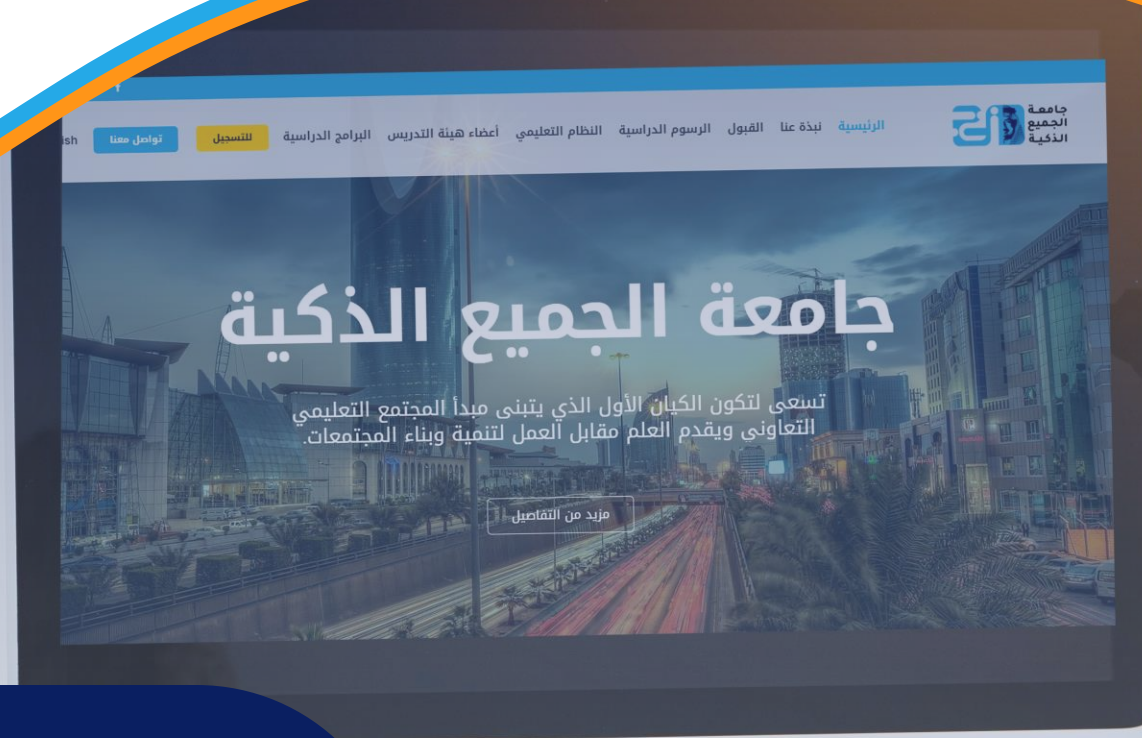




جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

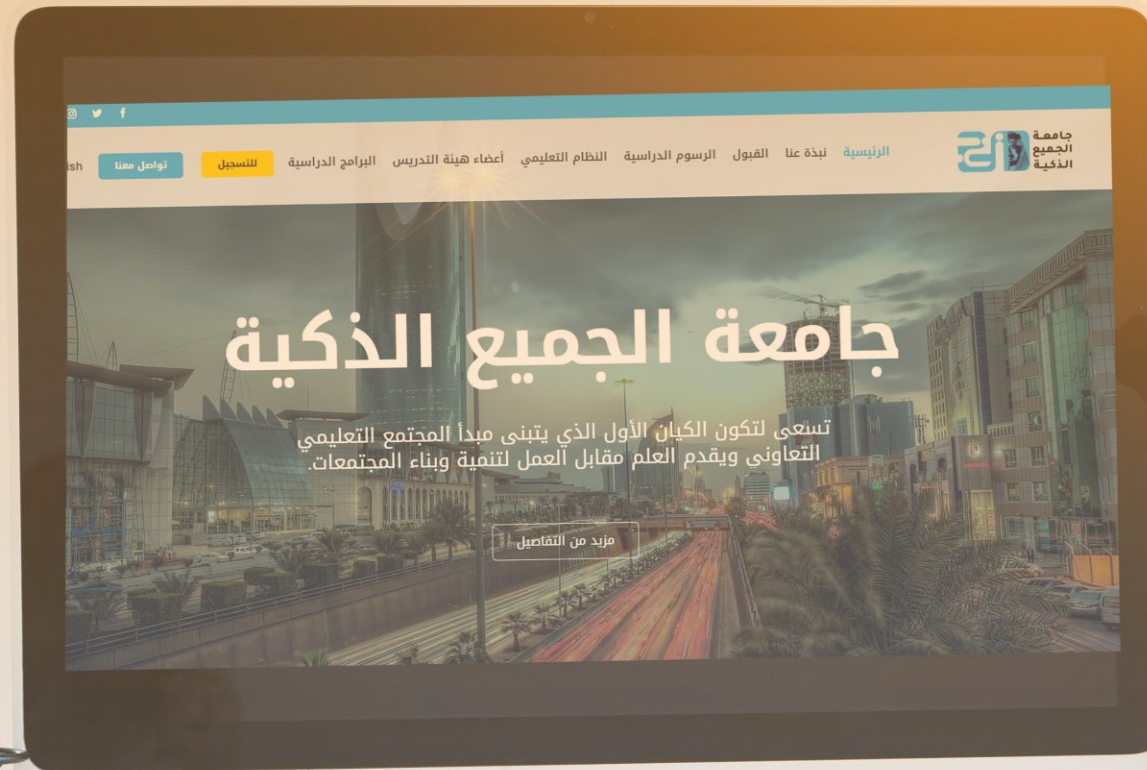


Prepared by
Dr. Hamada I. AbdulWakel
Lecture 3

CYB615
Network Programming



جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY



Module2: Network Programming using Python – Part II

- Application Program Interface (API).
- BeautifulSoup | Web Scraping.
- Socket Programming in Python.



- Geocoding is the process of converting addresses (like "1600 Amphitheatre Parkway, Mountain View, CA") into geographic coordinates (like latitude 37.423021 and longitude -122.083739), which you can use to place markers on a map or position the map.
- Reverse geocoding is the process of converting geographic coordinates into a human-readable address.
- Google map API simulation
- Use this URL.



```
import urllib.error, urllib.parse, urllib.request
import json
import ssl
```

```
api_key = False
```

```
if api_key is False:
```

```
    api_key = 42
```

```
    serviceurl = 'http://py4e-data.dr-chuck.net/json?'
```

```
else:
```

```
    serviceurl = 'https://maps.googleapis.com/maps/api/geocode/json?'
```



while True:

```
    address = input("Enter location: ")
```

```
    if len(address) < 1:
```

```
        break
```

```
    parms = dict()
```

```
    parms['address'] = address
```

```
    if api_key is not False: parms['key'] = api_key
```

```
    url = serviceurl + urllib.parse.urlencode(parms)
```

```
    print('Retrieving', url)
```



while True:

...

```
urllib_handler = urllib.request.urlopen(url, context=ctx)
```

```
data = urllib_handler.read().decode()
```

```
print('Retrieved', len(data), 'characters')
```

```
try:
```

```
    js = json.loads(data)
```

```
except:
```

```
    js = None
```



while True:

...

if 'status' not in js or js['status'] != 'OK':

 print('Fail to retrieve data'); print(data); continue

print(json.dumps(js, indent=4))

lat = js['results'][0]['geometry']['location']['lat']

lng = js['results'][0]['geometry']['location']['lng']

location = js['results'][0]['formatted_address']

print('lat', lat, 'lng', lng); print(location)



- When a program pretends to be a web browser and retrieves web pages, extracts information, and then looks at more pages.
- This is what the search engine do.

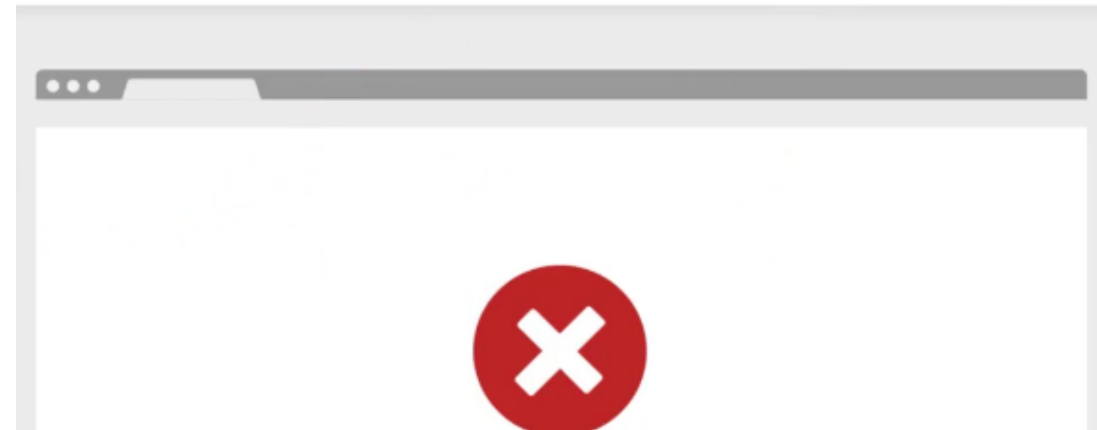


- Monitor a site for new information.
- Make a database for a search engine.
- Get your data from a system that has no export feature.
- Pull social data.



- There is some controversy about web scraping.
- Do not republishing copyrighted material.
- Do not violate terms of service.

Sorry, you have been blocked
You are unable to access youm7.com



- BeautifulSoup is a Python library for pulling data out of HTML and XML files.
- One of the important pieces of HTML document are tags, which may contain other tags/strings (tag's children). BeautifulSoup provides different ways to navigate and iterate over's tag's children.
- BeautifulSoup is the object created when we try to scrape a web resource. So, it is the complete document which we are trying to scrape.



Lecture 3

Beautiful Soup Module

Cont'd

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title></title>
```

```
  </head>
```

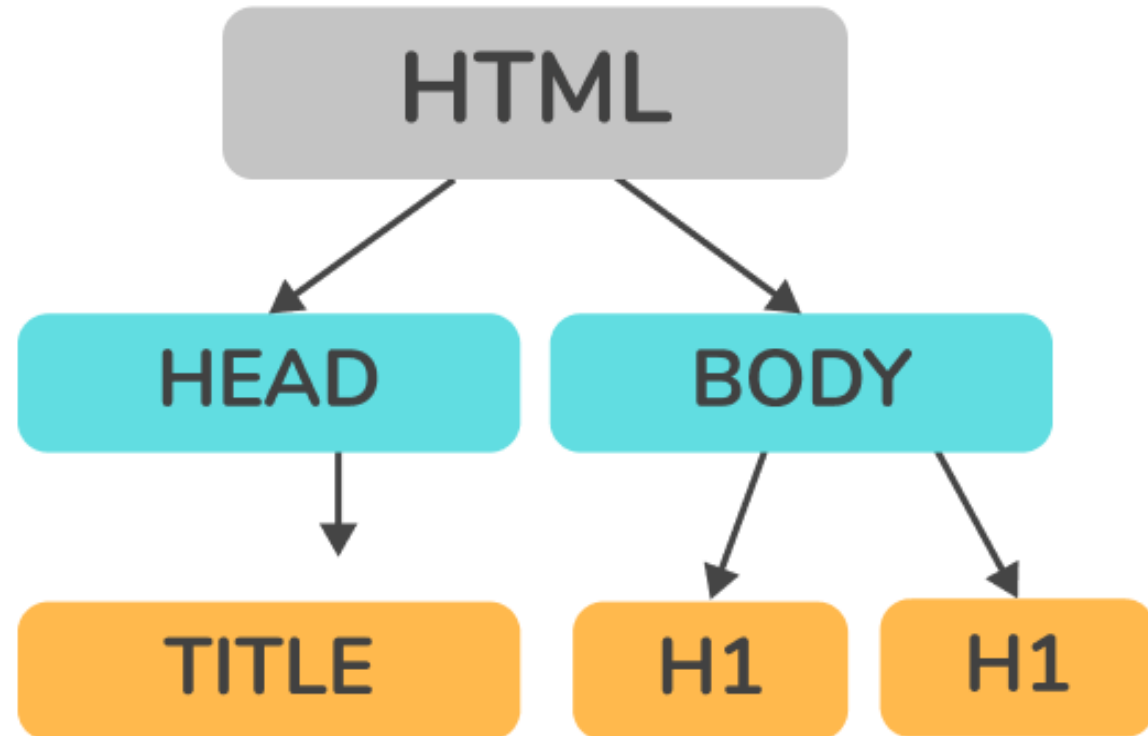
```
  <body>
```

```
    <h1></h1>
```

```
    <h1></h1>
```

```
  </body>
```

```
</html>
```



- Installation:
[PyPi](#)
- BeautifulSoup supports different HTML parser such as:
 1. Html.parser: included in Python's standard library
 2. lxml `pip install lxml`
 3. Html5lib `pip install html5lib`



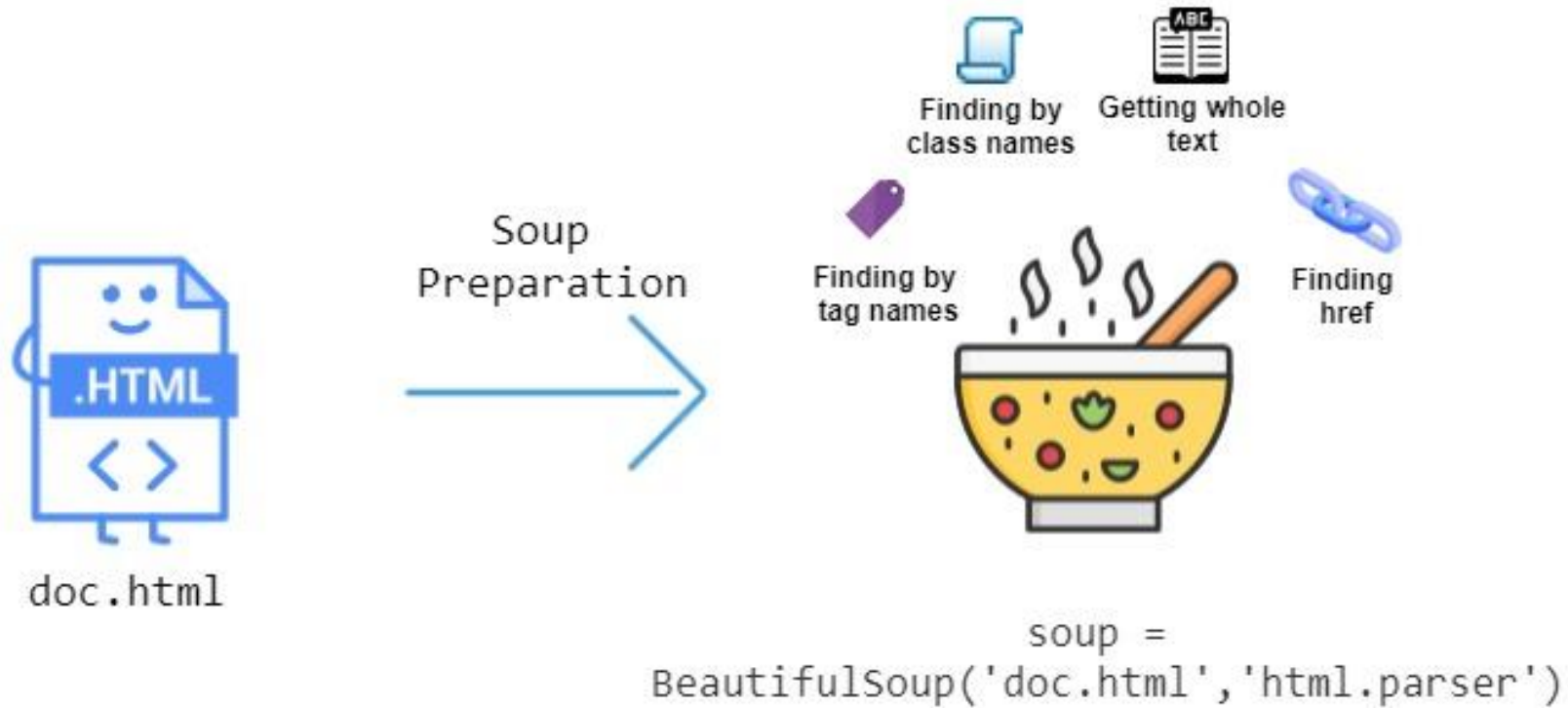
- **Inspect your data source:** to understand the web site structure to extract the information that's relevant for you.
Mac: Cmd+Alt+I
Windows/Linux: Ctrl+Shift+I
- **Scrape HTML content from a page:** to get the site's HTML code into your Python script so that you can interact with it.
- **Parse HTML code with BeautifulSoup:** to make it more accessible and pick out the data you want.



Lecture 3

Steps for Web Scraping

Cont'd



- Parse HTML code with BeautifulSoup:
 - Find element by ID:
`soup.find(id="idName")`
 - Find elements by HTML class name:
`soup.find_all("div", {"class":"cssClassName"})`
 - Extract text from HTML element:
`print(element.text)`
 - Find elements by class name and text content:
`soup.find_all("h2", string="Python")`



Lecture 3

Web Scraping Example

```
import urllib.request
from bs4 import BeautifulSoup
url = 'http://www.dr-chuck.com/page1.htm'
html = urllib.request.urlopen(url).read()
soup = BeautifulSoup(html, 'html.parser')
tags = soup('a') #parsing anchor tags
for tag in tags:
    print(tag.get('href'))
```



Lecture 3

Searching Job Example Using Wuzzuf

```
from itertools import zip_longest
import requests
from bs4 import BeautifulSoup
import csv
```

```
result = requests.get('https://wuzzuf.net/search/jobs/?q=python&a=hp')
source = result.content
soup = BeautifulSoup(source, 'html.parser')
job_titles = soup.find_all("h2", {"class": "css-m604qf"})
```



```
company_names = soup.find_all("a", {"class":"css-17s97q8"})
locations = soup.find_all("span", {"class":"css-5wys0k"})
skills = soup.find_all("div", {"class":"css-y4udm8"})
print(len(job_titles))
jobs = []; comapnies = []; locs = []; skill = []
for i in range(len(job_titles)):
    jobs.append(job_titles[i].text)
    comapnies.append(company_names[i].text)
    locs.append(locations[i].text)
    skill.append(skills[i].text)
```



```
fileLists = [jobs, comapnies, locs, skill]
exported = zip_longest(*fileLists)
with open("/path/to/create/jobsResults.csv", "w") as f:
    wr = csv.writer(f)
    wr.writerow(["job title", "company name", "location", "skills"])
    wr.writerows(exported)
```



- Sockets are the endpoints of a bidirectional communications channel.
- Sockets may communicate within a process, between processes on the same machine, or between processes on different continents.
- Sockets are a combinations of an IP address and a port.



Lecture 3

What are Sockets?

Cont'd

Protocol	Port Number	Python library	Function
HTTP	80	urllib	Web pages
FTP	20	ftplib	File transfer
SMTP	25	smtplib	Sending email
POP3	110	poplib	Fetching email
Gopher	70	gopherlib	Document transfer



- To create a socket, you must use the `socket.socket()` function available in socket module, which has the general syntax:

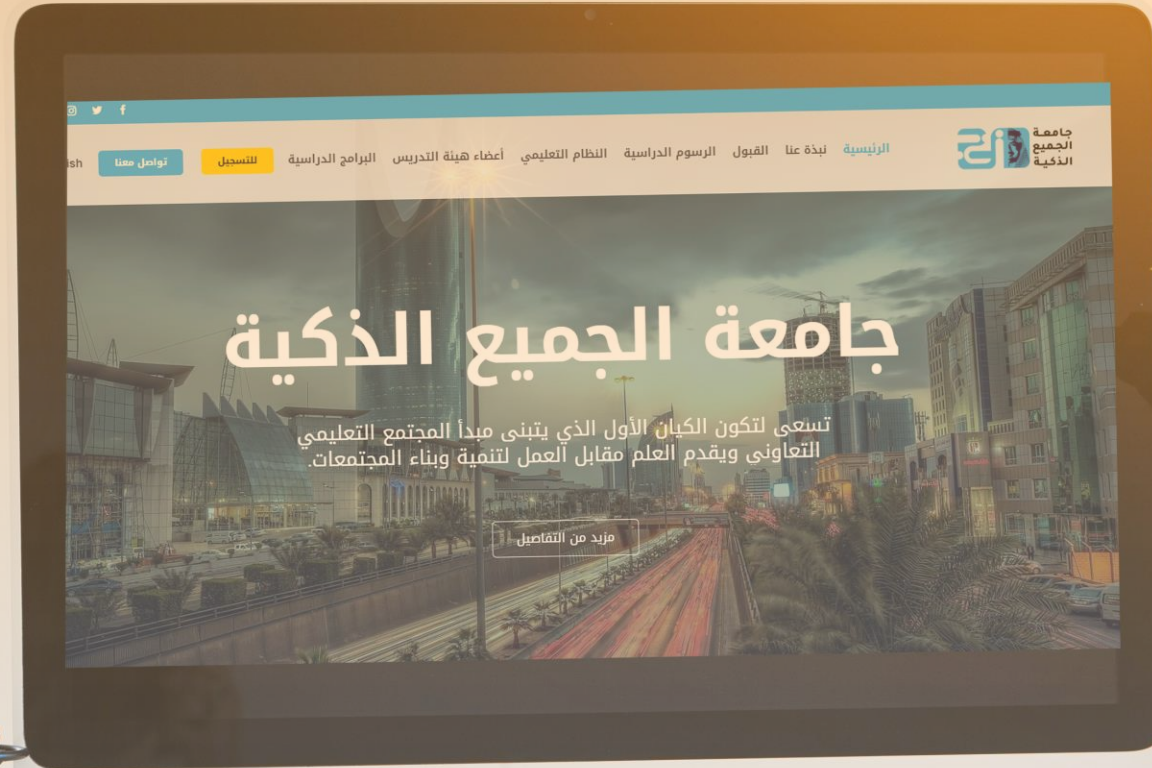
```
s = socket.socket (socket_family, socket_type, protocol=0)
```

- The description of the parameters:
 - `socket_family`: This is either `AF_UNIX` or `AF_INET`
 - `socket_type`: This is either `SOCK_STREAM` or `SOCK_DGRAM`.
 - `protocol`: This is usually left out, defaulting to 0.



Method	Description
<code>socket.socket()</code>	Used to create sockets one for the server and another for the client.
<code>socket.accept()</code>	Used to accept a connection
<code>socket.bind()</code>	Used to bind to the address that is specified as a parameter.
<code>socket.close()</code>	Used to mark the socket as closed.
<code>socket.connect()</code>	Used to connect to a remote address that is specified as a parameter.
<code>socket.listen()</code>	Used to enable the server to accept a connection.





جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

THANK YOU