The Python Mega Course: Build 10 Real World Applications

Tuesday, August 17, 2021 1:17 PM

https://www.udemy.com/course/the-python-mega-course/

Question 2: **Incorrect**

Check ALL true statements.

- [ ] Tuples can contain only numbers.
- [ ] Lists can contain any type of object. (Correct)
- [x] Dictionaries represent pairs of keys and values. (Correct)
- [x] List and tuple indexing start from zero. (Correct)

Explanation

The only wrong statement is the first one. That should not be checked. Tuples can contain any type of object. They are not homogenous.

Question 14: **Incorrect**

What does the code below output?

```
1  def eur_to_usd(euros, rate=0.8):
2      return euros * rate
3  print(eur_to_usd(10))
```

- ○ `TypeError: eur_to_usd() missing 1 argument: 'rate'`
- ◉ 8 (Incorrect)
- ○ 8.0 (Correct)

Explanation

8.0 is returned because `10 * 0.8` is actually the float 8.0. Multiplying an integer with a float gives a float.

Question 18: **Incorrect**

What would be the output of the following code if the user entered 10 as input?

```
weight = input("How many kg?")
price = weight * 2.5
print(price)
```

- ○ 12.5
- ◉ 25 (Incorrect)
- ○ TypeError: can't multiply sequence by non-int of type 'float' (Correct)

Explanation

Even if the user enters a number like 10, the input function converts that number into a string. For the code to behave correctly, the first line should be changed to `weight = float(input("How many kg?"))` so that the string `"10"` is converted to float `10.0`.

- You can **format strings** with (works both on Python 2 and 3):

```
name = "Sim"
experience_years = 1.5
print("Hi %s, you have %s years of experience." % (name, experienc
```

Output: `Hi Sim, you have 1.5 years of experience.`

- You can also **format strings** with:

```
name = "Sim"
experience_years = 1.5
print("Hi {}, you have {} years of experience".format(name, experi
```

Output: `Hi Sim, you have 1.5 years of experience.`

textpro.py

```
def sentence_maker(phrase):
    interrogatives = ("how", "what", "why")
    capitalized = phrase.capitalize()
    if phrase.startswith(interrogatives):
        return "{}?".format(capitalized)
    else:
        return "{}.".format(capitalized)

results = []
while True:
    user_input = input("Say something: ")
    if user_input == "\end":
        break
    else:
        results.append(sentence_maker(user_input))

print(results)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
Type "help", "copyright", "credits" or "license" for more inf
>>> def maker(phrase):
...     capitalized = phrase.capitalize()
ardit:app0 m$ python3 textpro.py
ardit:app0 m$ python3 textpro.py
How are you?
ardit:app0 m$ python3 textpro.py
Say something: how are you
Say something: weather is good
Say something: \end
['how are you', 'weather is good']
ardit:app0 m$ python3 textpro.py
```

This joint metho

- An ***args** parameter allows the function to be called with an arbitrary number of non-keyword arguments:

```
def find_max(*args):
    return max(args)
print(find_max(3, 99, 1001, 2, 8))
```

Output: `1001`

- An ****kwargs** parameter allows the function to be called with an arbitrary number of keyword arguments:

```
def find_winner(**kwargs):
    return max(kwargs, key = kwargs.get)

print(find_winner(Andy = 17, Marry = 19, Sim = 45, Kae = 34))
```

Output: `Sim`

Cheatsheett (File Processing)

In this section, you learned that:

- You can **read** an existing file with Python:

```
with open("file.txt") as file:
    content = file.read()
```

- You can **create** a new file with Python and **write** some text on it:

```
with open("file.txt", "w") as file:
    content = file.write("Sample text")
```

- You can **append** text to an existing file without overwriting it:

```
with open("file.txt", "a") as file:
    content = file.write("More sample text")
```

- You can both **append and read** a file with:

```
with open("file.txt", "a+") as file:
    content = file.write("Even more sample text")
    file.seek(0)
    content = file.read()
```

==========

Section 13: Application 1: Building an English Thesaurus

App1.py

```
import json
from difflib import get_close_matches

data = json.load(open("data.json"))

def translate(w):
    w = w.lower()
    if w in data:
        return data[w]
    elif len(get_close_matches(w, data.keys())) > 0:
        yn = input("Did you mean %s instead? Enter Y if yes, or N if no: " % get_close_matches(w, data.keys())[0])
        if yn == "Y":
            return data[get_close_matches(w, data.keys())[0]]
        elif yn == "N":
            return "The word doesn't exist. Please double check it."
        else:
            return "We didn't understand your entry."
    else:
        return "The word doesn't exist. Please double check it."

word = input("Enter word: ")
output = translate(word)
if type(output) == list:
    for item in output:
        print(item)
else:
    print(output)
```

Currently, when the user inputs a proper noun, such as Delhi or Paris, the program will 1) convert that string into lowercase and 2) it will look for the lowercase version (i.e., delhi or paris) in the dataset. However, the dataset doesn't have delhi or paris. It only has Delhi and Paris. Therefore, no definition is currently returned for proper nouns such as Delhi or Paris.

Please try to fix this issue. You can think of adding another conditional block. You can find the code we currently have attached in this lecture for your convenience.

有些专有名词，首字母大写，如何解决

```
6       if w in data:
7           return data[w]
8       elif w.title() in data: #if user entered "texas" this will che
9           return data[w.title()]
10      elif len(get_close_matches(w, data.keys())) > 0:
11          yn = input("Did you mean %s instead? Enter Y if yes, or N
```

加入line 8, 9

A student discovered another issue with the program. The program cannot return the definition of acronyms such as USA or NATO; therefore, add another conditional to make the program return the definition of such words.

```
6       if w in data:
7           return data[w]
8       elif w.title() in data:
9           return data[w.title()]
10      elif w.upper() in data: #in case user enters words like USA or
11          return data[w.upper()]
12      elif len(get_close_matches(w, data.keys())) > 0:
13          yn = input("Did you mean %s instead? Enter Y if yes, or N
```

加入line 10，11

==========

Section 14: Python and MySQL: Interactive English Dictionary

Source.py

```
1  import mysql.connector
2
3  con = mysql.connector.connect(
4  user = "ardit700_student",
5  password = "ardit700_student",
6  host = "108.167.140.122",
7  database = "ardit700_pm1database"
8  )
9
10 cursor = con.cursor()
11
12 word=input("Enter the word: ")
13
14 query = cursor.execute("SELECT Definition FROM Dictionary WHERE Expression = '%s'" % word)
15 results = cursor.fetchall()
16
17 if results:
18     for result in results:
19         print(result[0])
20 else:
21     print("No word found!")
```

从作者的服务器的数据库里读数据

==========

Section 15: Using Python with CSV, JSON, and Excel Files

Exercise: Loading JSON Files

In the previous lecture, you learned that you can load a CSV file with this code:

```
import pandas
df1 = pandas.read_csv("supermarkets.csv")
```

Try loading the `supermarkets.json` file for this exercise using `read_json` instead of `read_csv`.

The supermarkets.json file can be found inside the supermarkets.zip file attached in the previous lecture.

The code for loading the `supermarkets.json` file in Python with pandas would be this:

```
import pandas
df2 = pandas.read_json("supermarkets.json")
```

The `df2` dataframe should contain this data:

	ID	Address	City	State	Country	Name	Employees
0	1	3666 21st St	San Francisco	CA 94114	USA	Madeira	8
1	2	735 Dolores St	San Francisco	CA 94119	USA	Bready Shop	15
2	3	332 Hill St	San Francisco	California 94114	USA	Super River	25
3	4	3995 23rd St	San Francisco	CA 94114	USA	Ben's Shop	10
4	5	1056 Sanchez St	San Francisco	California	USA	Sanchez	12
5	6	551 Alvarado St	San Francisco	CA 94114	USA	Richvalley	20

Note on Loading Excel Files

In the next lecture, you will learn how to load Excel files in Python with *pandas*. For this, you need *pandas* (which you have already installed) and also two other dependencies that *pandas* needs for opening Excel files. You can install them with *pip*:

`pip3.9 install openpyxl` (needed to load Excel *.xlsx* files)

`pip3.9 install xlrd` (needed to load Excel old *.xls* files)

Note

We are going to use `Nominatim()` in the next video. `Nominatim()` currently has a bug. To fix this problem, whenever you see these lines in the next video:

```
from geopy.geocoders import Nominatim
nom = Nominatim()
```

change them to these

```
from geopy.geocoders import ArcGIS
nom = ArcGIS()
```

The rest of the code remains the same.

其他相关请参考 http://localhost:8888/lab/tree/Python_Mega_Course/sec15_supermarkets/pandas_read.ipynb

==========

Section 16: Numerical and Scientific Computing with Python and Numpy

http://localhost:8888/lab/tree/Python_Mega_Course/sec16_numpy/numpy_learn.ipynb

```
import numpy as np
import cv2
```

```
im_g = cv2.imread("sec16_smallgray.png", 0) #0 means read in gray scale, 1 means read in BGR
im_g
```

```
array([[187, 158, 104, 121, 143],
       [198, 125, 255, 255, 147],
       [209, 134, 255,  97, 182]], dtype=uint8)
```

```
cv2.imwrite("new_smallgray.png", im_g)
```

```
True
```

```
imh = np.hstack((im_g, im_g))
print(imh)
```

```
[[187 158 104 121 143 187 158 104 121 143]
 [198 125 255 255 147 198 125 255 255 147]
 [209 134 255  97 182 209 134 255  97 182]]
```

```
imv = np.vstack((im_g, im_g))
print(imv)
```

```
[[187 158 104 121 143]
 [198 125 255 255 147]
 [209 134 255  97 182]
 [187 158 104 121 143]
 [198 125 255 255 147]
 [209 134 255  97 182]]
```

```
lst_h = np.hsplit(imv, 5)
lst_h
```

```
lst_v = np.vsplit(imv, 3)
lst_v
```

```
[array([[187, 158, 104, 121, 143],
        [198, 125, 255, 255, 147]], dtype=uint8),
 array([[209, 134, 255,  97, 182],
        [187, 158, 104, 121, 143]], dtype=uint8),
 array([[198, 125, 255, 255, 147],
        [209, 134, 255,  97, 182]], dtype=uint8)]
```

==========

Section 17: Application 2: Making a Web Map of Volcanoes and Poplulation with Python

Note

In the next lecture, I use this in the code:

```
tiles = "Mapbox Bright"
```

Please use this instead:

```
tiles = "Stamen Terrain"
```

Mapbox Bright and Stamen Terrain are both types of basemaps, but Mapbox Bright doesn't work anymore. Stamen Terrain works great, and you will see it creates a beautiful relief map.

Adding HTML on Popups

Note that if you want to have stylized text (bold, different fonts, etc) in the popup window you can use HTML. Here's an example:

```
import folium
import pandas

data = pandas.read_csv("Volcanoes.txt")
lat = list(data["LAT"])
lon = list(data["LON"])
elev = list(data["ELEV"])

html = """<h4>Volcano information:</h4>
Height: %s m
"""

map = folium.Map(location=[38.58, -99.09], zoom_start=5, tiles="Mapbox Bright")
fg = folium.FeatureGroup(name = "My Map")

for lt, ln, el in zip(lat, lon, elev):
    iframe = folium.IFrame(html=html % str(el), width=200, height=100)
    fg.add_child(folium.Marker(location=[lt, ln], popup=folium.Popup(iframe), icon = folium.Icon(color = "green")))


map.add_child(fg)
map.save("Map_html_popup_simple.html")
```

You can even put links in the popup window. For example, the code below will produce a popup window with the name of the volcano as a link which does a Google search for that particular volcano when clicked:

```
import folium
import pandas

data = pandas.read_csv("Volcanoes.txt")
lat = list(data["LAT"])
lon = list(data["LON"])
elev = list(data["ELEV"])
name = list(data["NAME"])

html = """
Volcano name:<br>
<a href="https://www.google.com/search?q=%%22%s%%22" target="_blank">%s</a><br>
Height: %s m
"""

map = folium.Map(location=[38.58, -99.09], zoom_start=5, tiles="Mapbox Bright")
fg = folium.FeatureGroup(name = "My Map")

for lt, ln, el, name in zip(lat, lon, elev, name):
    iframe = folium.IFrame(html=html % (name, name, el), width=200, height=100)
```

```
    fg.add_child(folium.Marker(location=[lt, ln], popup=folium.Popup(iframe), icon = folium.Icon(color = "green")))

map.add_child(fg)
map.save("Map_html_popup_advanced.html")
```

==========

Section 21: Application 4: Personal Website with Python and Flask

Script1.py

```
# -*- coding: utf-8 -*-
"""
Created on Thu Aug 19 00:51:43 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com

Reference:
"""
#run in cmd: python .\script1.py

from flask import Flask, render_template

app = Flask(__name__)

@app.route('/') #http://localhost:5000/
def home():
    return render_template("home.html") #must put home.html under folder "template"

@app.route('/about/') #http://localhost:5000/about
def about():
    return render_template("about.html")

if __name__ == "__main__":
    app.run(debug=True)
```

Check the about.html, home.html, layout.html, main.css,

187. Creating a Python Virtual Environment

安装virtualenv package
Pip install virtualenv

我们写的文件放在Demo文件夹里面，把Demo放在mysite文件夹里面， **cd到mysite**

添加新的虚拟环境
Python -m venv virtual

查看python
Virtual\Scripts\python
Exit()

Install flask for new virtual env:
Virtual\Scripts\pip install flask

Run demo locally with new virtual env:
Virtual\Scripts\python Demo\script1.py

189. Deploying the Website to a Live Server

注册heroku账户，
下载安装heroku toolbelt

Cd 到mysite\Demo

Heroku login

Create apps:
heroku create app-name

List your apps:
heroku apps

Website address: app-name.herokuapp.com

Get a list of packages of the new virtual env
..\virtual\Scripts\pip freeze

Install gunicorn
..\virtual\Scripts\pip install gunicorn

Write packages info to requirements.txt
..\virtual\Scripts\pip freeze > requirements.txt

Create procfile, no file extension
Web: gunicorn script1:app

Create runtime.txt, specify the python version, check the website
python-3.9.6

Still in Demo folder:
Git init
Git add . # . Means files of current folder
Git commit -m "note message"
Heroku git:remote --app app-name #beroku git:remote --app yc-web1
git push heroku master

如何检查login成功：
heroku login
heroku info

做些更改之后
在vs code or Atom里，如果更改之后还没有push，文件会变为黄色
git add .
git commit -m "add changes"
git push heroku master
(如果第一次执行push可能会报错，需要授权
Git config --global user.email "ycheng22@hotmail.com"
Git config --blobal user.name "ycheng22"
完成上面两行，再次
git push heroku master
)

Troubleshooting

If you deployed your website on Heroku but when you visit the website on the browser you see an error, you probably did something wrong during the deployment.

No worries! You can see what you did wrong by looking at the server logs. You can access the server logs by running the following in your terminal:

```
heroku logs
```

This command will show a series of messages. Carefully read the logs to understand what went wrong. If you have trouble understanding the logs, feel free to post the logs in the Q&A.

==========

Section 22: Building Desktop Graphical User Interfaces (GUI) with Python

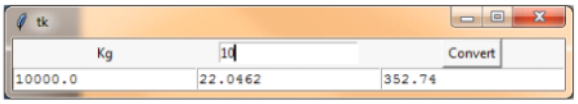
Learn Tkinter


```python
# -*- coding: utf-8 -*-
"""
Created on Fri Aug 20 16:36:46 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com

Reference:
"""
from tkinter import *

window = Tk()

def km2miles():
    print(e1_value.get())
    miles = float(e1_value.get())*1.6
    t1.insert(END, miles)

b1 = Button(window, text="Excute", command=km2miles)
#b1.pack()
b1.grid(row=0, column=0)

e1_value = StringVar()
e1 = Entry(window, textvariable=e1_value)
e1.grid(row=0, column=1)

t1 = Text(window, height=1, width=20)
t1.grid(row=0, column=2)

window.mainloop()
```

Exercise: Create a Multi-Widget GUI

Create a Python program that expects a kilogram input value and converts that value to grams, pounds, and ounces when the user pushes the *Convert* button.

The program will look similar to the one in the following picture:



Tip:

1 kg = 1000 grams

1 kg = 2.20462 pounds

1 kg = 35.274 ounces

```python
# -*- coding: utf-8 -*-
"""
Created on Fri Aug 20 17:03:56 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com
```

```
Reference:
"""
from tkinter import *

window = Tk()

def from_kg():

    gram = float(e2_value.get()) * 1000
    pound = float(e2_value.get()) * 2.20462
    ounce = float(e2_value.get()) * 35.274

    # Empty the Text boxes if they had text from the previous use and fill them again
    t1.delete("1.0", END) # Deletes the content of the Text box from start to END
    t1.insert(END, gram) # Fill in the text box with the value of gram variable
    t2.delete("1.0", END)
    t2.insert(END, pound)
    t3.delete("1.0", END)
    t3.insert(END, ounce)


#create Kg label
e1 = Label(window, text="Kg")
e1.grid(row=0, column=0)

#entry
e2_value = StringVar()
e2 = Entry(window, textvariable=e2_value)
e2.grid(row=0, column=1)

#button
b1 = Button(window, text="Convert", command=from_kg)
b1.grid(row=0, column=2)

#three text boxes
t1 = Text(window, height=1, width=20)
t1.grid(row=1, column=0)

t2 = Text(window, height=1, width=20)
t2.grid(row=1, column=1)

t3 = Text(window, height=1, width=20)
t3.grid(row=1, column=2)

# This makes sure to keep the main window open
window.mainloop()
```

==========

Section 23: Interacting with Databases

1. connect to a database
2. Create a cursor object
3. Write and SQL query
4. Commit changes
5. Close database connection

199. Connecting to an SQLite Database with Python

```
# -*- coding: utf-8 -*-
"""
Created on Fri Aug 20 17:16:11 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com

Reference:
"""
import sqlite3

def create_table():
    conn = sqlite3.connect("lite.db")
    cur = conn.cursor()
    cur.execute("CREATE TABLE IF NOT EXISTS store (item TEXT, quantity INTEGER, price REAL)")
    conn.commit()
    conn.close()
```

```
create_table()

def insert(item, quantity, price):
    conn = sqlite3.connect("lite.db")
    cur = conn.cursor()
    cur.execute("INSERT INTO store VALUES (?,?,?)", (item, quantity, price))
    conn.commit()
    conn.close()

insert("Coffe Cup", 10, 5)

def view():
    conn = sqlite3.connect("lite.db")
    cur = conn.cursor()
    cur.execute("SELECT * FROM store")
    rows = cur.fetchall()
    conn.close()
    return rows

def delete(item):
    conn = sqlite3.connect("lite.db")
    cur = conn.cursor()
    cur.execute("DELETE FROM store WHERE item=?", (item,))
    conn.commit()
    conn.close()

def update(quantity, price, item):
    conn = sqlite3.connect("lite.db")
    cur = conn.cursor()
    cur.execute("UPDATE store SET quantity=?, price=? WHERE item=?", (quantity, price, item))
    conn.commit()
    conn.close()

update(11, 6,  "Coffe Cup")
#delete("Wine Glass")
print(view())
```

201. PostgreSQL Database with Python

```
# -*- coding: utf-8 -*-
"""
Created on Fri Aug 20 17:50:05 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com

Reference:
"""
import psycopg2

def create_table():
    #conn = psycopg2.connect("dbname='db1' user='postgres' password='2020' host='localhost' port='5433'")
    conn = psycopg2.connect("host=127.0.0.1 port=5433 dbname=db1 user=postgres password=2020")
    cur = conn.cursor()
    cur = conn.cursor()
    cur.execute("CREATE TABLE IF NOT EXISTS store (item TEXT, quantity INTEGER, price REAL)")
    conn.commit()
    conn.close()

def insert(item, quantity, price):
    conn = psycopg2.connect("host=127.0.0.1 port=5433 dbname=db1 user=postgres password=2020")
    cur = conn.cursor()
    #cur.execute("INSERT INTO store VALUES ('%s','%s','%s')" % (item, quantity, price))
    cur.execute("INSERT INTO store VALUES (%s, %s, %s)", (item, quantity, price))
    conn.commit()
    conn.close()

def view():
    conn = psycopg2.connect("host=127.0.0.1 port=5433 dbname=db1 user=postgres password=2020")
    cur = conn.cursor()
    cur.execute("SELECT * FROM store")
    rows = cur.fetchall()
    conn.close()
    return rows

def delete(item):
    conn = psycopg2.connect("host=127.0.0.1 port=5433 dbname=db1 user=postgres password=2020")
    cur = conn.cursor()
    cur.execute("DELETE FROM store WHERE item=%s", (item,))
    conn.commit()
    conn.close()
```

```
def update(quantity, price, item):
    conn = psycopg2.connect("host=127.0.0.1 port=5433 dbname=db1 user=postgres password=2020")
    cur = conn.cursor()
    cur.execute("UPDATE store SET quantity=%s, price=%s WHERE item=%s", (quantity, price, item))
    conn.commit()
    conn.close()


create_table()
#insert("Orange", 10, 1.5)
#delete("Orange")
update(20, 26.5, "Apple")
print(view())
#insert("Coffe Cup", 10, 5)
#update(11, 6,  "Coffe Cup")
#delete("Wine Glass")
```

Working with MySQL Databases

In previous videos, I explained how to interact with PostGreSQL databases. If you prefer to work with MySQL instead of PostGreSQL, see the code further down.

I set up a remote MySQL database on a server with the IP address 108.167.140.122, so you don't have to install and set up a MySQL database yourself. To connect and query data from that remote database, you need a *username*, *password*, and the name of the *database*. These are written inside the Python script below.

You also need a Python library that interacts with MySQL databases. Many libraries are compatible, but I prefer *mysql.connector*. To install *mysql.connector*: simply execute `pip install mysql-connector` or `pip3 install mysql-connector` depending on whether you use *pip* or *pip3*. Once you install the library, try this working example:

```
import mysql.connector
word = input("Enter a word in English and press Enter: ")
con = mysql.connector.connect(
    user="ardit700_student",
    password = "ardit700_student",
    host="108.167.140.122",
    database = "ardit700_pm1database"
)
cursor = con.cursor()
query = cursor.execute("SELECT * FROM Dictionary WHERE Expression = '%s'" % word)
results = cursor.fetchall()
if results:
    for result in results:
        print(result[1])
else:
    print("We couldn't find any results about that.")
```

==========

Section 24: Application 5: Building a Desktop SQL Book Inventory GUI App with Python

Frontend.py

```
# -*- coding: utf-8 -*-
"""
Created on Fri Aug 20 18:40:16 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com

Reference:
```

```
Reference:

About:
A program that stores this book information:
Title, Author, Year, ISBN

User can:

View all records,
Search an entry
Add entry
Update entry
Delete
Close
"""
from tkinter import *
import backend

def get_selected_row(event): #what's' event for?
    try:
        global selected_tuple
        index = list1.curselection()[0] #if not [0], index would be like (0, ) (2, )
        selected_tuple = list1.get(index)
        e1.delete(0, END)
        e1.insert(END, selected_tuple[1])
        e2.delete(0, END)
        e2.insert(END, selected_tuple[2])
        e3.delete(0, END)
        e3.insert(END, selected_tuple[3])
        e4.delete(0, END)
        e4.insert(END, selected_tuple[4])
    except IndexError:
        pass

def view_command():
    list1.delete(0, END) #delete contents inside list1 before view all
    for row in backend.view():
        list1.insert(END, row)

def search_command():
    list1.delete(0, END)
    for row in backend.search(title_text.get(), author_text.get(), year_text.get(), isbn_text.get()):
        list1.insert(END, row)

def add_command():
    backend.insert(title_text.get(), author_text.get(), year_text.get(), isbn_text.get()) #add to database
    list1.delete(0, END)
    list1.insert(END, (title_text.get(), author_text.get(), year_text.get(), isbn_text.get())) #show in listbox

def delete_command():
    backend.delete(selected_tuple[0])

def update_command():
    backend.update(selected_tuple[0], title_text.get(), author_text.get(), year_text.get(), isbn_text.get())

window = Tk()
window.wm_title("BookStore")

l1 = Label(window, text="Title")
l1.grid(row=0, column=0)

l2 = Label(window, text="Author")
l2.grid(row=0, column=2)

l3 = Label(window, text="Year")
l3.grid(row=1, column=0)

l4 = Label(window, text="ISBN")
l4.grid(row=1, column=2)

title_text = StringVar()
e1 = Entry(window, textvariable=title_text)
e1.grid(row=0, column=1)

author_text = StringVar()
e2 = Entry(window, textvariable=author_text)
e2.grid(row=0, column=3)

year_text = StringVar()
e3 = Entry(window, textvariable=year_text)
e3.grid(row=1, column=1)

isbn_text = StringVar()
e4 = Entry(window, textvariable=isbn_text)
e4.grid(row=1, column=3)

list1 = Listbox(window, height=6, width=35)
```

```
list1 = Listbox(window, height=6, width=35)
list1.grid(row=2, column=0, rowspan=6, columnspan=2)

sb1 = Scrollbar(window)
sb1.grid(row=2, column=2, rowspan=6)

list1.configure(yscrollcommand=sb1.set) ##############
sb1.configure(command=list1.yview)  ##############

#bind() is used to bind a function to a widget event
list1.bind('<<ListboxSelect>>', get_selected_row)

b1 = Button(window, text="View all", width=12, command=view_command)
b1.grid(row=2, column=3)

b2 = Button(window, text="Search entry", width=12, command=search_command)
b2.grid(row=3, column=3)

b3 = Button(window, text="Add entry", width=12, command=add_command)
b3.grid(row=4, column=3)

b4 = Button(window, text="Update selected", width=12, command=update_command)
b4.grid(row=5, column=3)

b5 = Button(window, text="Delete selected", width=12, command=delete_command)
b5.grid(row=6, column=3)

b6 = Button(window, text="Close", width=12, command=window.destroy)
b6.grid(row=7, column=3)

window.mainloop()
```

Backend.py

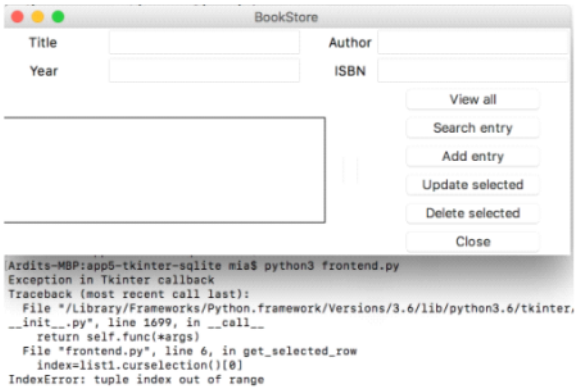
```
# -*- coding: utf-8 -*-
"""
Created on Fri Aug 20 23:07:22 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com

Reference:
"""
import sqlite3

def connect():
    conn = sqlite3.connect("books.db")
    cur = conn.cursor()
    cur.execute("CREATE TABLE IF NOT EXISTS book (id INTEGER PRIMARY KEY, title text, author text, year integer, isbn
integer)")
    conn.commit()
    conn.close()

def insert(title, author, year, isbn):
    conn = sqlite3.connect("books.db")
    cur = conn.cursor()
    cur.execute("INSERT INTO book VALUES (NULL,?,?,?,?)", (title, author, year, isbn))
    conn.commit()
    conn.close()

def view():
    conn = sqlite3.connect("books.db")
    cur = conn.cursor()
    cur.execute("SELECT * FROM book")
    rows = cur.fetchall()
    conn.close()
    return rows

def search(title="", author="", year="", isbn=""):
    conn = sqlite3.connect("books.db")
    cur = conn.cursor()
    cur.execute("SELECT * FROM book WHERE title=? OR author=? OR year=? OR isbn=?", (title, author, year, isbn))
    rows = cur.fetchall()
    conn.close()
    return rows

def delete(id):
    conn = sqlite3.connect("books.db")
    cur = conn.cursor()
    cur.execute("DELETE FROM book WHERE id=?", (id,))
    conn.commit()
    conn.close()

def update(id, title, author, year, isbn):
    conn = sqlite3.connect("books.db")
```

```
    cur = conn.cursor()
    cur.execute("UPDATE book SET title=?, author=?, year=?, isbn=? WHERE id=?", (title, author, year, isbn, id))
    conn.commit()
    conn.close()

connect()
#insert("The Sun", "John Smith", 1918, 913123132)
#delete(3)
#update(4, "The moon", "John Smooth", 1917, 99999)
#print(view())
#print(search(author="John Smith"))
```

Exercise: Fixing a Bug in Our Program

Exercise

If you haven't already noticed, the program has a bug. When the listbox is empty and the user clicks the listbox, an *IndexError* is generated in the terminal:



Why does this error happen?

Well, everything starts with the user clicking on the listbox. Clicking the listbox executes the following code:

```
list1.bind('<<ListboxSelect>>',get_selected_row)
```

That code calls the `get_selected_row` function:

```
1  def get_selected_row(event):
2      global selected_tuple
3      index=list1.curselection()[0]
4      selected_tuple=list1.get(index)
5      e1.delete(0,END)
6      e1.insert(END,selected_tuple[1])
7      e2.delete(0,END)
8      e2.insert(END,selected_tuple[2])
9      e3.delete(0,END)
10     e3.insert(END,selected_tuple[3])
11     e4.delete(0,END)
12     e4.insert(END,selected_tuple[4])
```

Since the listbox is empty, `list1.curselection()` will be an empty list with no items. Trying to access the first item on the list with `[0]` in line 3 will throw an error because there is no first item in the list.

Try to fix that bug. The next lecture contains the solution.

Solution: Fixing a Bug in Our Program

Solution

```
def get_selected_row(event):
    try:
        global selected_tuple
        index=list1.curselection()[0]
        selected_tuple=list1.get(index)
        e1.delete(0,END)
        e1.insert(END,selected_tuple[1])
        e2.delete(0,END)
        e2.insert(END,selected_tuple[2])
        e3.delete(0,END)
        e3.insert(END,selected_tuple[3])
        e4.delete(0,END)
        e4.insert(END,selected_tuple[4])
    except IndexError:
        pass
```

Explanation

The error was fixed by simply implementing a `try` and `except` block. When the `get_selected_row` function is called, Python will execute the indented block under `try`. If there is an *IndexError*, none of the lines under `try` will be executed; the line under `except` will be executed, which is `pass`. The `pass` statement means "do nothing". Therefore, the function will do nothing when there's an empty listbox.

212. Creating .exe and .app Executables from the Python Script

Install pyinstaller

In command, current folder:
Pyinstaller frontend.py
Or
Pyinstaller --onefile --windowed frontend.py (这个起作用了，具体命令细节待研究)

In folder dist: books.db, frontend.exe
这个books.db是空的，可以用之前生成的books.db替换

==========

Section 26: Object-Oriented Programming (OOP)

Backend.py

```
import sqlite3

class Database:

    def __init__(self, db):
        self.conn=sqlite3.connect(db)
        self.cur=self.conn.cursor()
        self.cur.execute("CREATE TABLE IF NOT EXISTS book (id INTEGER PRIMARY KEY, title text, author text, year integer,
isbn integer)")
        self.conn.commit()

    def insert(self,title,author,year,isbn):
        self.cur.execute("INSERT INTO book VALUES (NULL,?,?,?,?)",(title,author,year,isbn))
        self.conn.commit()

    def view(self):
        self.cur.execute("SELECT * FROM book")
        rows=self.cur.fetchall()
        return rows

    def search(self,title="",author="",year="",isbn=""):
        self.cur.execute("SELECT * FROM book WHERE title=? OR author=? OR year=? OR isbn=?", (title,author,year,isbn))
        rows=self.cur.fetchall()
```

```
        return rows

    def delete(self,id):
        self.cur.execute("DELETE FROM book WHERE id=?",(id,))
        self.conn.commit()

    def update(self,id,title,author,year,isbn):
        self.cur.execute("UPDATE book SET title=?, author=?, year=?, isbn=? WHERE id=?",(title,author,year,isbn,id))
        self.conn.commit()

    def __del__(self):
        self.conn.close()

#insert("The Sun","John Smith",1918,913123132)
#delete(3)
#update(4,"The moon","John Smooth",1917,99999)
#print(view())
#print(search(author="John Smooth"))
```

Frontend.py

```
from tkinter import *
from backend import Database

database=Database("books.db")

class Window(object):

    def __init__(self,window):

        self.window = window

        self.window.wm_title("BookStore")


        l1=Label(window,text="Title")
        l1.grid(row=0,column=0)

        l2=Label(window,text="Author")
        l2.grid(row=0,column=2)

        l3=Label(window,text="Year")
        l3.grid(row=1,column=0)

        l4=Label(window,text="ISBN")
        l4.grid(row=1,column=2)

        self.title_text=StringVar()
        self.e1=Entry(window,textvariable=self.title_text)
        self.e1.grid(row=0,column=1)

        self.author_text=StringVar()
        self.e2=Entry(window,textvariable=self.author_text)
        self.e2.grid(row=0,column=3)

        self.year_text=StringVar()
        self.e3=Entry(window,textvariable=self.year_text)
        self.e3.grid(row=1,column=1)

        self.isbn_text=StringVar()
        self.e4=Entry(window,textvariable=self.isbn_text)
        self.e4.grid(row=1,column=3)

        self.list1=Listbox(window, height=6,width=35)
        self.list1.grid(row=2,column=0,rowspan=6,columnspan=2)

        sb1=Scrollbar(window)
        sb1.grid(row=2,column=2,rowspan=6)

        self.list1.configure(yscrollcommand=sb1.set)
        sb1.configure(command=self.list1.yview)

        self.list1.bind('<<ListboxSelect>>',self.get_selected_row)

        b1=Button(window,text="View all", width=12,command=self.view_command)
        b1.grid(row=2,column=3)

        b2=Button(window,text="Search entry", width=12,command=self.search_command)
        b2.grid(row=3,column=3)

        b3=Button(window,text="Add entry", width=12,command=self.add_command)
        b3.grid(row=4,column=3)
```

```
        b4=Button(window,text="Update selected", width=12,command=self.update_command)
        b4.grid(row=5,column=3)

        b5=Button(window,text="Delete selected", width=12,command=self.delete_command)
        b5.grid(row=6,column=3)

        b6=Button(window,text="Close", width=12,command=window.destroy)
        b6.grid(row=7,column=3)

    def get_selected_row(self,event):
        index=self.list1.curselection()[0]
        self.selected_tuple=self.list1.get(index)
        self.e1.delete(0,END)
        self.e1.insert(END,self.selected_tuple[1])
        self.e2.delete(0,END)
        self.e2.insert(END,self.selected_tuple[2])
        self.e3.delete(0,END)
        self.e3.insert(END,self.selected_tuple[3])
        self.e4.delete(0,END)
        self.e4.insert(END,self.selected_tuple[4])

    def view_command(self):
        self.list1.delete(0,END)
        for row in database.view():
            self.list1.insert(END,row)

    def search_command(self):
        self.list1.delete(0,END)
        for row in database.search(self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isbn_text.get()):
            self.list1.insert(END,row)

    def add_command(self):
        database.insert(self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isbn_text.get())
        self.list1.delete(0,END)
        self.list1.insert(END,(self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isbn_text.get()))

    def delete_command(self):
        database.delete(self.selected_tuple[0])

    def update_command(self):

database.update(self.selected_tuple[0],self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isbn_text.get
())

window=Tk()
Window(window)
window.mainloop()
```

Acc.py

```
class Account:

    def __init__(self, filepath):
        self.filepath = filepath
        with(open(filepath, 'r')) as file:
            self.balance = int(file.read())

    def withdraw(self, amount):
        self.balance = self.balance - amount

    def deposit(self, amount):
        self.balance = self.balance + amount

    def commit(self):
        with open(self.filepath, 'w') as file:
            file.write(str(self.balance))


class Checking(Account):
    """This class generates checking account objects"""

    type = 'checking'

    def __init__(self, filepath, fee):
        Account.__init__(self, filepath)
        self.fee = fee

    def transfer(self, amount):
        self.balance = self.balance - amount - self.fee

jacks_checking = Checking('./jack.txt', 1)
jacks_checking.transfer(50)
print(jacks_checking.balance)
jacks_checking.commit()
print(jacks_checking.type)

johns_checking = Checking('./john.txt', 1)
```

```
johns_checking.transfer(50)
print(johns_checking.balance)
johns_checking.commit()
print(johns_checking.type)
```

==========

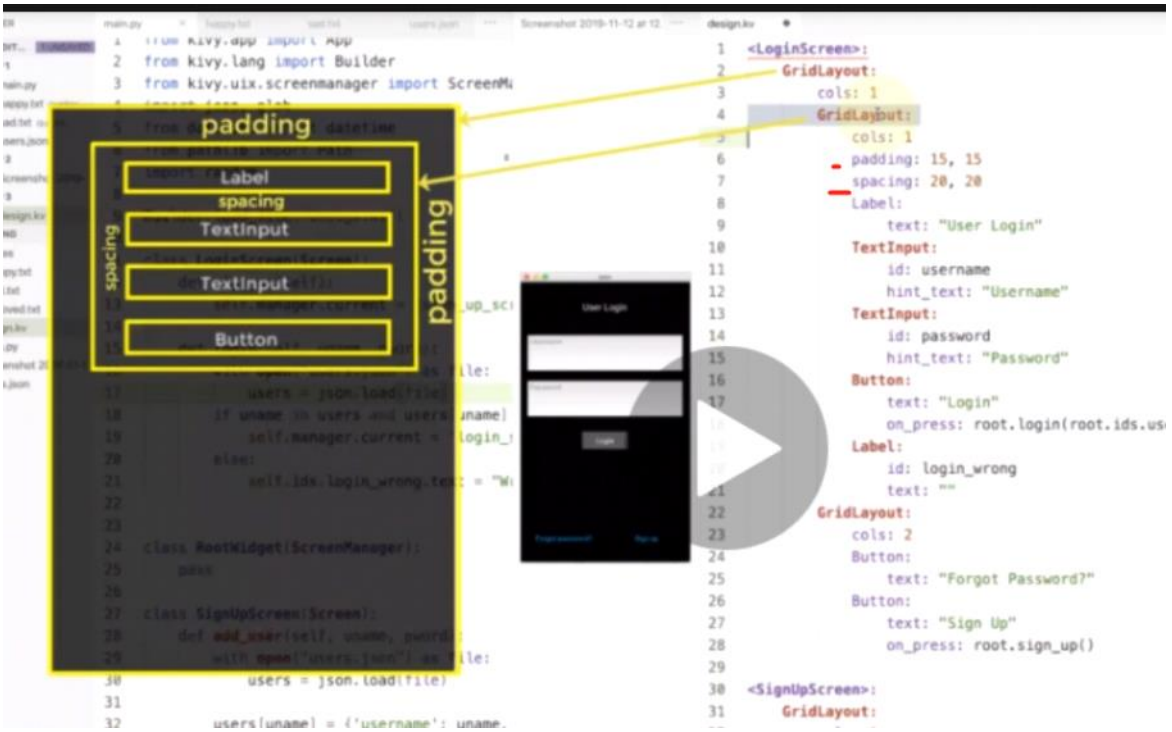
Section 27: App 6: Mobile App Development: Build a Feel-Good App

Installing the Library

We need the Python kivy library to build this GUI program. You can install kivy with:

pip install kivy

If you have issues with the installation, please check the kivy installation tips.



Design.kv

```
<LoginScreen>:

    GridLayout:
        cols: 1

        GridLayout:
            cols: 1
            padding: 15, 15
            spacing: 20, 20
            Label:
                text: "User Login"
                font_size: '20sp' #sp = space-independent pixels
            TextInput:
                id: username
                hint_text: "Username"
            TextInput:
                id: password
                password: True #hidden the password as *****
                hint_text: "Password"
            RelativeLayout:
                Button:
                    text: "Login"
```

```
                    text: "Login"
                    on_press: root.login(root.ids.username.text, root.ids.password.text)
                    size_hint: 0.3, 0.5 #sqeeze 0.3 in horizontal, 0.5 in vertical
                    pos_hint: {'center_x': 0.5, 'center_y': 0.6}
            Label:
                id: login_wrong
                text: ""

        GridLayout:
            cols: 2
            size_hint: 0.2, 0.2
            padding: 10, 10
            spacing: 10, 0
            Button:
                text: "Forgot Password?"
                background_color: 1, 1, 1, 0 #R, G, B, transparency
                opacity: 1 if self.state == 'normal' else 0.5 #when click, opacity is 0.5
                color: 0.1, 0.7, 1, 1
            Button:
                text: "Sign up"
                background_color: 1, 1, 1, 0 #R, G, B, transparency
                opacity: 1 if self.state == 'normal' else 0.5 #when click, opacity is 0.5
                color: 0.1, 0.7, 1, 1
                on_press: root.sign_up()

<SignUpScreen>:
    GridLayout:
        cols: 1
        padding: 20, 20
        spacing: 20, 20
        Label:
            text: "Sign up for a space journey!"
        TextInput:
            id: username
            hint_text: "Username"
        TextInput:
            id: password
            hint_text: "Password"
        Button:
            text: "Submit"
            on_press: root.add_user(root.ids.username.text, root.ids.password.text)

<SignUpScreenSuccess>:
    GridLayout:
        cols: 1
        Label:
            text: "Sign up successful!"
        Button:
            text: "Login page"
            on_press: root.go_to_login()

<LoginScreenSuccess>:
    GridLayout:
        cols: 1
        padding: 30, 30
        spacing: 30, 30
        RelativeLayout:
            ImageButton:
                on_press: root.log_out()
                source: 'logout_hover.png' if self.hovered else 'logout_nothover.png'
                size_hint: 0.35, 0.35
                pos_hint: {'center_x': 0.93, 'center_y': 0.8}
        Label:
            text: "How do you fell?"
        TextInput:
            id: feeling
            hint_Text: "Things to try: happy, sad, unloved..."
        Button:
            text: "Enlighten me"
            on_press: root.get_quote(root.ids.feeling.text)
        ScrollView:
            Label:
                id: quote
                text: ""
                text_size: self.width, None
                size_hint_y: None
                height: self.texture_size[1] #texture_size is a tuple: (text.width,text.height)

<RootWidget>:
    LoginScreen:
        name: "login_screen"
    SignUpScreen:
        name: "sign_up_screen"
    SignUpScreenSuccess:
        name: "sign_up_screen_success"
    LoginScreenSuccess:
        name: "login_screen_success"
```

Main.py

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Sep  8 17:43:01 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com

@Github: https://github.com/ycheng22

Reference:ex
"""
from kivy.app import App
from kivy.lang import Builder
from kivy.uix.screenmanager import ScreenManager, Screen
from kivy.uix.image import Image
from kivy.uix.behaviors import ButtonBehavior
import json, glob, random
from datetime import datetime
from pathlib import Path
from hoverable import HoverBehavior


Builder.load_file('design.kv')

class LoginScreen(Screen):
    def sign_up(self):
        self.manager.current = "sign_up_screen"

    def login(self, uname, pword):
        with open("users.json") as file:
            users = json.load(file)
        if uname in users and users[uname]['password'] == pword:
            self.manager.current  = "login_screen_success"
        else:
            self.ids.login_wrong.text = "Wrong username or password!"


class SignUpScreen(Screen):
    def add_user(self, uname, pword):
        with open("users.json") as file:
            users = json.load(file)

        users[uname] = {'username': uname,
                        'password': pword,
                        'created': datetime.now().strftime("%Y-%m-%d %H-%M-%S")}
        with open("users.json", "w") as file:
            json.dump(users, file)
        self.manager.current = "sign_up_screen_success"

class SignUpScreenSuccess(Screen):
    def go_to_login(self):
        self.manager.transition.direction = 'right'
        self.manager.current = 'login_screen'

class LoginScreenSuccess(Screen):
    def log_out(self):
        self.manager.transition.direction = "right"
        self.manager.current = "login_screen"

    def get_quote(self, feel):
        feel = feel.lower()
        available_feelings = glob.glob("quotes/*txt")

        #Path(filename).stem will return the filename except the extension
        available_feelings = [Path(filename).stem for filename in available_feelings]

        if feel in available_feelings:
            with open(f"quotes/{feel}.txt", "rb") as file:
                quotes = file.readlines()
            self.ids.quote.text = str(random.choice(quotes))
        else:
            self.ids.quote.text = "Try another feeling"

class ImageButton(ButtonBehavior, HoverBehavior, Image):
    pass


class RootWidget(ScreenManager):
    pass

class MainApp(App):
    def build(self):
        return RootWidget()
```

```
if __name__ == '__main__':
    MainApp().run()
```

Hoverable.py

```
"""Hoverable Behaviour (changing when the mouse is on the widget by O. Poyen.
License: LGPL
"""
__author__ = 'Olivier POYEN'


from kivy.properties import BooleanProperty, ObjectProperty
from kivy.core.window import Window

class HoverBehavior(object):
    """Hover behavior.

    :Events:
        `on_enter`
            Fired when mouse enter the bbox of the widget.
        `on_leave`
            Fired when the mouse exit the widget
    """

    hovered = BooleanProperty(False)
    border_point= ObjectProperty(None)
    '''Contains the last relevant point received by the Hoverable. This can
    be used in `on_enter` or `on_leave` in order to know where was dispatched the event.
    '''

    def __init__(self, **kwargs):
        self.register_event_type('on_enter')
        self.register_event_type('on_leave')
        Window.bind(mouse_pos=self.on_mouse_pos)
        super(HoverBehavior, self).__init__(**kwargs)

    def on_mouse_pos(self, *args):
        if not self.get_root_window():
            return # do proceed if I'm not displayed <=> If have no parent
        pos = args[1]
        #Next line to_widget allow to compensate for relative layout
        inside = self.collide_point(*self.to_widget(*pos))
        if self.hovered == inside:
            #We have already done what was needed
            return
        self.border_point = pos
        self.hovered = inside
        if inside:
            self.dispatch('on_enter')
        else:
            self.dispatch('on_leave')

    def on_enter(self):
        pass

    def on_leave(self):
        pass

from kivy.factory import Factory
Factory.register('HoverBehavior', HoverBehavior)

if __name__=='__main__':
    from kivy.uix.floatlayout import FloatLayout
    from kivy.lang import Builder
    from kivy.uix.label import Label
    from kivy.base import runTouchApp
    class HoverLabel(Label, HoverBehavior):
        def on_enter(self, *args):
            print("You are in, through this point", self.border_point)

        def on_leave(self, *args):
            print("You left through this point", self.border_point)

    Builder.load_string('''
<HoverLabel>:
    text: "inside" if self.hovered else "outside"
    pos: 200,200
    size_hint: None, None
    size: 100, 30
    canvas.before:
        Color:
            rgb: 1,0,0
        Rectangle:
            size: self.size
            pos: self.pos
```

```
    ''')
    fl = FloatLayout()
    fl.add_widget(HoverLabel())
    runTouchApp(fl)
```

==========

Section 28: Making an Android APK File from the Kivy App

244. Creating an APK file for Android

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/20145342#overview>

Download the files to linux system

Bash kivy-buildozer-installer.sh to install libries

Python3 main.py运行程序

Buildozer init

生成buildozer.spec
修改里面的参数:

```
3 # (str) Title of your application
4 title = How Do You Feel?

15 # (list) Source files to include (let empty to include all the files)
16 source.include_exts = py,png,jpg,kv,atlas,json,txt

37 # (list) Application requirements
38 # comma separated e.g. requirements = sqlite3,kivy
39 requirements = python3,kivy

51 # (str) Supported orientation (one of landscape, sensorLandscape, portrait or all)
52 orientation = all

256 # (str) The Android arch to build for, choices: armeabi-v7a, arm64-v8a, x86, x86_64
257 android.arch = arm64-v8a
```

生成apk file
~~Buildozer android debug~~
Buildozer android clean debug

最终还是生成apk失败，还未解决。

https://gofile.io/welcome

Deploying to iOS

Unfortunately, converting a kivy app to an iOS app requires you to have access to a Mac computer. The conversion cannot be done on a Windows or a Linux computer.

It's also currently not possible to do this with Python 3. You need to use Python 2.

I will replace these notes with a video once there's a stable method to convert kivy apps to iOS. Meanwhile, you can try the official instructions found here: https://kivy.org/doc/stable/guide/packaging-ios.html

or if you prefer video, you can watch this YouTube video from Erik Sandberg: https://www.youtube.com/watch?v=UAi3PG-qN2k

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/20145342#overview>

==========

Section 29: Web Scraping with Python & Beautiful Soup

https://pythonhow.com/example.html

Request Headers to Enable Web Scraping

Important note:

When I use this code in the next video:

```
r = requests.get("http://www.pythonhow.com/example.html")
```

please use this instead:

```
r = requests.get("https://pythonizing.github.io/data/example.html", headers={'User-agent': 'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:61.0) Gecko/20100101 Firefox/61.0'})
```

Explanation:

As you can see in the new code, we are changing the domain name from *pythonhow.com* to *pythonizing.github.io* and we are also adding a `headers` argument. We are changing the domain because the new domain now contains the data we want to scrape. And we are adding request headers because that allows the Python script to impersonate a web browser. Of course, you don't need the header argument for every website, but i's good to have it just in case.

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/14736692#overview>

==========

Section 30: App 7: Web Scraping -Scraping Properties for Sale from the Web

Preparing the Request Headers

Important note:

Whenever I use this code in the following videos:

```
r = requests.get("http://www.pythonhow.com/real-estate/rock-springs-wy/LCWYROCKSPRINGS/")
```

please use this code instead:

```
r = requests.get("https://pythonizing.github.io/data/real-estate/rock-springs-wy/LCWYROCKSPRINGS/", headers={'User-agent': 'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:61.0) Gecko/20100101 Firefox/61.0'})
```

Explanation:

As you can see in the new code, we are changing the domain name from *pythonhow.com* to *pythonizing.github.io* and we are also adding a `headers` argument. We are changing the domain because the new domain now contains the data we want to scrape. And we are adding request headers because that allows the Python script to impersonate a web browser. Of course, you don't need the header argument for every website, but i's good to have it just in case.

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/14736656#overview>

==========

Section 31: App 8: Flask and PostGreSQL - Build a Data Collector Web App

在本sec文件夹下:
Pip install virtualenv

Python -m venv virtual

安装flask, psycopg2, flask_sqlalchemy

Current folder> virtual\Scripts\pip install flask

也可以激活本地文件夹下的虚拟环境:
cd到D:\OneDrive - University Of Houston\Notebook_OneDrive\Python_Mega_Course\sec31_App8_Data Collector Web App\virtual\Scripts
打开cmd， 输入 activate， 回车即可（其他prompt不一定行，必须用自带的prompt）

264. Creating the PostGreSQL Database Model

跟着上面这个视频，无法生存table data,
在test_sqlalchemy.py中进行了一些成功尝试，
后在add_table_manually中手动添加table data,

App.py

```
from flask import Flask, render_template, request
#from flask.ext.sqlalchemy import SQLAlchemy
from flask_sqlalchemy import SQLAlchemy
from send_email import send_email
from sqlalchemy.sql import func

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = \
    'postgresql://postgres:1992@localhost/height_collector'
db = SQLAlchemy(app)
#db.create_all()

class Data(db.Model):
    __tablename__ = "data"
    id = db.Column(db.Integer, primary_key=True)
    email_ = db.Column(db.String(120), unique=True)
    height_ = db.Column(db.Integer)


    def __init__(self, email_, height_):
        self.email_ = email_
        self.height_ = height_

@app.route("/")
def index():
    return render_template('index.html')

@app.route("/success", methods=['POST'])
def success():
    if request.method == "POST":
        email = request.form["email_name"]
        height = request.form["height_name"]
        if db.session.query(Data).filter\
            (Data.email_==email).count() == 0:
            data = Data(email, height)
            db.session.add(data)
            db.session.commit()
            ave_height = db.session.query(func.avg(Data.height_)).scalar()
            ave_height = round(ave_height, 1)
            count = db.session.query(Data.height_).count()
            #send_email(email, height, ave_height, count)
            return render_template("success.html")
    return render_template('index.html',
        text="Email already exits, try different one!")

if __name__ == '__main__':
    app.debug = True
    app.run(port = 5001)
```

Send_email.py

```
from email.mime.text import MIMEText
import smtplib

def send_email(email, height, ave_height, count):
    from_email = "mygmail@gmail.com"
    from_passowrd = "mypassword"
    to email = email
```

```
        _

    subject = "Height data"
    message = "Hey there, your height is <strong>%s</strong>. \
        Average height of all is <strong>%s</strong> and that is calculated out \
        <strong>%s</strong> of people." % (height, ave_height, count)

    msg = MIMEText(message, 'html')
    msg['subject'] = subject
    msg['To'] = to_email
    msg['From'] = from_email

    gmail = smtplib.SMTP('smtp.gmail.com', 587)
    gmail.ehlo()
    gmail.starttls()
    gmail.login(from_email, from_passowrd)
    gmail.send_message(msg)
```

Index.html

```
<!DOCTYPE html>
<html lang="en">
    <title>Data Collector App</title>
    <head>
        <link href="../static/main.css" rel="stylesheet">
    </head>
    <body>
        <div class="container">
            <h1>Collecting height</h1>
            <h3>Please fill the entries to get population statistics on height</h3>
            <div class="message">
                {{text | safe}}
            </div>
            <form action="{{url_for('success')}}" method="POST">
                <input title="Your email will be safe with us"
                    placeholder="Enter your email address"
                    type="email" name="email_name" required> <br>
                <input title="Your data will be safe with us"
                    placeholder="Enter your height in cm"
                    type="number" min="50" max="300"
                    step="0.1" name="height_name" required> <br>
                <button type="submit"> Submit </button>
            </form>
        </div>
    </body>
</html>
```

Success.html

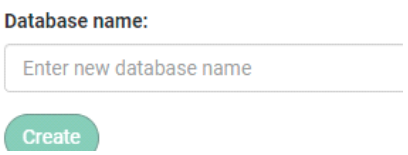
```
<!DOCTYPE html>
<html lang="en">
    <title>Data Collector App</title>
    <head>
        <link href="../static/main.css" rel="stylesheet">
    </head>
    <body>
        <div class="container">
            <p>Thank you for your submmisin! <br>
                You will receive an email with the survey results shortly.
            </p>
        </div>
    </body>
</html>
```

268. Deploying the Database Web App Online

https://www.pythonanywhere.com/

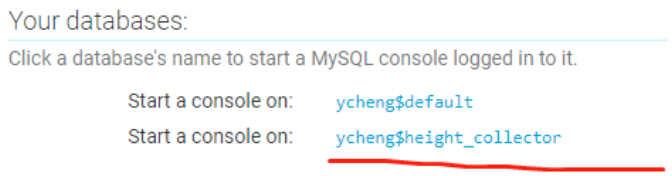
在files上传文件

在database create database, name is height_collector



Set database password: *******

The full database name would be like below,



click it to go to mysql command, create columns:
CREATE TABLE data (id SERIAL PRIMARY KEY, email_ VARCHAR(120), height_ INT);

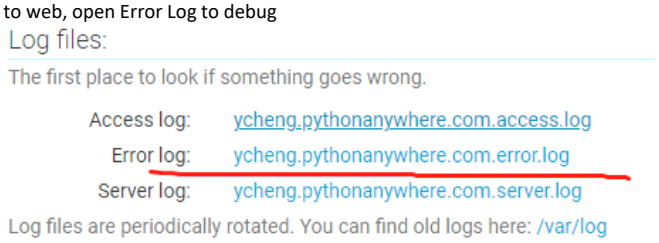
Connect to the database:

app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql+mysqlconnector://ycheng:19921019cyp@ycheng.mysql.pythonanywhere-services.com/ycheng$height_collector'

Note: no space in the string,

Go to webs, reload web, go the web address
Should work now.

Go to web, open Error Log to debug



部署在pythonanywhere.com上面flask_app.py

```
from flask import Flask, render_template, request
#from flask.ext.sqlalchemy import SQLAlchemy
from flask_sqlalchemy import SQLAlchemy
from send_email import send_email
from sqlalchemy.sql import func

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql+mysqlconnector://ycheng:19921019cyp@ycheng.mysql.pythonanywhere-
services.com/ycheng$height_collector'
db = SQLAlchemy(app)
#db.create_all()

class Data(db.Model):
    __tablename__ = "data"
    id = db.Column(db.Integer, primary_key=True)
    email_ = db.Column(db.String(120), unique=True)
    height_ = db.Column(db.Integer)

    def __init__(self, email_, height_):
        self.email_ = email_
        self.height_ = height_

@app.route("/")
def index():
    return render_template('index.html')

@app.route("/success", methods=['POST'])
def success():
    if request.method == "POST":
        email = request.form["email_name"]
        height = request.form["height_name"]
        if db.session.query(Data).filter\
            (Data.email_==email).count() == 0:
            data = Data(email, height)
            db.session.add(data)
            db.session.commit()
            ave_height = db.session.query(func.avg(Data.height_)).scalar()
            ave_height = round(ave_height, 1)
            count = db.session.query(Data.height_).count()
            #send_email(email, height, ave_height, count)
            return render_template("success.html")
    return render_template('index.html',
        text="Email already exits, try different one!")

if __name__ == '__main__':
    app.debug = True
    app.run()
```

269. Creating a Download-Upload Feature

App.py

```python
from flask import Flask, render_template, request, send_file
#from flask.ext.sqlalchemy import SQLAlchemy
from flask_sqlalchemy import SQLAlchemy
from send_email import send_email
from sqlalchemy.sql import func
from werkzeug.utils import secure_filename

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = \
    'postgresql://postgres:1992@localhost/height_collector'
db = SQLAlchemy(app)
#db.create_all()

class Data(db.Model):
    __tablename__ = "data"
    id = db.Column(db.Integer, primary_key=True)
    email_ = db.Column(db.String(120), unique=True)
    height_ = db.Column(db.Integer)

    def __init__(self, email_, height_):
        self.email_ = email_
        self.height_ = height_

@app.route("/")
def index():
    return render_template('index.html')

@app.route("/success", methods=['POST'])
def success():
    global file
    if request.method == "POST":
        file = request.files["file"]
        file.save(secure_filename("uploaded"+file.filename))
        with open("uploaded"+file.filename, 'a') as f:
            f.write("This was added later!")
        print(file)
        print(type(file))
        return render_template("index.html", btn="download.html")

@app.route('/download')
def download():
    return send_file("uploaded"+file.filename, attachment_filename="yourfile.csv", as_attachment=True)

if __name__ == '__main__':
    app.debug = True
    app.run(port = 5001)
```

Index.html

```html
<!DOCTYPE html>
<html lang="en">
    <title>Data Collector App</title>
    <head>
        <link href="../static/main.css" rel="stylesheet">
    </head>
    <body>
        <div class="container">
            <h1>Collecting height</h1>
            <h3>Please fill the entries to get population statistics on height</h3>
            <div class="message">
                {{text | safe}}
            </div>
            <form action="{{url_for('success')}}"
                method="POST" enctype="multipart/form-data">
                <input type="file", name="file" > <br>
                <button type="submit"> Submit </button>
            </form>
            {% if btn %}
                {% include btn %}
            {% endif %}
        </div>
    </body>
</html>
```

Success.html

```html
<!DOCTYPE html>
<html lang="en">
    <title>Data Collector App</title>
    <head>
```

```
        <link href="../static/main.css" rel="stylesheet">
    </head>
    <body>
        <div class="container">
            <p>Thank you for your submmisin! <br>
                You will receive an email with the survey results shortly.
            </p>
        </div>
    </body>
</html>
```

Download.html

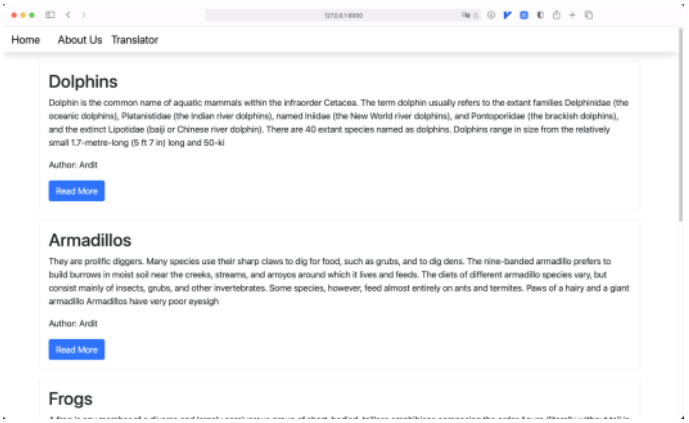
```
<!DOCTYPE html>
<html lang="en">
<div class="download">
<a href={{url_for('download')}} target="blank"> <button class="btn">  Download </button></a>
</div>
</html>
```

==========

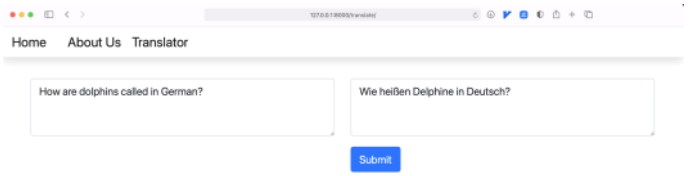
Section 32: App 9: Django & Bootstrap Blog and Translator App

Demo of the App

Welcome to a new Python app! This one will be a Django website which has two features. It has a blog feature with an admin interface through which content creators can write and publish new blog posts to the website. The website also has a translator app where people can translate text from one language to another. Here is how the website will look like after you code it:



And here is how the finished translator tool:



Let's get started!

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/28400318#questions>

在本地安装虚拟环境 python -m venv env
激活本地虚拟环境，安装django

初始化django：
django-admin startproject mysite .

Project name name是mysite, . Means 在当前文件夹下生成mysite

启动server:
Python manage.py runserver

根据warning, 执行： （this will create database and tables, djangon默认使用SQLLite3,可以用其他的sql）
python manage.py migrate
生成db.dqlite3

可以安装 db browser打开查看

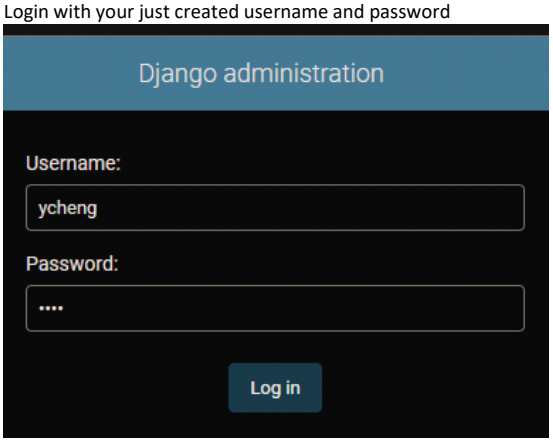
274. Creating a Superuser for the Project

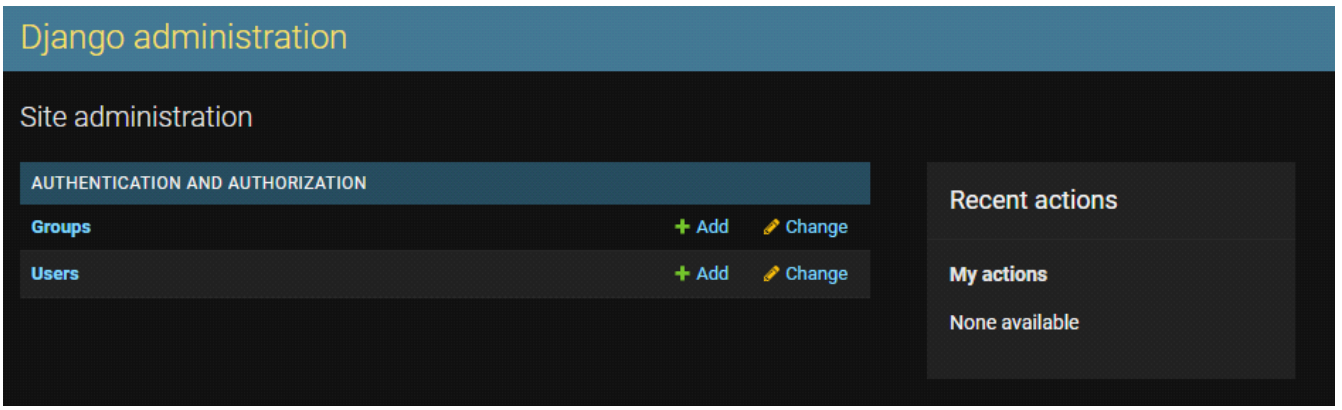
Create superuser:
Python manage.py createsuperuser

```
a_Course\sec32_App9 Django_Bootstrap_Blog_Translator> python manage
.py createsuperuser
Username (leave blank to use 'ychen'): ycheng
Email address: ycheng22@hotmail.com
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
This password is entirely numeric.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

Password: 1992

启动server,
Python manage.py runserver
Go to http://127.0.0.1:8000/admin
Login with your just created username and password





275. Setting up an Empty Django Blog App

生成django app, name是blog
python manage.py startapp blog

A folder blog is created,

在mysite/settings.py添加 'blog'

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'blog',
]
```

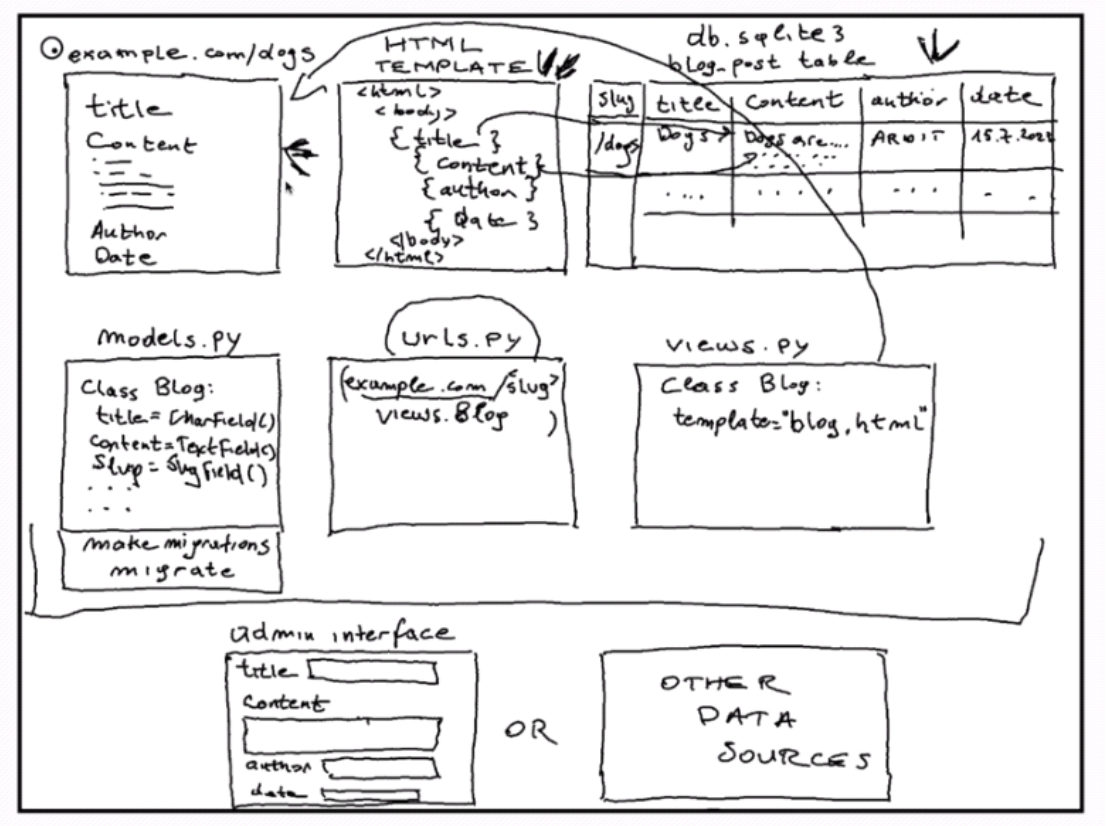
275. Setting up an Empty Django Blog App

修改models.py

```
from django.db import models
from django.contrib.auth.models import User

STATUS = ((0, 'Draft'), (1, 'Publish'))
# Create your models here.

class Post(models.Model):
    title = models.CharField(max_length=200)
    content = models.TextField()
    date_created = models.DateTimeField(auto_now_add=True)
    slug = models.SlugField(max_length=200, unique=True)
    #to means from other database,
    #on_delete=models.CASCADE means the post will be deleted if the use was deleted from database
    author = models.ForeignKey(to=User, on_delete=models.CASCADE)
    status = models.IntegerField(choices=STATUS, default=0)
```

之后运行
python manage.py makemigrations
python manage.py migrate

277. Overview of the Web App Architecture



278. HTML Templates

在本文件夹下新建templates, 在其下新建blog.html

Blog.html

```
<!DOCTYPE html>
<html>
    <body>
        <h1>{{object.title}}</h1>
        <p>{{object.content}}</p>
        <p>{{object.author}}</p>
    </body>
</html>
```

在mysite/settings.py添加：

TEMPLATES_DIR = os.path.join(BASE_DIR, 'templates')

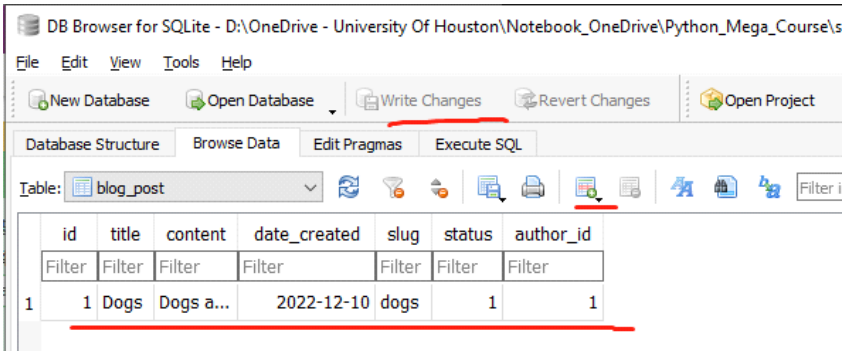
279. Django Views

修改blog/views.py

```
from django.shortcuts import render
from .models import Post

# Create your views here.

class BlogView:
    model = Post
    template_name = 'blog.html'
```

280. URL Patterns

在database中添加一行记录

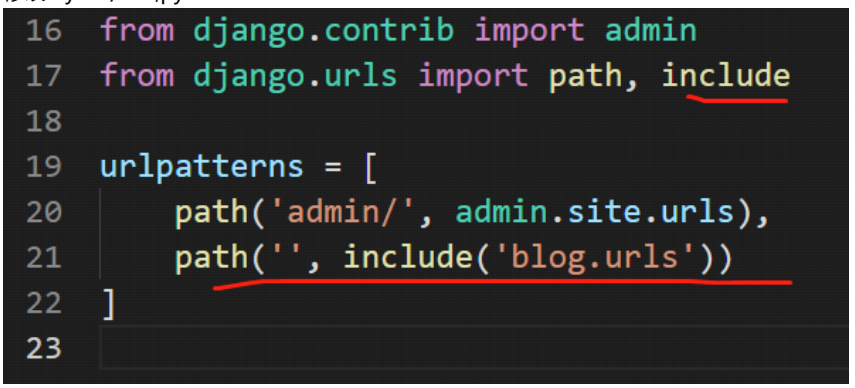


在blog新建urls.py

```
from . import views
from django.urls import path

urlpatterns = [
    path('<slug:slug>', views.BlogView.as_view(), name='blog_view')
]
```

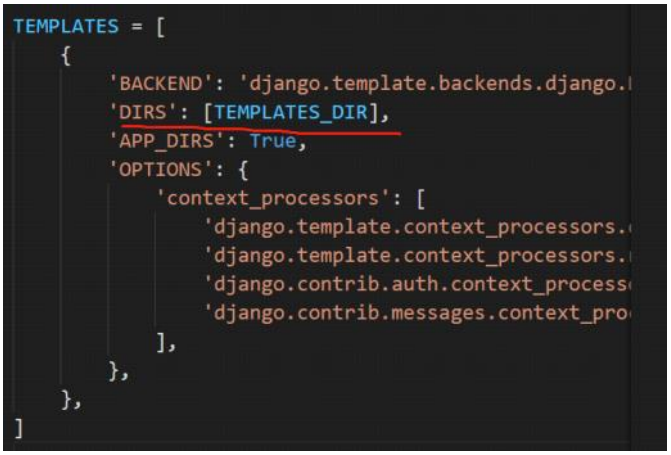
```
path('<slug:slug>' will search slug in each row in the database
```

修改mysite/urls,py



```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('blog.urls'))
]
```

修改mysite/settings.py：



```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.
        'DIRS': [TEMPLATES_DIR],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.
                'django.template.context_processors.
                'django.contrib.auth.context_process
                'django.contrib.messages.context_pro
            ],
        },
    },
]
```

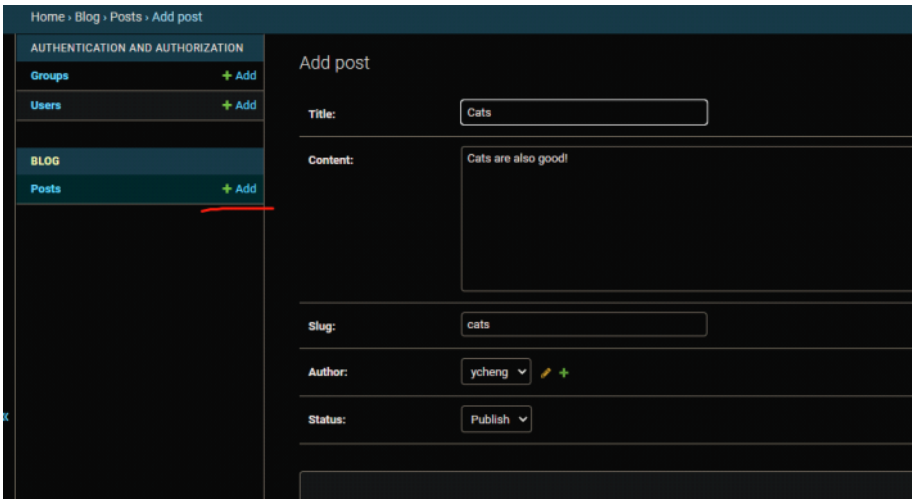
281. Creating Admin Interface Views

Revise blog/admin.py

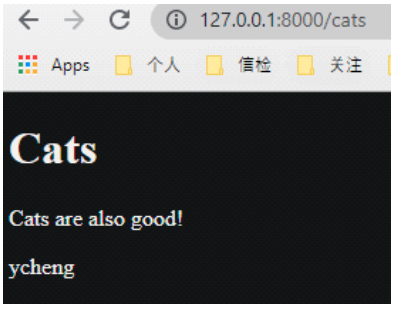
```
from django.contrib import admin
from .models import Post

# Register your models here.
admin.site.register(Post)
```

进入http://127.0.0.1:8000/admin/

Note: amin后面必须带/， 否则page not found

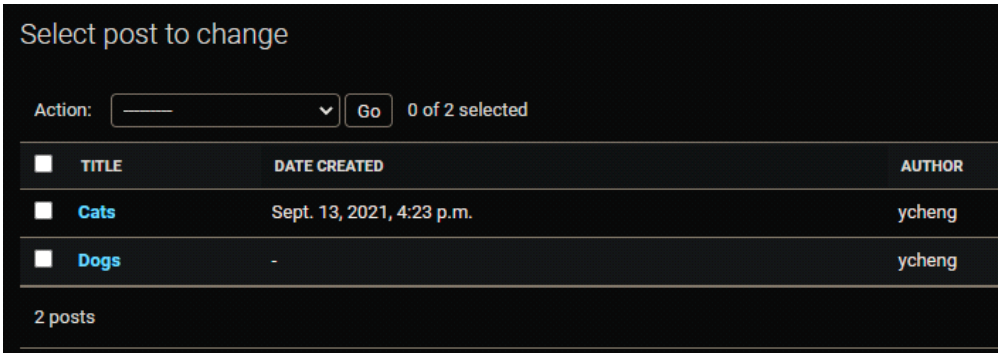


Save



Revise blog/admin.py

```
from django.contrib import admin
from .models import Post

class PostAdmin(admin.ModelAdmin):
    list_display = ('title', 'date_created', 'author')

# Register your models here.
admin.site.register(Post, PostAdmin)
```



282. Creating a Homepage

Create index.html under templates:

```
<!DOCTYPE html>
<html>
    <body>
        This is the homepage!
    </body>
```

```
</html>
```

Revise blog/urls.py

```
from . import views
from django.urls import path

urlpatterns = [
    path('<slug:slug>', views.BlogView.as_view(), name='blog_view'),
    path('', views.HomeView.as_view(), name='home_view')
]
```

Review blog/views.py

```
from django.shortcuts import render
from .models import Post
from django.views import generic

# Create your views here.

class BlogView(generic.DetailView):
    model = Post
    template_name = 'blog.html'

#TemplateView used when you only need to render a
#template without getting data from model
class HomeView(generic.TemplateView):
    template_name = 'index.html'
```

284. Listing Blog Posts on the Homepage

Index.html

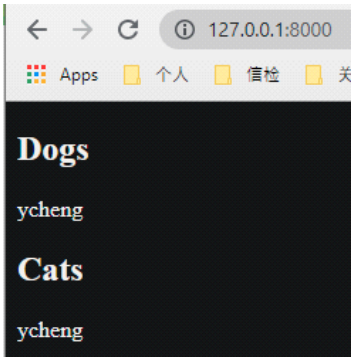
```
<!DOCTYPE html>
<html>
    <body>
        {% for post in post_list %}
            <h2>{{post.title}}</h2>
            <p>{{post.author}}</p>
        {% endfor %}
    </body>
</html>
```

Urls.py

```
from . import views
from django.urls import path

urlpatterns = [
    path('<slug:slug>', views.BlogView.as_view(), name='blog_view'),
    path('about/', views.AboutView.as_view(), name='home_view'),
    path('', views.PostList.as_view(), name='home')
]
```

Views.py

```
from django.shortcuts import render
from .models import Post
from django.views import generic

# Create your views here.

class BlogView(generic.DetailView):
    model = Post
    template_name = 'blog.html'

#TemplateView used when you only need to render a
#template without getting data from model
class AboutView(generic.TemplateView):
    template_name = 'about.html'

#order_by('-date_created'), - means reversed order
class PostList(generic.ListView):
    queryset = Post.objects.filter(status=1).order_by('-date_created')
    template_name = 'index.html'
```



285. Creating Links

Revise blog/index.html:

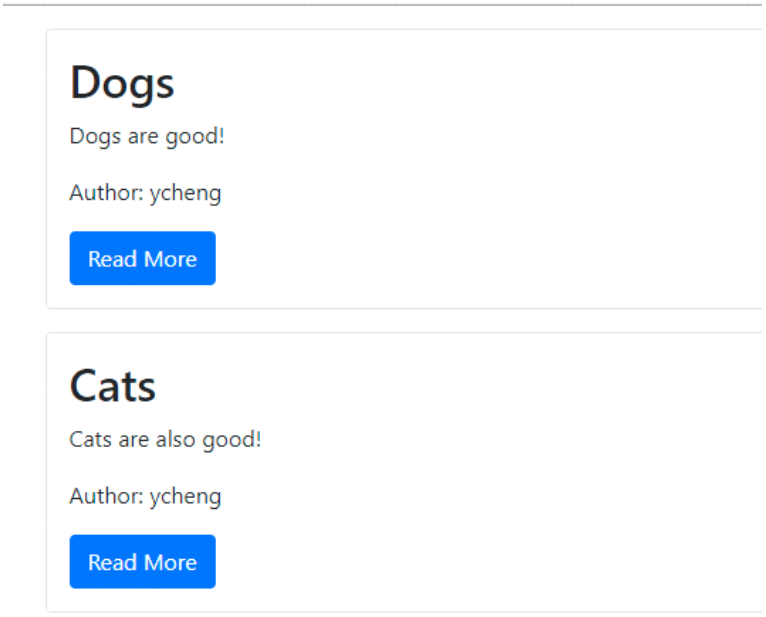
```
<!DOCTYPE html>
<html>
    <body>
        {% for post in post_list %}
            <a href="{% url 'blog_view' post.slug %}">
                <h2>
                    {{post.title}}
                </h2>
            </a>
            <p>{{post.author}}</p>
        {% endfor %}
    </body>
</html>
```

286. Adding Bootstrap to Django

https://getbootstrap.com/docs/5.1/getting-started/introduction/

Revise blog/index.html:

```
<!DOCTYPE html>
<html lang='en'>
    <head>
        <!-- Required meta tags -->
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <!-- CSS only -->
        <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-F3w7mX95PdgyTmZZMECAngseQB83DfGTowi0iMjiWaeVhAn4FJkqJByhZMI3AhiU" crossorigin="anonymous">
        <title>Blog Posts</title>
    </head>
    <body>
        <div class="container">
            {% for post in post_list %}
                <div class="card m-3">
                    <div class="card-body">
                        <h2 class="card-title">
                            {{post.title}}
                        </h2>
                        <p class="card-text">
                            {{post.content}}
                        </p>
                        <p class="card-text">
                            Author: {{post.author}}
                        </p>
                        <a class="btn btn-primary" href="{% url 'blog_view' post.slug %}">
                                Read More
                        </a>
                    </div>
                </div>
            {% endfor %}
        </div>
        <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/js/bootstrap.bundle.min.js"
integrity="sha384-/bQdsTh/da6pkI1MST/rWKFNjaCP5gBSY4sEBT38Q/9RBh9AH40zEOg7Hlq2THRZ" crossorigin="anonymous"></script>
    </body>
</html>
```

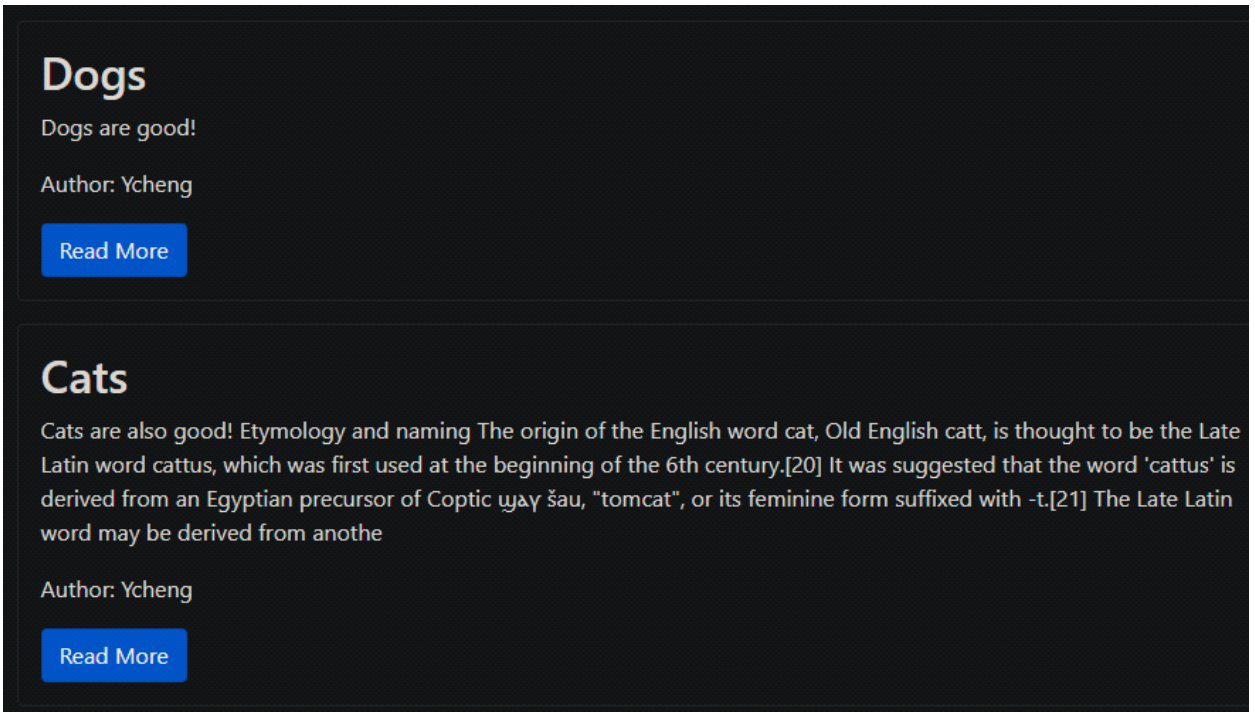


287. Django Template Filters

Goal: 显示content的前几百字，把作者名字首字母大写

Django template filters:
https://docs.djangoproject.com/en/3.2/ref/templates/builtins/

```
<p class="card-text">
    {{post.content | slice:"400"}}
</p>
<p class="card-text">
    Author: {{post.author | title}}
</p>
```

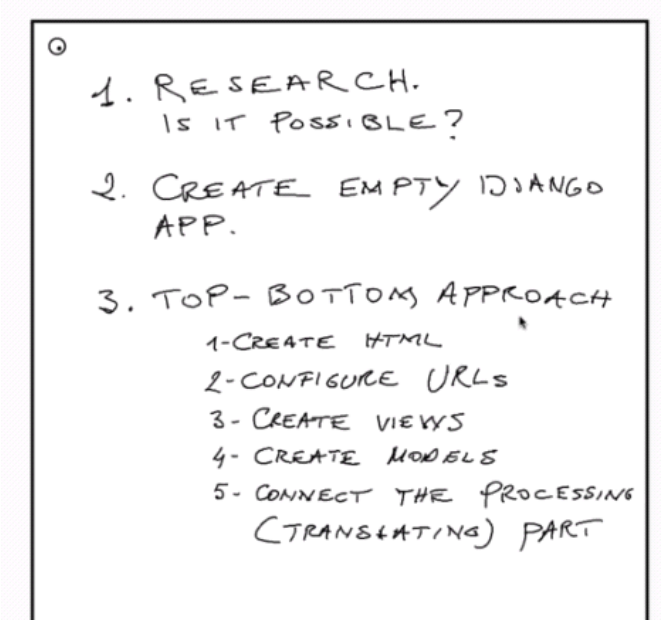


288. Template Inheritance

主要介绍在base.html中，思想是创建base.html,之后在其他html文件中插入改模板
具体见视频

理解
{% block content %}
{% endblock content %}
https://stackoverflow.com/questions/53383602/what-is-block-content-and-endblock-content-for-in-django
https://tutorial.djangogirls.org/zh/template_extending/

291. The Steps of Django App Development



292. Creating an Empty App Structure of the Translator

python manage.py startapp translator

Mysite/settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'blog',
    'translator',
]
```

293. Creating an HTML Form in Django

Add templates/translator.html

```
1  {% extends 'base.html' %}
2  {% block content %}
3  <div class="container">
4      <form action="{% url 'translator_view' %}" method="post">
5          {% csrf_token %}
6          <div class="row">
```

```
            <div class="col-sm-6 mt-3 left">
                <textarea class="form-control" rows="3" name="my_textarea">{{original_text}}</textarea>
            </div>
            <div class="col-sm-6 mt-3 left">
                <textarea class="form-control" rows="3">{{output_text}}</textarea>
                <input class="btn btn-primary ml-3 mt-3" type="submit" value="Submit">
            </div>
        </div>
    </form>
</div>
{% endblock content %}
```

294. Configuring the URLs

In mysite/urls.py, add:

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('blog.urls')),
    path('translate/', include('translator.urls'))
]

```

Create tranlator/urls.py

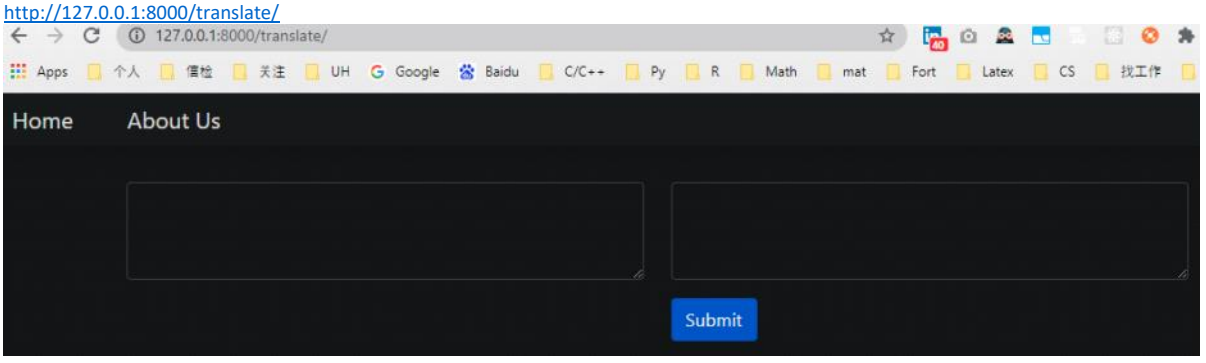
```
from . import views
from django.urls import path

urlpatterns = [
    path('', views.translator_view, name='translator_view')
]
```

295. Creating a Form

Translate/Views.py

```
from django.shortcuts import render

# Create your views here.

def translator_view(request):
    return render(request, 'translator.html')
```

http://127.0.0.1:8000/translate/



296. Getting and Processing User Input Through a Form

Revise tranlate/views.py

```
from django.shortcuts import render

# Create your views here.

def translator_view(request):
    if request.method == 'POST':
        original_text = request.POST['my_textarea']
        output = original_text.upper()
        return render(request, 'translator.html',
            {'output_text':output, 'original_text':original_text})
```

```
    else:
        return render(request, 'translator.html')
```

297. Completing the Translator App

pip install googletran==4.0.0-rc1

Add translate/translate.py

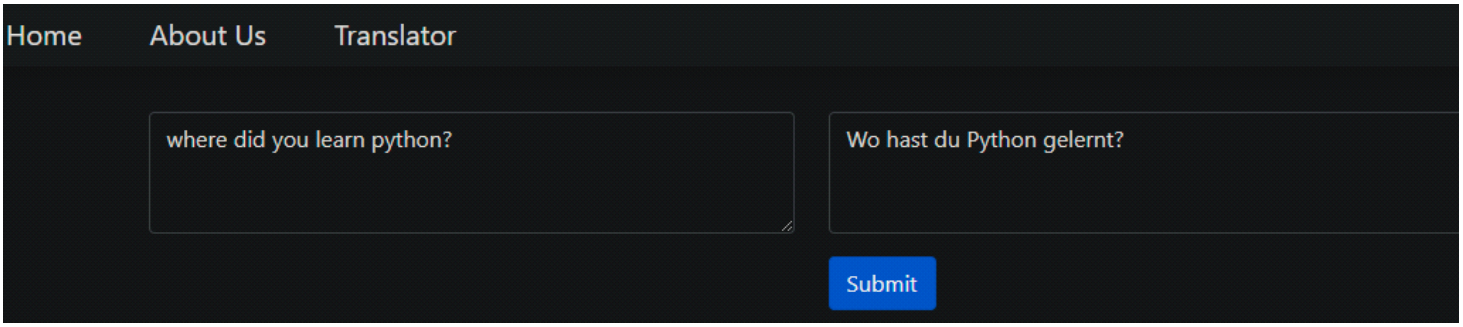
```
from googletrans import Translator

def translate(text):
    translator = Translator()
    translation = translator.translate(text=text, dest='de')
    return translation.text
```

Revise views.py

```
from django.shortcuts import render
from . import translate
# Create your views here.

def translator_view(request):
    if request.method == 'POST':
        original_text = request.POST['my_textarea']
        output = translate.translate(original_text)
        return render(request, 'translator.html',
            {'output_text':output, 'original_text':original_text})
    else:
        return render(request, 'translator.html')
```

Add translator on navbar in base.html

```
<nav class="navbar navbar-expand-lg navbar-light bg-light shadow mb-3" id="mainNav">
            <div class="collapse navbar-collapse" id="navbarResponsive">
                <ul class="navbar-nav ml-auto">
                    <li class="nav-item">
                        <a class="navbar-brand p-3" href="{% url 'home' %}">Home</a>
                    </li>
                    <li>
                        <a class="navbar-brand p-3" href="{% url 'about_view' %}">About Us</a>
                    </li>
                    <li>
                        <a class="navbar-brand p-3" href="{% url 'translator_view' %}">Translator</a>
                    </li>
                </ul>
            </div>
        </nav>
```



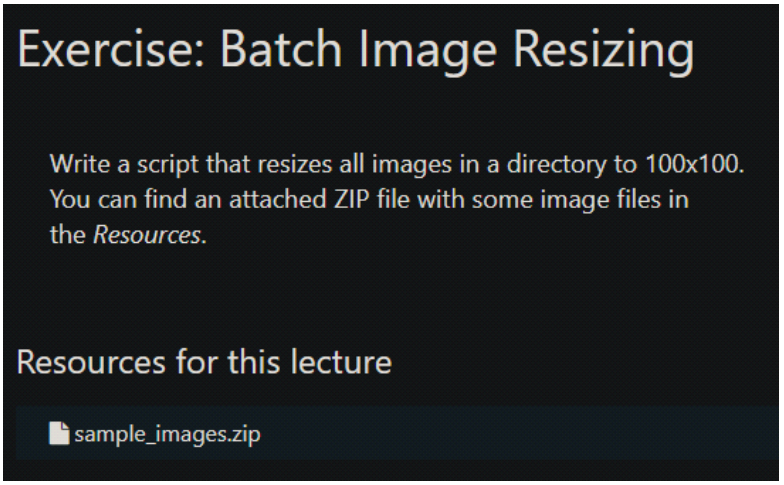
==========

Section 33: App 10: (Project) - Build a Geocoder Web App with Flask and Pandas

==========

Section 17: Image and Video Processing with Python

The order of the sections 17 and 18 were revised by author.



Solution: Batch Image Resizing

I first created a list containing the image file paths and then iterated through the aforementioned list.

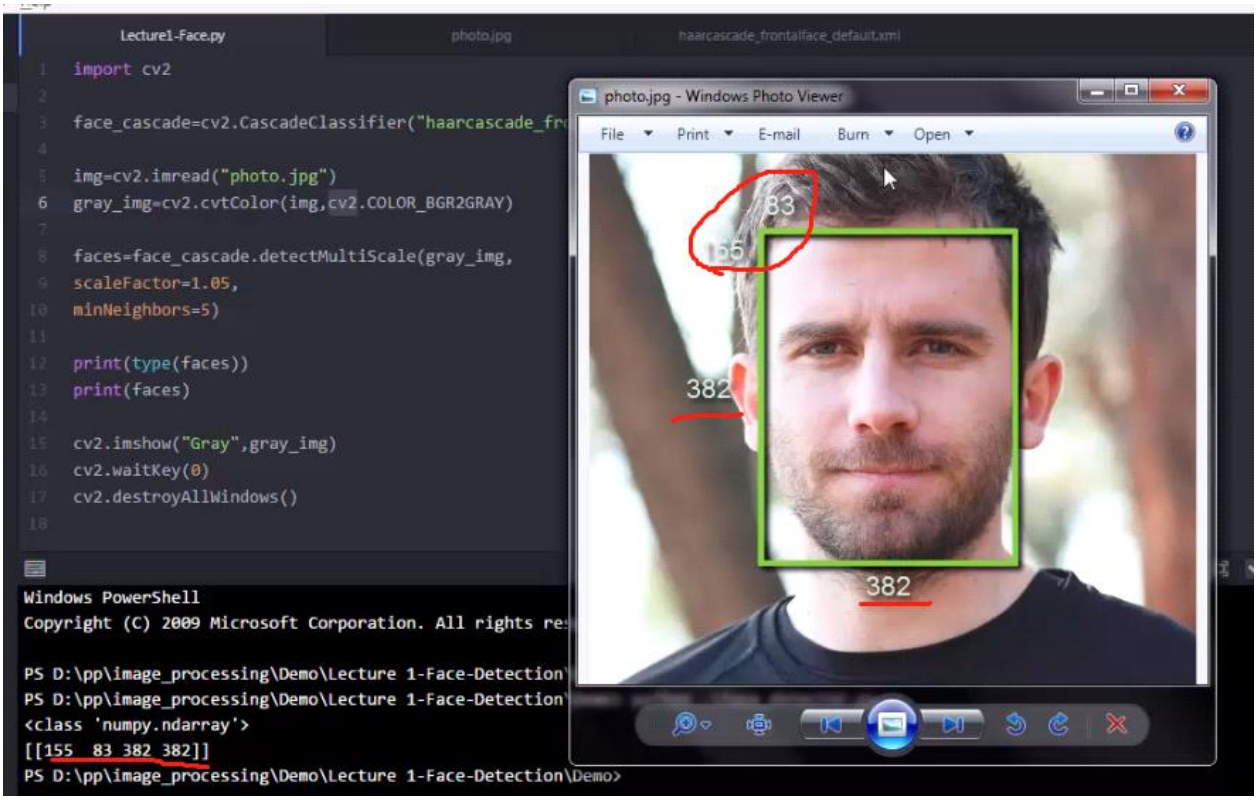
The loop: reads each image, resizes, displays the image, waits for the user input key, closes the window once the key is pressed, and writes the resized image. The name of the resized image will be "resized" plus the existing file name of the original image.

```
import cv2
import glob

images=glob.glob("*.jpg")

for image in images:
    img=cv2.imread(image,0)
    re=cv2.resize(img,(100,100))
    cv2.imshow("Hey",re)
    cv2.waitKey(500)
    cv2.destroyAllWindows()
    cv2.imwrite("resized_"+image,re)
```

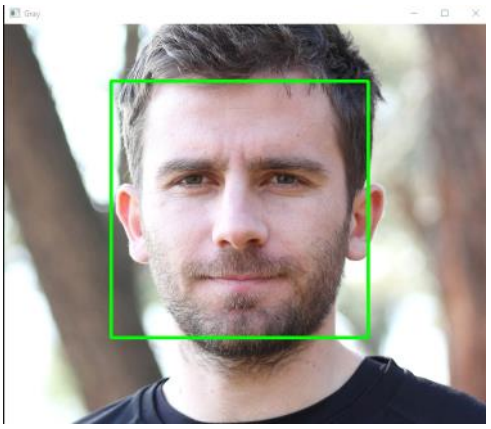
From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/5214910#overview>

141. Detecting Faces in Images

Face_cascade 会探测face的左上角坐标，以及宽度，高度




```
for x, y, w, h in faces:
    #(x,y) is left upper corner, (x+w,y+h) right bottom coner,
    #(0,255,0) line color, 3 line width
    img = cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 3)
```



face_detector.py

```
import cv2

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

img = cv2.imread('news.jpg')
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

faces = face_cascade.detectMultiScale(gray_img,
                                      scaleFactor=1.05,
                                      minNeighbors=5)

for x, y, w, h in faces:
    #(x,y) is left upper corner, (x+w,y+h) right bottom coner,
    #(0,255,0) line color, 3 line width
    img = cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 3)

```

```
print(type(faces))
print(faces)

resized = cv2.resize(img, (int(img.shape[1]/3), int(img.shape[0]/3)))

cv2.imshow('Gray', img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

效果不是很好，可以尝试调整参数

```
scaleFactor=1.05, minNeighbors=5
```

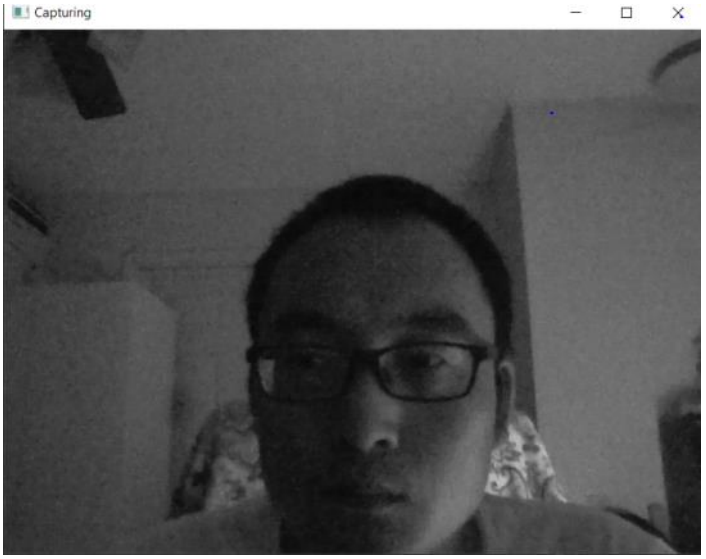
142. Capturing Video with Python

Learn cv2.VideoCapture

https://docs.opencv.org/4.5.1/dd/d43/tutorial_py_video_display.html

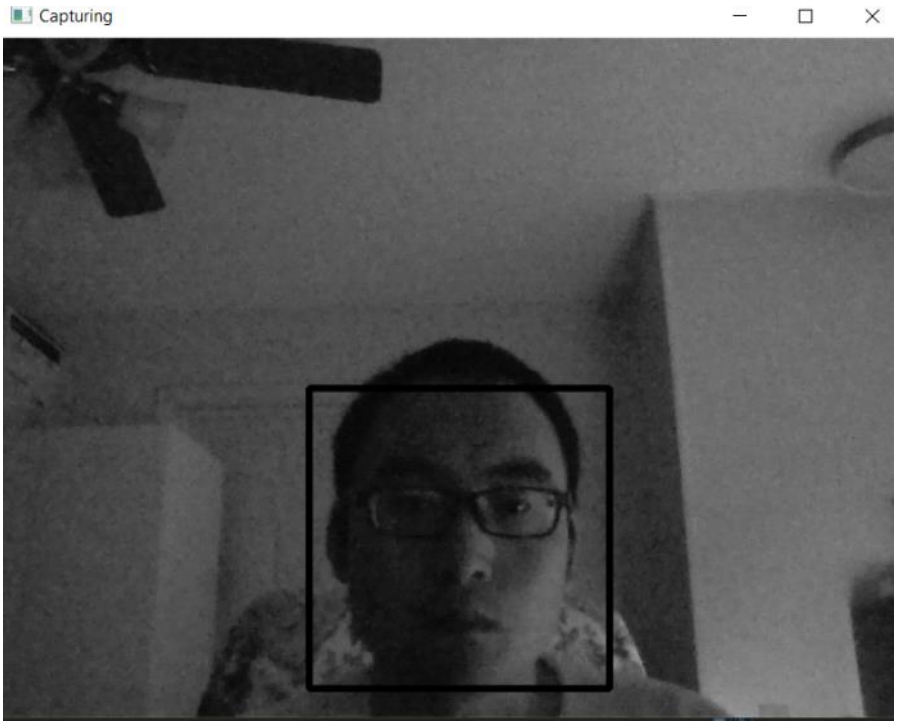
Capture.py

```
import cv2, time

video = cv2.VideoCapture(0)
#specify a number means use which camera,
#or specify the file name, eg:'movie.mp4
a = 1
while True:
    a += 1
    check, frame = video.read()

    print(check)
    print(frame)

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    #time.sleep(3)
    cv2.imshow('Capturing', gray)

    key = cv2.waitKey(1)
    if key == ord('q'):
        break

print(a)
video.release()
cv2.destroyAllWindows()
```



capture_with_rectangle.py

```
import cv2, time
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')


video = cv2.VideoCapture(0)
#specify a number means use which camera,
#or specify the file name, eg:'movie.mp4
a = 1
while True:
    a += 1
    check, frame = video.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = face_cascade.detectMultiScale(gray,
                                          scaleFactor=1.05,
                                          minNeighbors=5)
    for x, y, w, h in faces:
        img = cv2.rectangle(gray, (x,y), (x+w,y+h), (0,255,0), 3)

    print(check)
    print(frame)

    #time.sleep(3)
    cv2.imshow('Capturing', img)

    key = cv2.waitKey(1)
    if key == ord('q'):
        break

print(a)
video.release()
cv2.destroyAllWindows()
```



==========

Section 18: App 2: Controlling the Webcam and Detecting Objects

144. Detecting Moving Objects from the Webcam

Learn cv2.threshold:
Image Thresholding From <https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html>

Learn cv2.dilate:
https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html

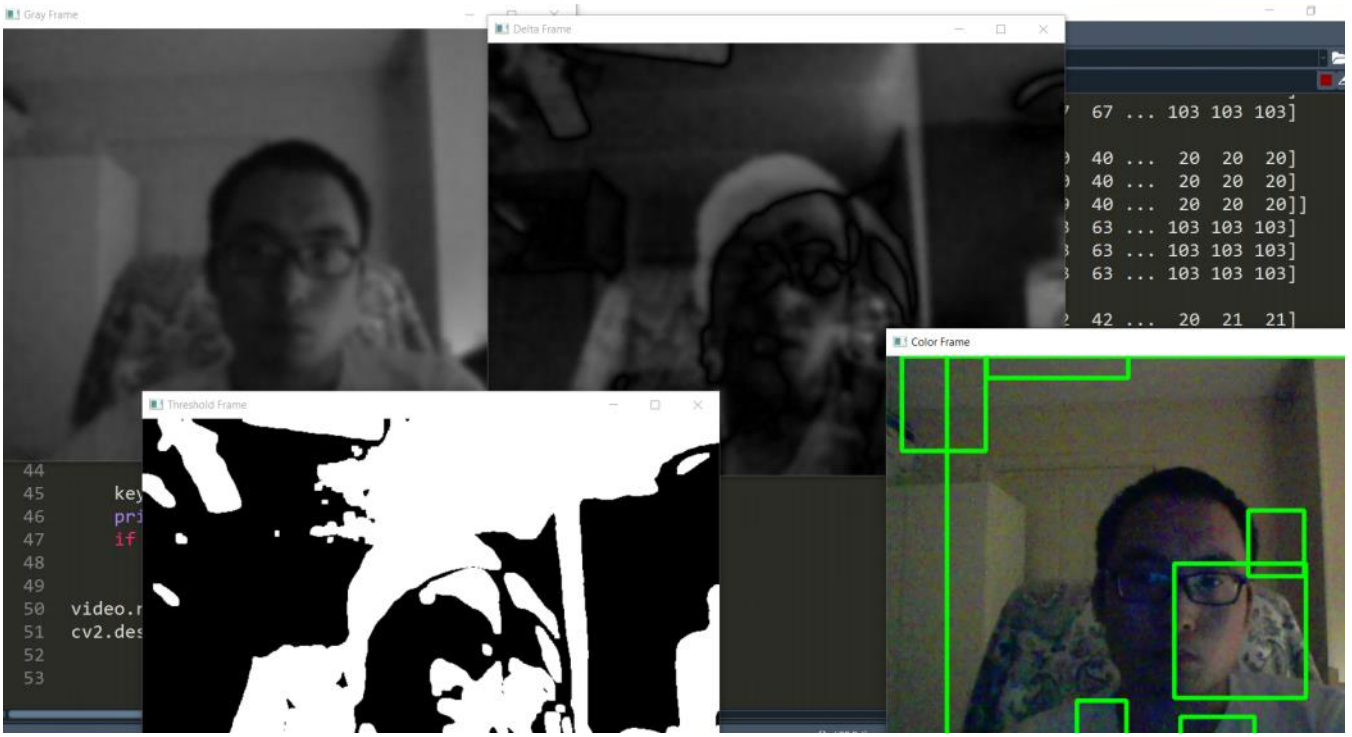
Learn cv2.findContours
https://docs.opencv.org/master/d4/d73/tutorial_py_contours_begin.html
https://pythonexamples.org/python-opencv-cv2-find-contours-in-image/

Learn cv2.boundingRect
https://docs.opencv.org/3.4/dd/d49/tutorial_py_contour_features.html
https://www.pythonpool.com/cv2-boundingrect/

Motion_detector.py

```
import cv2, time

first_frame = None

video = cv2.VideoCapture(0)

while True:
    check, frame = video.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    gray = cv2.GaussianBlur(gray, (21,21), 0)

    if first_frame is None:
        first_frame = gray
        continue

    delta_frame = cv2.absdiff(first_frame, gray)
    thresh_frame = cv2.threshold(delta_frame, 30, 255, cv2.THRESH_BINARY)[1]
    thresh_frame = cv2.dilate(thresh_frame, None, iterations=2)

    (contours, _) = cv2.findContours(thresh_frame.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    for contour in contours:
        if cv2.contourArea(contour) < 1000:
            continue
        (x, y, w, h) = cv2.boundingRect(contour)
        cv2.rectangle(frame, (x,y), (x+w,y+h), (0,255,0), 3)
```

```
        cv2.rectangle(frame, (x,y), (x+w,y+h), (0,255,0), 3)

    cv2.imshow('Gray Frame', gray)
    cv2.imshow('Delta Frame', delta_frame)
    cv2.imshow('Threshold Frame', thresh_frame)
    cv2.imshow('Color Frame', frame)

    key = cv2.waitKey(1)
    print(gray)
    if key == ord('q'):
        break

video.release()
cv2.destroyAllWindows()
```



145. Storing Object Detection Timestamps in a CSV File

Motion_detector.py

```
# -*- coding: utf-8 -*-
"""
Created on Mon Sep 13 23:28:14 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com

@Github: https://github.com/ycheng22

Reference:
"""
import cv2, time
from datetime import datetime
import pandas as pd

first_frame = None
status_list = [None, None]
times = []
df = pd.DataFrame(columns=["Start", "End"])

video = cv2.VideoCapture(0)

while True:
    check, frame = video.read()
    status = 0

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    gray = cv2.GaussianBlur(gray, (21,21), 0)

    if first_frame is None:
        first_frame = gray
```

```
        continue

    delta_frame = cv2.absdiff(first_frame, gray)
    thresh_frame = cv2.threshold(delta_frame, 30, 255, cv2.THRESH_BINARY)[1]
    thresh_frame = cv2.dilate(thresh_frame, None, iterations=2)

    (contours, _) = cv2.findContours(thresh_frame.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    for contour in contours:
        if cv2.contourArea(contour) < 10000:
            continue
        status = 1
        (x, y, w, h) = cv2.boundingRect(contour)
        cv2.rectangle(frame, (x,y), (x+w,y+h), (0,255,0), 3)
    status_list.append(status)
    if status_list[-1] == 1 and status_list[-2] == 0:
        times.append(datetime.now())
    if status_list[-1] == 0 and status_list[-2] == 1:
        times.append(datetime.now())
    cv2.imshow('Gray Frame', gray)
    cv2.imshow('Delta Frame', delta_frame)
    cv2.imshow('Threshold Frame', thresh_frame)
    cv2.imshow('Color Frame', frame)

    key = cv2.waitKey(1)
    if key == ord('q'):
        #when quiting, if last frame status is 1, record it
        #otherwise, you will lose the last one time record
        if status == 1:
            times.append(datetime.now())
        break
print(status_list)
print(times)
for i in range(0, len(times), 2):
    df = df.append({"Start": times[i], "End": times[i+1]}, ignore_index=True)
df.to_csv("Times.csv")
video.release()
cv2.destroyAllWindows()
```

status_list and times:

```
[None, None, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0]
[datetime.datetime(2021, 9, 14, 9, 39, 58, 909246), datetime.datetime(2021, 9, 14, 9,
40, 1, 211188), datetime.datetime(2021, 9, 14, 9, 40, 2, 351139),
datetime.datetime(2021, 9, 14, 9, 40, 2, 716163), datetime.datetime(2021, 9, 14, 9,
40, 3, 309578), datetime.datetime(2021, 9, 14, 9, 40, 5, 149656)]
```

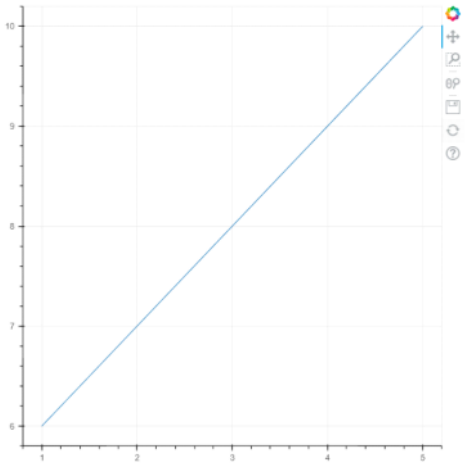
==========

Section 19: Interactive Data Visualization with Python and Bokeh

148. Your First Bokeh Plot

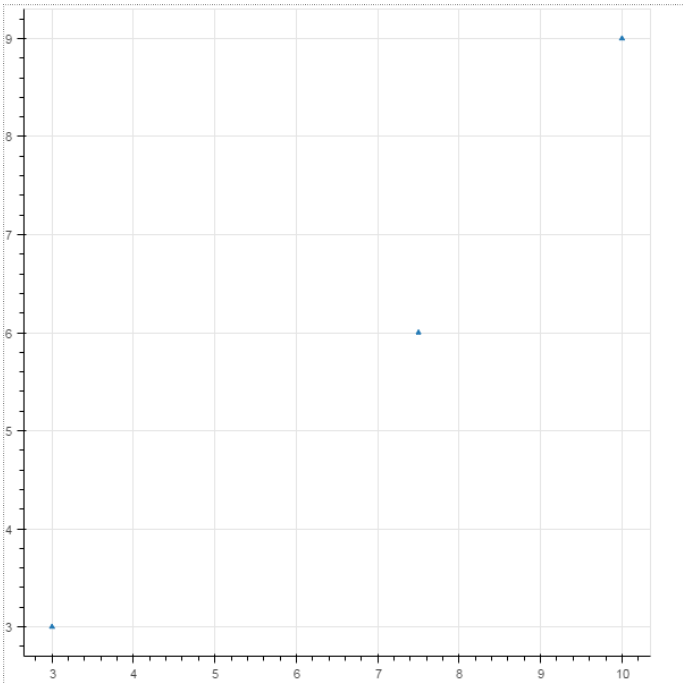
```
#making a basic Bokeh line graph
from bokeh.io import output_file, show
from bokeh.plotting import figure


x = [1,2,3,4,5]
y = [6,7,8,9,10]

output_file("Line.html")

f = figure()

f.line(x, y)
```

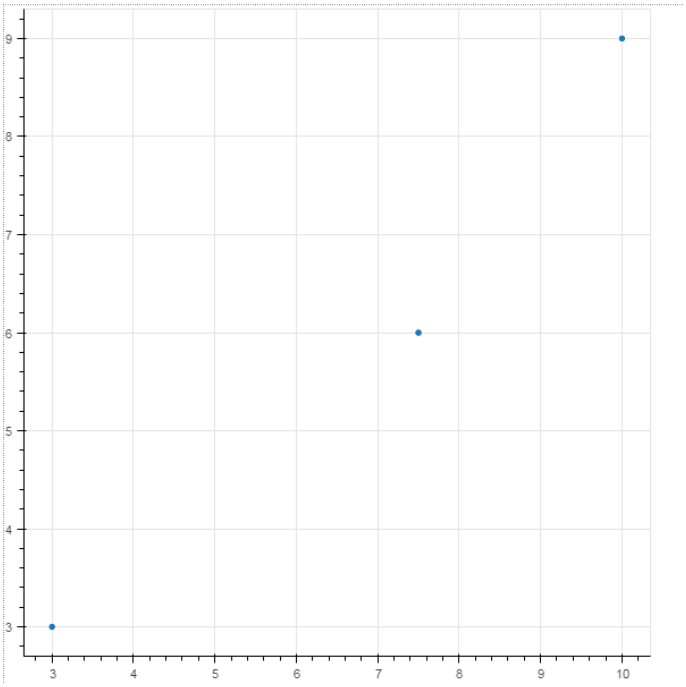
```
15
show(f)
```



Exercise: Plotting Triangles and Circles

Write two code snippets, each producing the following graphs. The first graph has triangles as glyphs and the second graph has circles as glyphs. You can use triangle , and circle instead of line. You should have the same coordinates, as shown in the plots below.





From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/9439116#overview>

Solution: Plotting Triangles and Circles

#Snippet producing the triangle-based plot

```
#Making a basic Bokeh line graph

#importing Bokeh
from bokeh.plotting import figure
from bokeh.io import output_file, show

#prepare some data
x=[3,7.5,10]
y=[3,6,9]

#prepare the output file
output_file("Line.html")

#create a figure object
f=figure()

#create line plot
f.triangle(x,y)

#write the plot in the figure object
show(f)
```

#Snippet producing the circle based plot

```
#Making a basic Bokeh line graph

#importing Bokeh
from bokeh.plotting import figure
from bokeh.io import output_file, show

#prepare some data
x=[3,7.5,10]
```

```
y=[3,6,9]

#prepare the output file
output_file("Line.html")

#create a figure object
f=figure()

#create line plot
f.circle(x,y)

#write the plot in the figure object
show(f)
```

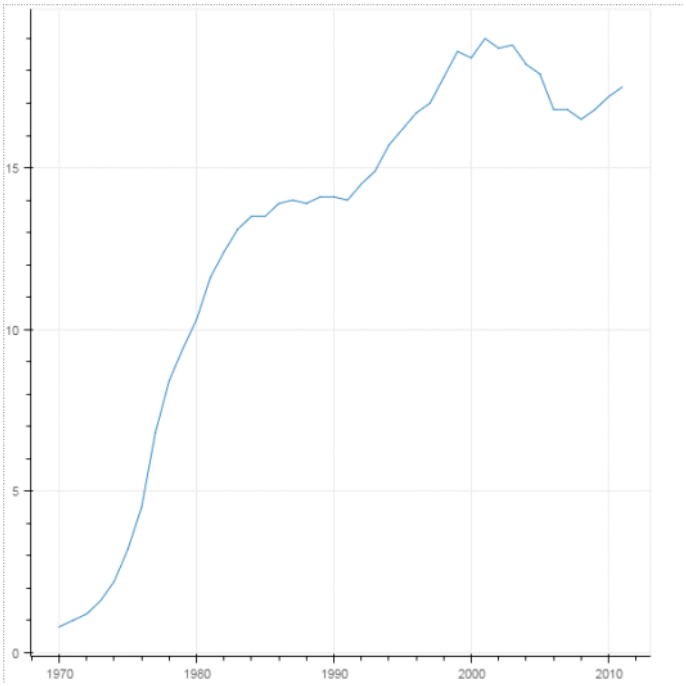
From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/9439122#overview>

Exercise: Plotting Education Data

The following line graph shows the percentage of women who have received a bachelor's degree over the years in the USA. The graph was produced from the `Year` and `Engineering` columns of the CSV file provided in the following link:

http://pythonhow.com/data/bachelors.csv

Try to reproduce the graph using Bokeh.



From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/9439156#overview>

Solution: Plotting Education Data

```
#Plotting percentage of women who received an engineering degree over years

#importing bokeh and pandas
from bokeh.plotting import figure
from bokeh.io import output_file, show
import pandas

#prepare some data
```

```
df=pandas.read_csv("http://pythonhow.com/data/bachelors.csv")
x=df["Year"]
y=df["Engineering"]

#prepare the output file
output_file("Line_from_bachelors.html")

#create a figure object
f=figure()

#create line plot
f.line(x,y)

#write the plot in the figure object
show(f)
```

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/9439172#overview>

Note on Loading Excel Files

In the next lecture, you will learn how to load Excel files in Python with *pandas*. For this, you need *pandas* (which you have already installed) and also two other dependencies that *pandas* needs for opening Excel files. You can install them with *pip*:

`pip3.9 install openpyxl` (needed to load Excel *.xlsx* files)

`pip3.9 install xlrd` (needed to load Excel old *.xls* files)

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/7828328#overview>

Changing Plot Properties

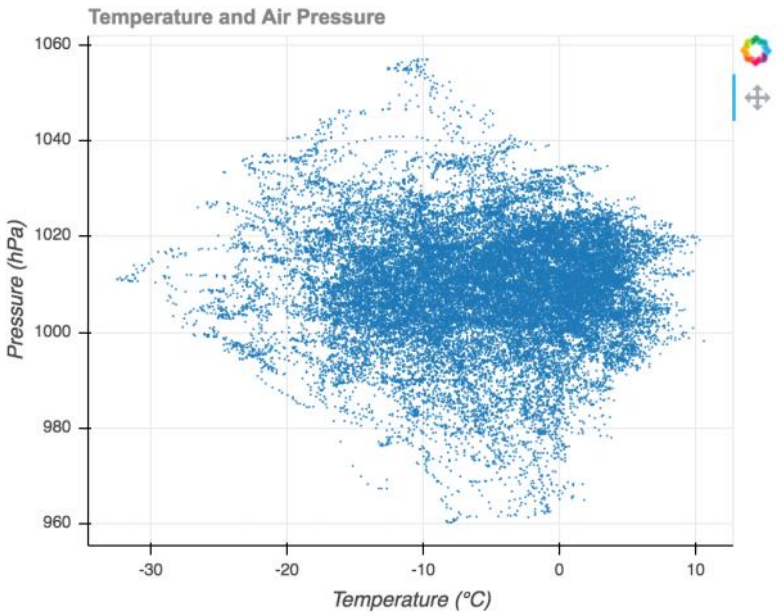
You can add a title to the plot, set the figure width and height, change title font, etc. Below is a summary of properties which can be added to change the style of the plot:

```
import pandas
from bokeh.plotting import figure, output_file, show
p=figure(plot_width=500,plot_height=400, tools='pan',logo=None)
p.title.text="Cool Data"
p.title.text_color="Gray"
p.title.text_font="times"
p.title.text_font_style="bold"
p.xaxis.minor_tick_line_color=None
p.yaxis.minor_tick_line_color=None
p.xaxis.axis_label="Date"
p.yaxis.axis_label="Intensity"
p.line([1,2,3],[4,5,6])
output_file("graph.html")
show(p)
```

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/10398136#overview>

Exercise: Plotting Weather Data

Produce the following graph using the data from this Excel file: http://pythonhow.com/data/verlegenhuken.xlsx



Some notes:

Temperature and pressure values in the Excel file have a scale factor of 10; you'll have to divide those values by 10 to get the actual observations.

And, yes, you can set your own fonts and colors, but be accurate with the rest of the plot elements.

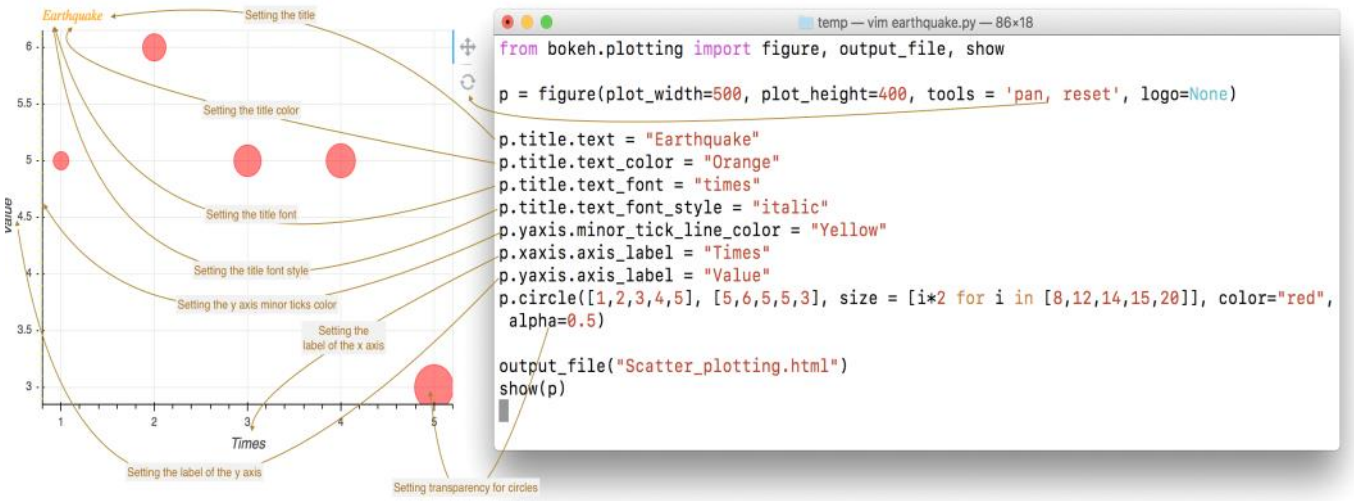
Solution: Plotting Weather Data

```
import pandas

from bokeh.plotting import figure, output_file, show

df=pandas.read_excel("http://pythonhow.com/data/verlegenhuken.xlsx",sheet_name=0)
df["Temperature"]=df["Temperature"]/10
df["Pressure"]=df["Pressure"]/10

p=figure(plot_width=500,plot_height=400,tools='pan')

p.title.text="Temperature and Air Pressure"
p.title.text_color="Gray"
p.title.text_font="arial"
p.title.text_font_style="bold"
p.xaxis.minor_tick_line_color=None
p.yaxis.minor_tick_line_color=None
p.xaxis.axis_label="Temperature (° C)"
p.yaxis.axis_label="Pressure (hPa)"

p.circle(df["Temperature"],df["Pressure"],size=0.5)
output_file("Weather.html")
show(p)
```

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/5220348#overview>

Changing Visual Attributes

Once you have built a basic plot, you can customize its visual attributes, including changing the title color and font, adding labels for xaxis and yaxis, changing the color of the axis ticks, etc. All these properties are illustrated in the diagram below:



And here is the code if you want to play around with it:

```
from bokeh.plotting import figure, output_file, show
p = figure(plot_width=500, plot_height=400, tools = 'pan, reset')
p.title.text = "Earthquakes"
p.title.text_color = "Orange"
p.title.text_font = "times"
p.title.text_font_style = "italic"
p.yaxis.minor_tick_line_color = "Yellow"
p.xaxis.axis_label = "Times"
p.yaxis.axis_label = "Value"
p.circle([1,2,3,4,5], [5,6,5,5,3], size = [i*2 for i in [8,12,14,15,20]], color="red", alpha=0.5)
output_file("Scatter_plotting.html")
show(p)
```

For a complete list of visual attributes, see the [Styling Visual Attributes] documentation page of Bokeh.

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/9439608#overview>

159. Creating a Time-series Plot

```
df = pd.read_csv("http://www.google.com/finance/historical?q=NASDAQ:ADBE&startdate=Jan+01%2C+2009&enddate=Aug+2%2C+2012
&output=csv", parse_dates=["Date"])

p = figure(plot_width=500, plot_height=500, x_axis_type="datetime")

p.line(df['Date'], df['Close'], color='Orange', alpha=0.5)

output_file("Timeseries.html")
show(p)
```

这个有bug，

161. Plotting Time Intervals from the Data Generated by the Webcam App

./sec18_App2 Controlling the Webcam/Plotting.py

```
#from motion_detector import df
#from bokeh.plotting import figure, show, output_file
from bokeh.io import output_file, show
from bokeh.plotting import figure

import pandas as pd
df = pd.read_csv('Times.csv')

p = figure(x_axis_type ='datetime', plot_width=500, plot_height=400)
#p.yaxis.minor_tick_line_color = None
#p.ygrid[0].ticker.desired_num_ticks = 1
p.quad(left=df['Start'], right=df['End'], bottom=0, top=1, color='Green')

output_file("Graph.html")
show(p)
```

有bug

162. Implementing a Hover Feature

```
from motion_detector import df
from bokeh.plotting import figure, show, output_file
from bokeh.models import HoverTool, ColumnDataSource

df["Start_string"] = df['Start'].dt.strftime("%Y-%m-%d %H:%M:%S")
df["End_string"] = df['End'].dt.strftime("%Y-%m-%d %H:%M:%S")

cds = ColumnDataSource(df)

import pandas as pd
df = pd.read_csv('Times.csv')

p = figure(x_axis_type ='datetime', plot_width=500, plot_height=400, title="Motion Graph")
p.yaxis.minor_tick_line_color = None
p.ygrid[0].ticker.desired_num_ticks = 1

hover = HoverTool(tooltips=[("Start", "@Start_string"), ("End", "@End_string")])
p.add_tools(hover)

p.quad(left='Start', right='End', bottom=0, top=1, color='Green', source=cds)

output_file("Graph.html")
show(p)
```

有bug

==========

Section 35: Bonus App: Building a Website Blocker

Host file on
Windows: C:\Windows\System32\drivers\etc
linux, mac: /etc/hosts

==========

Section 36: Bonus App: Data Visualization Dashboard with Bokeh

331. Downloading Datasets with Python

Pandas_datareader 提供了很多有用的数据接口，值得学习

https://pandas-datareader.readthedocs.io/en/latest/index.html

```
from pandas_datareader import data
from datetime import datetime
```

```
data.DataReader?
```

```
...
```

```
# start = datetime(2016,3,1)
# end = datetime(2016,3,10)
start = datetime(2021,9,10)
end = datetime(2021,9,14)
df = data.DataReader(name='AAPL', data_source="yahoo", start=start, end=end)
```

```
df
```

Date	High	Low	Open	Close	Volume	Adj Close
2021-09-10	155.479996	148.699997	155.000000	148.970001	140646400	148.970001
2021-09-13	151.419998	148.750000	150.630005	149.550003	102404300	149.550003
2021-09-14	151.070007	146.910004	150.350006	148.119995	105455815	148.119995

Mplfinance can plot candlestick

From <https://github.com/matplotlib/mplfinance>

Finance 也可以

From <https://github.com/highfestiva/finplot>

Pandas-Bokeh 看起来更好用

From <https://github.com/PatrikHlobil/Pandas-Bokeh#lineplot>

339. Note

Depending on your version of Bokeh, you may get an `IndexError: list index out of range error` in the next video. If that is the case please see this thread here on how to easily fix the issue.

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/16036746#questions>

340. Embedding the Bokeh Chart in a Webpage

From <https://www.udemy.com/course/the-python-mega-course/learn/lecture/5189318#overview>

把之前做flask presonal web的代码（mysiteDemo）copy到本section文件夹一份，重新安装虚拟环境，
把之前的记录copy如下：

---

187. Creating a Python Virtual Environment

安装virtualenv package
Pip install virtualenv

我们写的文件放在Demo文件夹里面，把Demo放在mysite文件夹里面， **cd到mysite**

添加新的虚拟环境
Python -m venv virtual
激活虚拟环境

查看python
Virtual\Scripts\python
Exit()

Install flask for new virtual env:
Virtual\Scripts\pip install flask
此处还要安装bokeh, pandas, pandas_datareader

Run demo locally with new virtual env:
Virtual\Scripts\python Demo\main.py

189. Deploying the Website to a Live Server

注册heroku账户，
下载安装heroku toolbelt

Cd 到mysite\Demo

Heroku login

Create apps:
heroku create app-name

List your apps:
heroku apps

Website address: app-name.herokuapp.com

Get a list of packages of the new virtual env
..\virtual\Scripts\pip freeze

Install gunicorn
..\virtual\Scripts\pip install gunicorn

Write packages info to requirements.txt
..\virtual\Scripts\pip freeze > requirements.txt

Create Procfile, no file extension
Web: gunicorn script1:app

Create runtime.txt, specify the python version, check the website
https://devcenter.heroku.com/articles/python-runtimes#supported-python-runtimes

python-3.9.6

Still in Demo folder:
Git init
Git add . # . Means files of current folder
Git commit -m "note message"
Heroku git:remote --app app-name #heroku git:remote --app yc-web1
git push heroku master

如何检查login成功：
heroku login

heroku info

做些更改之后
在vs code or Atom里，如果更改之后还没有push，文件会变为黄色
git add .
git commit -m "add changes"
git push heroku master
(如果第一次执行push可能会报错，需要授权
Git config --global user.email "ycheng22@hotmail.com"
Git config --blobal user.name "ycheng22"
完成上面两行，再次
git push heroku master
)

本地测试：

340. Embedding the Bokeh Chart in a Webpage

Main.py

```
# -*- coding: utf-8 -*-
"""
Created on Thu Aug 19 00:51:43 2021

@author: Yunpeng Cheng

@E_mail: ycheng22@hotmail.com

Reference:
"""
#run in cmd: python .\script1.py

from flask import Flask, render_template

app = Flask(__name__)

@app.route('/plot/')
def plot():
    from pandas_datareader import data
    from datetime import datetime
    from bokeh.plotting import figure, show, output_file
    from bokeh.embed import components
    from bokeh.resources import CDN

    start = datetime(2020,5,10)
    end = datetime(2020,6,10)
    df = data.DataReader(name='GOOG', data_source="yahoo", start=start, end=end)
    def inc_dec(c, o):
        if c > o:
            value ="Increase"
        elif c < o:
            value = "Decrease"
        else:
            value = "Equal"
        return value
    df["Status"] = [inc_dec(c, o) for c, o in zip(df.Close, df.Open)]
    df["Middle"] = (df.Open+df.Close)/2
    df["Height"] = abs(df.Open-df.Close)
    p = figure(x_axis_type='datetime', width=1000, height=300, sizing_mode="scale_width")
    p.title.text = "Candlestick Chart"
    p.grid.grid_line_alpha = 0.3

    hours_12 = 12*60*60*1000 #hour, mininute, second, millisecond, total 12 hours
    df_inc = df[df.Status == "Increase"]
    df_dec = df[df.Status == "Decrease"]
    df_equ = df[df.Status == "Equal"]
    p.segment(df.index, df.High, df.index, df.Low, color="black")
    #increase days
    p.rect(df_inc.index, df_inc.Middle, hours_12,
           df_inc.Height, fill_color="#00ff40", line_color="black")
    #decrease days
    p.rect(df_dec.index, df_dec.Middle, hours_12,
           df_dec.Height, fill_color="#FF3333", line_color="black")

    script1, div1 = components(p)
    cdn_js = CDN.js_files[0] #only need first one now
    #cdn_css = CDN.css_files #it's empty,
    return render_template("plot.html", script1=script1,
           div1=div1, cdn_js=cdn_js)

@app.route('/') #http://localhost:5000/
```

```python
@app.route('/') #http://localhost:5000/
def home():
    return render_template("home.html") #must put home.html under folder "template"

@app.route('/about/') #http://localhost:5000/about
def about():
    return render_template("about.html")

if __name__ == "__main__":
    app.run(debug=True)
```

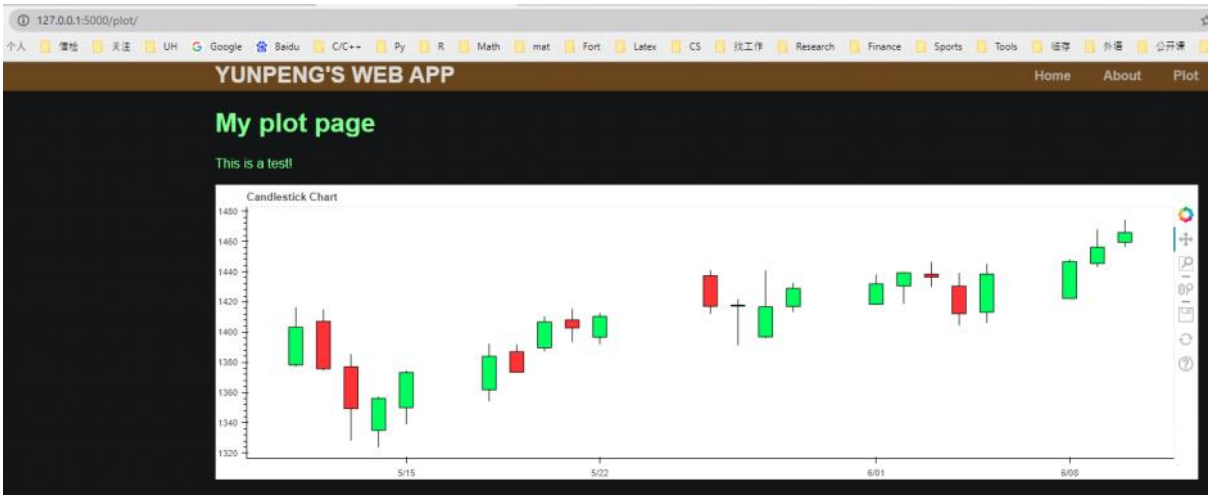
Plot.html

```
{%extends "layout.html"%}
{%block content%}
<script type="text/javascript" src={{cdn_js | safe}}></script>

<div>
    <h1>My plot page</h1>
    <p>This is a test!</p>
</div>

{{script1 | safe}}
{{div1 | safe}}

{%endblock%}
```

Palyout.html加上一行

```
<li><a href="{{ url_for('home') }}">Home</a></li>
<li><a href="{{ url_for('about') }}">About</a></li>
<li><a href="{{ url_for('plot') }}">Plot</a></li>
```

效果如下:

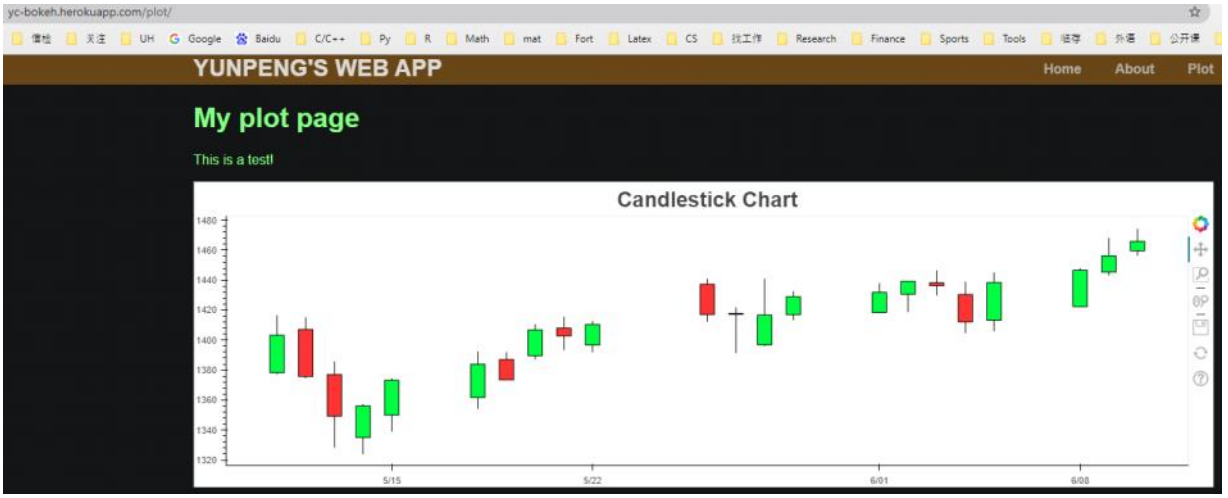


部署到heroku:

341. Deploying the Chart Website to a Live Server

新建Procfile的时候，web: gunicorn script1:app, 冒号和app之间不能有空格，否则即使部署成功， 打开会遇到page error

App home page:
https://yc-bokeh.herokuapp.com/


yc-bokeh.herokuapp.com/plot/
YUNPENG'S WEB APP
Home
About
Plot
My plot page
This is a test!
Candlestick Chart
1480
1460
1440
1420
1400
1380
1360
1340
1320
5/15
5/22
6/01
6/08

==========

==========

==========

==========

==========

==========

==========