

Web scraping with urllib

Import basic functions	<pre>from urllib.request import urlopen, urlretrieve, Request</pre>
Request webpage 'example.com'	<pre>raw_request = Request('https://example.com')</pre>
Add headers (I)	<pre>raw_request.add_header('User-Agent', 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:78.0) Gecko/20100101 Firefox/78.0')</pre>
Add headers (II)	<pre>raw_request.add_header('Accept', 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,/*;q=0.8')</pre>
Get HTML code as a string	<pre>html = urlopen(raw_request).read().decode("utf-8")</pre>
Download file	<pre>urlretrieve(fileURL, 'file_name_in_destination')</pre>

Python libraries

urllib: URL handling	<code>import urllib</code>
BeautifulSoup: HTML/XML parser	<code>from bs4 import BeautifulSoup</code>
Regular expressions: pattern matching	<code>import re</code>
NetworkX: network analysis	<code>import networkx as nx</code>

Websites of interest

Stanford Network Analysis Project	http://snap.stanford.edu/
Koblenz Network Collection	http://konect.cc/
Network Repository	https://networkrepository.com/

Graph dataset formats

GML (.gml)	<i>Custom structure</i>	<code>G = nx.read_gml(path)</code>
Pajek (.net)	<i>Custom structure</i>	<code>G = nx.read_pajek(path)</code>
JSON (.json)	<i>Custom structure</i>	<code>G = nx.read_json(path)</code>
Plain text	<code>node1 node2 weight</code>	Manual
Multilayer v1 (.)	<code>layer node1 node2 weight</code>	Manual
Multilayer v2 (.)	<code>layer1 layer2 node1 node2 weight</code>	Manual

Those formats without a specific function are usually easy to parse manually.

Beautiful Soup HTML parsing

String slicing tricks

Split by character	<code>'a_string'.split('_')</code>
Join with a character	<code>'_'.join(['a', 'string'])</code>
Capitalize	<code>foo.capitalize()</code>
Capitalize, lower, upper	<code>foo.capitalize(), foo.lower(), foo.upper()</code>
Find, count	<code>foo.find('e'), foo.count('e')</code>
Replace	<code>'a_string.replace('string', 'banana')</code>

Regular Expressions

