Web crawling:

* wget localhost:8080/index.html
  + only fetch the HTML file without any associated resources like CSS/JS/img
* wget -p localhost:8080/index.html
  + download a web page along with all its necessary assets like CSS/JS/img
* wget -r -l 1 localhost:8080/index.html
  + follow links on a page and download those resources (including CSS/JS files if they are directly linked from that page ) too – but only 1 level deep
* wget -m localhost:8080/index.html
  + create local copy of the entire website
  + download all files necessary to properly display the website offline
* wget -i inputfile
  + download all URLs in the input file
* wget -i inputfile --force-html
  + treat the input file as html and parse it and wget the links it finds
* wget --spider -r -l 3 <http://example.com>
  + not downloading pages, but mainly used to check for broken links/issues, 3 levels deep recursively

Beautiful Soup:

from bs4 import BeautifulSoup

file = "cattax/index.html"

soup = BeautifulSoup(open(file, 'r'), 'html.parser')

* soup.get\_text()
  + extracts all visible text
* sou.find\_all()
  + locate all instances of a specific tag
    - strong\_tags = soup.find\_all(‘strong”)
    - for tag in strong\_tags:
      * print(tag.get\_text())
* divs = soup.find\_all(‘div’, class\_=’container’)
  + find all <div> elements with a specific class, you can pass arguments to find\_all(). This works even if a <div> has multiple classes.

if not soup.find('div', class\_='container'):

print("This is a leaf node.")

* The above code identifies leaf node from the website by attemping to find a <div> element with container class. If not found, arrives at leaf node.

**When to Scrape**

The text concludes with important ethical and legal considerations regarding web scraping:

* **Respect robots.txt**: This file on websites specifies which parts should not be accessed programmatically.
* **Prefer APIs over Scraping**: If the website offers an API, it’s better to use that for data extraction.
* **Be Cautious with Authenticated Content**: Scraping data that requires a login can lead to privacy and legal issues.
* **Do Not Republish Without Permission**: Accessing data doesn’t automatically grant the rights to republish it.

**Robots.txt file**

is a text file created by the designer to prevent the search engines and bots to crawl up their sites. It contains the list of allowed and disallowed sites and whenever a bot wants to access the website, it checks the robots.txt file and accesses only those sites that are allowed. It doesn’t show up the disallowed sites in search results.

Base URL/robots.txt