Lecture 4b
Web Scraping

Pierre Biscaye
Université Clermont Auvergne

Data Science for Economics

Note: Materials for this lecture are drawn from the UC Berkeley D-Lab's Python Web Scraping course.

Why web scraping?

- Abundance of web data useful for social scientists
  - social media
  - news media
  - government publications
  - organizational records
- Two kinds of ways to get data off the web
  - Web scraping: pulling information from user-facing websites for humans
  - APIs: application-facing structured information access, for computers

Web scraping vs. APIs

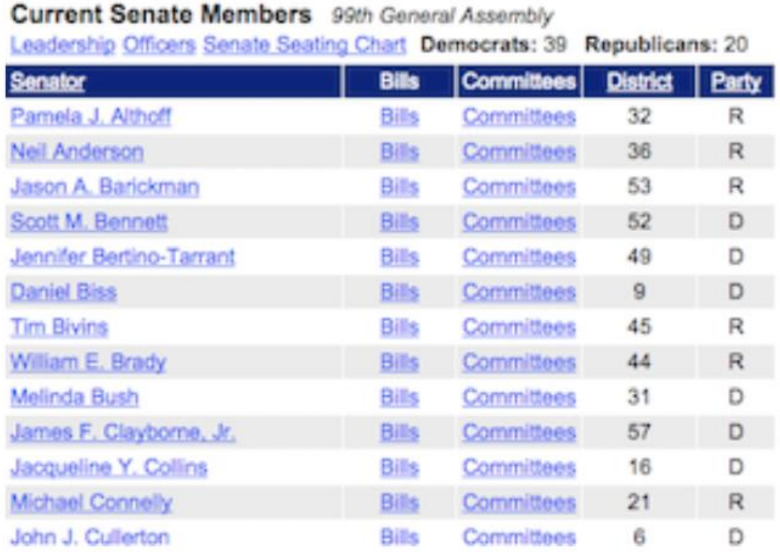
- Web scraping benefits
  - Any content that can be viewed on a website can be scraped
  - No API needed
  - No rate-limiting or authentication (usually)
- Web scraping challenges
  - Rarely tailored for information accessibility/organization
  - Your IP can be blocked
  - Messy, unstructured, inconsistent
  - Entirely site-dependent
- Rule of thumb: Check for API first. If not available, scrape.

Some disclaimers

- Check a site's terms and conditions before scraping.
    - Don’t scrape any personal/sensitive data.
    - Respect intellectual property.
    - Be transparent about sources of scraped data.
- Be nice - don't hammer the site's server.
    - Add a time delay in between batches of scraping.
    - Many requests from same IP in short amount of time can lead to a ban.
- Sites change their layout all the time. This can break your scraper.

What is a website?

- Some combination of codebase, database.
- The "front end" product is HTML + CSS stylesheets + javascript.
- Browser turns the left image into the right.



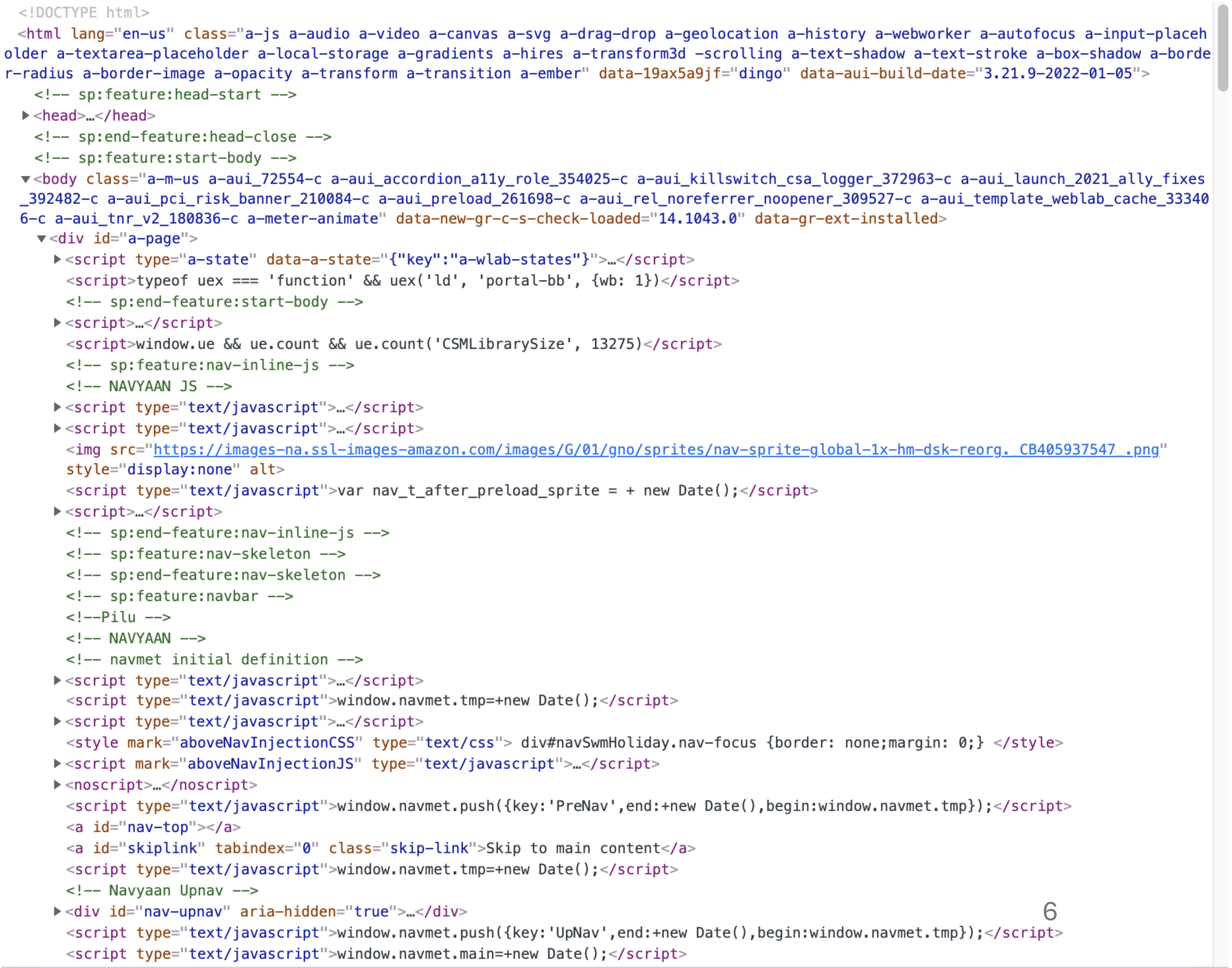
Current Senate Members *99th General Assembly*

Leadership Officers Senate Seating Chart **Democrats:** 39 **Republicans:** 20

Senator	Bills	Committees	District	Party
Pamela J. Althoff	Bills	Committees	32	R
Neil Anderson	Bills	Committees	36	R
Jason A. Barickman	Bills	Committees	53	R
Scott M. Bennett	Bills	Committees	52	D
Jennifer Bertino-Tarrant	Bills	Committees	49	D
Daniel Biss	Bills	Committees	9	D
Tim Bivins	Bills	Committees	45	R
William E. Brady	Bills	Committees	44	R
Melinda Bush	Bills	Committees	31	D
James F. Clayborne, Jr.	Bills	Committees	57	D
Jacqueline Y. Collins	Bills	Committees	16	D
Michael Connelly	Bills	Committees	21	R
John J. Cullerton	Bills	Committees	6	D

Web scraping returns HTML

- It's easy to pull HTML from a website using Python tools.
- It's much more difficult to find the information you want from that HTML.
- So we have to learn how to parse HTML to find the data we want.

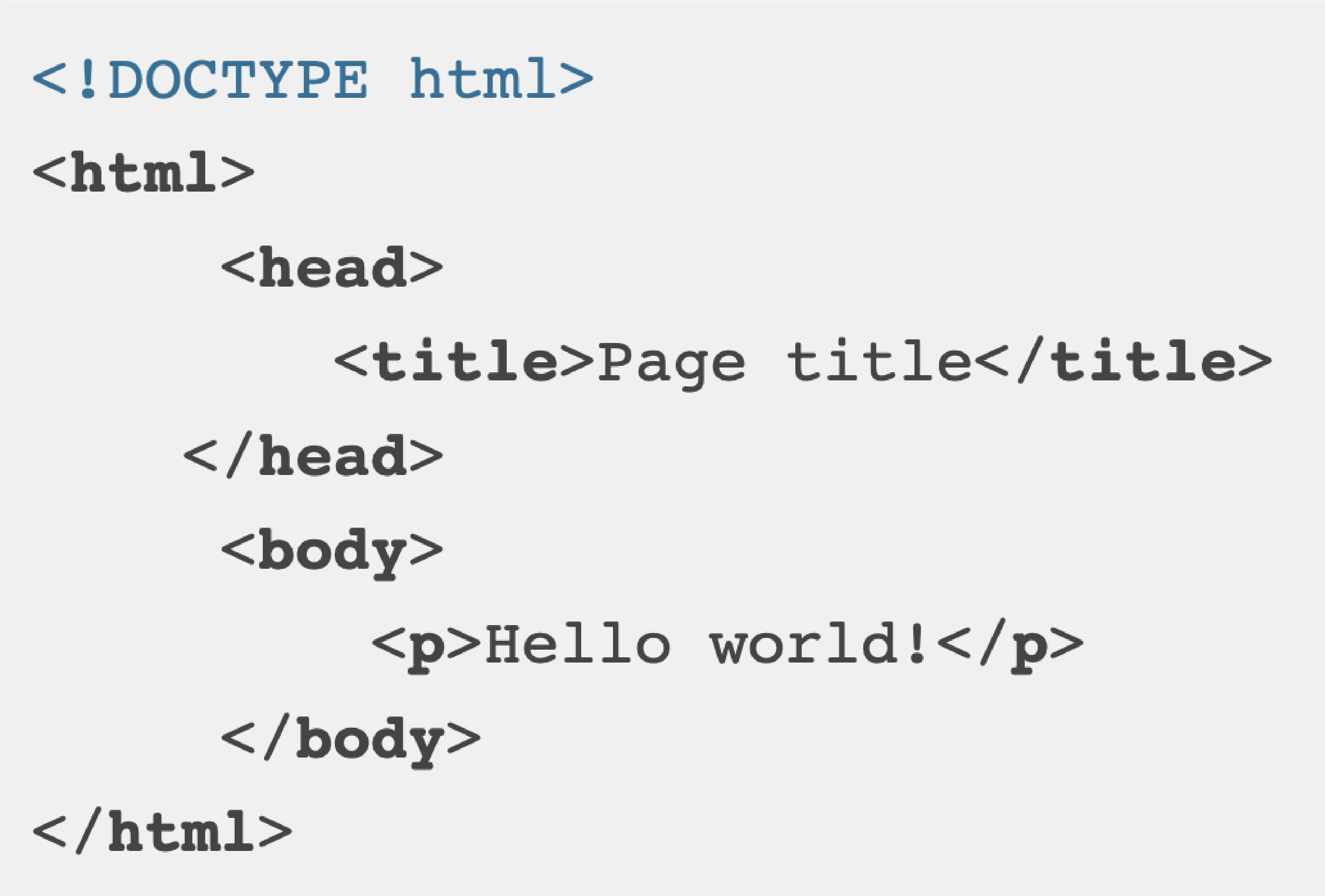


```
<!DOCTYPE html>
<html lang="en-us" class="a-js a-audio a-video a-canvas a-svg a-drag-drop a-geolocation a-history a-webworker a-autofocus a-input-placeh
older a-textarea-placeholder a-local-storage a-gradients a-hires a-transform3d -scrolling a-text-shadow a-text-stroke a-box-shadow a-borde
r-radius a-border-image a-opacity a-transform a-transition a-ember" data-19ax5a9jf="dingo" data-aui-build-date="3.21.9-2022-01-05">
  <!-- sp:feature:head-start -->
 ▶<head>…</head>
  <!-- sp:end-feature:head-close -->
  <!-- sp:feature:start-body -->
 ▼<body class="a-m-us a-aui_72554-c a-aui_accordion_a11y_role_354025-c a-aui_killswitch_csa_logger_372963-c a-aui_launch_2021_ally_fixes
 _392482-c a-aui_pci_risk_banner_210084-c a-aui_preload_261698-c a-aui_rel_noreferrer_noopener_309527-c a-aui_template_weblab_cache_33340
 6-c a-aui_tnr_v2_180836-c a-meter-animate" data-new-gr-c-s-check-loaded="14.1043.0" data-gr-ext-installed>
   ▼<div id="a-page">
     ▶<script type="a-state" data-a-state="{"key":"a-wlab-states"}">…</script>
      <script>typeof uex === 'function' && uex('ld', 'portal-bb', {wb: 1})</script>
      <!-- sp:end-feature:start-body -->
     ▶<script>…</script>
      <script>window.ue && ue.count && ue.count('CSMLibrarySize', 13275)</script>
      <!-- sp:feature:nav-inline-js -->
      <!-- NAVYAAN JS -->
     ▶<script type="text/javascript">…</script>
     ▶<script type="text/javascript">…</script>
      <img src="https://images-na.ssl-images-amazon.com/images/G/01/gno/sprites/nav-sprite-global-1x-hm-dsk-reorg._CB405937547_.png"
      style="display:none" alt>
      <script type="text/javascript">var nav_t_after_preload_sprite = + new Date();</script>
     ▶<script>…</script>
      <!-- sp:end-feature:nav-inline-js -->
      <!-- sp:feature:nav-skeleton -->
      <!-- sp:end-feature:nav-skeleton -->
      <!-- sp:feature:navbar -->
      <!--Pilu -->
      <!-- NAVYAAN -->
      <!-- navmet initial definition -->
     ▶<script type="text/javascript">…</script>
      <script type="text/javascript">window.navmet.tmp=+new Date();</script>
     ▶<script type="text/javascript">…</script>
      <style mark="aboveNavInjectionCSS" type="text/css"> div#navSwmHoliday.nav-focus {border: none;margin: 0;} </style>
     ▶<script mark="aboveNavInjectionJS" type="text/javascript">…</script>
     ▶<noscript>…</noscript>
      <script type="text/javascript">window.navmet.push({key:'PreNav',end:+new Date(),begin:window.navmet.tmp});</script>
      <a id="nav-top"></a>
      <a id="skiplink" tabindex="0" class="skip-link">Skip to main content</a>
      <script type="text/javascript">window.navmet.tmp=+new Date();</script>
      <!-- Navyaan Upnav -->
     ▶<div id="nav-upnav" aria-hidden="true">…</div>
      <script type="text/javascript">window.navmet.push({key:'UpNav',end:+new Date(),begin:window.navmet.tmp});</script>
      <script type="text/javascript">window.navmet.main=+new Date();</script>
```

Basic strategy of web scraping

- Find out what kind of HTML element your data is in. Use your browser‘s “inspector”.
- Think about how you can differentiate those elements from other, similar elements in the webpage using HTML/CSS anatomy.
    - This requires some basic knowledge of HTML/CSS.
    - We will go over some basic concepts below; just enough to get your started.
- Use Python and add-on modules like BeautifulSoup to extract just that data.

HTML: Basic structure



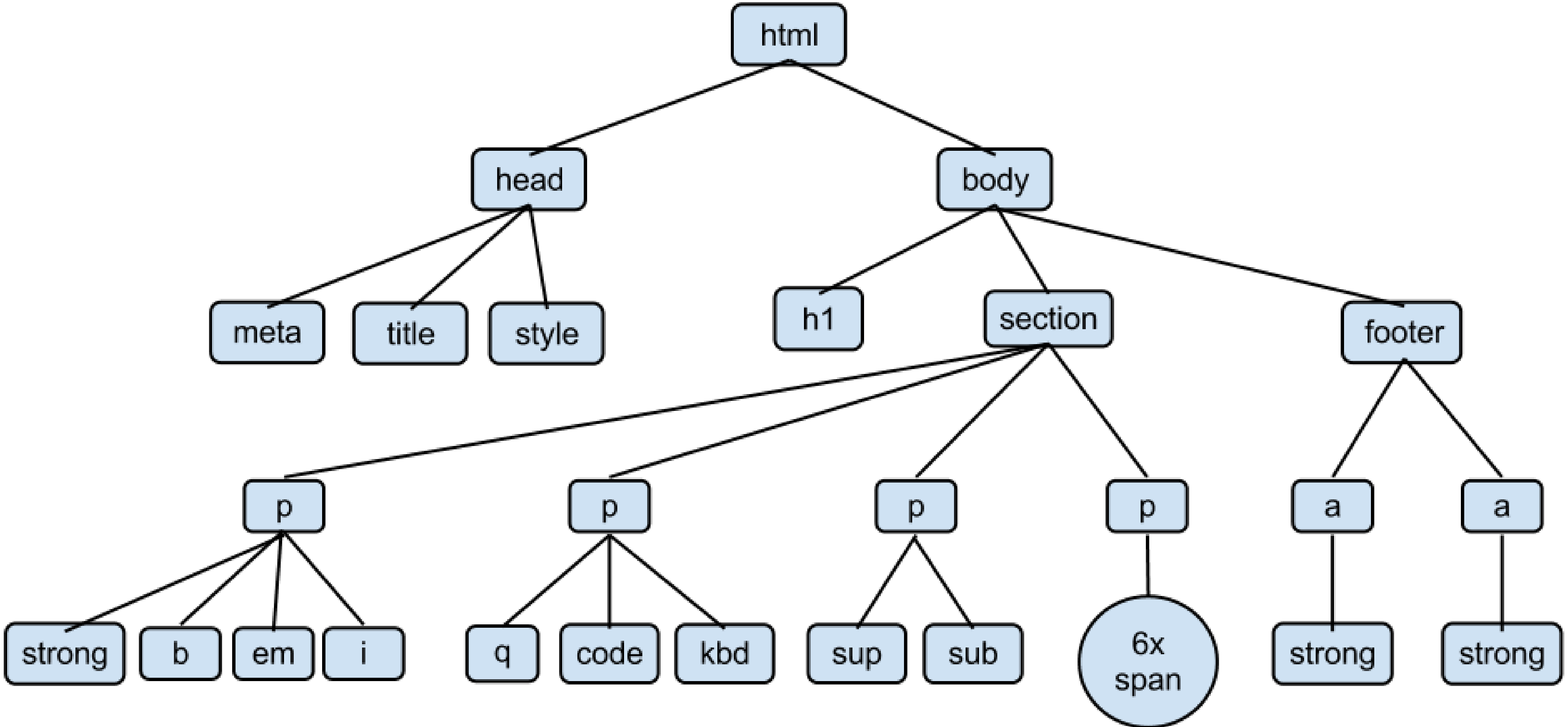
```
<!DOCTYPE html>
<html>
     <head>
        <title>Page title</title>
    </head>
     <body>
         <p>Hello world!</p>
     </body>
</html>
```

HTML: Basic structure

- The **head** contains metadata; data about the website. Things like the document title, styles, scripts.
- Examples:
  - <title>: The title of the document, which is required in all HTML/XHTML documents and is shown in the browser's title bar or on the page's tab.
  - <style>: CSS styles that are included directly within the HTML document.
  - <script>: Used to include JavaScript or link to external JavaScript files.
- The **body** element contains the content of an HTML document, such as text, images, links, tables, lists, etc. This is the section that is visible to the user in the web browser.
- Examples:
  - <h1> to <h6>: Header tags that define headings.
  - <p>: Defines a paragraph.
  - <a>: Defines a hyperlink.
  - <img>: Embeds an image.

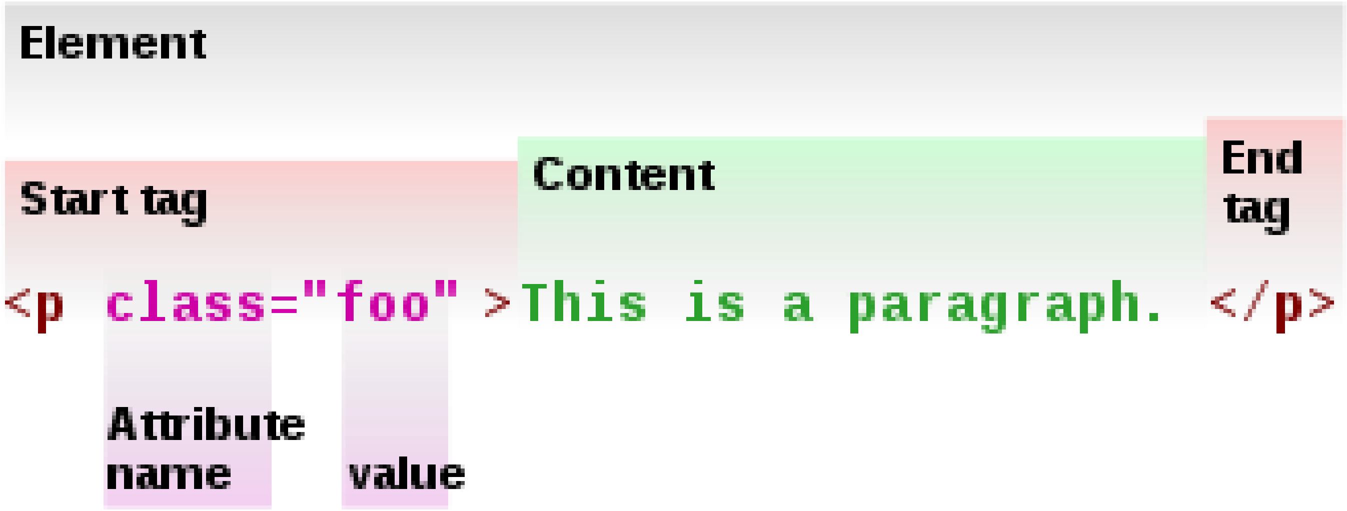
HTML as a tree

- Each branch of the tree is called an element



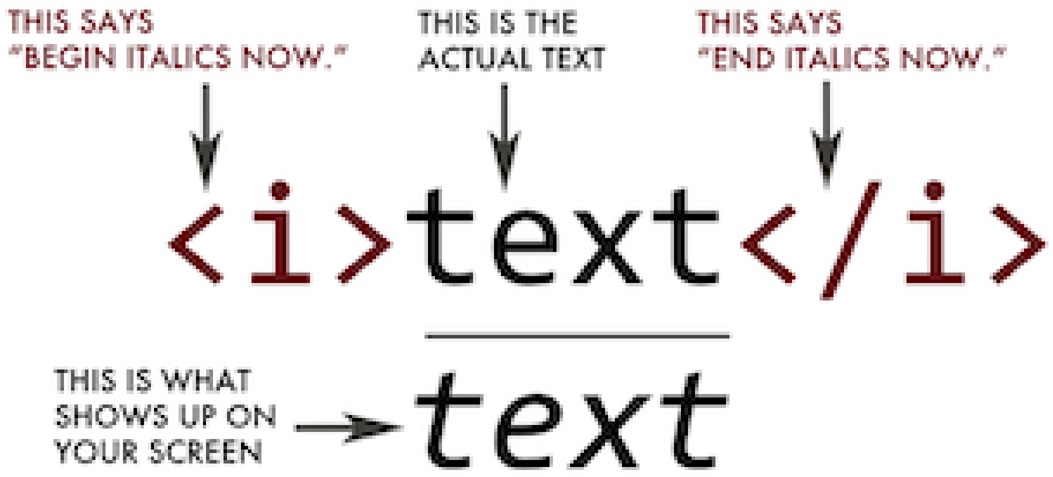
Three general components of HTML elements

- Tags (starting and ending the element)
- Attributes (giving information about the element)
- Text, or Content (the text inside the element)



```
<p class="foo" >This is a paragraph. </p>
```

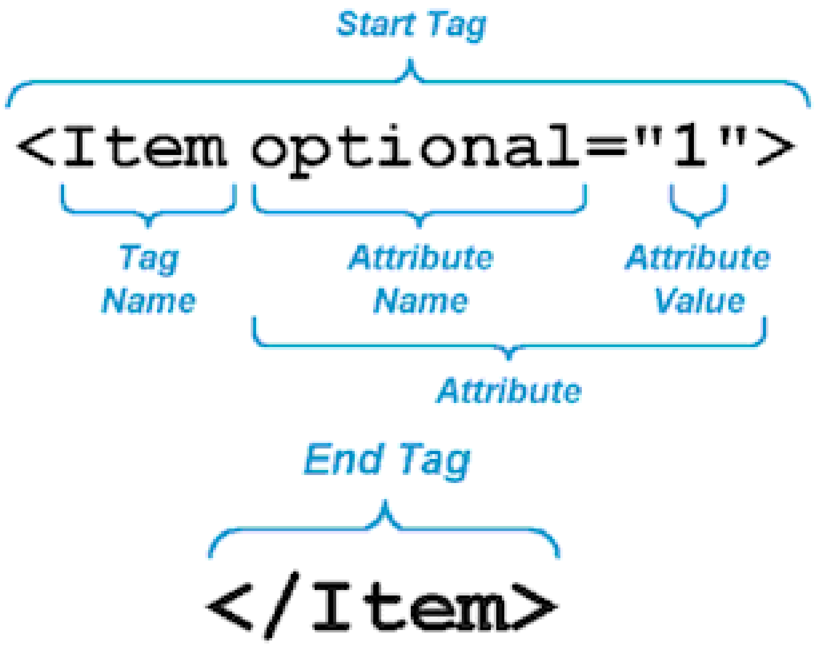
HTML tags


THIS SAYS "BEGIN ITALICS NOW."
THIS IS THE ACTUAL TEXT
THIS SAYS "END ITALICS NOW."
<i>text</i>
THIS IS WHAT SHOWS UP ON YOUR SCREEN
text

Tag	Meaning
<head>	page header (metadata, etc
<body>	holds all of the content
<p>	regular text (paragraph)
<h1>,<h2>,<h3>	header text, levels 1, 2, 3
<ol>,<ul>,<li>	ordered list, unordered list, list item
<a href="page.html">	link to "page.html"
<table>,<tr>,<td>	table, table row, table item
<div>,<span>	general containers (can contain CSS, JavaScript, etc.)

HTML attributes

- HTML elements can have attributes.
- Attributes provide additional information about an element.
- Attributes are always specified in the start tag.
- Attributes come in name/value pairs like: name="value"



Finding data in HTML

- Sometimes we can find the data we want just by using HTML tags or attributes (e.g., all the <a> tags)
- More often, this isn't enough: There might be 1000 <a> tags on a page. But maybe we want only the <a> tags inside of a <p> tag.
- This is where CSS comes in.

CSS (Cascading Style Sheet)

- CSS defines how HTML elements are to be displayed.
- HTML came first. But it was only meant to define content, not format it.
- CSS was created to display content on a webpage. Now, one can change the look of an entire website just by changing one file.
- Most web designers litter the HTML markup with tons of classes and ids to provide "hooks" for their CSS.
- You can piggyback on these "hooks" to jump to the parts of the HTML markup that contain the data you need.

CSS Selectors & Declarations

- Selector: a
- Property: background-color
- Value: blue

Type	HTML	CSS Selector
Element	<a>	a p a
Class	<a class="blue">	.blue a.blue
ID	<a id="blue">	#blue a#blue

Web scraping practice

- Two exercises today
- Scraping and organizing hyperlinks and text: World Bank Development Impact JMP Blogs
- Scraping data files from a database: country administrative boundaries