HTML page or in a separated file

Greeting

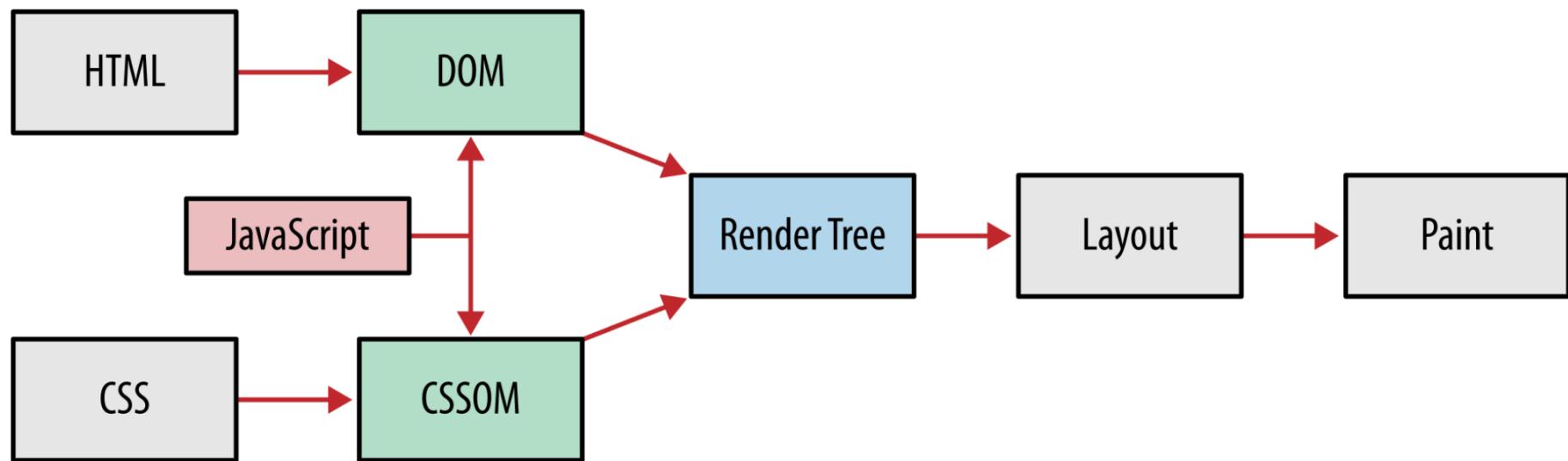
Hello, my name is **Natawut Nupairoj**. Nice to meet you!



CSS Selector

- CSS Selector is a pattern of elements and other terms that tell the browser which HTML elements should be selected to have the CSS property values inside the rule applied to them
 - Elements with a specific tag e.g. all H1 elements
 - Elements with a specific class / id / attribute e.g. all elements with class = “special”
 - Combination of selectors e.g.
 - all H1 elements with class = “special”
 - all H1 elements or elements with class = “special”
- A selector may select multiple elements at once

Web page mechanism (object models)



https://www.w3schools.com/howto/tryit.asp?filename=tryhow_css_example_website

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Page Title</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
* {
  box-sizing: border-box;
}

/* Style the body */
body {
  font-family: Arial, Helvetica, sans-serif;
  margin: 0;
}

/* Header/logo Title */
.header {
  padding: 80px;
  text-align: center;
  background: #1abc9c;
  color: white;
}

/* Increase the font size of the heading */
.header h1 {
  font-size: 40px;
}

/* Sticky navbar - toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed). The sticky value is not supported in IE or Edge 15 and earlier versions. However, for these versions the navbar will inherit default position */
.navbar {
  overflow: hidden;
  background-color: #333;
  position: sticky;
  position: -webkit-sticky;
  top: 0;
}

/* Style the navigation bar links */
.navbar a {
  float: left;
  display: block;
  color: white;
  text-align: center;
  padding: 14px 20px;
  text-decoration: none;
}
```

The screenshot shows a responsive website layout. At the top, a teal header bar contains the title 'My Website' and a subtitle 'A responsive website created by me.'. Below the header is a dark navigation bar with three items: 'Home', 'Link', and 'Link'. To the right of the navigation bar, there is a 'Link' placeholder. The main content area is divided into two columns. The left column, titled 'About Me', features a placeholder image labeled 'Image' and some sample text: 'Some text about me in culpa qui officia deserunt mollit anim..'. The right column, titled 'TITLE HEADING', also has a placeholder image labeled 'Image' and sample text: 'Some text.. Sunt in culpa qui officia deserunt mollit anim id est laborum consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco.'. Both columns have a 'More Text' section with placeholder text: 'Lorem ipsum dolor sit ame.' and 'Image' placeholders.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Page Title</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
* {
  box-sizing: border-box;
}

/* Style the body */
body {
  font-family: Arial, Helvetica, sans-serif;
  margin: 0;
}

/* Header/logo Title */
.header {
  padding: 80px;
  text-align: center;
  background: #1abc9c;
  color: white;
}

/* Increase the font size of the heading */
.header h1 {
  font-size: 40px;
}

/* Sticky navbar - toggles between relative and fixed, depending on the scroll position. It is positioned relative until
a given offset position is met in the viewport - then it "sticks" in place (like position:fixed). The sticky value is
not supported in IE or Edge 15 and earlier versions. However, for these versions the navbar will inherit default
position */
.navbar {
  overflow: hidden;
  background-color: #333;
  position: sticky;
  position: -webkit-sticky;
  top: 0;
}
```

simple_page.html

- 1) header
- 2) style (css)
- 3) body

```

<body>

<div class="header">
  <h1>My Website</h1>
  <p>A <b>responsive</b> website created by me.</p>
</div>

<div class="navbar">
  <a href="#" class="active">Home</a>
  <a href="#">Link</a>
  <a href="#">Link</a>
  <a href="#" class="right">Link</a>
</div>

<div class="row">
  <div class="side">
    <h2>About Me</h2>
    <h5>Photo of me:</h5>
    <div class="fakeimg" style="height:200px;">Image</div>
    <p>Some text about me in culpa qui officia deserunt mollit anim..</p>
    <h3>More Text</h3>
    <p>Lorem ipsum dolor sit ame.</p>
    <div class="fakeimg" style="height:60px;">Image</div><br>
    <div class="fakeimg" style="height:60px;">Image</div><br>
    <div class="fakeimg" style="height:60px;">Image</div>
  </div>
  <div class="main">
    <h2>TITLE HEADING</h2>
    <h5>Title description, Dec 7, 2017</h5>
    <div class="fakeimg" style="height:200px;">Image</div>
    <p>Some text..</p>
    <p>Sunt in culpa qui officia deserunt mollit anim id est laborum consectetur adipisciing elit, sed
do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
exercitation ullamco.</p>
    <br>
    <h2>TITLE HEADING</h2>
    <h5>Title description, Sep 2, 2017</h5>
    <div class="fakeimg" style="height:200px;">Image</div>
    <p>Some text..</p>
    <p>Sunt in culpa qui officia deserunt mollit anim id est laborum consectetur adipisciing elit, sed
do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
exercitation ullamco.</p>
  </div>
</div>

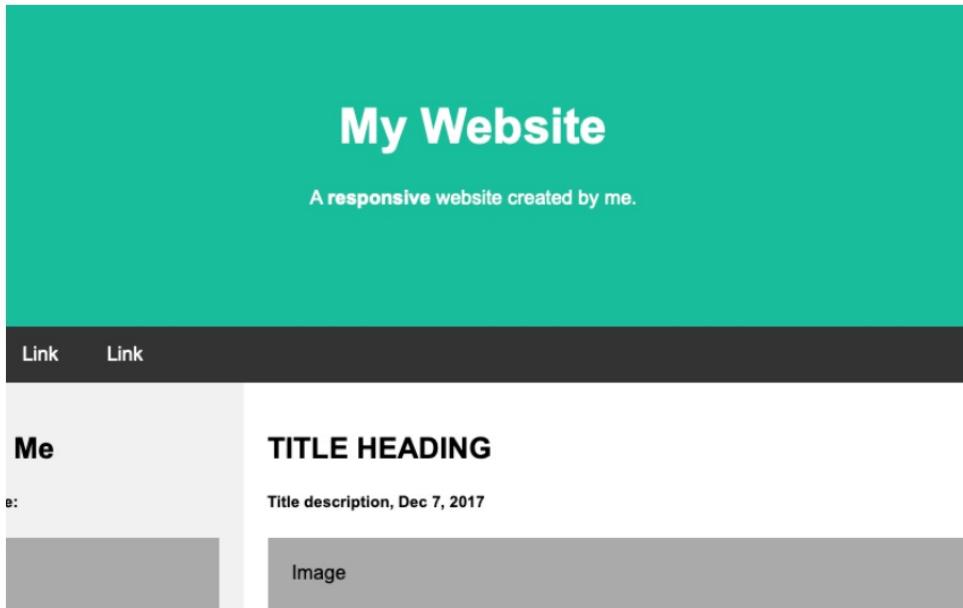
<div class="footer">
  <h2>Footer</h2>
</div>

```

1) header

2) style (css)

3) body



css

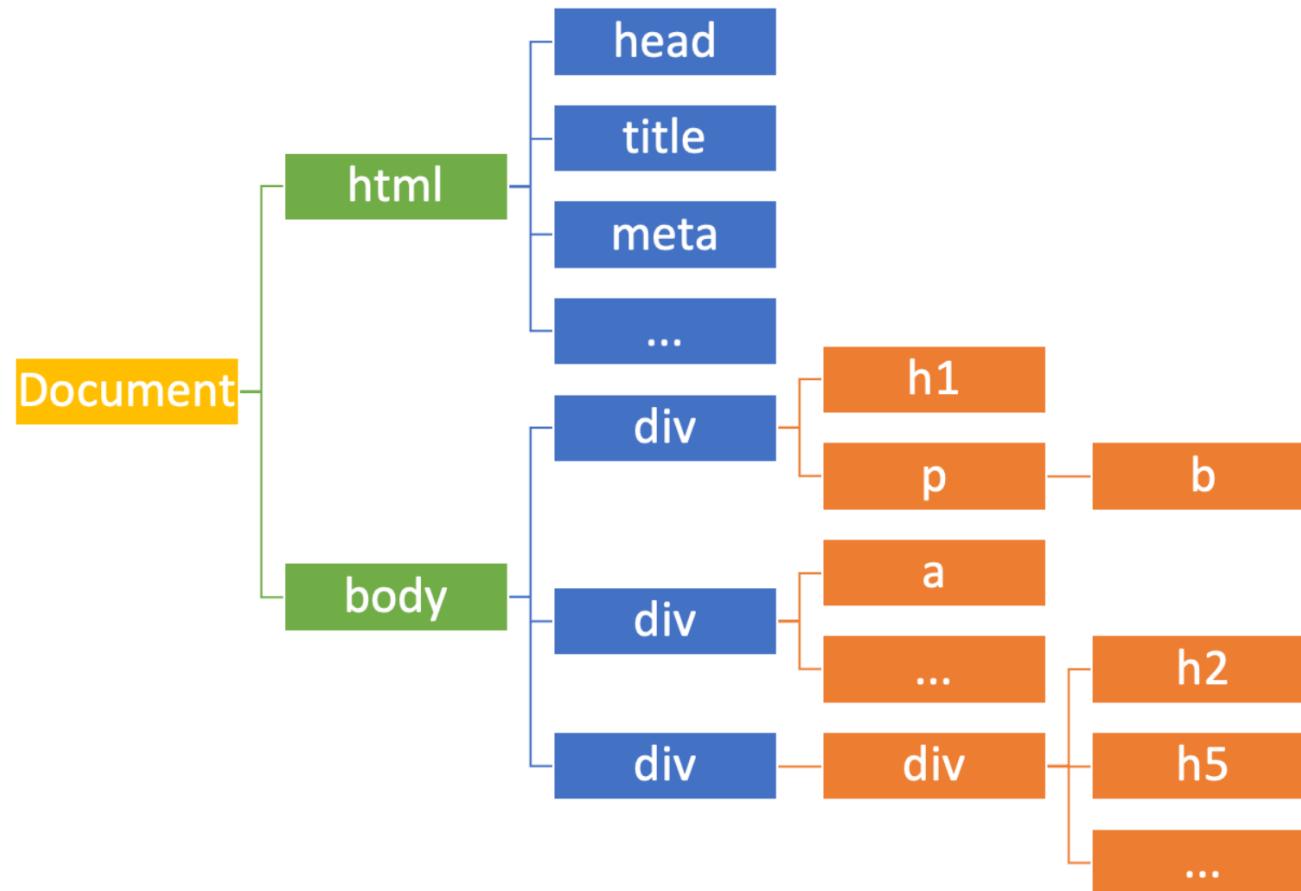
```
/* Header/logo Title */
.header {
  padding: 80px;
  text-align: center;
  background: #1abc9c;
  color: white;
}

/* Increase the font size of the heading */
.header h1 {
  font-size: 40px;
}
```

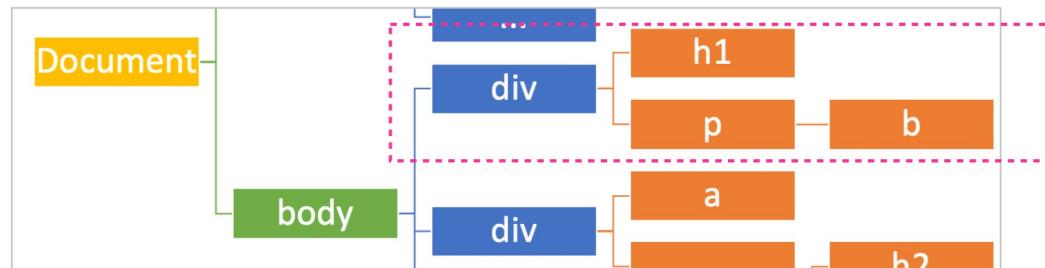
body

```
<div class="header">
  <h1>My Website</h1>
  <p>A <b>responsive</b> website created by me.</p>
</div>
```

DOM tree structure



DOM node example



```
<div class="header">
  <h1>My Website</h1>
  <p>A <b>responsive</b> website created by me.</p>
</div>
```

- Node ‘div’ has 2 children (h1 and p) and 1 attribute (key=‘class’, value=‘header’)

Parsing A Simple Webpage

- Applicable for a simple page only
 - No content rendered by javascript
 - Parse HTML with BeautifulSoup
- BeautifulSoup allows
 - Parsing HTML
 - Accessing and Navigating the DOM tree
 - Getting information from nodes

mentation »

Beautiful Soup Documentation

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.

These instructions illustrate all major features of Beautiful Soup 4, with examples. I show you what the library is good for, how it works, how to use it, how to make it do what you want, and what to do when it violates your expectations.

This document covers Beautiful Soup version 4.9.1. The examples in this documentation should work the same way in Python 2.7 and Python 3.2.

You might be looking for the documentation for [Beautiful Soup 3](#). If so, you should know that Beautiful Soup 3 is no longer being developed and that support for it will end on December 31, 2020. If you want to learn about the differences between Beautiful Soup 3 see [Porting code to BS4](#).

This documentation has been translated into other languages by Beautiful Soup users:

- 这篇文档当然还有中文版.
- このページは日本語で利用できます(外部リンク)
- 이 문서는 한국어 번역도 가능합니다.
- Este documento também está disponível em Português do Brasil.
- Эта документация доступна на русском языке.

Getting help

If you have questions about Beautiful Soup, or run into problems, send mail to the discussion group. If your question involves parsing an HTML document, be sure to mention what the `parse()` function did to the document.

Quick Start

Here's an HTML document I'll be using as an example throughout this document. It's part of Lewis Carroll's *Alice's Adventures in Wonderland*:

```
html_doc = """
<html><head><title>The Dormouse's story</title></head>
<body>
<p class="title"><b>The Dormouse's story</b></p>
```



CO 1_basic_web_scraping.ipynb Share

File Edit View Insert Runtime Tools Help

+ Code + Text Copy to Drive Connect

Basic Webpage Scraping

Webpage scraping consists of two steps: crawling and parsing. In this tutorial, we focus on parsing HTML data. BeautifulSoup is a powerful tool to process static HTML. More details can be found at <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

To simplify our learning, we will use a simple example from W3Schools: https://www.w3schools.com/howto/tryit.asp?filename=tryhow_css_example_website

This simple example contains in a single HTML for simplicity and has been saved in an html file, simple_page.html.

Parsing a webpage

First we read the content from simple_page.html file, store content in variable 'html'

```
[ ] 1 !wget "https://raw.githubusercontent.com/pvateekul/2190513_DS-ICE_2024s1/main/code/Week09/webscraping/simple_page.html"
```

```
[ ] 1 with open('simple_page.html') as f:  
2     html = f.read()
```

Understand CSS Selector Syntax

- CSS Selector is a powerful way to navigate DOM tree and locate DOM node(s)
- Find node(s) by tag, id, class, attributes, states (active, focus, hover, etc.), and much more
- Support hierarchical structure e.g. descendant, sibling, order of children, etc.
- In the example, we want to select
 - an anchor element (a) that is _____
 - inside an unordered list (ul), which is _____
 - inside a div element whose class is 'div-col' _____

```
a_list = soup.select('div.div-col ul a')
```

CSS Selector is very powerful for node searching. We can search by tag name, id, class, and combination of criteria. The CSS Selector includes:

- **string**: select node with the specific tag e.g. div for node with tag 'div'
- **.class**: select node with the specific class
- **#id**: select node with the specific id
- **tag[attr]**: select node with the specific tag and attr

CSS Selector is very powerful for node searching. We can search by tag name, id, class, and combination of criteria. The CSS Selector includes:

- **string**: select node with the specific tag e.g. div for node with tag 'div'
- **.class**: select node with the specific class
- **#id**: select node with the specific *id*
- **tag[attr]**: select node with the specific tag and attr

Selector	Example	Example description
<u>.class</u>	.intro	Selects all elements with class="intro"
<u>.class1.class2</u>	.name1.name2	Selects all elements with both <i>name1</i> and <i>name2</i> set within its class attribute
<u>.class1 .class2</u>	.name1 .name2	Selects all elements with <i>name2</i> that is a descendant of an element with <i>name1</i>
<u>#id</u>	#firstname	Selects the element with id="firstname"
<u>*</u>	*	Selects all elements
<u>element</u>	p	Selects all <p> elements
<u>element.class</u>	p.intro	Selects all <p> elements with class="intro"
<u>element,element</u>	div, p	Selects all <div> elements and all <p> elements
<u>element element</u>	div p	Selects all <p> elements inside <div> elements
<u>element>element</u>	div > p	Selects all <p> elements where the parent is a <div> element
<u>element+element</u>	div + p	Selects the first <p> element that is placed immediately after <div> elements
<u>element1~element2</u>	p ~ ul	Selects every element that is preceded by a <p> element
<u>[attribute]</u>	[target]	Selects all elements with a target attribute
<u>[attribute=value]</u>	[target=_blank]	Selects all elements with target="_blank"
<u>[attribute~=value]</u>	[title~=flower]	Selects all elements with a title attribute containing the word "flower"
<u>[attribute =value]</u>	[lang =en]	Selects all elements with a lang attribute value equal to "en" or starting with "en-"
<u>[attribute^=value]</u>	a[href^="https"]	Selects every <a> element whose href attribute value begins with "https"

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** The notebook is titled "2_wiki_scraping_example.ipynb".
- Toolbar:** Includes standard options like File, Edit, View, Insert, Runtime, Tools, and Help.
- Left Sidebar:** Features icons for file operations: + Code, + Text, Copy to Drive, Search, {x}, and a folder icon.
- Section Header:** A section titled "Wikipedia page data extraction" is expanded.
- Description:** Text stating, "In this tutorial, we will learn how to extract a static page and convert it into useful information."
- Text:** "We first get a wikipedia page using requests."
- Code Cells:** Several code cells are visible:
 - Cell 1:

```
1 import requests  
2 import re  
3 from bs4 import BeautifulSoup
```
 - Cell 2:

```
[ ] 1 bigdata = requests.get('https://en.wikipedia.org/wiki/Big_data')
```
 - Cell 3:

```
[ ] 1 len(bigdata.text)
```

 Output: 532562
 - Cell 4:

```
[ ] 1 bigdata.text
```
 - Cell 5:

```
[ ] Show hidden output
```

Example: Extracting references from a wiki page

Not logged in Talk Contributions Create account Log in

Article Talk Read Edit View history Search Wikipedia

Big data

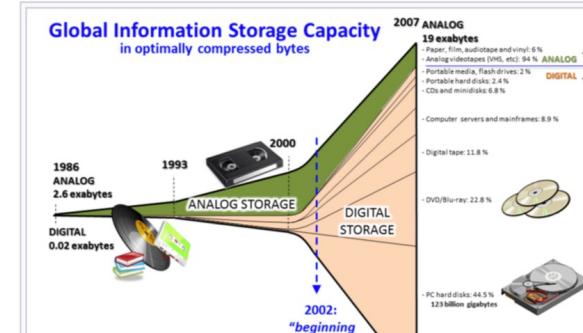
From Wikipedia, the free encyclopedia

This article is about large collections of data. For the band, see [Big Data \(band\)](#). For buying and selling of personal and consumer data, see [Surveillance capitalism](#).

 This article may contain an excessive number of citations. Please consider removing references to unnecessary or disreputable sources, merging citations where possible, or, if necessary, flagging the content for deletion. In particular many references are "spammed" here for promotional purposes. These need to be removed.. (November 2019) ([Learn how and when to remove this template message](#))

Big data is a field that treats ways to analyze, systematically extract information from, or otherwise deal with **data sets** that are too large or complex to be dealt with by traditional **data-processing application software**. Data with many cases (rows) offer greater **statistical power**, while data with higher complexity (more attributes or columns) may lead to a higher **false discovery rate**.^[2] Big data challenges include capturing data, data storage, data analysis, search, sharing, transfer, visualization, querying, updating, information privacy and data source. Big data was

Global Information Storage Capacity
in optimally compressed bytes



The infographic illustrates the exponential growth of global information storage capacity over two decades. It shows the shift from analog to digital storage technologies. In 1986, storage was entirely analog, with 2.6 exabytes in paper, film, audiotape and vinyl. By 1993, digital storage (CDs and minidisks) began to appear. In 2000, the transition to digital storage was complete. In 2002, it is noted as the "beginning of the digital age". By 2007, the total capacity had reached 19 exabytes, with digital storage accounting for 94% of the total. The digital storage categories include portable media players, mobile phones, PCs, servers, and optical media like DVD/Blu-ray.

Contents [\[hide\]](#)

[\(Top\)](#)

> [Definition](#)

[Characteristics](#)

[Architecture](#)

[Technologies](#)

> [Applications](#)

> [Case studies](#)

> [Research activities](#)

> [Critique](#)

[See also](#)

[References](#)

[Further reading](#)

[External links](#)

- Encouraging members of society to abandon interactions with institutions that would create a digital trace, thus creating obstacles to social inclusion

If these potential problems are not corrected or regulated, the effects of big data policing may continue to shape societal hierarchies. Conscientious usage of big data policing could prevent individual level biases from becoming institutional biases, Brayne also notes.

See also [\[edit\]](#)

For a list of companies, and tools, see also: [Category:Big data](#)

- [Big data ethics](#) – Ethics of mass data analytics
- [Big data maturity model](#) – Aspect of computer science
- [Big memory](#) – A large amount of random-access memory
- [Data curation](#) – work performed to ensure meaningful and enduring access to data
- [Data defined storage](#) – Marketing term for managing data by combining application, information and storage tiers
- [Data engineering](#) – Software engineering approach to designing and developing information systems
- [Data lineage](#) – Origins and events of data
- [Data philanthropy](#) – Aspect of culture
- [Data science](#) – Interdisciplinary field of study on deriving knowledge and insights from data
- [Datafication](#) – Technological trend
- [Document-oriented database](#) – Type of computer program
- [List of big data companies](#)
- [Very large database](#) – type of database containing a very large amount of data, so much that it can require specialized architectural, management, processing and maintenance methodologies
- [XLDB](#) – annual conference series on databases, data management and analytics

References [\[edit\]](#)

1. ^ Hilbert, Martin; López, Priscila (2011). "The World's Technological Capacity to Store, Communicate, and Compute Information". *Science*. **332** (6025): 60–65. Bibcode:2011Sci...332...60H. doi:10.1126/science.1200970. PMID 21310967. S2CID 206531385. Archived from the original on 14 April 2016. Retrieved 13 April 2016.
2. ^ Breur, Tom (July 2016). "Statistical Power Analysis and the contemporary "crisis" in social sciences". *Journal of Marketing Analytics*. London, England: Palgrave Macmillan. 4 (2–3): 61–65. doi:10.1057/s411270-016-0001-3. ISSN 2050-3218.
104. ^ Josh Rogin (2 August 2018). "Ethnic cleansing makes a comeback – in China". No. Washington Post. Archived from the original on 31 March 2019. Retrieved 4 August 2018. "Add to that the unprecedented security and surveillance state in Xinjiang, which includes all-encompassing monitoring based on identity cards, checkpoints, facial recognition and the collection of DNA from millions of individuals. The authorities feed all this data into an artificial-intelligence machine that rates people's loyalty to the Communist Party in order to control every aspect of their lives."

metaphors, conscientious usage or big data policy

Brayne also notes.

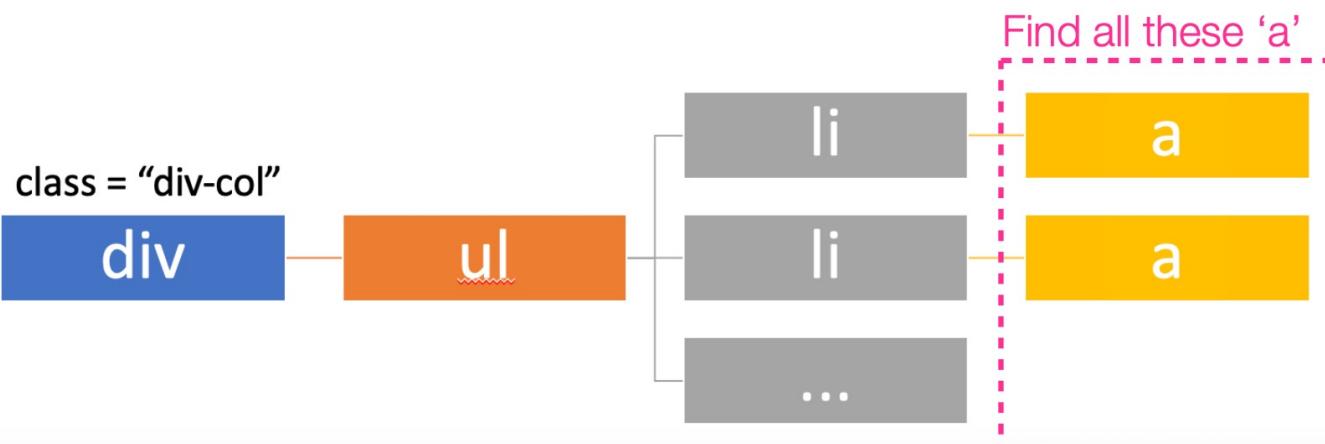
See also [edit]

- a 91.84 x 16 panies, and tools, see also: Categories
- [Big data ethics](#) – Ethics of mass data analytics
- [Big data maturity model](#) – Aspect of computer science
- [Big memory](#) – A large amount of random-access memory
- [Data curation](#) – work performed to ensure meaningful and enduring access to data
- [Data defined storage](#) – Marketing term

The screenshot shows the browser's developer tools with the 'Elements' tab selected. The DOM tree is displayed, starting with an

element. Inside the element, there is a element with the class "mw-headline" and id "See_also" containing the text "See also". Below it is a element with the class "mw-editsection" containing three dots (...). The element has a note div with a note icon and the text "Aspect of computer science". There is also a style element with a duplicate ID r1147244281. A element with the class "div-col" contains a list of items. Each item in the list consists of a - element containing an element with the href "/wiki/Big_data_ethics" and title "Big data ethics", followed by a text node " – Ethics of mass data a", and a link href "https://en.wikipedia.org/wiki/Big_data".

```
<p>...</p>
...
<h2> == $0
  <span class="mw-headline" id="See_also">See also</span>
  <span class="mw-editsection">...</span>
</h2>
<link rel="mw-deduplicated-inline-style" href="mw-data:TemplateStyle096">
<div role="note" class="hatnote navigation-not-searchable selfref">Aspect of computer science</div>
<style data-mw-duplicate="TemplateStyles:r1147244281">...</style>
<div class="div-col" style="column-width: 15em;">
  <ul>
    <li>
      <a href="/wiki/Big_data_ethics" title="Big data ethics">Big
        </a>
      &nbsp;– Ethics of mass data a https://en.wikipedia.org/wiki/Big_data
    </li>
    <li>...</li>
    <li>...</li>
    <li>...</li>
  </ul>
</div>
```





3_REST_API_extraction.ipynb

File Edit View Insert Runtime Tools Help



+ Code



+ Text

Copy to Drive



REST API Data Extraction

Gathering data from a REST API is quite typical. Most Single-Page-Application (SPA) and AJAX dynamic pages rely on REST APIs. In addition, most vendor-specific APIs such as Facebook, Twitter, etc., base on REST.

The most important step of extracting data via REST API is to identify the endpoint.

```
[ ] 1 import requests  
2 import json  
3 import pprint
```

ROBOT trade by
Settrade Open API

Get Started

API Reference

Use Case

Release Notes

Feedback

FAQ

settrade.com

สมัครเข้าใช้งาน

เข้าสู่ระบบ

Call REST API

After we investigate the main page of settrade.com, we can

```
[ ] 1 api_url = 'http://api.settrade.com/api/mar'  
  
[ ] 1 data_info = requests.get(api_url)
```

ROBOT trade by Settrade Open API

หากคุณเป็นเทรดเดอร์ที่มี Trading Algorithm สุดเจ๋ง
มาเพิ่มชีดความสามารถการเทรดบนตลาด
สร้าง ROBOT trade ของคุณเองด้วย Settrade Open API

Settrade Open API (Application Programming Interface) เพิ่มโอกาสให้
เทรดเดอร์ที่มีความสามารถในการเขียนโปรแกรม สามารถนำแนวคิดในการซื้อ^{ขาย}หลักทรัพย์หรืออนุพันธ์ (Trading Algorithm) ของตนออกมาประยุกต์ใช้กับ^{ตลาดหลักทรัพย์แห่งประเทศไทย (SET)} เพื่อสร้าง ROBOT trade ทั้งหุ้นและ^{อนุพันธ์}



</>



```
// 20230611215914
// https://www.settrade.com/api/set/index/info/list?type=INDEX

1  {
2      "indexIndustrySectors": [
3          {
4              "symbol": "SET",
5              "nameEN": "SET",
6              "nameTH": "SET",
7              "prior": 1559.500000,
8              "open": 1558.940000,
9              "high": 1559.430000,
10             "low": 1549.570000,
11             "last": 1555.110000,
12             "change": -4.390000,
13             "percentChange": -0.280000,
14             "volume": 12299648049,
15             "value": 43301220472,
16             "querySymbol": "SET",
17             "marketStatus": "Closed",
18             "marketDateTime": "2023-06-10T03:20:11.564264874+00:00",
19             "marketName": "SET",
20             "industryName": "",
21             "sectorName": "",
22             "level": "INDEX"
23         },
24         {
25             "symbol": "SET50",
26             "nameEN": "SET50",
27             "nameTH": "SET50",
28             "prior": 945.950000,
29             "open": 944.800000,
30             "high": 945.280000,
31             "low": 938.260000,
32             "last": 941.580000,
33             "change": -4.370000,
34             "percentChange": -0.460000,
35             "volume": 1240610800,
36             "value": 29122205677.810000,
37             "querySymbol": "SET50",
38             "marketStatus": "Closed"
39         }
40     ]
41 }
```

<https://www.settrade.com/th/home>

settrade
www.settrade.com

หุ้น อุปทานร์ กองทุนรวม บทวิเคราะห์ ข่าวสาร/บทความ บริการ/เครื่องมือ

ค้นหาชื่อหุ้น ค้นหา

จุดประกายไอเดียที่ใช่ เพื่อการลงทุนในแบบของคุณ

Get Quote ใส่ชื่อยอหักหัวพย. (หุ้น, อุปทานร์, กองทุนรวม)

SET mai TFEX Global

SET 1,470.10 +1.58 (+0.11%)

SET50 936.64 +1.40 (+0.15%)

SET50FF 906.13 -0.15 (-0.02%)

SET100 2,083.73 +2.11 (+0.10%)

SET100FF 1,982.67 -0.24 (-0.01%)

sSET 851.76 -1.67 (-0.20%)

SETCLMV 819.89 -6.97 (-0.84%)

SETHD 1,230.35 +11.80 (+0.97%)

SETESG 939.09 -0.25 (-0.03%)

สูบป่าจะดีลด สรุปเช็ขายลับบัน ข้อมูลสถิติ

สถานะตลาด Closed ข้อมูลล่าสุด 12 ต.ค. 2567 03:04:46

SET ▲ 1,470.10 +1.58 (+0.11%)

สูงสุด 1,482.02 ต่ำสุด 1,470.10

ปริมาณ ('000 หุ้น) 8,675,897 ยอดค่า (ล้านบาท) 60,196.60

Most Active Volume

SET	mai	ปริมาณ AOM ('000 หุ้น)
TTB	1.89	+0.02 (+1.07%) 253,832
SDC	0.03	-0.01 (-25.00%) 241,782
CCET	4.62	+0.08 (+1.76%) 162,560
KTB	21.60	+0.50 (+2.37%) 134,539
BTS	4.58	+0.06 (+1.33%) 133,826
PTT	34.25	+0.75 (+2.24%) 111,200
BANPU	6.75	-0.05 (-0.74%) 112,865
AAV	2.70	0.00 (0.00%) 81,200
BDMS	28.75	-1.00 (-3.36%) 77,319

https://www.settrade.com/th/home

 4_selenium.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text Copy to Drive

☰ {x} 🔒

▼ Data Extraction with Selenium

In this tutorial, we discuss how to use Selenium to extract data from the web. Please see <https://selenium-python.readthedocs.io> for more details.

▼ Installation

We first install selenium package.

```
pip install selenium
```

```
[ ] 1 from selenium import webdriver  
[ ] 2 import time  
[ ] 3 import os
```

```
[ ] 1 chrome_options = webdriver.ChromeOptions()  
[ ] 2 # Uncomment these two lines below if you run in colab. Though, you won't get interactive browser.
```

<https://www.settrade.com/th/home>

settrade www.settrade.com หุ้น อันดับรวม กองทุนรวม บกอ/เคราะห์ ข่าวสาร/บทความ บริการ/เครื่องมือ ค้นหาชื่อย่อ  

จุดประกายไอเดียที่ใช่ เพื่อการลงทุนในแบบของคุณ

 Get Quote ใส่ชื่อย่อหักห้าม (หุ้น, อันดับ, กองทุนรวม)

SET **mai** **TFEX** **Global**

▲ สรุปภาวะตลาด  สรุปซื้อขายสัปดาห์  ข้อมูลสถิติ

สถานะตลาด Closed  ข้อมูลล่าสุด 12 ต.ค. 2567 03:04:46

SET **▲ 1,470.10** **+1.58 (+0.11%)**

สูงสุด **1,482.02** ต่ำสุด **1,470.10**

ปัจจุบัน ('000 หุ้น) **8,675,897** มูลค่า (ล้านบาท) **60,196.60**

กราฟ SET 10:00 - 16:00 10:00 12:00 14:00 16:00 1,485 1,480 1,475 1,470 1,465

▲ Gainers 187 | ↔ Unchanges 215 | ▾ Losers 263

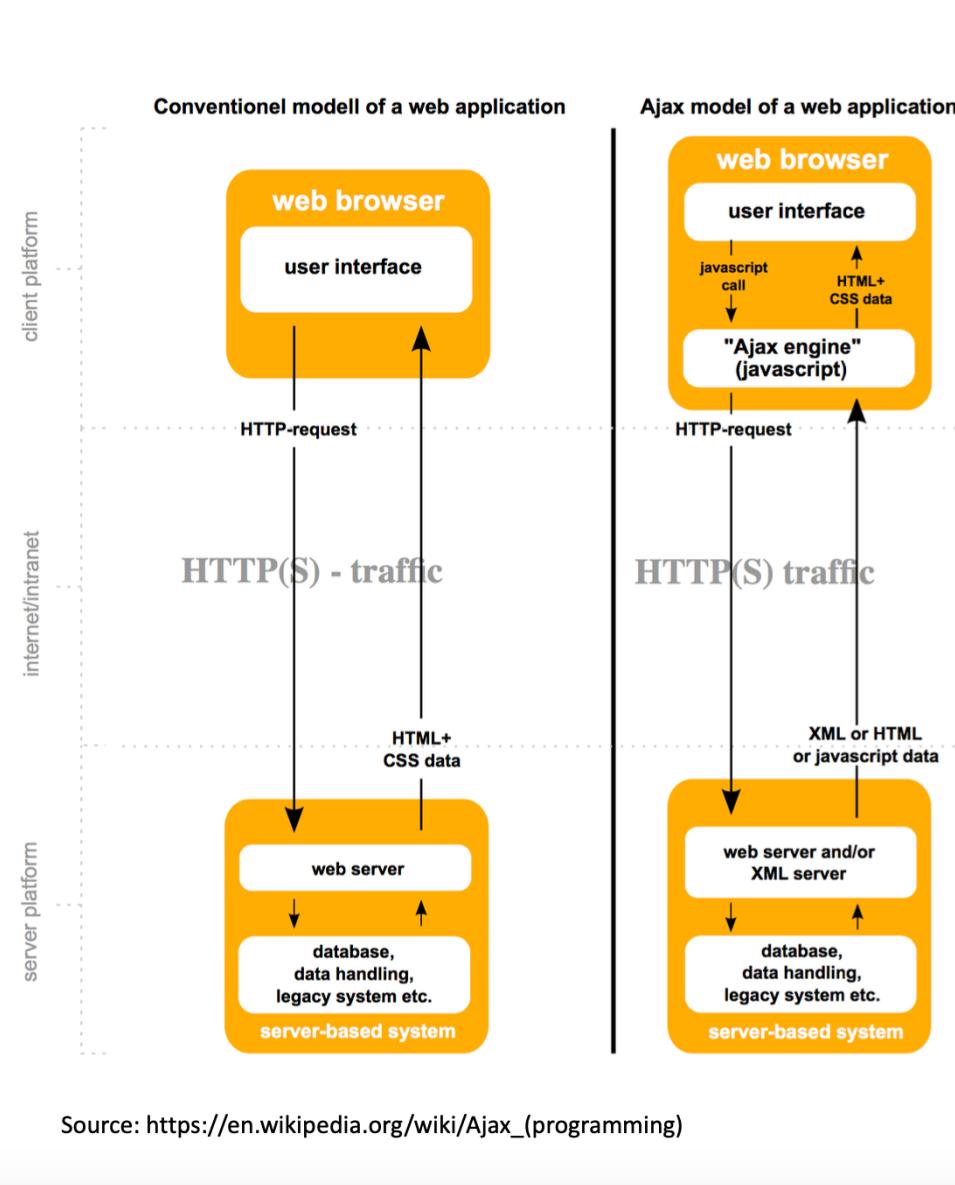
Most Active Volume

SET	mai	ปริมาณ AOM ('000 หุ้น)
TTB	1.89	+0.02 (+1.07%) 253,832
SDC	0.03	-0.01 (-25.00%) 241,782
CCET	4.62	+0.08 (+1.76%) 162,560
KTB	21.60	+0.50 (+2.37%) 134,539
BTS	4.58	+0.06 (+1.33%) 133,826
PTT	34.25	+0.75 (+2.24%) 118,865
BANPU	6.75	-0.05 (-0.74%) 112,865
AAV	2.70	0.00 (0.00%) 80,196.60
BDMS	28.75	-1.00 (-3.36%) 77,319

<https://www.settrade.com/th/home> 42

Extracting AJAX-based Webpage

- Modern web applications are AJAX-based e.g. SPA
- Typical web request approach does not work



DevTools - www.settrade.com/th/home

Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder Performance insights AngularJS

Preserve log Disable cache No throttling

Filter Invert Hide data URLs All Fetch/XHR JS CSS Img Media Font Doc WS Wasm Manifest Other Has blocked cookies Blocked Requests 3rd-party requests

20000 ms 40000 ms 60000 ms 80000 ms 100000 ms 120000 ms 140000 ms 160000 ms 180000 ms 200000 ms 220000 ms 240000 ms 260000 ms 280000 ms 300000 ms 320000 ms 340000 ms

Name Headers Payload Preview Response Initiator Timing Cookies

```

1   "indexIndustrySectors": [
      {
        "symbol": "SET",
        "nameEN": "SET",
        "nameTH": "SET",
        "prior": 1559.50000,
        "open": 1558.94000,
        "high": 1559.43000,
        "low": 1549.57000,
        "last": 1555.11000,
        "change": null,
        "percentChange": null,
        "volume": null,
        "value": null,
        "querySymbol": "SET",
        "marketStatus": "Closed",
        "marketDateTime": "2023-06-10T03:20:11.564264874+07:00",
        "marketName": "SET",
        "industryName": "",
        "sectorName": "",
        "level": "INDEX"
      },
      {
        "symbol": "SET50",
        "nameEN": "SET50",
        "nameTH": "SET50",
        "prior": 945.95000,
        "open": 944.80000,
        "high": 945.28000,
        "low": 938.26000,
        "last": 941.58000,
        "change": -4.37000,
        "percentChange": -0.46000,
        "volume": 1240610800,
        "value": 29122205677.81000,
        "querySymbol": "SET50",
        "marketStatus": "Closed",
        "marketDateTime": "2023-06-10T03:20:11.564264874+07:00",
        "marketName": "SET",
        "industryName": "",
        "sectorName": "",
        "level": "INDEX"
      },
      {
        "symbol": "SET100",
        "nameEN": "SET100",
        "nameTH": "SET100",
        "prior": 2104.43000,
        "open": 2102.36000,
        "high": 2103.88000,
        "low": 2088.55000,
        "last": 2096.69000,
        "change": -7.74000,
        "percentChange": -0.37000,
        "volume": 1822239600,
        "value": 34185571732.51000,
        "querySymbol": "SET100",
        "marketStatus": "Closed",
        "marketDateTime": "2023-06-10T03:20:11.564264874+07:00",
        "marketName": "SET",
        "industryName": "",
        "sectorName": "",
        "level": "INDEX"
      }
    ]
  
```

The screenshot shows the SET website's market summary page. It features a header with tabs for SET, mai, TFEX, and Global. Below the tabs, there are four main sections: SET, SET50, SET100, and sSET. Each section displays the current value, change, and percentage change. To the right, there are two additional boxes: one for SET showing a value of 1,555.11 and a change of -4.39 (-0.28%), and another for SETCLMV showing a value of 12,299,648 and a change of -1.38 (-0.16%).

Market	Value	Change	Percentage Change
SET	1,555.11	-4.39	-0.28%
SET50	941.58	-4.37	-0.46%
SET100	2,096.69	-7.74	-0.37%
sSET	1,001.14	+2.23	+0.22%
SETCLMV	888.25	-1.38	-0.16%
SETCLMV (Value)	12,299,648		
SETCLMV (Value)	43,301.22		

Extract complicated web page with Selenium

Selenium architecture: Need browser driver!



Selenium capabilities

- Process any website similar to normal web browser
 - Access a web page using normal URL
 - Grab HTML and other resource files in a particular web page
 - Find components based on criteria e.g. CSS selector, etc.
 - Interacting with components in a web page e.g. **click, input keyboard**
 - Navigate within a web page and across multiple pages
- During operation, Selenium can be normal mode or headless mode
- Checkout <https://www.selenium.dev/documentation/> for more information

Example: Selenium on Google

1. Visit <https://www.google.com>
2. Get HTML of this web page
3. Find a query box
4. Enter “ประเทศไทย” in the query box and press ‘enter’
5. Search result links (‘a’ elements under an element with id=‘search’)
6. Click the first link