Other Python Libraries: Pandas, Requests, Beautiful Soup

About me

Trayan Iliev

– CEO of **IPT – Intellectual Products & Technologies**
http://www.iproduct.org

– Oracle® certified programmer 15+ Y

– end-to-end reactive fullstack apps with Java, ES6+, TypeScript, Angular, React and Vue.js

– 12+ years IT trainer: Spring, Java EE, Node.js, Express, GraphQL, SOA, REST, DDD & Reactive Microservices

– Voxxed Days, jPrime, Java2Days, jProfessionals, BGOUG, BGJUG, DEV.BG speaker

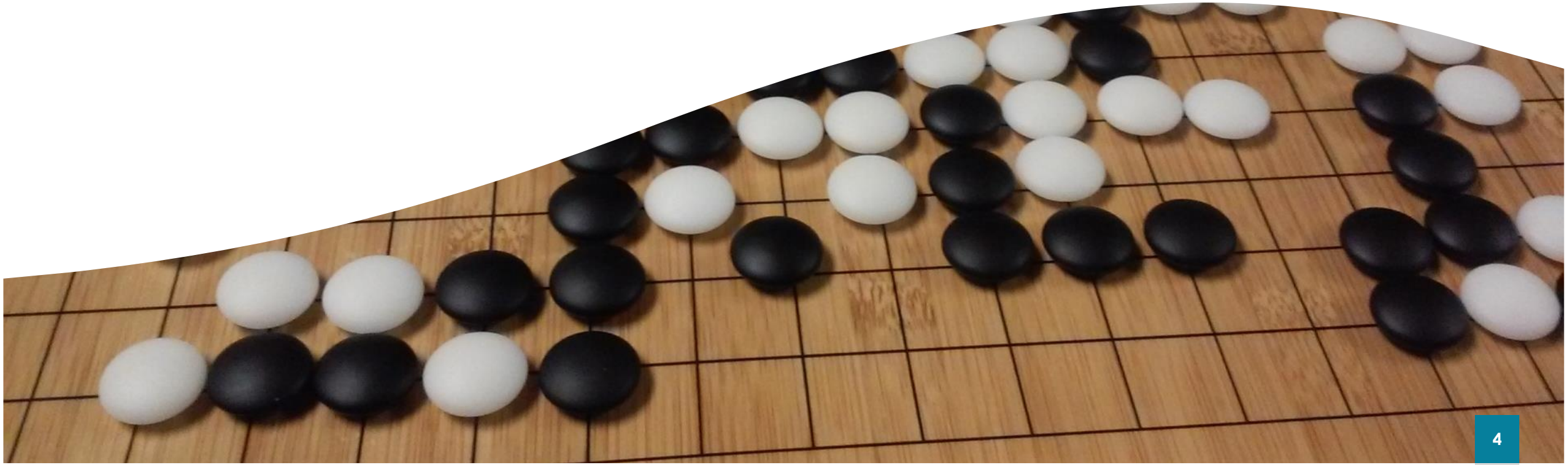
– Organizer RoboLearn hackathons and IoT enthusiast

Where to Find The Code and Materials?

https://github.com/iproduct/intro-python

Pandas

Working with dataframes

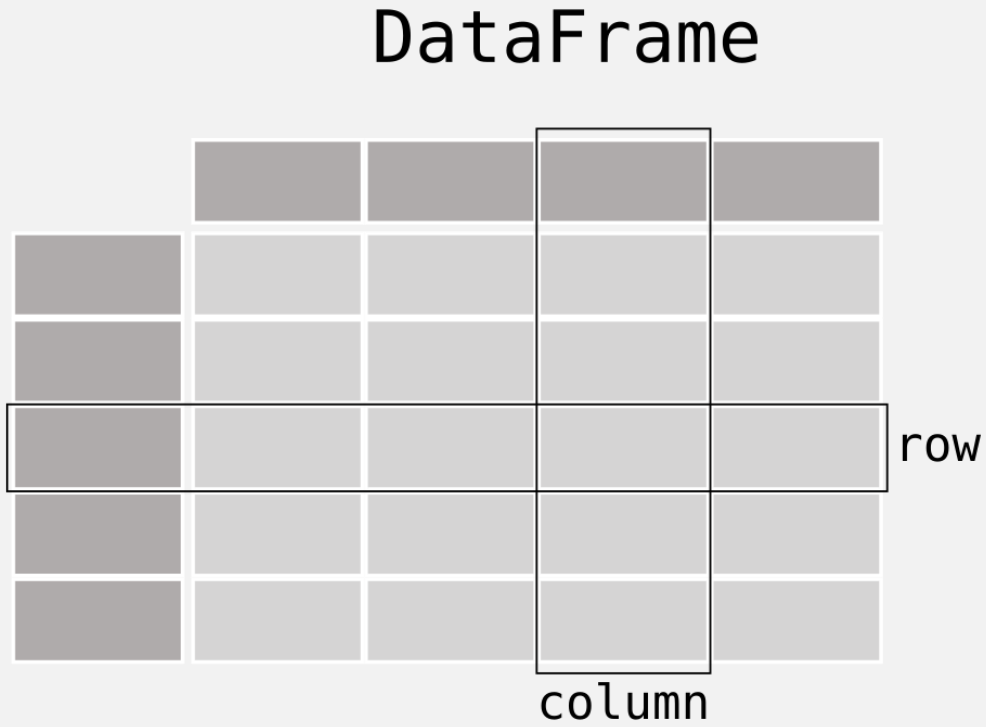


Pandas [https://www.w3schools.com/python/pandas/default.asp]

- Pandas is a Python library used for working with data sets.
- Fast, flexible, and expressive data structures designed to make working with 'relationa' or 'labeled' data both easy and intuitive. It aims to be the fundamental high-level building block for real world data analysis.
- It has functions for cleaning, exploring, analyzing, and manipulating data.
- "Pandas" name has a reference to "Panel Data", and "Python Data Analysis" - created by Wes McKinney in 2008.
- Pandas allows us to analyze big data and make conclusions based on statistical theories – allows us to answer questions like:
  - is there a correlation between two columns?
  - what is average / max /min value?
  - clean datasets
  - etc.

Pandas DataFrame Example

```
df = pd.DataFrame(
    {
        "Name": [
            "Braund, Mr. Owen Harris",
            "Allen, Mr. William Henry",
            "Bonnell, Miss. Elizabeth",
        ],
        "Age": [22, 35, 58],
        "Sex": ["male", "male", "female"],
    }
)
```

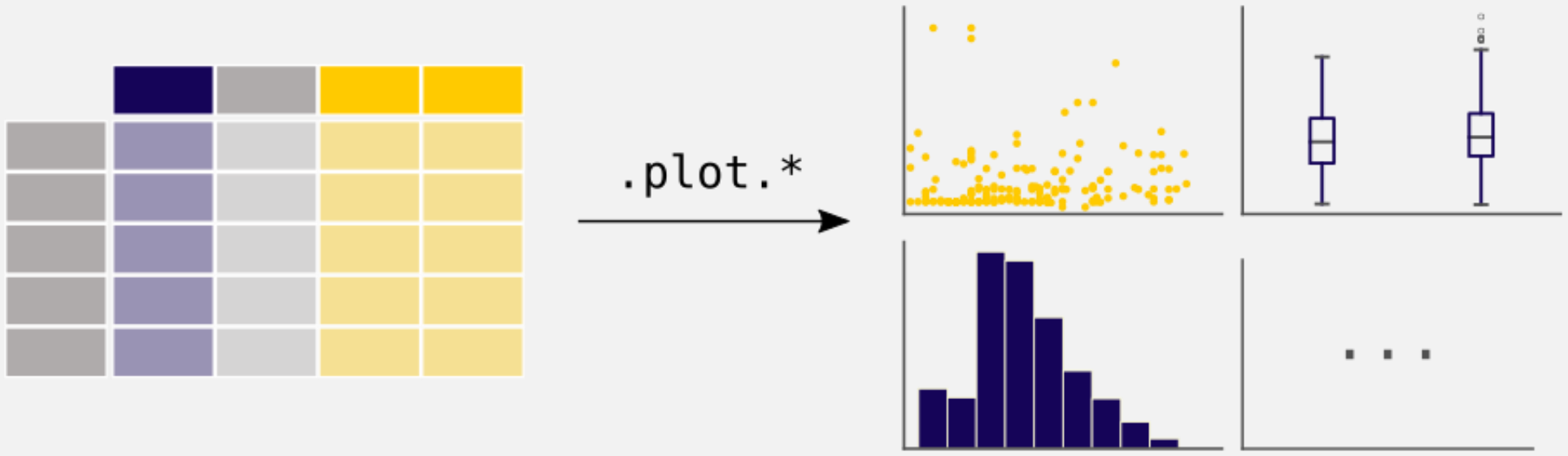


Pandas official documentation: https://pandas.pydata.org/docs/index.html

Pandas Tutorial in W3Schools: https://www.w3schools.com/python/pandas/default.asp

Pandas Exercises and problems: https://www.w3resource.com/python-exercises/pandas/index.php

Pandas plot with Mathplotlib



https://matplotlib.org/stable/
https://www.w3schools.com/python/matplotlib_intro.asp

Beautiful Soup

Beautiful Soup

[https://www.crummy.com/software/BeautifulSoup/bs4/doc/]

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.

Beautiful Soup 4 Basic Examples I

[https://www.crummy.com/software/BeautifulSoup/bs4/doc/]

```
from bs4 import BeautifulSoup

html_doc = """<html><head><title>The Dormouse's story</title></head>
<body>
<p class="title"><b>The Dormouse's story</b></p>

<p class="story">Once upon a time there were three little sisters; and their names were
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>;
and they lived at the bottom of a well.</p>

<p class="story">...</p>
"""

soup = BeautifulSoup(html_doc, 'html.parser')

print(soup.prettify())
```

Beautiful Soup 4 Basic Examples II

```
soup.title
# <title>The Dormouse's story</title>

soup.title.name
# u'title'

soup.title.string
# u'The Dormouse's story'

soup.title.parent.name
# u'head'

soup.p
# <p class="title"><b>The Dormouse's story</b></p>

soup.p['class']
# u'title'

soup.a
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>
```

Beautiful Soup 4 Basic Examples III

```
soup.find_all('a')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]

soup.find(id="link3")
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>

for link in soup.find_all('a'):
    print(link.get('href'))
# http://example.com/elsie
# http://example.com/lacie
# http://example.com/tillie

print(soup.get_text())
# The Dormouse's story
#
# The Dormouse's story
# ...
```

Requests for Humans

Requests – Features: [https://docs.python-requests.org/en/latest/]

- Keep-Alive & Connection Pooling
- International Domains and URLs
- Sessions with Cookie Persistence
- Browser-style SSL Verification
- Automatic Content Decoding
- Basic/Digest Authentication
- Elegant Key/Value Cookies
- Automatic Decompression
- Unicode Response Bodies
- HTTP(S) Proxy Support
- Multipart File Uploads
- Streaming Downloads
- Connection Timeouts
- Chunked Requests
- .netrc Support

Requests [https://docs.python-requests.org/en/latest/]

```
>>> r = requests.get('https://api.github.com/user', auth=('user', 'pass'))

>>> r.status_code

200

>>> r.headers['content-type']

'application/json; charset=utf8'

>>> r.encoding

'utf-8'

>>> r.text

'{"type":"User"...'

>>> r.json()

{'private_gists': 419, 'total_private_repos': 77, ...}
```

Requests Basic Example

```
import requests

if __name__ == '__main__':
    r = requests.get('https://api.github.com/events')
    # for event in r.json():
    #     print(event)

    r = requests.get('https://docs.python-requests.org/en/latest/user/quickstart/')
    print(r.text[:1000])
```

Requests + Beautiful Soup 4 Example

```
import requests
from bs4 import BeautifulSoup

if __name__ == '__main__':
    r = requests.get('https://docs.python-requests.org/en/latest/user/quickstart/')
    # print(r.text[:1000])
    soup = BeautifulSoup(r.text, 'html.parser')
    # print(soup.prettify())
    print("soup.title = ", soup.title)
    print("soup.title.name = ", soup.title.name)
    print("soup.title.string = ", soup.title.string)
    print("soup.title.parent.name = ", soup.title.parent.name)
    print("soup.p = ", soup.p)
    print("soup.a = ", soup.a)
    for link in soup.find_all('a'):
        print(link, "->", link.get('href'))
    print("\nTEXT CONTENT:")
    lines = [line.strip() for line in soup.body.get_text().split("\n") if len(line.strip()) > 0]
    for l in lines:
        print(l)
```

Problem: Fetch http://python.org and print all widget texts

Write a program using Resquests and Beautiful Soup 4 (BS4) libraries, with following functionality:

1. Using GET method of HTTP protocol, fetch the http://python.org web page document.
2. Parse the web page using BS4 and print the <title> tag content.
3. Print the text content of all menus in the <nav> tag of the page.
4. Print the alt attribute of the image in the <header> tag section of the page.
5. Print the text content of all divisions of class "small-widget" or "medium-widget" in the page

Thank's for Your Attention!

Trayan Iliev

IPT – Intellectual Products & Technologies

http://iproduct.org/

https://github.com/iproduct

https://twitter.com/trayaniliev

https://www.facebook.com/IPT.EACAD