DePaul University

AUTOMATED DATA COLLECTION WITH PYTHON

Ivan Hernandez, Ph.D

DePaul University

GOALS FOR THE SESSION

- **Discuss the Growing Interest in Data**
- **Introduce Automated Data Collection Methodology**
- **Describe the Process of Automating Data Collection**
- **Present Methods to Extract Data from the Web**

All Session Materials available at: github.com/ivanhrndz

DATA TODAY

CHANGING PERSPECTIVES ON DATA

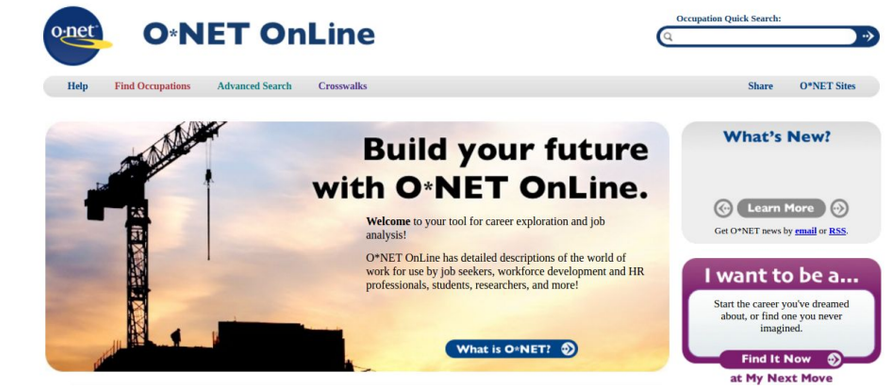
- **Data driven decisions** being emphasized

- **Age of Big Data**
  - Larger
  - More Frequent
  - More Varied

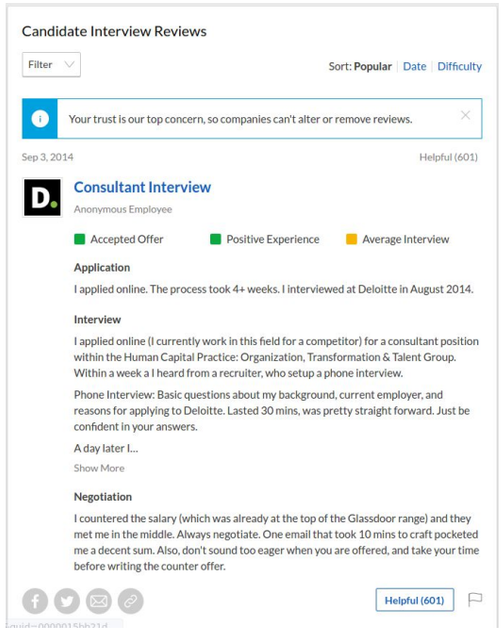
- **Where to access this data?**

SOURCES OF BIG DATA

Job Characteristics

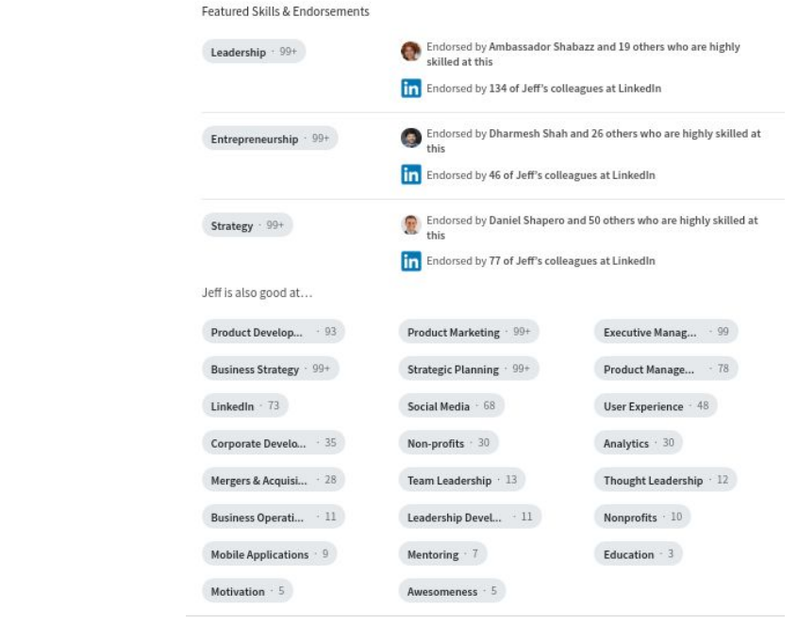


Workplace Attitudes

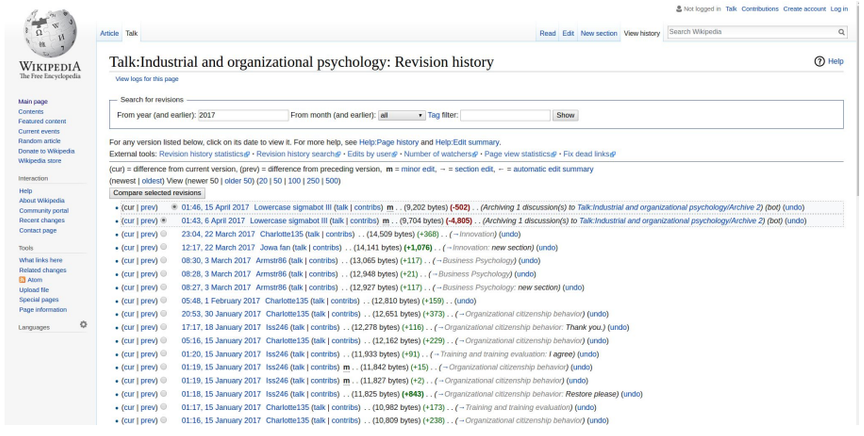


Team Processes and Performance

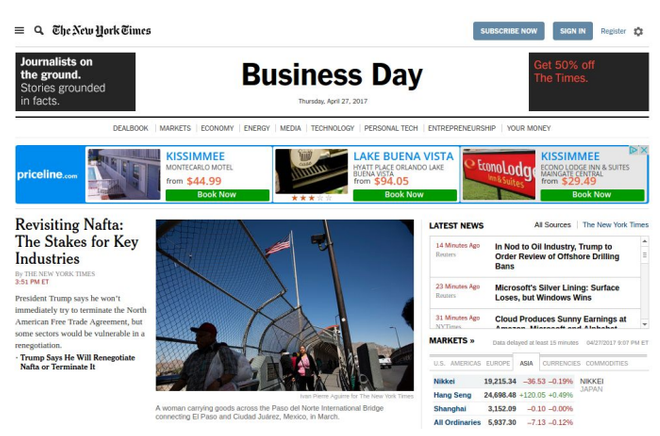
Employee History and Attributes



Collaboration



Business Trends



BENEFITS OF ACQUIRING BIG DATA

- **Web-based data can also facilitate <u>market intelligence and and examining both collective and individual behavior in social settings</u>**

- **Provides the following knowledge benefits**
  - Pricing analysis
  - Competitive intelligence
  - Events
  - Product data
  - Popularity
  - Reputation

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 for other data and other websites

COLLECTING BIG DATA

- **Limitations of Human Collection**:
  - Menial
  - Mental Demands
  - Inaccuracy
  - Cost
  - Scalability

COLLECTING BIG DATA

- Consider the following analogy:
- “The Sorcerer’s Apprentice”

- Mickey Mouse is tasked with helping a sorcerer
- Needs to clean an entire castle

COLLECTING BIG DATA

- **The required job is:**
  - Menial
  - Demanding
  - Requires precision
  - Costly (in time)
  - Not scalable

COLLECTING BIG DATA

- **Mickey solves problem by taking something inanimate, and giving it the ability to perform the task, as well the instructions it needs to follow**

COLLECTING BIG DATA

- The inanimate objects completes the task autonomously
- Mickey is free to spend his time in more productive ways

- The process is easily scaled
- Can conduct the task more efficiently, with little additional effort

AUTOMATED DATA COLLECTION

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**

- **Goal: 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?

HOW TO TALK TO A COMPUTER

- **We can tell a computer what to do using programming languages:**
  - Python
  - R
  - C
  - Java

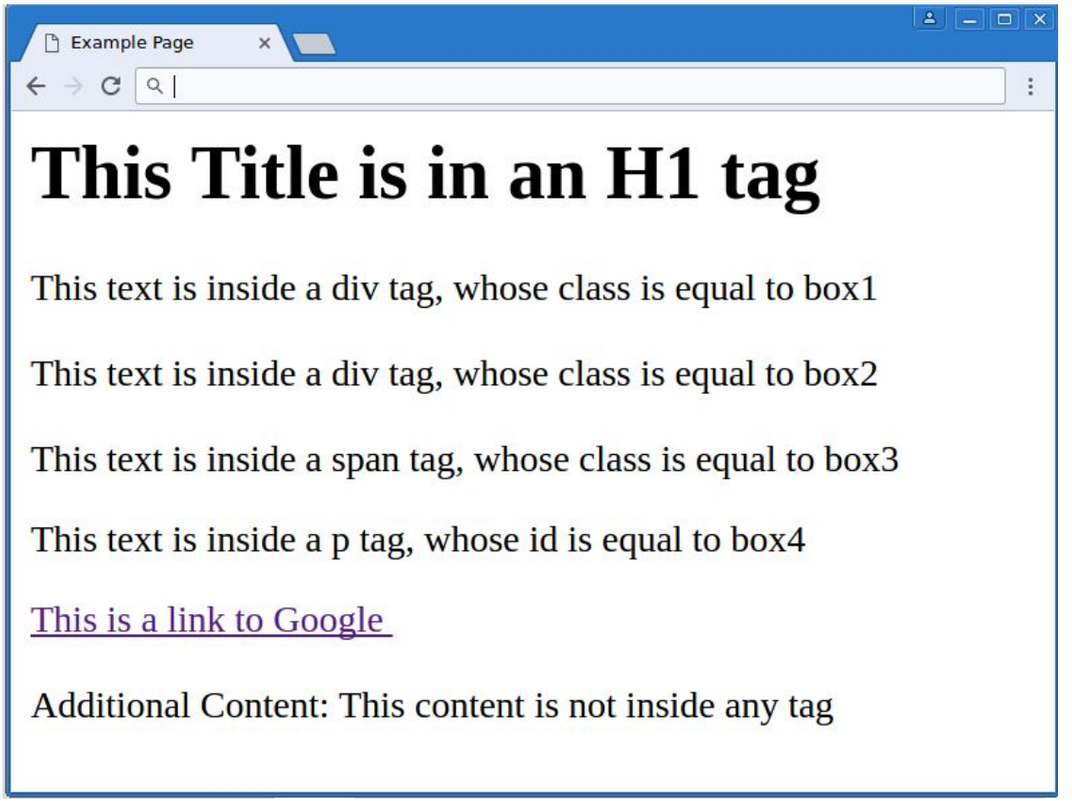
- **To tell a computer what to do using a programming language requires:**

  - Understanding how a computer sees things

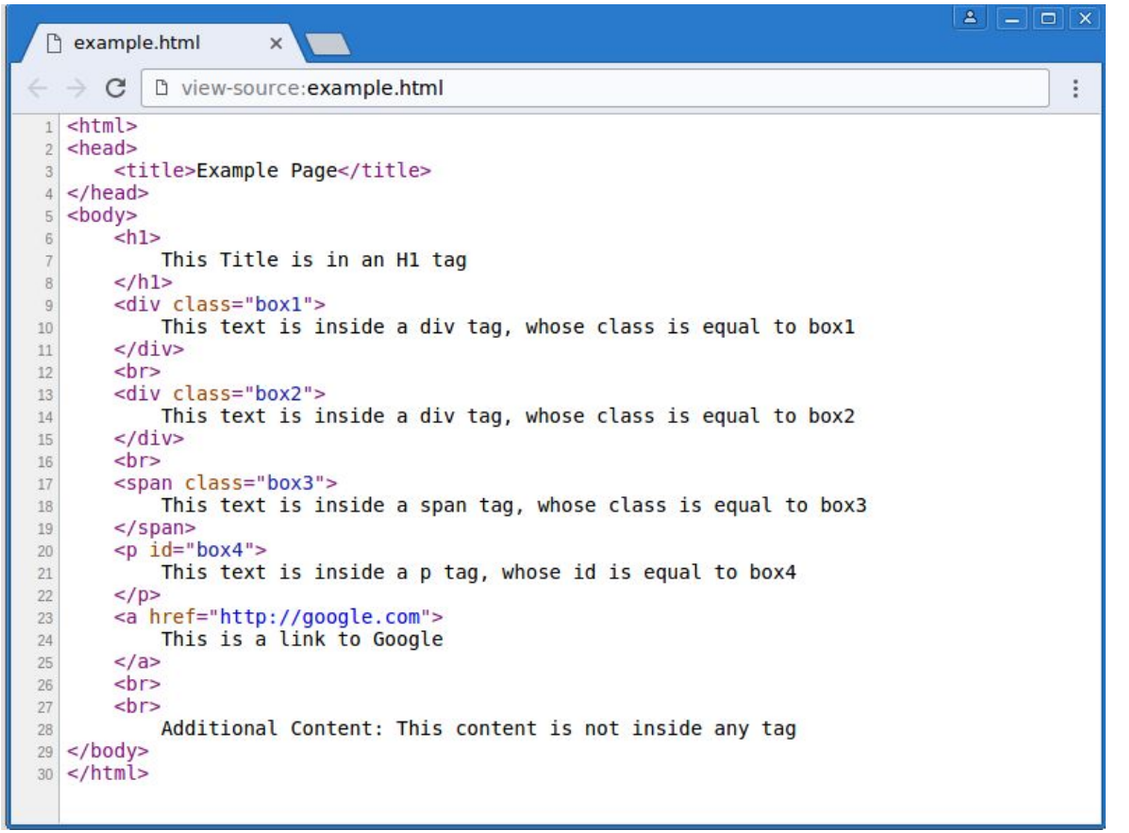
  - Understanding what the functions that are available

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

- Know the functions/instruction that are available from the programming language
- 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 <u>would</u> do, and how your computer <u>can</u> do it.**

- **<u>First, think how would YOU download the latest stock prices for Apple?</u>**
  - 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

- **Next, think about how can you have your COMPUTER do those same 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:**

```
import urllib
page = urllib.urlopen(“https://www.google.com/finance?q=APPL”)
```

  - **R Command**

```
library(RCurl)
page <- getURL(“https://www.google.com/finance?q=APPL”)
```

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 <u>extracting the content</u>**
  - **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

EXTRACTING CONTENT FROM WEB SITES

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>
    - *Hello*
    - **Hello**

VIEWING THE STRUCTURE OF A WEBSITE

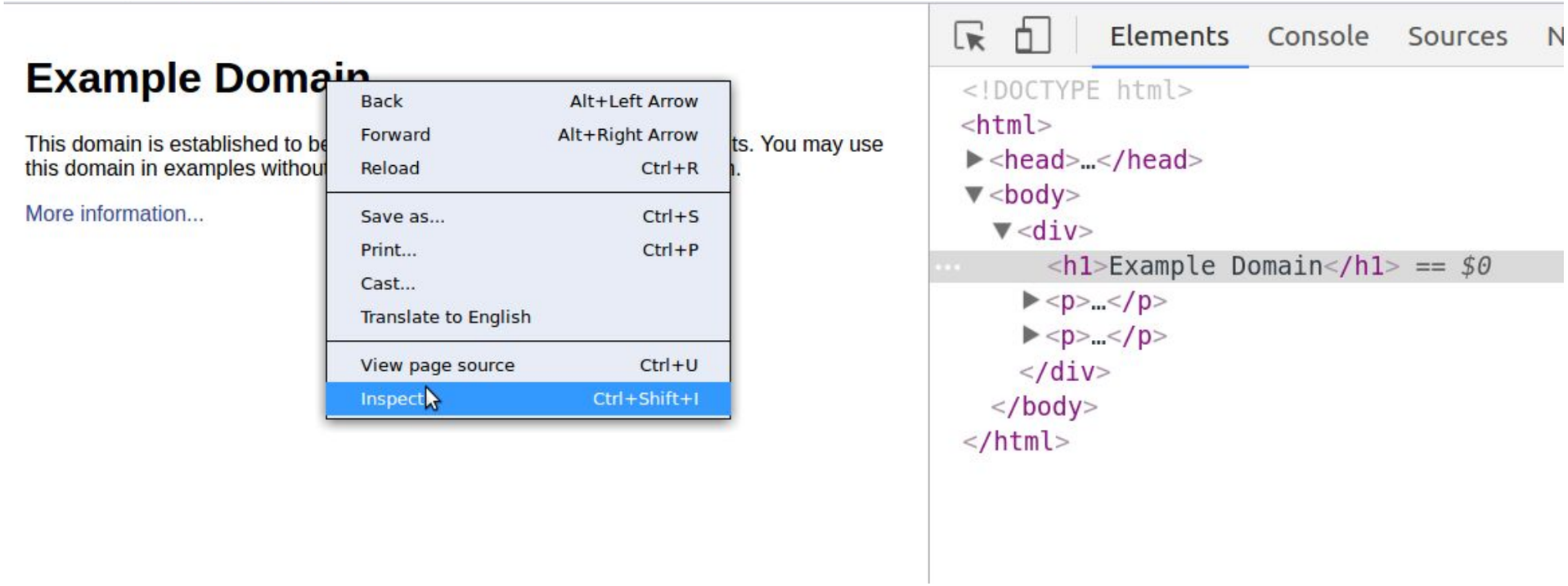
- To view the raw HTML of a website (i.e., the source), you can
  - Chrome/Firefox/Opera/Internet Explorer: Ctrl + U
  - Safari: Command + Option + U

```
<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>
```

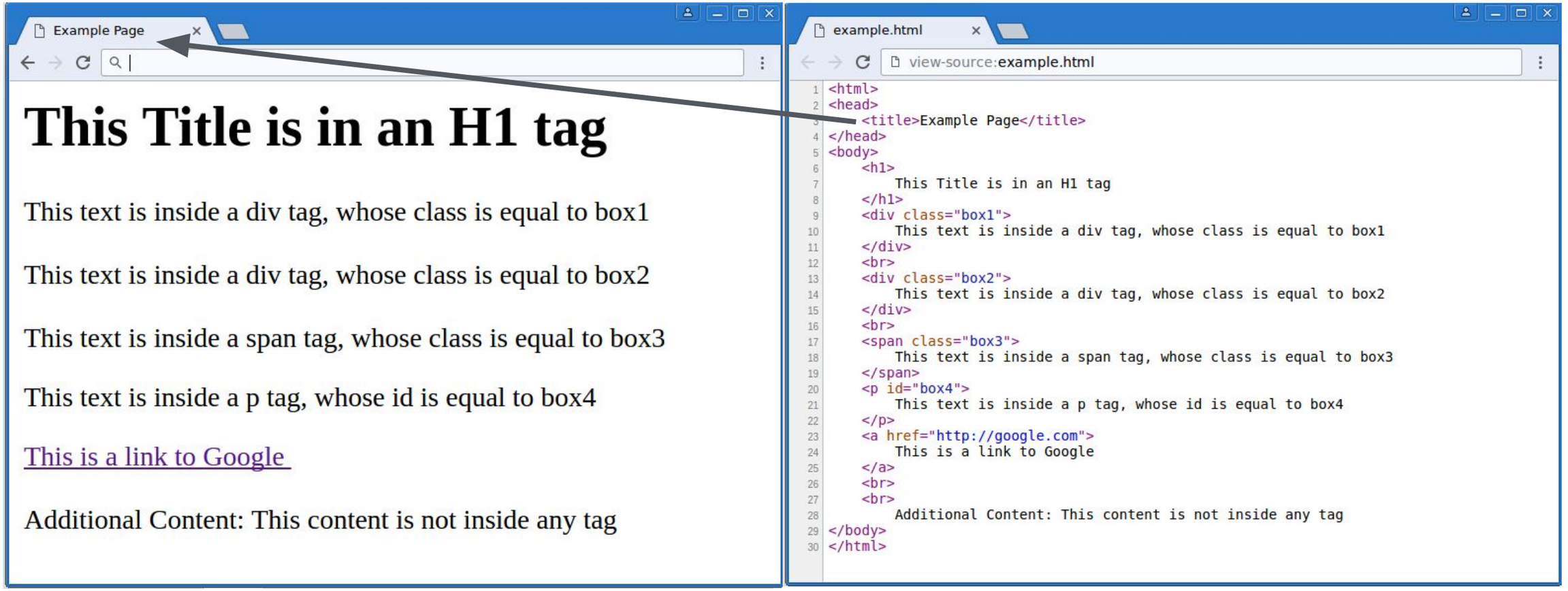
The HTML source of http://example.com

VIEWING THE STRUCTURE OF A WEBSITE

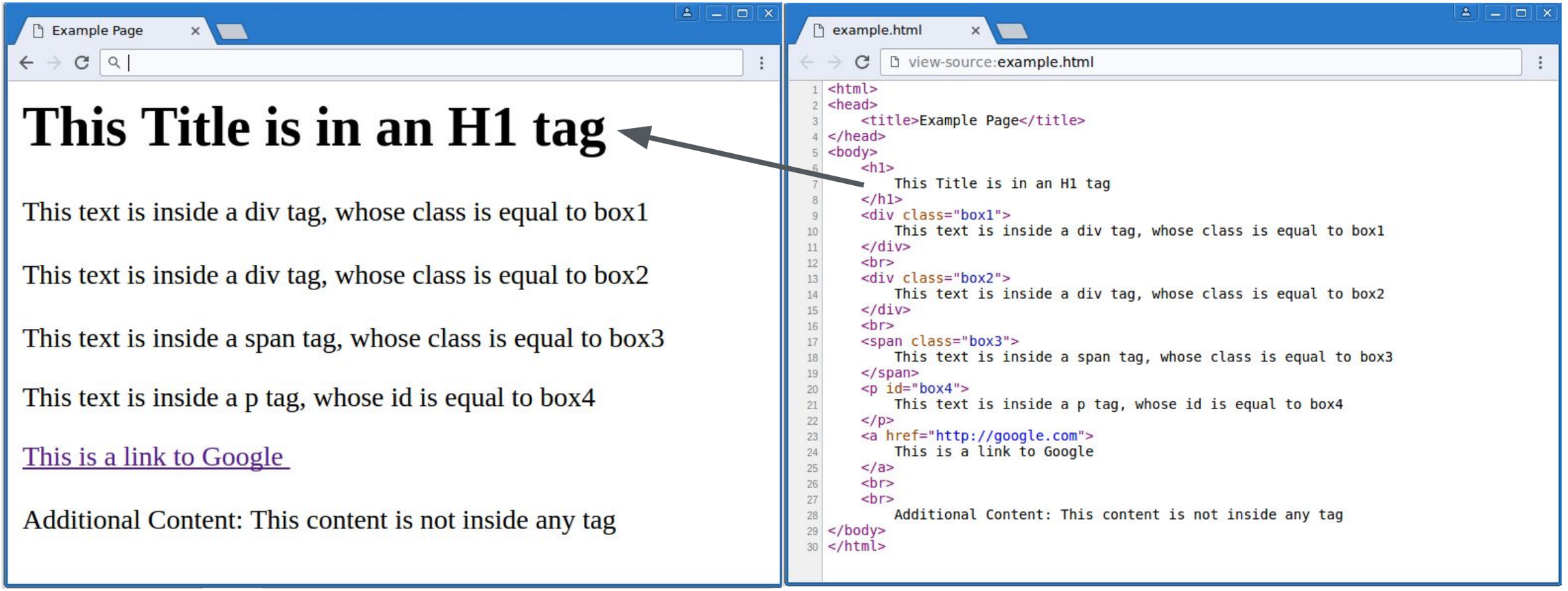
- You can also right-click on a specific part of a website and select "Inspect" to more easily examine a specific part of the HTML



READING THE STRUCTURE OF A WEBSITE

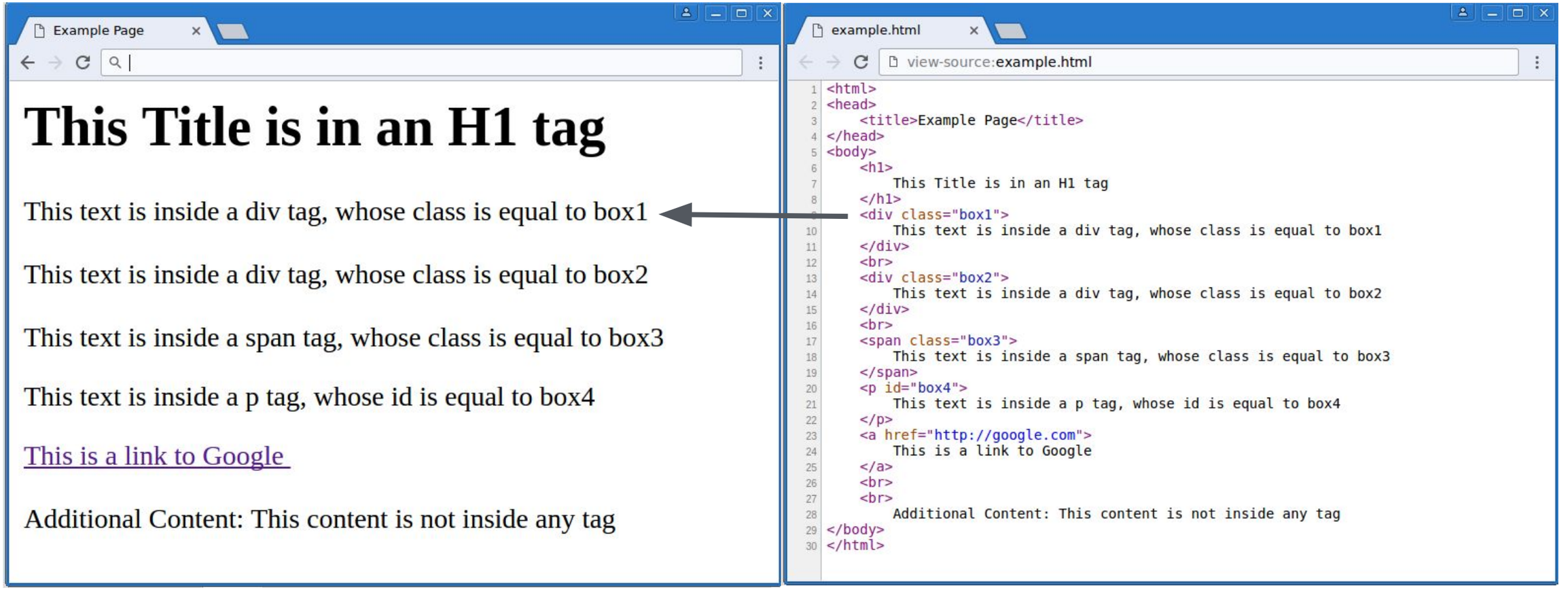


READING THE STRUCTURE OF A WEBSITE



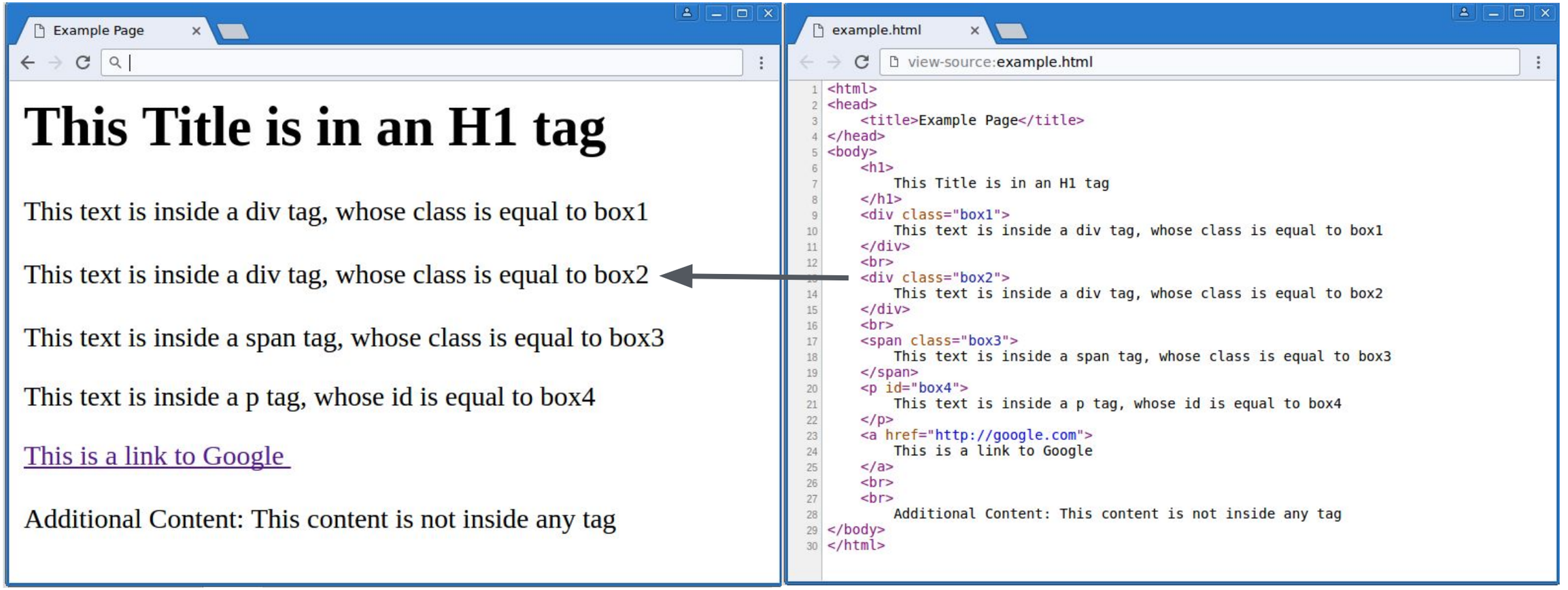
The main heading is inside of an <h1> tag

READING THE STRUCTURE OF A WEBSITE



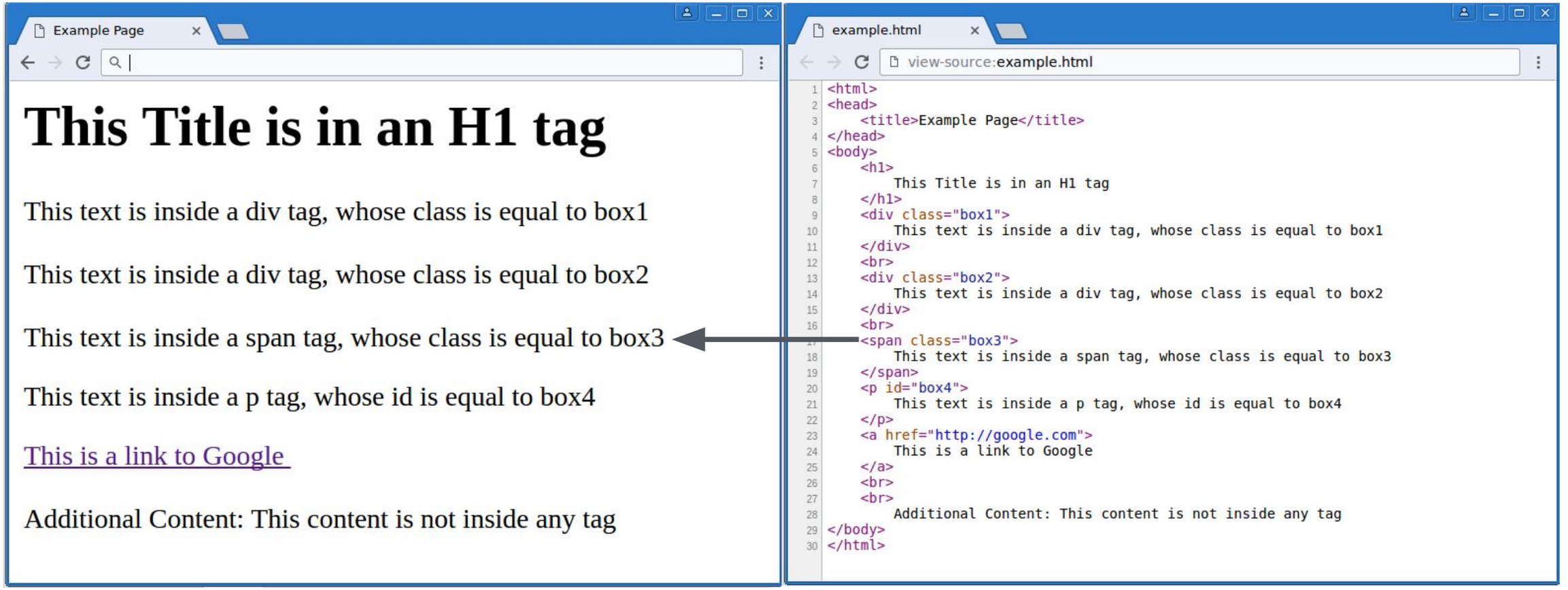
The second line is inside a tag with a class equal to “box1”

READING THE STRUCTURE OF A WEBSITE



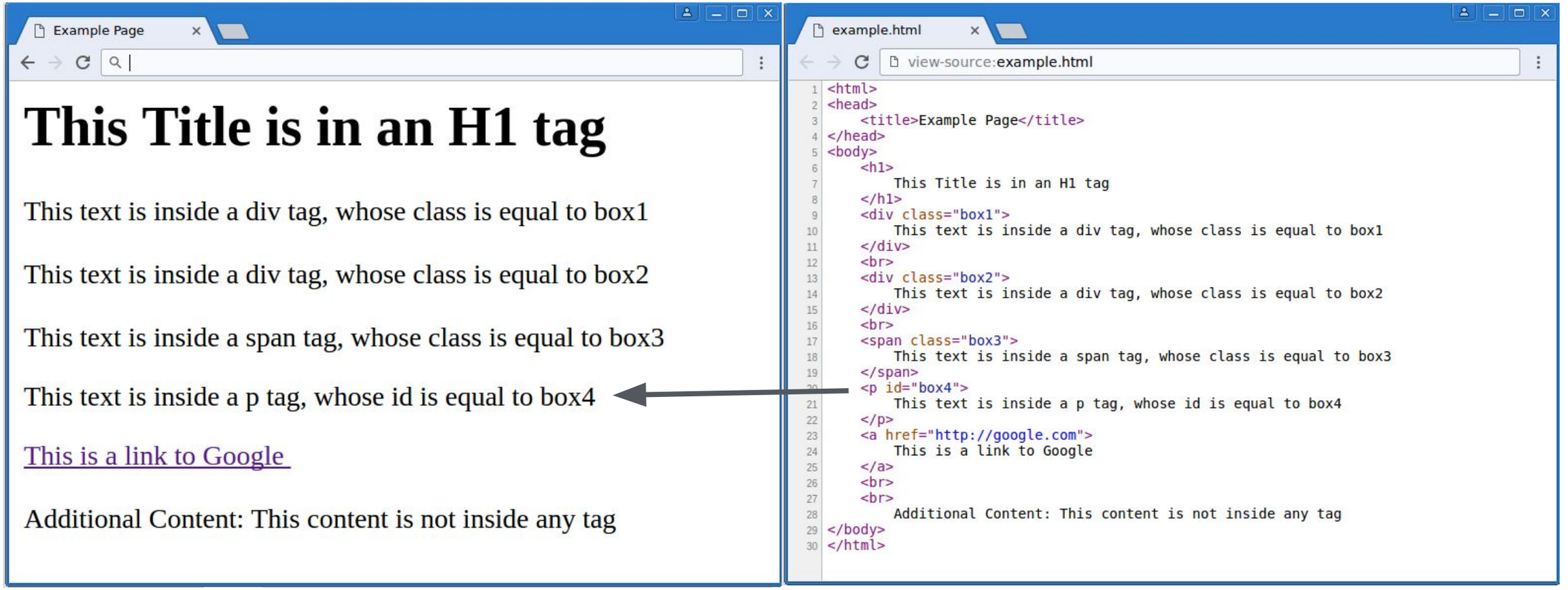
The third line is inside a tag with a class equal to “box2”

READING THE STRUCTURE OF A WEBSITE



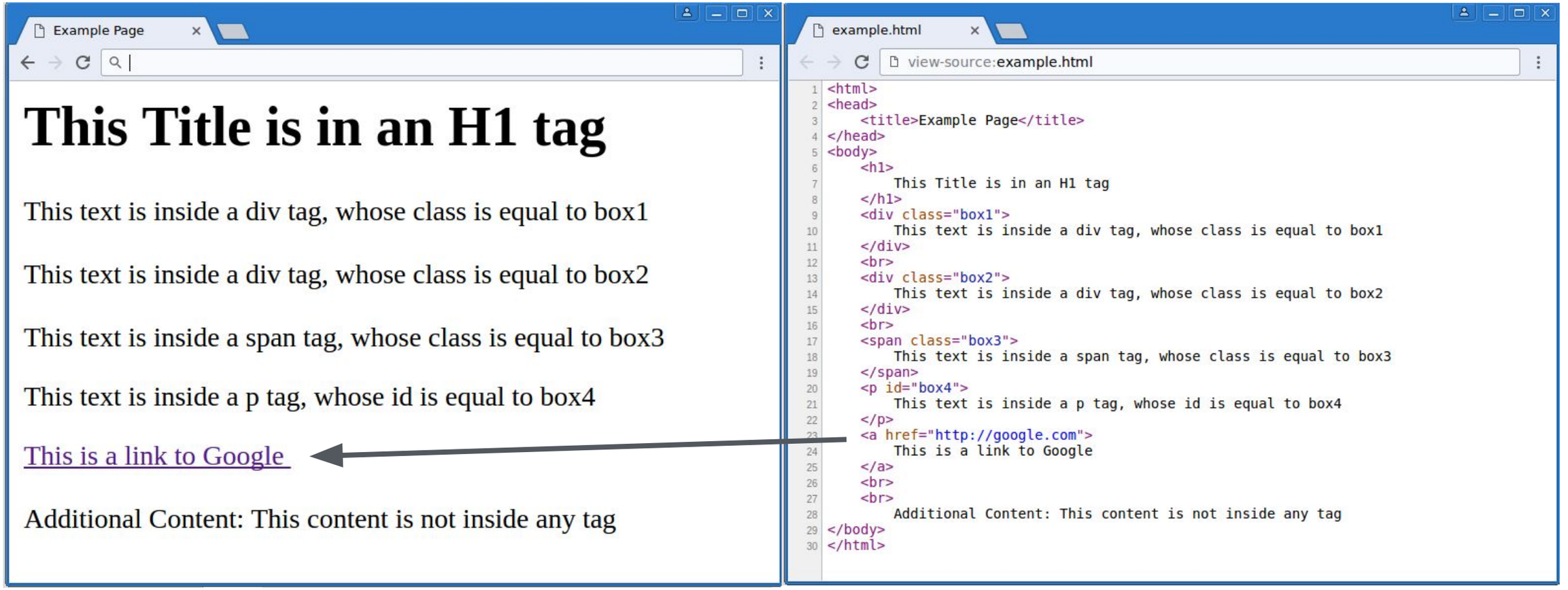
The fourth line is inside a tag with a class equal to “box3”

READING THE STRUCTURE OF A WEBSITE



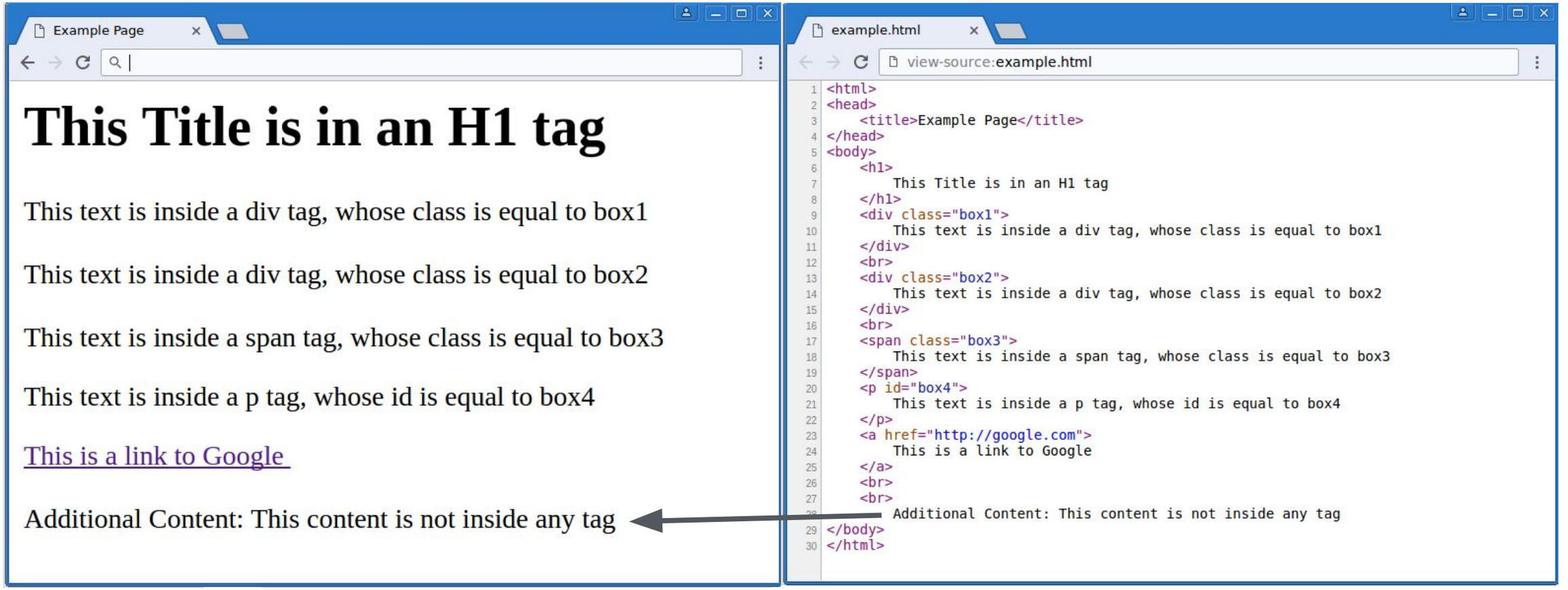
The fourth line is inside a <p> tag with an id equal to “box4”

READING THE STRUCTURE OF A WEBSITE



The fifth line is inside an <a> tag with an href that directs to google.com

READING THE STRUCTURE OF A WEBSITE



The fifth line is NOT inside any tags

EXTRACTING THE CONTENT

- **Extracting Content from a Web Page**
  - **When you have the HTML source of a website, you need to examine where in the source is the content you want to extract**
    - 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.

DEMONSTRATION OF DATA EXTRACTION

EXTRACTING THE CONTENT

- Walkthrough of How to Extract Web Page Content With Python:

- https://github.com/ivanhrndz/SIOP2017/blob/master/notebooks/Automated%20Data%20Collection.ipynb

SUMMARY

SUMMARY

- There's a growing interest in the benefits of "Big Data"
- The internet provides a vast source of data
- Data can be collected from the internet at scale through automation
- Automated data collection involves thinking of the steps a human would take when collecting the data, and translating those steps to procedures a computer can understand
- Using the urllib and BeautifulSoup libraries, Python provides a method for automating data collection from the internet.

CONTACT INFORMATION

For questions & comments:

Ivan Hernandez, Ph.D

DePaul University

ivan.hernandez@depaul.edu

github.com/ivanhrndz

SIOP encourages you to rate the sessions using the whova app or desktop site:

whova.com/webapp/e/siop_201704