

AUTOMATED DATA COLLECTION WITH PYTHON

Ivan Hernandez, Ph.D
DePaul University

CHANGING PERSPECTIVES ON DATA

- **Data driven decisions** being emphasized

- **Age of Big Data**


 - Larger

 - More Frequent

 - More Varied

- **Where to access this data?**

SOURCES OF BIG DATA




O*NET OnLine

Occupation Quick Search:

→

HelpFind OccupationsAdvanced SearchCrosswalks

ShareO*NET Sites



Build your future with O*NET OnLine.

Welcome to your tool for career exploration and job analysis!

O*NET OnLine has detailed descriptions of the world of work for use by job seekers, workforce development and HR professionals, students, researchers, and more!

What is O*NET? →

What's New?

Learn More

Get O*NET news by [email](#) or [RSS](#).

I want to be a...


Start the career you've dreamed about, or find one you never imagined.

Find it Now →


at My Next Move

Featured Skills & Endorsements

Leadership · 99+




Endorsed by Ambassador Shabazz and 19 others who are highly skilled at this




Endorsed by 134 of Jeff's colleagues at LinkedIn

Entrepreneurship · 99+




Endorsed by Dharmesh Shah and 26 others who are highly skilled at this




Endorsed by 46 of Jeff's colleagues at LinkedIn

Strategy · 99+



Endorsed by Daniel Shapero and 50 others who are highly skilled at this



Endorsed by 77 of Jeff's colleagues at LinkedIn

Jeff is also good at...

Product Develop... · 93

Product Marketing · 99+

Executive Manag... · 99

Business Strategy · 99+

Strategic Planning · 99+

Product Manage... · 78

LinkedIn · 73

Social Media · 68

User Experience · 48

Corporate Develo... · 35

Non-profits · 30

Analytics · 30

Mergers & Acquisi... · 28

Team Leadership · 13

Thought Leadership · 12

Business Operati... · 11

Leadership Devel... · 11

Nonprofits · 10

Mobile Applications · 9

Mentoring · 7

Nonprofits · 10

Motivation · 5

Awesomeness · 5

Education · 3

Candidate Interview Reviews

Filter

Sort: Popular | Date | Difficulty

1

Your trust is our top concern, so companies can't alter or remove reviews.

×

Sep 3, 2014Helpful (601)

D.

Consultant Interview

Anonymous Employee

Accepted Offer

Positive Experience

Average Interview

Application

I applied online. The process took 4+ weeks. I interviewed at Deloitte in August 2014.

Interview

I applied online (I currently work in this field for a competitor) for a consultant position within the Human Capital Practice: Organization, Transformation & Talent Group. Within a week a I heard from a recruiter, who setup a phone interview.

Phone Interview: Basic questions about my background, current employer, and reasons for applying to Deloitte. Lasted 30 mins, was pretty straight forward. Just be confident in your answers.

A day later I...

Show More

Negotiation

I countered the salary (which was already at the top of the Glassdoor range) and they met me in the middle. Always negotiate. One email that took 10 mins to craft pocketed me a decent sum. Also, don't sound too eager when you are offered, and take your time before writing the counter offer.


f

t

e

l

Helpful (601)



WIKIPEDIA
The Free Encyclopedia

Main pageContentsFeatured contentCurrent eventsRandom articleDonate to WikipediaWikipedia store

InteractionHelpAbout WikipediaCommunity portalRecent changesContact page

ToolsWhat links hereRelated changesUpload fileSpecial pagesPage informationLanguages

ArticleTalk

View logs for this page

ReadEditNew sectionView historySearch Wikipedia

Search for revisions

From year (and earlier): 2017From month (and earlier): allTag filter:Show

For any version listed below, click on its date to view it. For more help, see [Help:Page history](#) and [Help:Edit summary](#).
External tools: [Revision history statistics](#) · [Revision history search](#) · [Edits by user](#) · [Number of watchers](#) · [Page view statistics](#) · [Fix dead links](#)
(cur) = difference from current version, (prev) = difference from preceding version, m = minor edit, → = section edit, → = automatic edit summary (newest | oldest) View (newer 50 | older 50) (20 | 50 | 100 | 250 | 500)
[Compare selected revisions](#)

•

(cur | prev)

•

01:46, 15 April 2017 Lowercase signabot III (talk | contribs) m... (9,202 bytes) (**+602**)... ([Archiving 1 discussion\(s\) to Talk:Industrial and organizational psychology/Archive 2](#)) (bot) (undo)

•

(cur | prev)

•

01:43, 5 April 2017 Lowercase signabot III (talk | contribs) m... (9,704 bytes) (**+4,805**)... ([Archiving 1 discussion\(s\) to Talk:Industrial and organizational psychology/Archive 2](#)) (bot) (undo)

•

(cur | prev)

•

23:04, 22 March 2017 Charlotte135 (talk | contribs)... (14,509 bytes) (+969)... ([←](#) [Innovation](#)) (undo)

•

(cur | prev)

•

12:17, 22 March 2017 Jowa fan (talk | contribs)... (14,141 bytes) (**+1,679**)... ([←](#) [Innovation: new section](#)) (undo)

•

(cur | prev)

•

08:30, 3 March 2017 Armstr86 (talk | contribs)... (13,065 bytes) (+117)... ([←](#) [Business Psychology](#)) (undo)

•

(cur | prev)

•

08:28, 3 March 2017 Armstr86 (talk | contribs)... (12,948 bytes) (+21)... ([←](#) [Business Psychology](#)) (undo)

•

(cur | prev)

•

08:27, 3 March 2017 Armstr86 (talk | contribs)... (12,927 bytes) (+117)... ([←](#) [Business Psychology: new section](#)) (undo)

•

(cur | prev)

•

05:48, 1 February 2017 Charlotte135 (talk | contribs)... (12,810 bytes) (+159)... (undo)

•

(cur | prev)

•

20:53, 30 January 2017 Charlotte135 (talk | contribs)... (12,651 bytes) (+373)... ([←](#) [Organizational citizenship behavior](#)) (undo)

•

(cur | prev)

•

17:17, 18 January 2017 Is246 (talk | contribs)... (12,278 bytes) (+116)... ([←](#) [Organizational citizenship behavior: Thank you](#)) (undo)

•

(cur | prev)

•

05:16, 15 January 2017 Charlotte135 (talk | contribs)... (12,162 bytes) (+229)... ([←](#) [Organizational citizenship behavior](#)) (undo)

•

(cur | prev)

•

01:20, 15 January 2017 Is246 (talk | contribs)... (11,993 bytes) (+493)... ([←](#) [Training and training evaluation: Agreed](#)) (undo)

•

(cur | prev)

•

01:19, 15 January 2017 Is246 (talk | contribs) m... (11,842 bytes) (+115)... ([←](#) [Organizational citizenship behavior](#)) (undo)

•

(cur | prev)

•

01:19, 15 January 2017 Is246 (talk | contribs) m... (11,825 bytes) (+12)... ([←](#) [Organizational citizenship behavior](#)) (undo)

•

(cur | prev)

•

01:18, 15 January 2017 Is246 (talk | contribs)... (11,825 bytes) (**+643**)... ([←](#) [Organizational citizenship behavior: Restore please](#)) (undo)

•

(cur | prev)

•


01:17, 15 January 2017 Charlotte135 (talk | contribs)... (10,982 bytes) (+173)... ([←](#) [Training and training evaluation](#)) (undo)

•

(cur | prev)

•


01:16, 15 January 2017 Charlotte135 (talk | contribs)... (10,809 bytes) (+238)... ([←](#) [Organizational citizenship behavior](#)) (undo)



BASKETBALL REFERENCE

PlayersTeamsSeasonsLeadersScores²Playoffs

« Previous Season



CHICAGO BULLS

2016-17 Chicago Bulls Schedule and Results

Record: 41-41, 8th in [NBA Eastern Conference](#)
Last Game: [Friday, Apr. 28 vs. BOS](#)
Coach: [Fred Holberg](#) (41-41)
PTS/G: 102.9 (23rd of 30) [Opp](#) PTS/G: 102.4 (6th of 30)
SRS: 0.03 (14th of 30) [Pace](#): 95.3 (20th of 30)
[Off Rtg](#): 107.4 (20th of 30) [Def Rtg](#): 107.0 (6th of 30)
[Expected W-L](#): 42-40 (14th of 30)


More Team Info ▼

Bulls Franchise IndexTeam ClubhouseRoster & StatsSchedule & ResultsTransactionsMore 2016-17 Bulls F

Regular Season

Share & more ▼Glossary

G	Date	Box Score	Opponent	W	L	Opp	W	L	Streak	Notes
1	Thu, Oct 27, 2016	8:00p ET	Boston Celtics	W	105	99	1	0	W 1	
2	Sat, Oct 29, 2016	8:00p ET	Indiana Pacers	W	118	101	2	0	W 2	
3	Mon, Oct 31, 2016	7:30p ET	Brooklyn Nets	@	118	88	3	0	W 3	
4	Wed, Nov 2, 2016	8:00p ET	Boston Celtics	L	100	107	3	1	L 1	
5	Fri, Nov 4, 2016	8:00p ET	New York Knicks	L	104	117	3	2	L 2	
6	Sat, Nov 5, 2016	7:00p ET	Indiana Pacers	L	94	111	3	3	L 3	
7	Mon, Nov 7, 2016	8:00p ET	Orlando Magic	W	112	80	4	3	W 1	
8	Wed, Nov 9, 2016	7:30p ET	Atlanta Hawks	L	107	115	4	4	L 1	
9	Thu, Nov 10, 2016	8:00p ET	Miami Heat	W	98	95	5	4	W 1	
10	Sat, Nov 12, 2016	8:00p ET	Washington Wizards	W	106	95	6	4	W 2	
11	Tue, Nov 15, 2016	10:00p ET	Portland Trail Blazers	W	113	88	7	4	W 3	
12	Thu, Nov 17, 2016	10:30p ET	Utah Jazz	W	85	77	8	4	W 4	
13	Sat, Nov 19, 2016	10:30p ET	Los Angeles Clippers	L	95	102	8	5	L 1	
14	Sun, Nov 20, 2016	9:30p ET	Los Angeles Lakers	W	118	110	9	5	W 1	
15	Tue, Nov 22, 2016	9:00p ET	Denver Nuggets	L	107	110	9	6	L 1	
16	Fri, Nov 25, 2016	7:30p ET	Philadelphia 76ers	W	105	89	10	6	W 1	
17	Wed, Nov 30, 2016	8:00p ET	Los Angeles Lakers	L	96	96	10	7	L 1	
18	Fri, Dec 2, 2016	8:00p ET	Cleveland Cavaliers	W	111	105	11	7	W 1	
19	Sat, Dec 3, 2016	8:30p ET	Dallas Mavericks	L	87	107	11	8	L 1	



THE NEW YORK TIMES

SUBSCRIBE NOWSIGN INRegister

Get 50% off The Times.

Journalists on the ground. Stories grounded in facts.

DEALBOOKMARKETSECONOMYENERGYMEDIA TECHNOLOGYPERSONAL TECHENTREPRENEURSHIPYOUR MONEY

priceline.com

KISSIMMEE MONTECARLO MOTEL from \$44.99 Book Now

LAKE BUENA VISTA WYATT PLACE ORLANDO LAKE from \$94.05 Book Now


KISSIMMEE ECONOLodge Inn & Suites from \$29.49 Book Now

Revisiting Nafta: The Stakes for Key Industries

By THE NEW YORK TIMES 3:51 PM ET

President Trump says he won't immediately try to terminate the North American Free Trade Agreement, but some sectors would be vulnerable in a renegotiation.

Trump Says He Will Renegotiate Nafta or Terminate It



A woman carrying goods across the Paso del Norte International Bridge connecting El Paso and Ciudad Juárez, Mexico, in March.

LATEST NEWS

All Sources | The New York Times

14 Minutes Ago Reuters In Nod to Oil Industry, Trump to Order Review of Offshore Drilling Bans

23 Minutes Ago Reuters Microsoft's Silver Lining: Surface Loses, but Windows Wins

31 Minutes Ago NYT News Cloud Produces Sunny Earnings at Amazon, Facebook and Alphabet

MARKETS

Data delayed at least 15 minutes 04/27/2017 8:07 PM ET

U.S. AMERICAS EUROPE ASIA CURRENCIES COMMODITIES

Nikkei 19,215.34 -36.53 -0.19% NIKKEI JAPAN

Hang Seng 24,696.48 +120.05 +0.49%

Shanghai 3,152.09 -0.10 -0.00%

All Ordinaries 5,937.30 -7.13 -0.12%

COLLECTING BIG DATA

‣ How to collect this available data?

‣ **Human collection method:**

- Sit in front of a computer
- Go to a website of interest
- Copy the relevant data
- Paste into a common file
- Repeat 1,000,000 times



COLLECTING BIG DATA

‣ **Limitations of Human Collection:**

- Mental Demands

- Cost

- Inaccuracy

- Scalability

COLLECTING BIG DATA



COLLECTING BIG DATA



COLLECTING BIG DATA

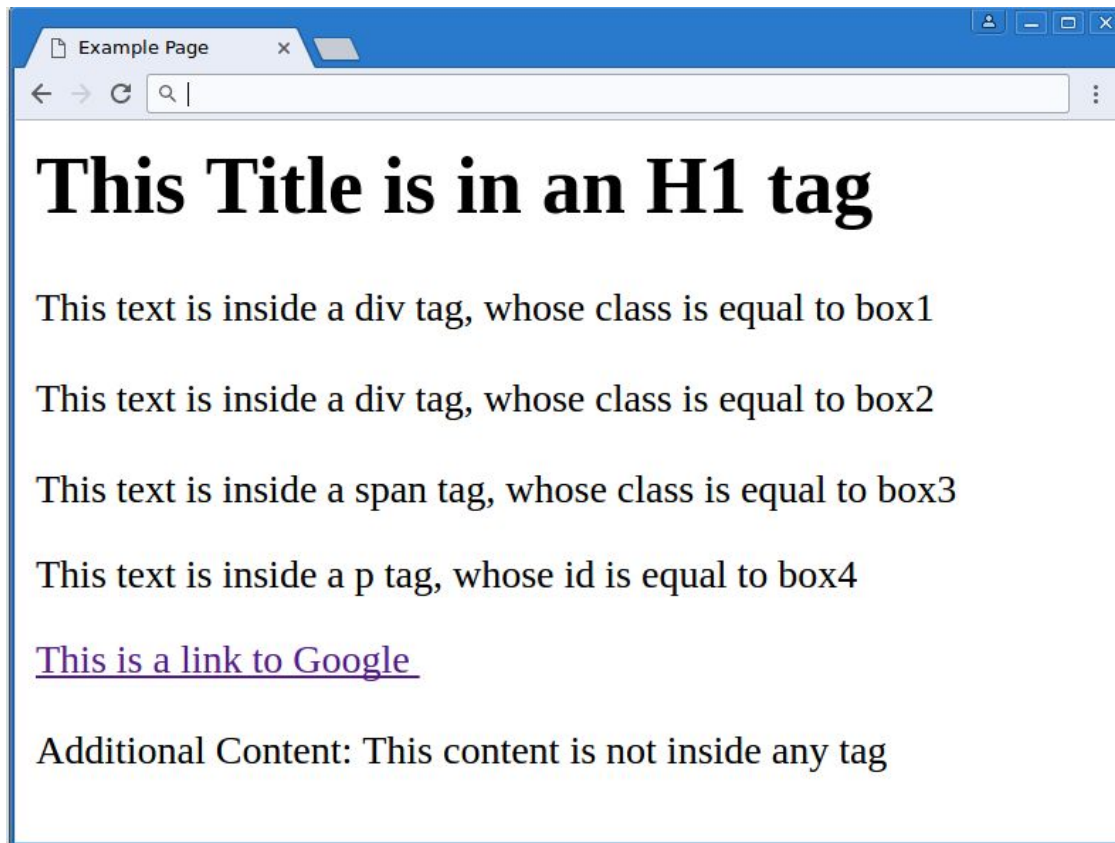


AUTOMATED DATA COLLECTION EXAMPLE

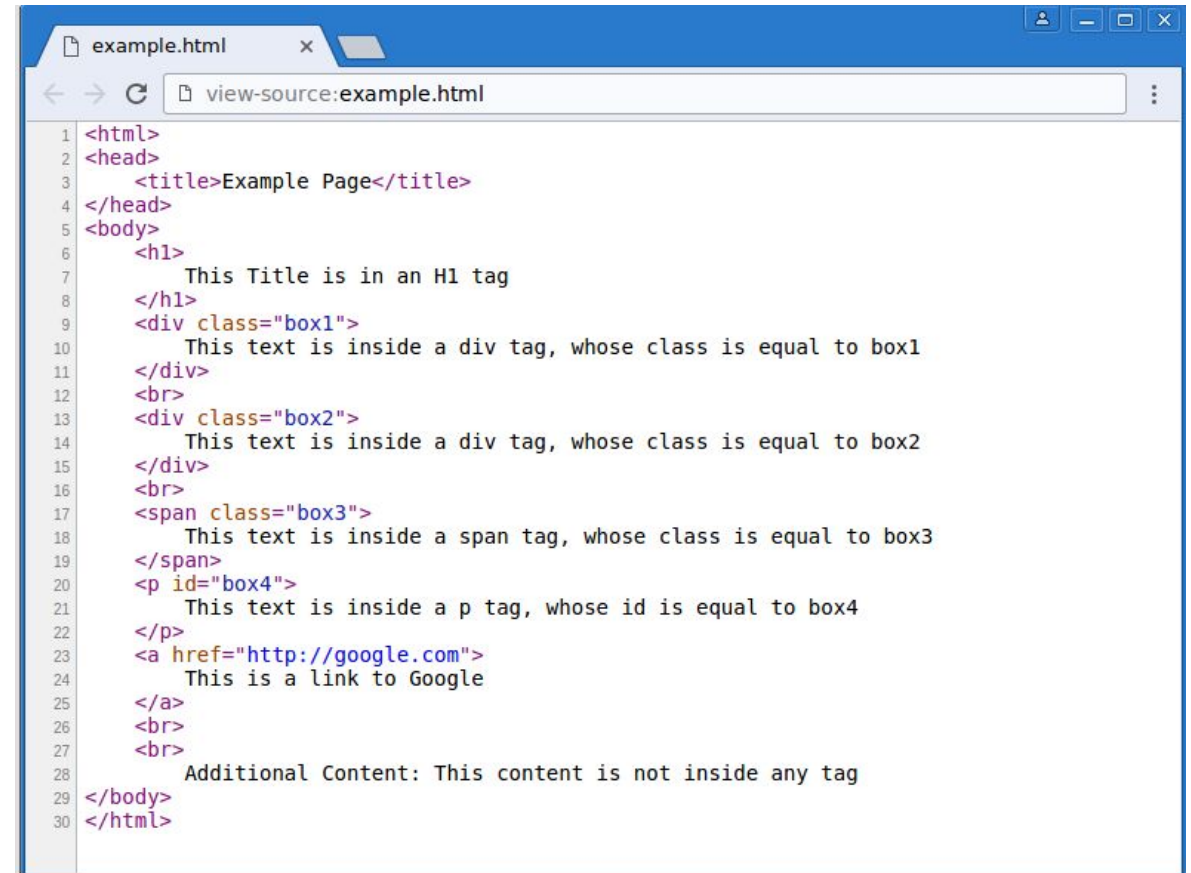
- Automated Data collection is about being able to translate what **you would do** as a human collecting the data to what your **computer can do**
- **Give a computer a set of instructions to follow**
 - **First do this**
 - **Then do that**
 - **Finally do this**
- Let the computer carry-out those instructions, and you come back to a completed project
- How do you talk to a computer?

THINKING LIKE A COMPUTER

- ▶ Automating requires you to consider what are the capabilities and limitations of a computer



This is what you see



This is what your computer sees

AUTOMATED DATA COLLECTION EXAMPLE

‣ Automated Data collection is about being able to translate what **you would do** as a human collecting the data to corresponding steps of what your **computer can do**

‣ **Example: Download the Main Headline from the New York Times**

‣ **What you would do:**

- Go to the New York Times website
- Look at the text in the main heading
- Copy that headline with the mouse
- Open a text file called “data.txt”
- Paste the copied text in the file
- Save it

‣ **What your computer can do:**

```
‣page = requests.get("http://nyt.com").text
‣headline = page.find("h1")
‣text = headline.text
‣file = open("data.txt","wb")
‣file.write(text)
‣file.close()
```

THINK ABOUT HOW YOU WOULD DO IT FIRST

- You have to think about everything you would do, and how your computer can do it.
- First, think how would you download the latest stock prices for Apple?
 - I would go to Google Finance (<https://www.google.com/finance>)
 - I would type in “Apple” at the search bar
 - I would look for the bold number
 - I would copy the price
 - I would open a text file
 - I would paste the price into the file
 - I would save the file and close it



TRANSLATING TO A COMPUTER

‣How can you have your computer do those steps:

‣It would be hard to have a computer type in a search box, so I have to think of a way for it to access a stock another way - THINK ABOUT WHAT A COMPUTER CAN DO

‣Notice that the url for Apple's stock price page is:

‣<https://www.google.com/finance?q=APPL>

‣The stock name always comes after “q=”

‣If I know the stock name, I can tell a computer to go to that page

‣I can tell a computer to look for text tagged as bold

‣I can tell a computer to save the bold text as a variable called “price”

‣I can tell the computer to open a file

‣I can tell the computer to write the stock price variable in the file

‣I can tell the computer to save and close the file

The underlined
text are all things
that your computer
knows how to do

FOUR STEPS OF AUTOMATED DATA COLLECTION

- Four Steps to Automatically Collecting Data (Scraping)
 - **Download the HTML source of a page**
 - **Extract the content from the HTML**
 - **Save the content**
 - **Repeat the process on a different Page**
- Each of those steps has specific commands in Python (and R) associated with it
- Successfully collecting data requires chaining those commands together

STEP 1: DOWNLOAD THE HTML SOURCE

‣ Download the HTML source of a page

‣ **Python command:**

```
page = requests.get("http://nyt.com").text
```

‣ **R Command**

```
library(RCurl)
```

```
page <- getURL("http://nyt.com")
```

STEP 2: EXTRACTING THE CONTENT

▸ **Extract the content**

We'll get to this part in a minute...

STEP 3: SAVE THE CONTENT

▸ Save the Content

▸ Python command:

```
textfile = open("data.txt", "a")  
textfile.write(content)  
textfile.close()
```

▸ R Command

```
write(content, "data.txt", append=TRUE)
```

STEP 4: REPEAT THE PROCESS

▸ Repeat the Processes

▸ Python command:

```
stocks = ["AAPL", "GOOGL", "MSFT"]  
for stock in stocks:  
    *** extract content ***
```

▸ R Command

```
stocks <- c("AAPL", "GOOGL", "MSFT")  
for (stock in stocks){  
    *** extract content ***  
}
```

STEP 2: EXTRACTING THE CONTENT

- The hardest part of automated data collection is extracting the content
- Code must be customized to your particular situation
- Depends on:
 - How much content is needed (one thing or many?)
 - The structure of the HTML (is it bold?, is it a heading?, is it italicized?)
 - The kind of content (is it text?, is it a url?, is it an image?)
- We will go over the major cases/situations that you could have

THE STRUCTURE OF A WEBSITE

- **Extracting content from a website requires understanding how websites are written**
- **Websites are written in HTML**
 - Text is formatted by putting it in between “tags”, which describe the way it should be displayed in a browser
 - Typically each tag has an opening tag and a closing tag, which isolate the specific text to be formatted
 -
 - **Example:**
 - `<h1>Hello</h1>`
 - `<i>Hello</i>`
 - `Hello`

VIEWING THE STRUCTURE OF A WEBSITE

‣To view the raw HTML of a website (i.e., the source), you can

‣Chrome/Firefox/Opera/Internet Explorer: Ctrl + U

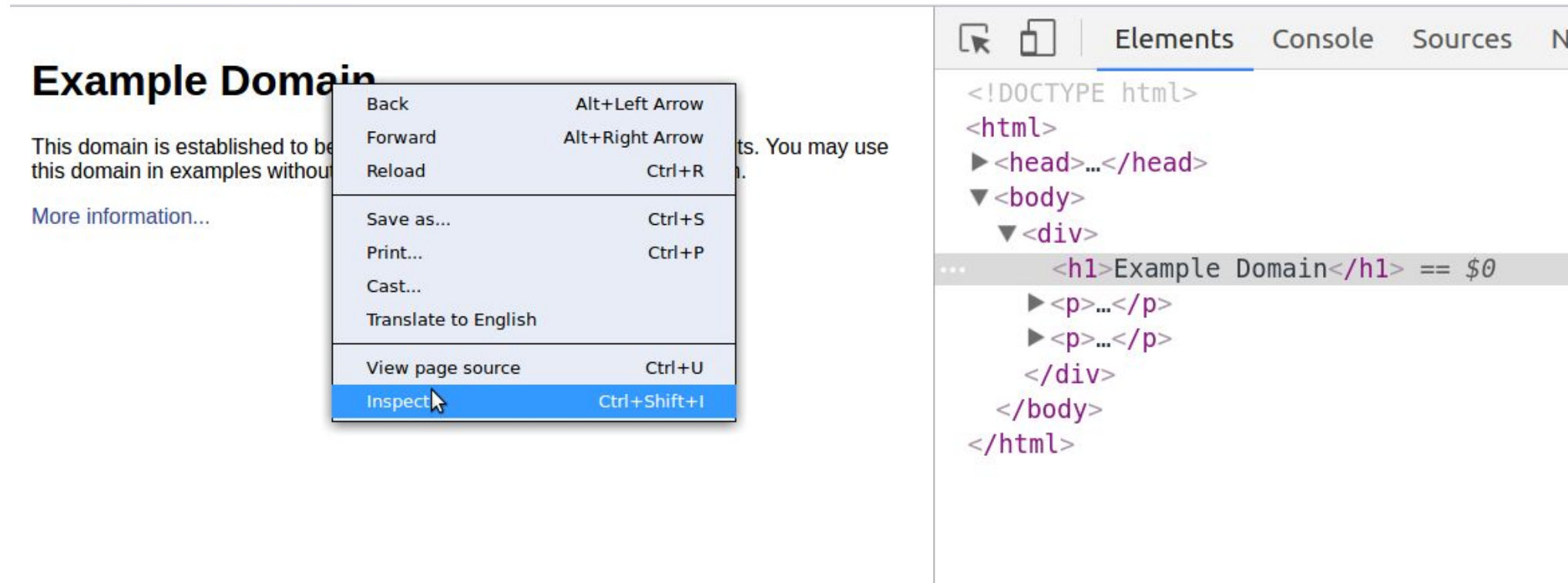
‣Safari: Command + Option + U

```
<html>
▼<head>
  <title>Example Domain</title>
</head>
▼<body>
  ▼<div>
    <h1>Example Domain</h1>
    ▼<p>
      "This domain is established to be used for illustrative examples in documents.
      You may use this
      domain in examples without prior coordination or asking for permission."
    </p>
    ▼<p>
      <a href="http://www.iana.org/domains/example">More information...</a>
    </p>
  </div>
</body>
</html>
```

The HTML source of a
<http://example.com>

VIEWING THE STRUCTURE OF A WEBSITE

- You can also right-click on a specific part of a website and select “Inspect” to more easily examine a specific part of the HTML



READING THE STRUCTURE OF A WEBSITE

The image illustrates the relationship between the visual presentation of a website and its underlying HTML structure. It is divided into two main panels.

Left Panel (Browser View): Shows a web browser window titled "Example Page". The content includes:

- A large heading: **This Title is in an H1 tag**
- Text: This text is inside a div tag, whose class is equal to box1
- Text: This text is inside a div tag, whose class is equal to box2
- Text: This text is inside a span tag, whose class is equal to box3
- Text: This text is inside a p tag, whose id is equal to box4
- A link: [This is a link to Google](http://google.com)
- Text: Additional Content: This content is not inside any tag

Right Panel (Source Code View): Shows the source code for "example.html". The code is as follows:

```
1 <html>
2 <head>
3   <title>Example Page</title>
4 </head>
5 <body>
6   <h1>
7     This Title is in an H1 tag
8   </h1>
9   <div class="box1">
10     This text is inside a div tag, whose class is equal to box1
11   </div>
12   <br>
13   <div class="box2">
14     This text is inside a div tag, whose class is equal to box2
15   </div>
16   <br>
17   <span class="box3">
18     This text is inside a span tag, whose class is equal to box3
19   </span>
20   <p id="box4">
21     This text is inside a p tag, whose id is equal to box4
22   </p>
23   <a href="http://google.com">
24     This is a link to Google
25   </a>
26   <br>
27   <br>
28   Additional Content: This content is not inside any tag
29 </body>
30 </html>
```

An arrow points from the title "This Title is in an H1 tag" in the browser view to the `<h1>` tag in the source code.

READING THE STRUCTURE OF A WEBSITE

The image illustrates the relationship between the visual presentation of a website and its underlying HTML structure. It is divided into two panels:

- Left Panel (Browser View):** Shows a web browser window titled "Example Page". The content includes:
 - A large heading: **This Title is in an H1 tag**
 - Text: "This text is inside a div tag, whose class is equal to box1"
 - Text: "This text is inside a div tag, whose class is equal to box2"
 - Text: "This text is inside a span tag, whose class is equal to box3"
 - Text: "This text is inside a p tag, whose id is equal to box4"
 - A link: [This is a link to Google](http://google.com)
 - Text: "Additional Content: This content is not inside any tag"
- Right Panel (Source Code View):** Shows the source code for "example.html". The code is as follows:

```
1 <html>
2 <head>
3   <title>Example Page</title>
4 </head>
5 <body>
6   <h1>
7     This Title is in an H1 tag
8   </h1>
9   <div class="box1">
10     This text is inside a div tag, whose class is equal to box1
11   </div>
12   <br>
13   <div class="box2">
14     This text is inside a div tag, whose class is equal to box2
15   </div>
16   <br>
17   <span class="box3">
18     This text is inside a span tag, whose class is equal to box3
19   </span>
20   <p id="box4">
21     This text is inside a p tag, whose id is equal to box4
22   </p>
23   <a href="http://google.com">
24     This is a link to Google
25   </a>
26   <br>
27   <br>
28   Additional Content: This content is not inside any tag
29 </body>
30 </html>
```

An arrow points from the `<h1>` tag in the source code (line 6) to the large heading text in the browser view, demonstrating how the browser interprets the HTML tags to render the page's structure.

READING THE STRUCTURE OF A WEBSITE

The image illustrates the relationship between the visual presentation of a website and its underlying HTML structure. It is divided into two panels:

Left Panel (Rendered Page): Shows a web browser window titled "Example Page". The content includes:

- This Title is in an H1 tag** (Large, bold black text)
- This text is inside a div tag, whose class is equal to box1
- This text is inside a div tag, whose class is equal to box2
- This text is inside a span tag, whose class is equal to box3
- This text is inside a p tag, whose id is equal to box4
- [This is a link to Google](http://google.com) (Underlined, purple text)
- Additional Content: This content is not inside any tag

Right Panel (Source Code): Shows the "view-source:example.html" page. The HTML code is as follows:

```
1 <html>
2 <head>
3   <title>Example Page</title>
4 </head>
5 <body>
6   <h1>
7     This Title is in an H1 tag
8   </h1>
9   <div class="box1">
10     This text is inside a div tag, whose class is equal to box1
11   </div>
12   <br>
13   <div class="box2">
14     This text is inside a div tag, whose class is equal to box2
15   </div>
16   <br>
17   <span class="box3">
18     This text is inside a span tag, whose class is equal to box3
19   </span>
20   <p id="box4">
21     This text is inside a p tag, whose id is equal to box4
22   </p>
23   <a href="http://google.com">
24     This is a link to Google
25   </a>
26   <br>
27   <br>
28   Additional Content: This content is not inside any tag
29 </body>
30 </html>
```

An arrow points from the first line of the source code (line 7) to the first line of the rendered page, highlighting the connection between the code and the visual output.

READING THE STRUCTURE OF A WEBSITE

The image illustrates the relationship between the visual rendering of a web page and its underlying HTML structure. It is divided into two panels:

- Left Panel (Browser View):** Shows the rendered page with the following content:
 - This Title is in an H1 tag** (Large heading)
 - This text is inside a div tag, whose class is equal to box1
 - This text is inside a div tag, whose class is equal to box2
 - This text is inside a span tag, whose class is equal to box3
 - This text is inside a p tag, whose id is equal to box4
 - [This is a link to Google](http://google.com)
 - Additional Content: This content is not inside any tag
- Right Panel (Source Code View):** Shows the raw HTML code for the page, with line numbers 1 through 30. The code structure is as follows:
 - 1: `<html>`
 - 2: `<head>`
 - 3: `<title>Example Page</title>`
 - 4: `</head>`
 - 5: `<body>`
 - 6: `<h1>`
 - 7: `This Title is in an H1 tag`
 - 8: `</h1>`
 - 9: `<div class="box1">`
 - 10: `This text is inside a div tag, whose class is equal to box1`
 - 11: `</div>`
 - 12: `
`
 - 13: `<div class="box2">`
 - 14: `This text is inside a div tag, whose class is equal to box2`
 - 15: `</div>`
 - 16: `
`
 - 17: ``
 - 18: `This text is inside a span tag, whose class is equal to box3`
 - 19: ``
 - 20: `<p id="box4">`
 - 21: `This text is inside a p tag, whose id is equal to box4`
 - 22: `</p>`
 - 23: ``
 - 24: `This is a link to Google`
 - 25: ``
 - 26: `
`
 - 27: `
`
 - 28: `Additional Content: This content is not inside any tag`
 - 29: `</body>`
 - 30: `</html>`

An arrow points from the text "This text is inside a div tag, whose class is equal to box2" in the browser view to the corresponding HTML code in the source view, highlighting the mapping between the visual content and the underlying structure.

READING THE STRUCTURE OF A WEBSITE

The image illustrates the relationship between the visual presentation of a website and its underlying HTML structure. It consists of two side-by-side browser windows.

Left Window (Example Page): Displays the rendered content of the website. It includes a large heading, several paragraphs of text, a link, and a final line of content not enclosed in any tag.

Right Window (example.html): Displays the raw HTML source code of the page. The code is numbered from 1 to 30. It shows the structure of the document, including the head, body, and various HTML tags used to format the content.

Annotations: Arrows point from descriptive text in the left window to the corresponding HTML code in the right window:

- An arrow points from the heading "This Title is in an H1 tag" to the `<h1>` tag on line 6.
- An arrow points from the text "This text is inside a div tag, whose class is equal to box1" to the `<div class="box1">` tag on line 9.
- An arrow points from the text "This text is inside a div tag, whose class is equal to box2" to the `<div class="box2">` tag on line 13.
- An arrow points from the text "This text is inside a span tag, whose class is equal to box3" to the `` tag on line 17.
- An arrow points from the text "This text is inside a p tag, whose id is equal to box4" to the `<p id="box4">` tag on line 20.

HTML Source Code (example.html):

```
1 <html>
2 <head>
3   <title>Example Page</title>
4 </head>
5 <body>
6   <h1>
7     This Title is in an H1 tag
8   </h1>
9   <div class="box1">
10     This text is inside a div tag, whose class is equal to box1
11   </div>
12   <br>
13   <div class="box2">
14     This text is inside a div tag, whose class is equal to box2
15   </div>
16   <br>
17   <span class="box3">
18     This text is inside a span tag, whose class is equal to box3
19   </span>
20   <p id="box4">
21     This text is inside a p tag, whose id is equal to box4
22   </p>
23   <a href="http://google.com">
24     This is a link to Google
25   </a>
26   <br>
27   <br>
28   Additional Content: This content is not inside any tag
29 </body>
30 </html>
```


READING THE STRUCTURE OF A WEBSITE

The image illustrates the relationship between the visual rendering of a web page and its underlying HTML structure. It is divided into two panels:

- Left Panel (Browser View):** Shows the rendered page with the following content:
 - This Title is in an H1 tag** (Large bold heading)
 - This text is inside a div tag, whose class is equal to box1
 - This text is inside a div tag, whose class is equal to box2
 - This text is inside a span tag, whose class is equal to box3
 - This text is inside a p tag, whose id is equal to box4
 - [This is a link to Google](http://google.com) (Underlined link)
 - Additional Content: This content is not inside any tag
- Right Panel (Source Code View):** Shows the raw HTML code for the page, with line numbers 1 through 30. The code structure is as follows:

```
1 <html>
2 <head>
3   <title>Example Page</title>
4 </head>
5 <body>
6   <h1>
7     This Title is in an H1 tag
8   </h1>
9   <div class="box1">
10     This text is inside a div tag, whose class is equal to box1
11   </div>
12   <br>
13   <div class="box2">
14     This text is inside a div tag, whose class is equal to box2
15   </div>
16   <br>
17   <span class="box3">
18     This text is inside a span tag, whose class is equal to box3
19   </span>
20   <p id="box4">
21     This text is inside a p tag, whose id is equal to box4
22   </p>
23   <a href="http://google.com">
24     This is a link to Google
25   </a>
26   <br>
27   <br>
28   Additional Content: This content is not inside any tag
29 </body>
30 </html>
```

An arrow points from the text "This text is inside a p tag, whose id is equal to box4" in the browser view to the corresponding HTML code in the source view, specifically to the line: `<p id="box4">`.

READING THE STRUCTURE OF A WEBSITE

The image illustrates the relationship between the visual rendering of a web page and its underlying HTML structure. It is divided into two panels:

Left Panel (Browser View): Shows the rendered page with the following content:

- This Title is in an H1 tag** (Large bold text)
- This text is inside a div tag, whose class is equal to box1
- This text is inside a div tag, whose class is equal to box2
- This text is inside a span tag, whose class is equal to box3
- This text is inside a p tag, whose id is equal to box4
- [This is a link to Google](http://google.com) (Underlined link)
- Additional Content: This content is not inside any tag

Right Panel (Source Code View): Shows the raw HTML code for the page, with line numbers 1 through 30. The code is as follows:

```
1 <html>
2 <head>
3   <title>Example Page</title>
4 </head>
5 <body>
6   <h1>
7     This Title is in an H1 tag
8   </h1>
9   <div class="box1">
10     This text is inside a div tag, whose class is equal to box1
11   </div>
12   <br>
13   <div class="box2">
14     This text is inside a div tag, whose class is equal to box2
15   </div>
16   <br>
17   <span class="box3">
18     This text is inside a span tag, whose class is equal to box3
19   </span>
20   <p id="box4">
21     This text is inside a p tag, whose id is equal to box4
22   </p>
23   <a href="http://google.com">
24     This is a link to Google
25   </a>
26   <br>
27   <br>
28   Additional Content: This content is not inside any tag
29 </body>
30 </html>
```

An arrow points from the rendered link in the browser view to the corresponding HTML code in the source code view, highlighting the connection between the visual element and its underlying structure.

READING THE STRUCTURE OF A WEBSITE

The image displays two side-by-side browser windows illustrating the relationship between a website's visual presentation and its underlying HTML structure.

Left Window (Example Page): Shows the rendered content of the page. It features a large heading, several paragraphs of text, a link, and a final line of text.

Right Window (example.html): Shows the source code of the page. The code is numbered 1 through 30, corresponding to the lines of HTML. It includes the `<html>`, `<head>`, `<title>`, `<body>`, `<h1>`, `<div>`, ``, `<p>`, and `<a>` tags, along with their closing tags and attributes.

Mapping: An arrow points from the source code to the browser view, indicating the mapping between the two. The source code includes comments explaining the purpose of each tag and attribute.

Source Code (example.html):

```
1 <html>
2 <head>
3   <title>Example Page</title>
4 </head>
5 <body>
6   <h1>
7     This Title is in an H1 tag
8   </h1>
9   <div class="box1">
10     This text is inside a div tag, whose class is equal to box1
11   </div>
12   <br>
13   <div class="box2">
14     This text is inside a div tag, whose class is equal to box2
15   </div>
16   <br>
17   <span class="box3">
18     This text is inside a span tag, whose class is equal to box3
19   </span>
20   <p id="box4">
21     This text is inside a p tag, whose id is equal to box4
22   </p>
23   <a href="http://google.com">
24     This is a link to Google
25   </a>
26   <br>
27   <br>
28   Additional Content: This content is not inside any tag
29 </body>
30 </html>
```

EXTRACTING THE CONTENT

‣ Extracting Content

- **When you have the HTML source of a website, you can see where the content you want is in that source**
 - What are its closest tags?
 - Are those tags unique to the content?
 - Does the tag have an id or class name?
 - Does some specific word or character always precede the content of interest?
- When you know the answers to the above questions, you direct Python to extract the content based on the identifying information.

SUMMARY