Introductory Programming Using Python

Day 2

By Tan Kok Cheng/ Vincent Ng

Republic Polytechnic

Admin Information

1. Software Installation if you have not done so
    - Python 3.8.x
    - Wing IDE
2. Download course material at https://bit.ly/rp-oct20
    - Link can be found in the chat
3. Please change your log in screen name to resemble the registered name for the course
4. Set microphone to muted when you are not speaking

Trainers

Tan Kok Cheng

tan_kok_cheng@rp.edu.sg

Vincent Ng

Vincent_ng@rp.edu.sg

Outline for the day

Time	Agenda
9.00am – 10.30am	
10.30am – 10.45am	Break
10.45am – 12.30pm	
12.30pm – 1.30pm	Lunch
1.30pm – 3.15pm	
3.15pm – 3.30pm	Break
3.30pm – 4.45pm	
4.45pm – 5.00pm	Wrap up, Q&A

Programme Day Two

Morning

- Read and writing files
- Copying, moving and deleting files and folders
- Working with Excel
- Image Processing

Afternoon

- Connecting to the Web
- Sending emails
- Generating PDF

File Paths

Absolute file paths are notated by a **leading forward slash or drive label**.
For example,
`/home/example_user/example_directory`

or

`C:/system32/cmd.exe`

An absolute file path describes how to access a given file or directory, starting from the root of the file system.

Relative file paths are notated by a **lack of a leading forward slash**.
For example,
`example_directory`.

A relative file path is interpreted from the perspective your current working directory. If you use a relative file path from the wrong directory, then the path will refer to a different file than you intend, or it will refer to no file at all..

Read files

```
1  # make sure you have a hello.txt in your current working director
2  # same directory as your python script
3  helloFile = open("hello.txt")
4  content = helloFile.read()
5  print(content)
6  helloFile.close()
7
8  # make sure you have a hello.txt in the specified director
9  # same directory as your python script
10 helloFile = open("hello.txt")
11
12 content = helloFile.readlines()
13 print(content)
14
```

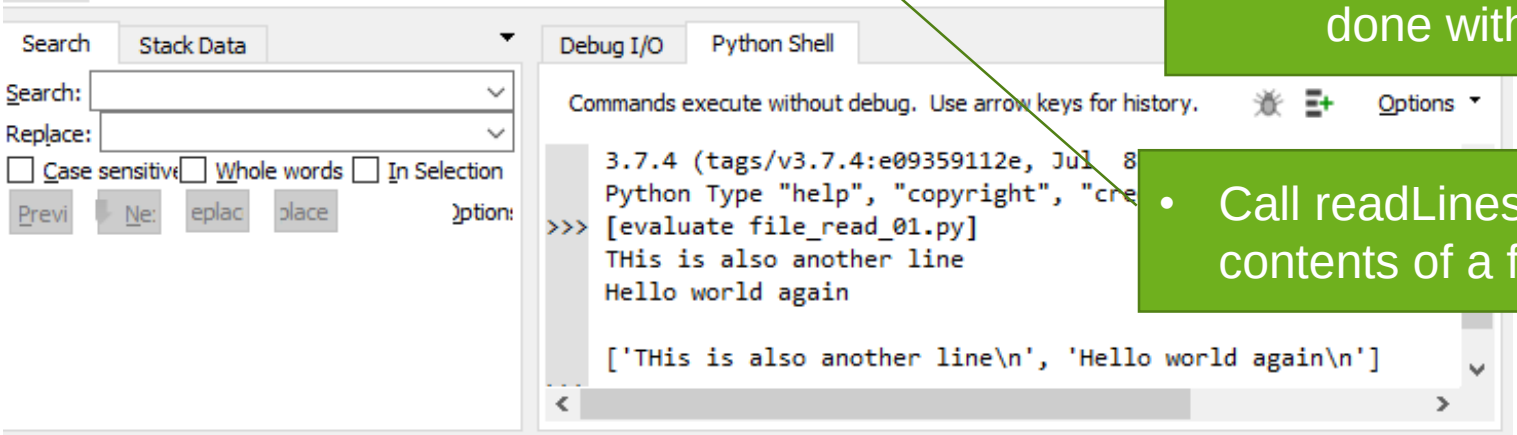
Open() will return a file object which has reading and writing related methods

Pass ‘r’ (or nothing) to open() to open the file in read mode.

Call read() to read the contents of a file

Call close() when you are done with the file.

- Call readLines() to read the contents of a file



file_read.py

Write files

```
1  # make sure you have a hello.txt in your current working director
2  # same directory as your python script
3  helloFile = open("hello.txt", "w")
4  helloFile.write("This is also another line\n")
5  helloFile.close()
6
7  # reopen to display content
8  helloFile = open("hello.txt")
9  print(helloFile.read())
10 helloFile.close()
11
12 # open the file for adding next text
13 helloFile = open("hello.txt", "a")
14 helloFile.write("Hello world again\n")
15 helloFile.close()
16
17 # reopen to display content
18 helloFile = open("hello.txt")
19 print(helloFile.read())
20 helloFile.close()
```

Pass ‘w’ to open() to open the file in write mode or ‘a’ for append mode.

⚠ Opening a non-existent file in write or append mode will create that file

Call write() to write a string to a file.

Search | Stack Data

Search:

Replace:

Case sensitive | Whole words | In Selection

Previ | Ne: | eplac | place | Option:

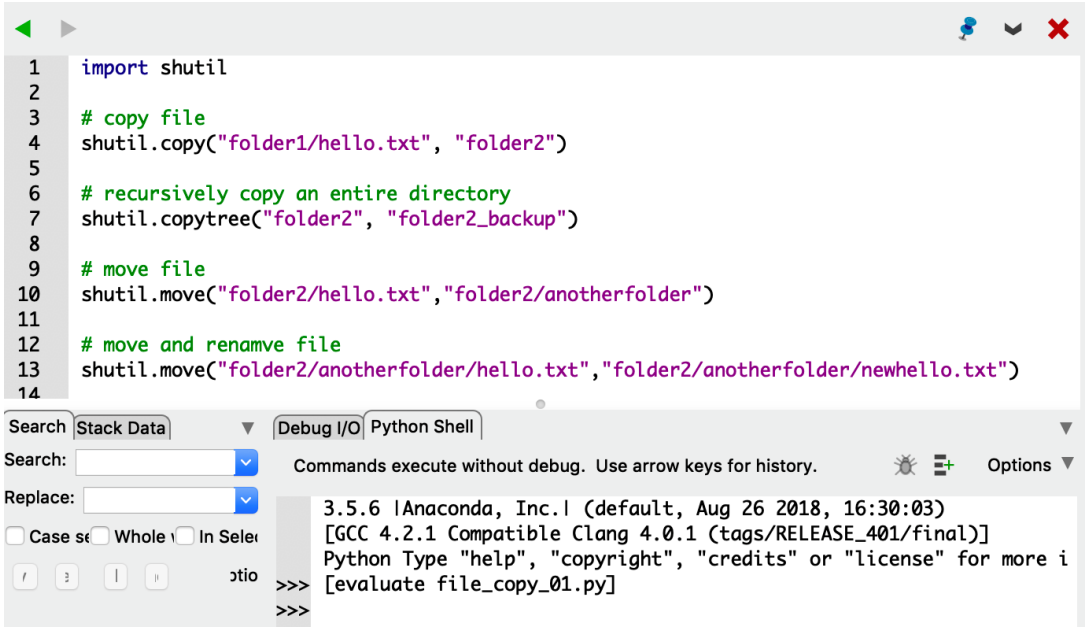
Debug I/O | Python Shell

Commands execute without debug. Use arrow keys for history.

```
3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC
Python Type "help", "copyright", "credits" or "license" for
>>> [evaluate file_write..py]
THis is also another line

THis is also another line
Hello world again
```

file_write.py

Copy and moving files



```
1  import shutil
2
3  # copy file
4  shutil.copy("folder1/hello.txt", "folder2")
5
6  # recursively copy an entire directory
7  shutil.copytree("folder2", "folder2_backup")
8
9  # move file
10 shutil.move("folder2/hello.txt","folder2/anotherfolder")
11
12 # move and renamve file
13 shutil.move("folder2/anotherfolder/hello.txt","folder2/anotherfolder/newhello.txt")
14
```

```
Search | Stack Data
Search:
Replace:
Case se  Whole  In Sele
                otio

Debug I/O | Python Shell
Commands execute without debug.  Use arrow keys for history.   Options

3.5.6 |Anaconda, Inc.| (default, Aug 26 2018, 16:30:03)
[GCC 4.2.1 Compatible Clang 4.0.1 (tags/RELEASE_401/final)]
Python Type "help", "copyright", "credits" or "license" for more i
>>> [evaluate file_copy_01.py]
>>>
```

- shutil.copy(src, dst) – Copy the file *src* to the file or directory *dst*
- shutil.copytree(src, dst) - Recursively copy an entire directory tree rooted at *src*.
- shutil.move(src, dst) - Recursively move a file or directory (*src*) to another location (*dst*).

https://docs.python.org/3/library/shutil.html

file_copy.py

Deleting files

```python
import os

# error if file do not exist
os.unlink("hello.txt")

# get current working directory
print(os.getcwd())

# delete directory (can only delete empty folder)
os.rmdir("folder3")

import shutil
# delete directory (with content)
# error if folder is not found
shutil.rmtree("folder3")
```

- os.unlink() will delete a file
- os.rmdir() will delete a folder (but folder must be empty)
- shutil.rmtree() will delete a folder and all its contents

e.g. To delete all .docx file in the current folder

```python
import os

for filename in os.listdir():
    if filename.endswith(".docx"):
        print(filename)
        os.unlink(filename)
```

Deleting can be dangerous, so do a dry run first

file_delete.py

Exercise

- **Write code to achieve the following:**

1. Create a file named: “myfile.txt”.
2. Write the following line of text into the file:
   - Programming is fun!
3. Close the file
4. Create a folder called “myfolder”
   - Use os.mkdir() command
5. Copy myfile.txt to myfolder

Use Cases

- System administrators can use these commands to
  - Copy and backup files to other hard-disks
  - Delete folders/ files at fixed schedules
    - End of financial year?
    - End of semester?

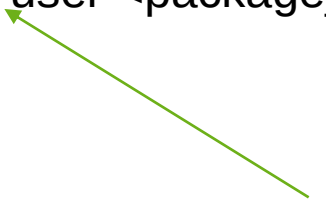
- Others use
  - Check timestamp of files, and delete all files created before a certain date

Python Package Index

- https://pypi.org/
- A repository of software for the Python Programming Language
- Python Installation provides the core libraries needed for the common tasks
  - Additional packages can be found at the website and installed as extension
    - E.g. send2trash, openpyxl, pillow etc
- Installation is easy done with the following command
  - **pip install <software_package>**
- Installed packages can be found at:
  - C:\python38\Lib\site-packages

Using pip install

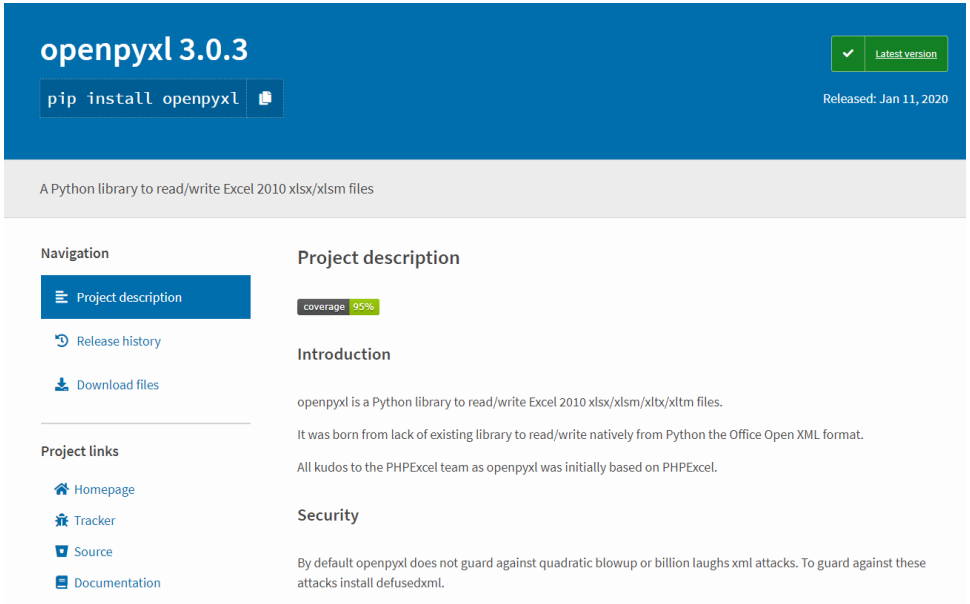
- **For all windows users by default**
  - Open command prompt
  - pip install <package_name>
- **For Mac User**
  - Open terminal
  - **pip3** install <package_name>
- **For RP staff using RP issued laptop**
  - Open command prompt
  - pip install --user <package_name>



Excel Spreadsheet Manipulation with Python

Working with Excel

- Install openpyxl module using "pip install openpyxl"
- Full openpyxl documentation: https://openpyxl.readthedocs.io/en/stable/index.html



Typical Workflow for Excel Automation

1. You are given some data in a spreadsheet
2. You want to do some or all of the following
   - Analyze the data
   - Manipulate the data
3. Output the processed data in another spreadsheet

Reading Excel file

1) Import openpyxl

```
import openpyxl

workbook = openpyxl.load_workbook("bmi.xlsx")
sheet=workbook["Sheet1"]
```

2) Load Excel content into “workbook“ object by specifying the file name

3) Get the worksheet named "Sheet1"

```
name = sheet.cell(row=2, column=1).value
weight = sheet.cell(row=2, column=2).value
height = sheet.cell(row=2, column=3).value
```

4) Get the value of each cell by specifying the row and column

```
print("name:%s \tweight: %d \theight: %f " % (name, weight, height))
```

5) Get the value of each cell by specifying the row and column

excel_read_bmi.py

Reading Excel file

The typical workflow for reading excel file is to use a ***for*** loop to go through each row to read the data

```
import openpyxl

workbook = openpyxl.load_workbook("bmi.xlsx")
sheet=workbook["Sheet1"]


max_row = sheet.max_row # get number of rows

#loop through every row
for i in range(2, max_row + 1):

    #read cell
    name = sheet.cell(row=i, column=1).value
    weight = sheet.cell(row=i, column=2).value
    height = sheet.cell(row=i, column=3).value

    print("name:%s \tweight: %d \theight: %f " % (name, weight, height))
```

1) Get the number of rows and columns

2) Use For loop to go through every row

3) Extract the status at Column C to check for attendance

excel_read_bmi.py

Update Excel file

```
import openpyxl

workbook = openpyxl.load_workbook("bmi.xlsx")
sheet=workbook["Sheet1"]

max_row = sheet.max_row # get number of rows

# add a column header for bmi
sheet.cell(row=1, column=4).value = "bmi"

#loop through every row
for i in range(2, max_row + 1):

    #read cell
    name = sheet.cell(row=i, column=1).value
    weight = sheet.cell(row=i, column=2).value
    height = sheet.cell(row=i, column=3).value

    bmi = weight / (height * height)

    sheet.cell(row=i, column=4).value = bmi

    print("name:%s \tBMI: %f" % (name, bmi))

#save the file
workbook.save("bmi_update.xlsx")
```

2) Load file into memory & get the sheet

1) Perform calculation with values taken from the excel files

2) Add comments to cell

5) Save the spreadsheet

Create Excel file

If you have data in nested python list, you can write the data into an excel file.

```
import openpyxl

workbook = openpyxl.Workbook()

#get the default sheet
sheet=workbook["Sheet"]

#create a list of tuples as data source
data = [
    [225.7,'Gone with the Wind','Victor Fleming'],
    [194.4, 'Star Wars', 'George Lucas'],
    [161.0, 'ET: The Extraterrestrial', 'Steven Spielberg']
]

for row in data:
    sheet.append(row)

#save the spreadsheet
workbook.save("movies.xlsx")
```

1) Some data in nested list

2) Using for loop to add each row of data into the excel sheet

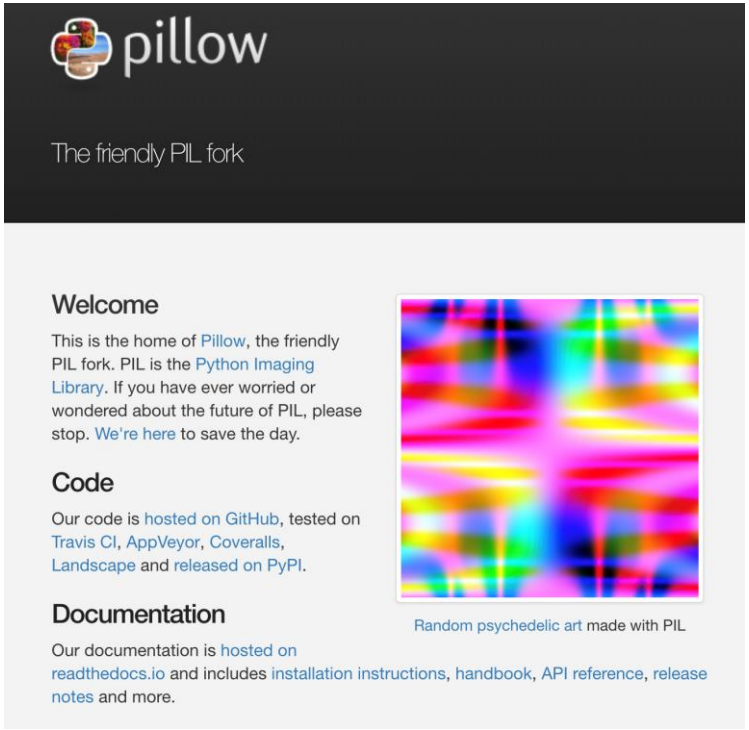
3) Save the spreadsheet

Typical Workflow for Excel Automation

1. You are given some data in a spreadsheet
2. You want to do some or all of the following
   - Analyze the data
   - Manipulate the data
   - Create visualization (Charts, Pivot Table etc)
3. Output the processed data in another spreadsheet

Image Processing with Python

Image Processing



For the next section we are going to use the Python Image Library, or in short Pillow.

Install using the following command:
pip install pillow

The documentation is at:
https://pillow.readthedocs.io/en/stable/handbook/overview.html

Image Processing

```
import os
from PIL import Image

filename = "img/clungup.jpg"

im = Image.open(filename)
print ("%s - %s" % (im.size, im.mode))

# show the image
im.show()

# close the file
im.close()
```

```
Debug I/O | Python Shell | Messages | OS Commands

Debug I/O (stdin, stdout, stderr) appears below

(800, 600) - RGB
```

As a start we need to import it:
import Image

We can open images with
im = Image.open(fullname)

Then we can display the image using
im.show()

Print some info about the image using
im.size and im.mode

Size: 800 x 600, Mode: RGB

image_show_info.py

Image Processing

```
import os
from PIL import Image, ImageFilter

filename = "img/clungup.jpg"

im = Image.open(filename)

out = im.filter(ImageFilter.BLUR)

im.show()
out.show()
```

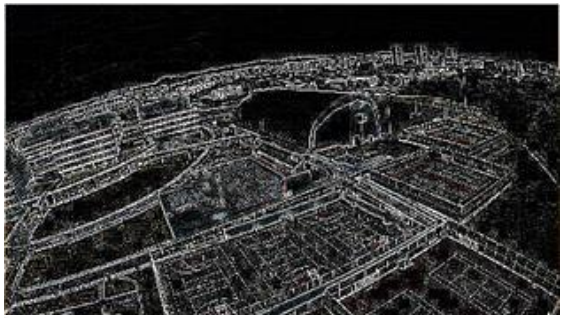
Pillow has many conversion and filters, to use filters we need to extend our import:
from PIL import Image, ImageFilter

The way you can apply filters is :
out = im.filter(ImageFilter.BLUR)

Try other different filters!

image_blur_filter.py

Image processing - filters



```
image = image.filter(ImageFilter.FIND_EDGES)
```

```
image = ImageOps.grayscale(image)
```

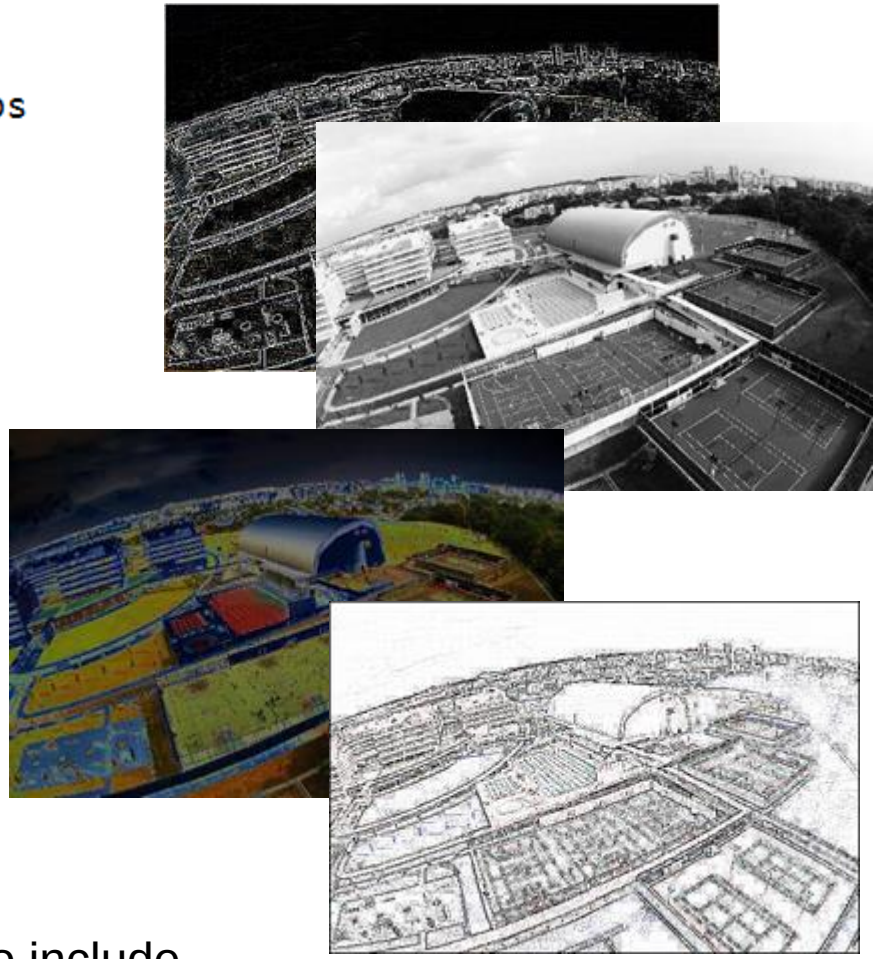
```
image = ImageOps.solarize(image)
```

```
image = image.filter(ImageFilter.CONTOUR)
```

⚠ * Remember to include ImageOps in your import statement

Image processing - filters

```
import os
from PIL import Image, ImageFilter, ImageOps

filename = "img/clungup.jpg"

im = Image.open(filename)

# Filter
#out = im.filter(ImageFilter.BLUR)
#out = im.filter(ImageFilter.FIND_EDGES)
#out = im.filter(ImageFilter.CONTOUR)

# ImageOps
out = ImageOps.grayscale(im)
#out = ImageOps.solarize(im)

im.show()
out.show()
```



* Remember to include ImageOps in your import statement

image_other_filters.py

Image Processing - Rotating

Flipping the image horizontally or vertically
out = im.transpose(Image.FLIP_LEFT_RIGHT)
out = im.transpose(Image.FLIP_TOP_BOTTOM)

Flip images

Rotating the image
out = im.transpose(Image.ROTATE_90)
out = im.transpose(Image.ROTATE_180)
out = im.transpose(Image.ROTATE_270)

Rotate images

Contrast
First add ImageEnhance to our imports:
from PIL import Image, ImageFilter, ImageEnhance

Then:

enh = ImageEnhance.Contrast(im)
out = enh.enhance(1.3)

make image brighter by changing the contrast

Image Processing - Writing

```
import os
from PIL import Image, ImageFilter, ImageOps

filename = "clungup.jpg"

src_folder = "img/"
out_folder = "out/"

im = Image.open(src_folder + filename) # img/clungup.jpg
out = im.filter(ImageFilter.BLUR)

outFilename = out_folder + filename # out/clungup.jpg

out.save(outFilename)
```

You can see the image, but it's not being saved !

All you need to do to save the images in the "out" folder is: out.save(the name of the output file)

image_save.py

Image processing – Watermark

Create the mark image
You can reduce the size to 100,100

```
mark = Image.open("img\\watermark.png")
mark = mark.resize((100,100))
```

Create a new function called

```
def watermark(im, mark, position):
        ....
```

It takes the original image, the watermark image and the desired position that we want the watermark to appear. The function will return the result.

We can use this function like:

```
watermark(im, mark, (0, 50)).show()
```

or

```
imOut = watermark(im, mark, (0,50))
imOut.save(fileOut)
```

Maybe you want to leave a small footprint on your images, called watermark.

In this case we can use the \\img\\watermark.png
and place it in each image on the bottom right.

Image processing – Watermark

```
from PIL import Image

def watermark(im, mark, position):
    layer = Image.new("RGBA", im.size, (0,0,0,0))
    layer.paste(mark, position)
    return Image.composite(layer, im, layer)

im = Image.open("img\\clungup.jpg")
mark = Image.open("img\\watermark.png")
mark = mark.resize((100,100))
mark.putalpha(128)

out = watermark(im, mark, (0,0))
out.show()
```

First we need to create a new layer with the size of the original image.

Then we paste the watermark image at the desired position and we return the composite.

Finally we merge the image and the layer together and return the result.

Then you can use it like this:

image_watermark.py

Use Case I: Batch Resize

1. Find all the files in “img” folder with “.jpg” extension

1. Resize all the file to 60 x 90.

1. Save all the files to the resized folder

```
import os
from PIL import Image, ImageFilter, ImageOps

files = os.listdir('img')
size = 60, 90

for file in files:
    if file.lower().endswith(".jpg"):
        im = Image.open("img/" + file)
        im.thumbnail(size, Image.ANTIALIAS)
        im.save("resized/" + file, "JPEG")
```

image_batch_resize.py

Use Case II: Batch Rename

1. Find all the files in “img” folder with “.jpg” extension

1. Copy all the files to the renamed folder

1. Rename all the files with the “s-” prefix.

```
import os
import shutil

files = os.listdir('img')

for file in files:
    if file.lower().endswith(".jpg"):
        shutil.copyfile("img/" + file, "renamed/s-" + file[:-4] + ".jpg")
```

image_batch_rename.py

REPUBLIC POLYTECHNIC
DISCOVER. TRANSFORM. ACHIEVE

Lunch Break

Web Automation with Python

Connecting to the Web

- requests – download files and web pages from the Web

pip install requests

```
1  import requests
2
3  url = "https://api.data.gov.sg/v1/environment/2-hour-weather-forecast"
4  req = requests.get(url)
5  print(req.text)
```

Get the required information from the given URL

web_request.py

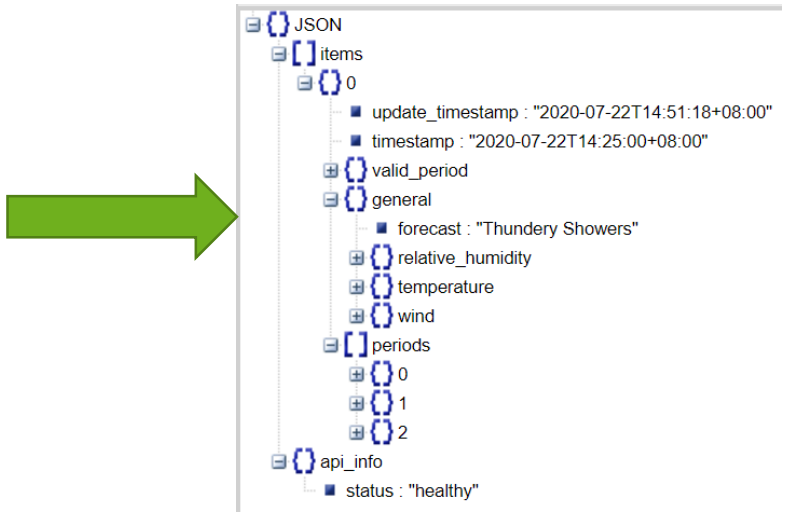
Connecting to the Web

- Data is in JSON format
- Use a JSON formatter tool to present the data in a nicer form

http://jsonviewer.stack.hu/

```
import requests

url="https://api.data.gov.sg/v1/environment/24-hour-weather-forecast"
req=requests.get(url)
print(req.text)
```


```
{"items":[{"update_timestamp":"2020-07-22T14:51:18+08:00","timesta
"2020-07-22T14:25:00+08:00","valid_period":{"start":"2020-07-22T1
"end":"2020-07-23T12:00:00+08:00"},"general":{"forecast":"Thunder
"relative_humidity":{"low":70,"high":95},"temperature":{"low":22,"
"wind":{"speed":{"low":10,"high":20},"direction":"ESE"}},"periods"
"start":"2020-07-22T12:00:00+08:00","end":"2020-07-22T18:00:00+08
:{"west":"Moderate Rain","east":"Moderate Rain","central":"Light R
"Light Rain","north":"Light Rain"}},{"time":{"start":"2020-07-22T1
,"end":"2020-07-23T06:00:00+08:00"},"regions":{"west":"Partly Clou
"east":"Partly Cloudy (Night)","central":"Partly Cloudy (Night)","
"Partly Cloudy (Night)","north":"Partly Cloudy (Night)"}},{"time":
"2020-07-23T06:00:00+08:00","end":"2020-07-23T12:00:00+08:00"},"re
```

Connecting to the Web

- To work with JSON data, import json first
- Use json.loads() to load the data in JSON format
- Extract and retrieve the required data

```
import json
import requests

url="https://api.data.gov.sg/v1/environment/24-hour-weather-forecast"
req=requests.get(url)

data = json.loads(req.text)

# print update timestamp
update_time = data["items"][0]["update_timestamp"]
print("Update time: " + update_time)

# print forecast
forecast = data["items"][0]["general"]["forecast"]
print("Forecast: " + forecast)
```

```
Update time: 2020-07-22T14:51:18+08:00
Forecast: Thundery Showers
```

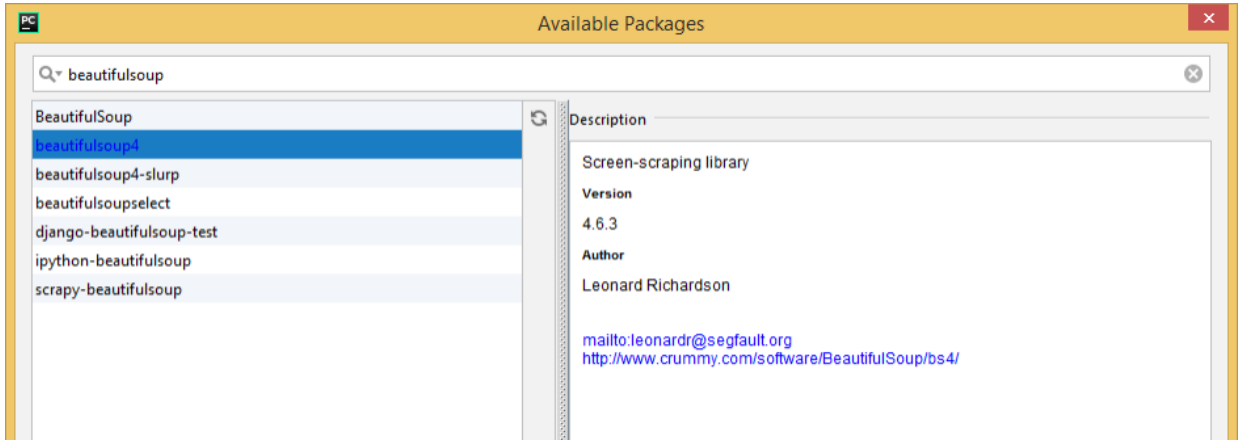
web_json.py

Connecting to the Web

- Beautiful Soup – a third party module that parses HTML (web pages)

Web Scraping – download and process Web content

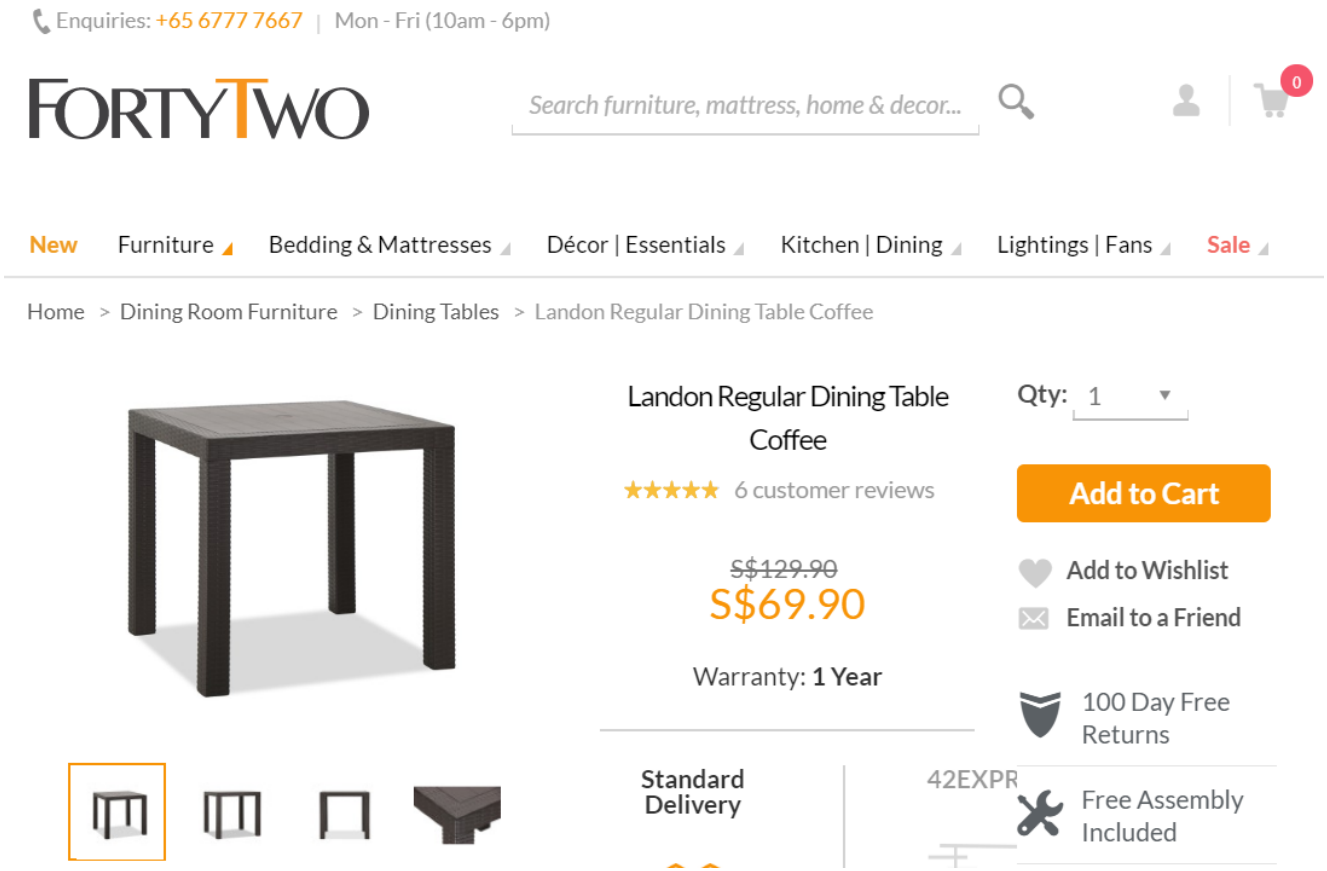
- Install Beautiful Soup 4 - **pip install beautifulsoup4**



Connecting to the Web

- What's the URL?

https://www.fortytwo.sg/dining/dining-tables/landon-regular-dining-table-coffee.html

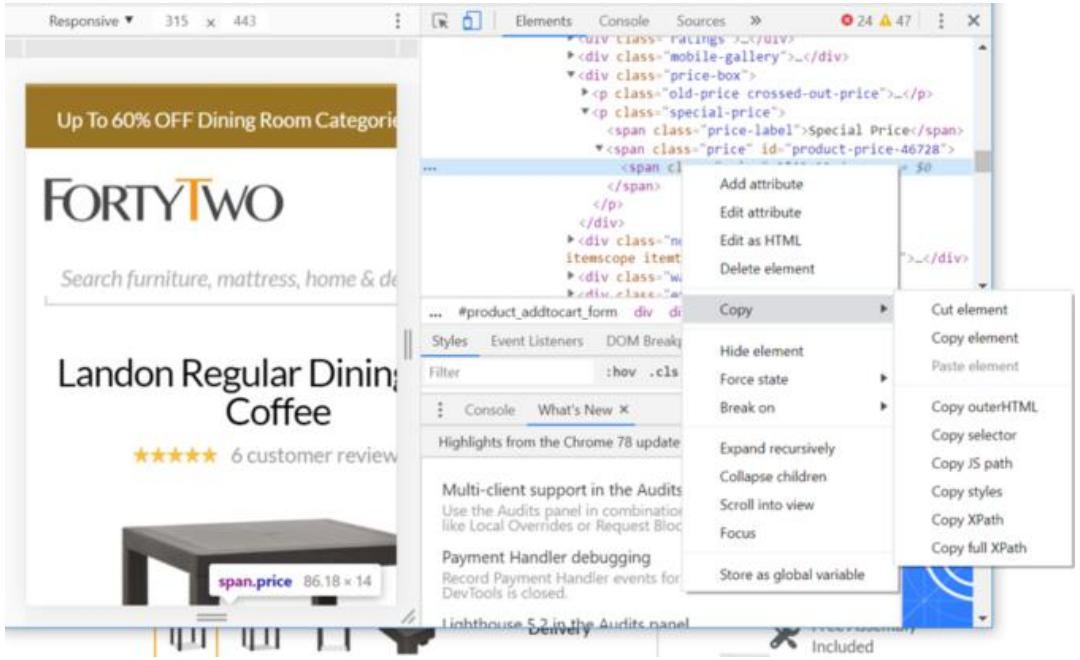


Connecting to the Web

- Get the url

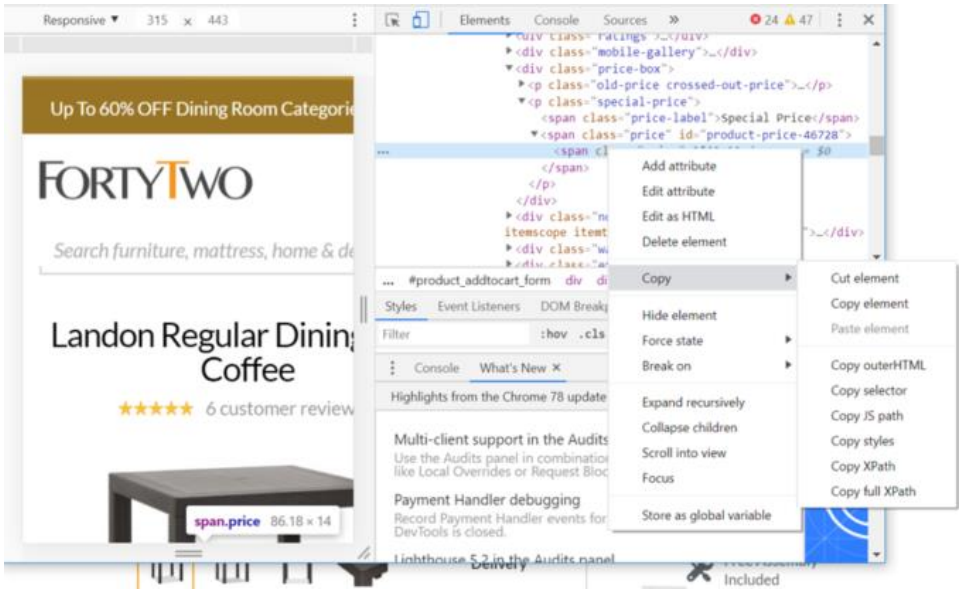
https://www.fortytwo.sg/dining/dining-tables/landon-regular-dining-table-coffee.html

- Select the element to extract, right-click "Inspect"
- Right-click "Copy" □ "Copy selector



Connecting to the Web

- Get the url
- Select the element to extract, right-click "Inspect"
- Right-click "Copy“ → "Copy selector"



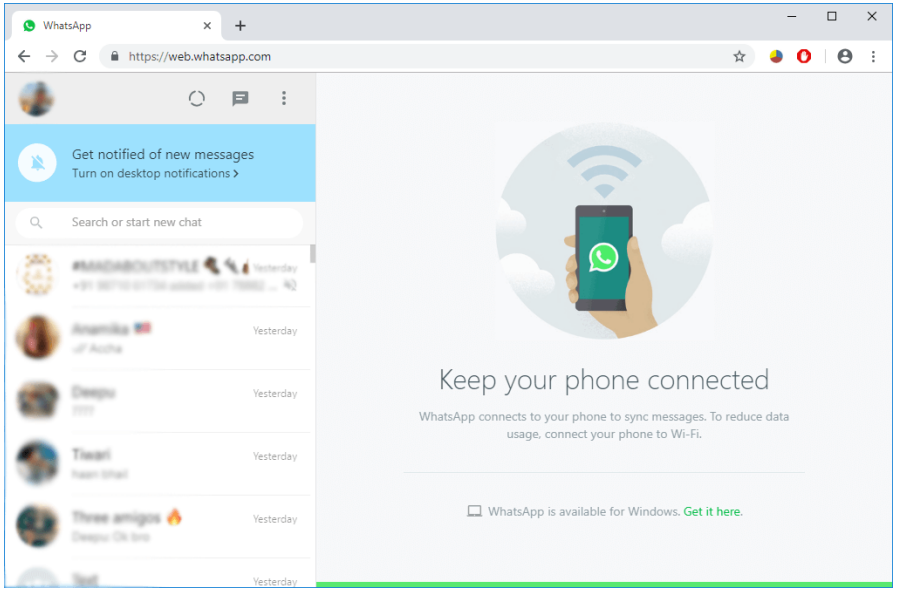

```
import bs4
import requests

requestObj = requests.get("https://www.fortytwo.sg/dining/dining-tables/landon-regular-dining-table-coffee.html")
requestObj.raise_for_status()
soup = bs4.BeautifulSoup(requestObj.text, 'html.parser')


elements = soup.select("#product-price-46728") # $69.90
print("Current Price: " + elements[0].text)

elements = soup.select("#old-price-46728")  #  $129.90
print("\nOld Price: " + elements[0].text)
```

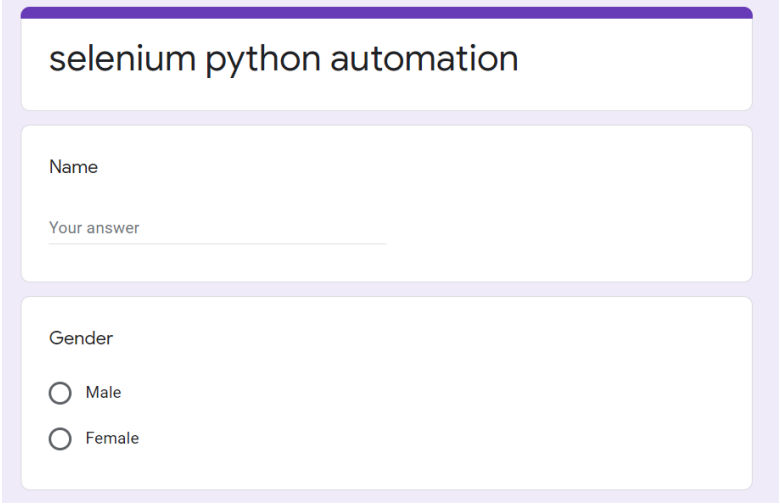
```
Debug I/O | Python Shell | Messages | OS Commands
Debug I/O (stdin, stdout, stderr) appears below
Current Price:  S$69.90

Old Price:  S$129.90
```

web_scrap.py

Sharing other Use Cases

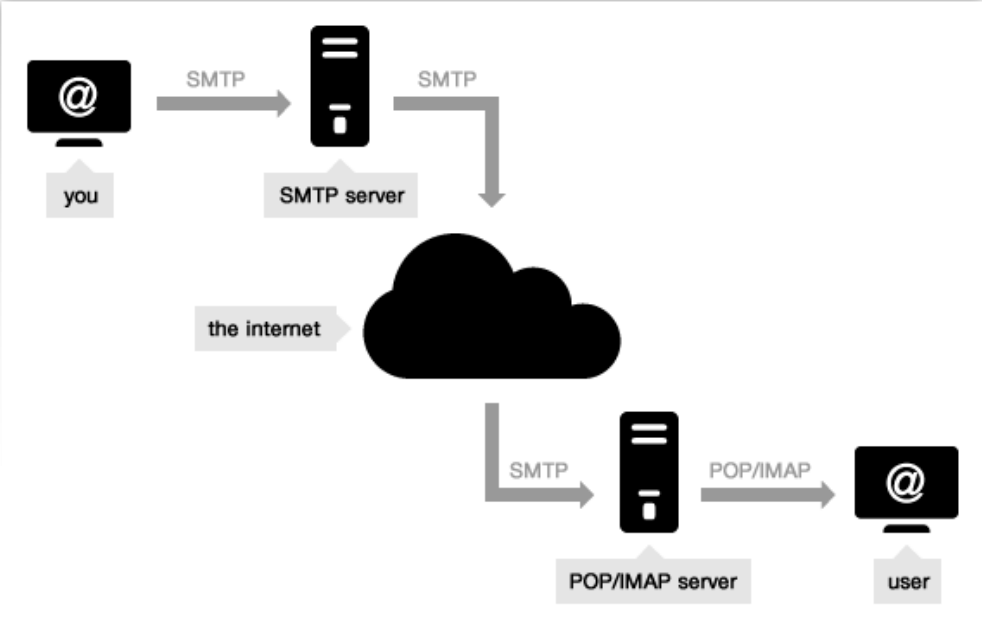
- Using another library: selenium
  - Filling up google form
  - Sending WhatsApp message





Email Automation with Python

Send Email

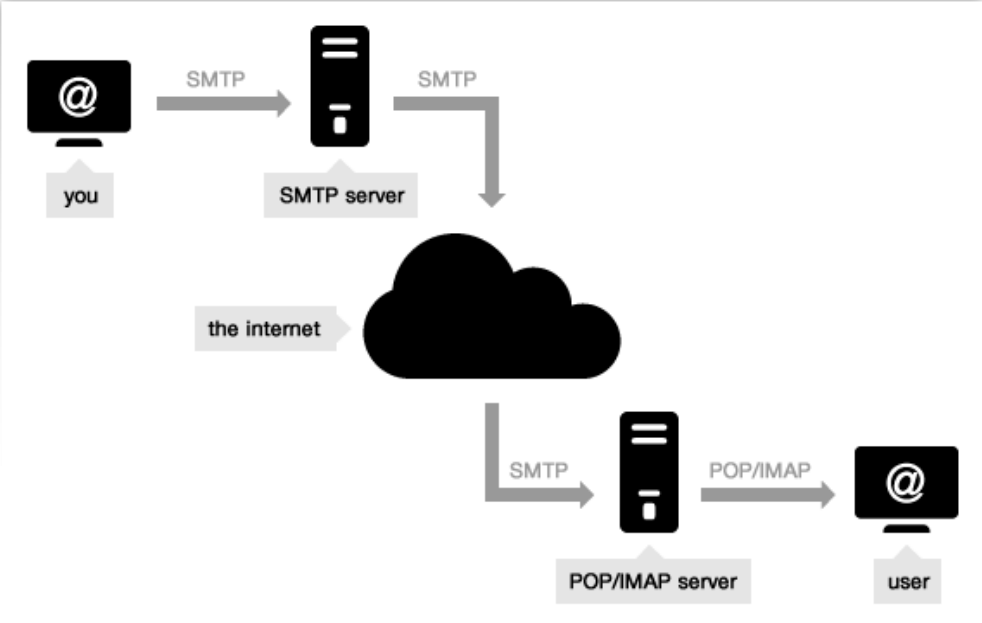


- SMTP (Simple Mail Transfer Protocol) is used for sending and delivering from a client to a server via port 25, 465 or 587: it's the **outgoing server**.
- IMAP and POP are two methods to access email. IMAP is the recommended method when you need to check your emails from several different devices, such as a phone, laptop, and tablet.

https://www.mailgun.com/blog/which-smtp-port-understanding-ports-25-465-587/

https://serversmtp.com/what-is-smtp-server/

Send Email



- **Note**: The SMTP servers used when you send your emails- Hotmail, Gmail , Yahoo Mail – are **shared among users**
- Common providers establish some **strict limits** on the number of emails you can send (e.g. Yahoo's restriction is 100 emails per hour).
- If you plan to send a bulk email or set up an email campaign you should opt for a professional outgoing email server like turboSMTP,
- which guarantees a controlled IP and ensure that all your messages reach their destination.

Send Email using Gmail

Incoming Mail (IMAP) Server	imap.gmail.com Requires SSL: Yes Port: 993
Outgoing Mail (SMTP) Server	smtp.gmail.com Requires SSL: Yes Requires TLS: Yes (if available) Requires Authentication: Yes Port for SSL: 465 Port for TLS/STARTTLS: 587
Full Name or Display Name	Your name
Account Name, User name, or Email address	Your full email address
Password	Your Gmail password

Note: If you are using your office network, most port numbers, including 587, may be blocked.

Send Email using Gmail

- Import smtplib module
- Specify Gmail email & password, receiver's email address, email title & content
- Connect to SMTP server using Port 587
- Call starttls() to enable encryption for your connection
- Login using email and password
- Call sendmail()
- Call quit() to disconnect from the SMTP server

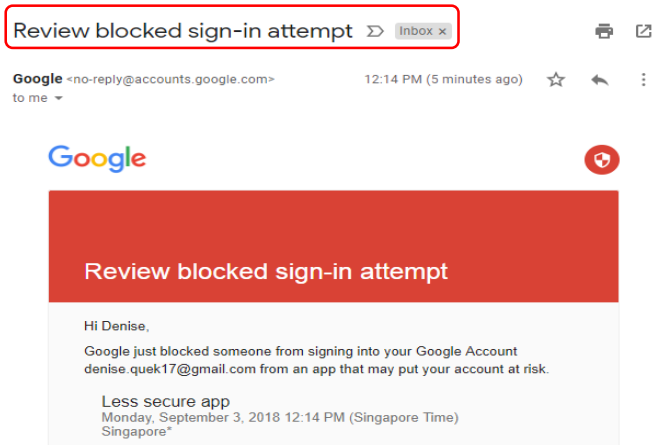
```
import smtplib

sender_email_address = "your_email_address@gmail.com"
sender_email_password = "xxxxxxxxxxxxxx"
receiver_email_address = "another_email_address@gmail.com"
email_title_content = "Subject: Sending Email Using Python\nThis is a test email."

email_title_content = "Subject: Sending Email Using Python\nThis is a test email."
```

➢ The start of the email body must begin with "Subject: " for the subject line. The "\n" newline character separates the subject line from the main body content.

```
print("Trying to connect to Gmail SMTP server")
smtpObj = smtplib.SMTP("smtp.gmail.com", 587)
smtpObj.starttls()

print("Connected. Logging in...")
smtpObj.login(sender_email_address, sender_email_password)

smtpObj.sendmail(sender_email_address, receiver_email_address, email_title_content)
print("Email sent successfully...")

smtpObj.quit()
```

Send Email using Gmail

- Google may block attempted sign-in from unknown devices that don't meet their security standards!




```
C:\Users\denise_quek\AppData\Local\Programs\Python\Python37\python.exe D:/CET_Python/Denise/TestEmail.py
Trying to connect to Gmail SMTP server
Connected. Logging in...
Traceback (most recent call last):
  File "D:/CET_Python/Denise/TestEmail.py", line 13, in <module>
    smtpObj.login(sender_email_address, sender_email_password)
  File "C:\Users\denise_quek\AppData\Local\Programs\Python\Python37\lib\smtplib.py", line 730, in login
    raise last_exception
  File "C:\Users\denise_quek\AppData\Local\Programs\Python\Python37\lib\smtplib.py", line 721, in login
    initial_response_ok=initial_response_ok)
  File "C:\Users\denise_quek\AppData\Local\Programs\Python\Python37\lib\smtplib.py", line 642, in auth
    raise SMTPAuthenticationError(code, resp)
smtplib.SMTPAuthenticationError: (534, b'5.7.9 Application-specific password required. Learn more at\n5.7.9

Process finished with exit code 1
```

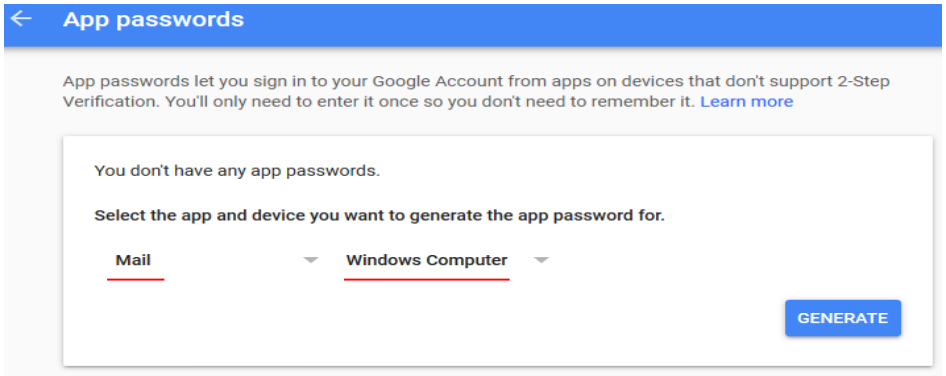
Send Email using Gmail

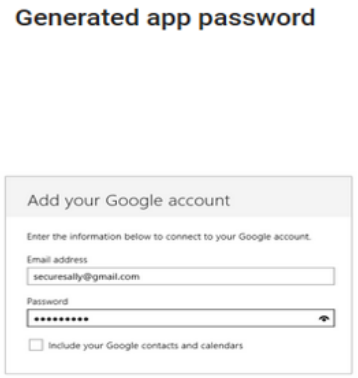
Steps To Create Google App Password

Step 1: Login to Gmail. Go to Account □ Signing in to Google

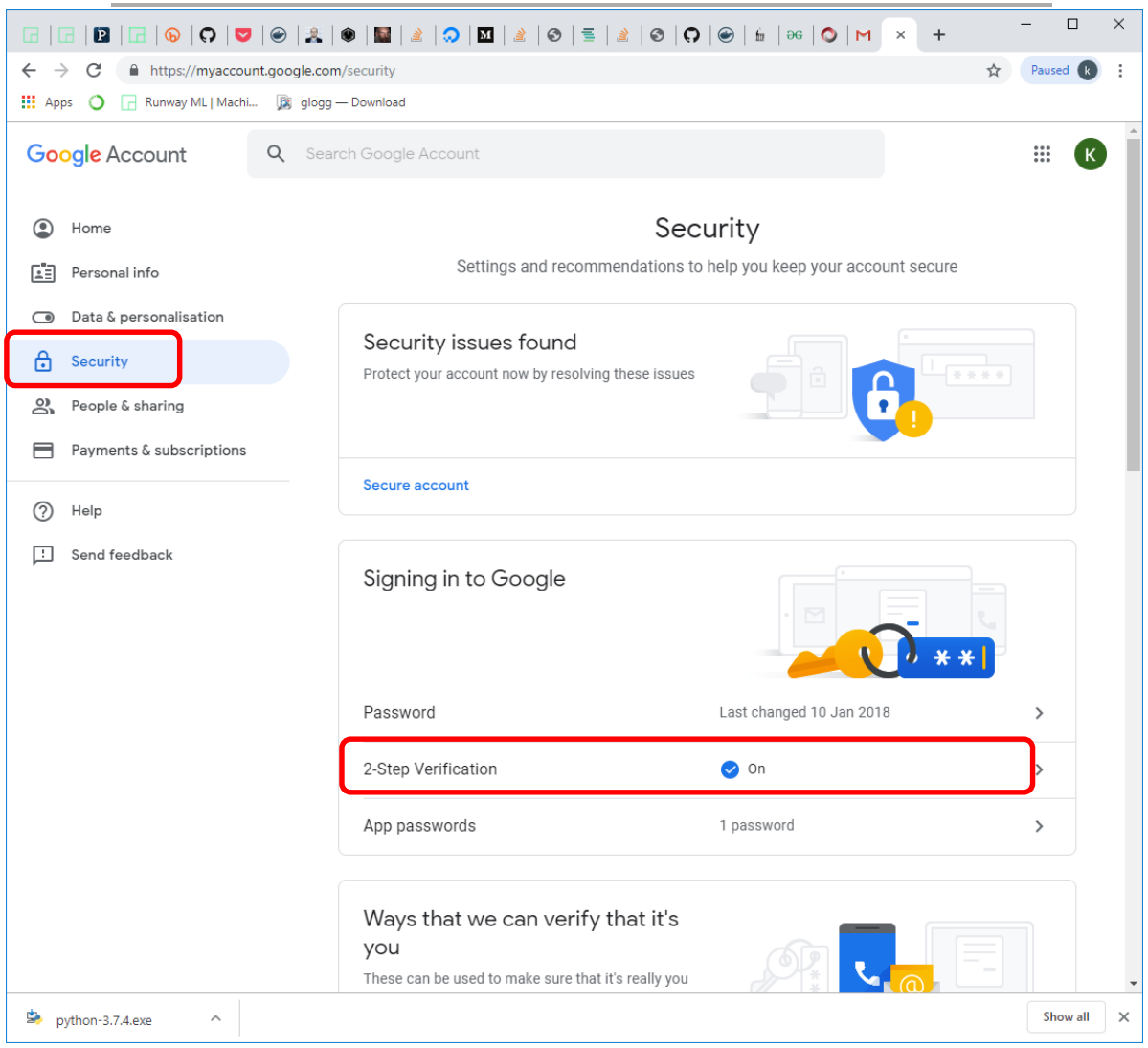
Step 2: Make sure that 2-Step Verification is on

Step 3: Create an App password





Send Email using Gmail



Send Email using Gmail

- Replace your actual password with the App password

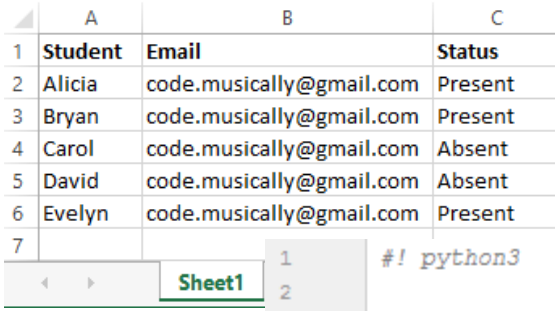
```
import smtplib

sender_email_address = "your_email_address@gmail.com"
sender_email_password = "xxxxxxxxxxxxxxx"
receiver_email_address = "another_email_address@gmail.com"
email_title_content = "Subject: Sending Email Using Python\nThis is a test email."
```

- Run your email program

```
C:\Users\denise_quek\AppData\Local\Programs\Python\Python37\python.exe D:/CET_Python/Denise/TestEmail.py
Trying to connect to Gmail SMTP server
Connected. Logging in...
Email sent successfully...

Process finished with exit code 0
```

Use Case: Send emails to students

- Send email to students who were absent



	A	B	C
1	**Student**	**Email**	**Status**
2	Alicia	code.musically@gmail.com	Present
3	Bryan	code.musically@gmail.com	Present
4	Carol	code.musically@gmail.com	Absent
5	David	code.musically@gmail.com	Absent
6	Evelyn	code.musically@gmail.com	Present
7			

Sheet1

```python
#! python3

import openpyxl, smtplib

def sendEmail(name, emailTo):
    email_body = "Subject: Your attendance. \nDear %s, \nYou were absent for class.\n" %(name)

    smtpObj = smtplib.SMTP("smtp.gmail.com", 587)
    smtpObj.starttls()
    smtpObj.login("code.musically@gmail.com", "xxxxxxxxxxxx")
    smtpObj.sendmail('code.musically@gmail.com', emailTo, email_body)

    smtpObj.quit()
```

Use Case: Send Emails to Students

- Open an Excel file
- Send email to students who were absent

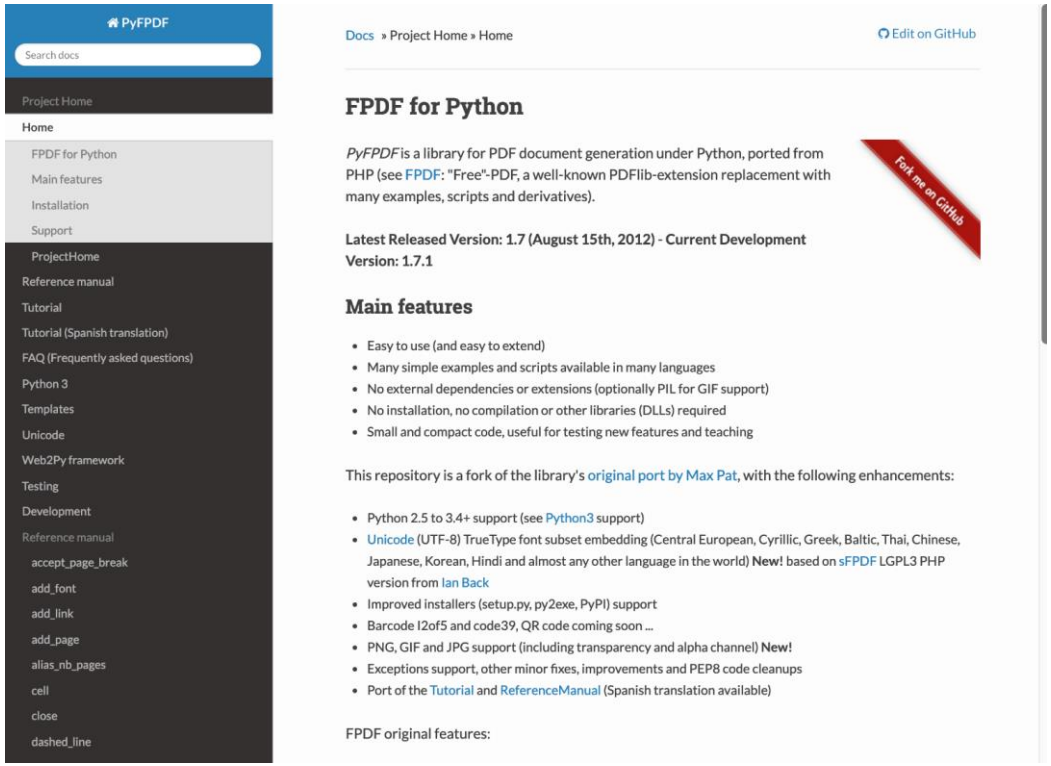
```
16  workbook = openpyxl.load_workbook("D:\CET_Python\students_attendance.xlsx")
17  sheet = workbook["Sheet1"]
18
19  max_row = sheet.max_row
20  max_column = sheet.max_column
21
22  for i in range(1, max_row+1):
23
24      attendance = sheet.cell(row=i, column=3).value
25
26      if attendance == "Absent":
27          name = sheet.cell(row=i, column=1).value
28          email = sheet.cell(row=i, column=2).value
29
30          print(name + " is absent.")
31          sendEmail(name, email)
32          print("Email sent to " + email)
33          print()
34
```

Sharing other Use Cases

- Sending Emails using Outlook
- Create Appointment using Outlook

Generate PDF Report with Python

PDF



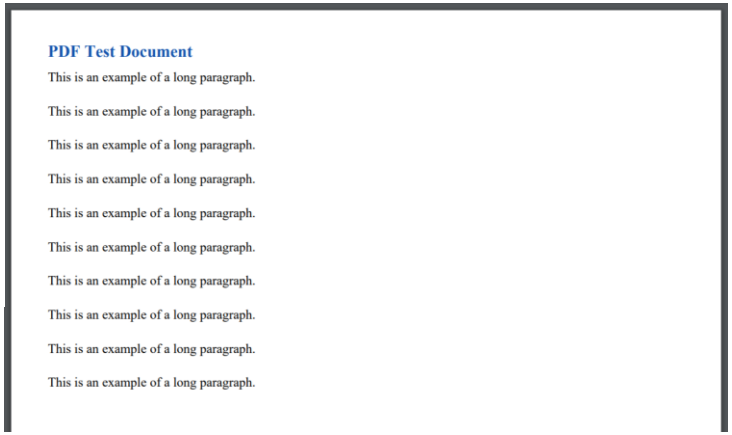
- Install fpdf
  - pip install fpdf

https://pyfpdf.readthedocs.io/en/latest/Tutorial/index.html

PDF – Basic document

```python
import fpdf

#create a new pdf
document = fpdf.FPDF()

#define font and color for title and add the first page
document.set_font("Times","B", 14)
document.set_text_color(19,83,173)
document.add_page()

#write the title of the document
document.cell(0,5,"PDF Test Document")
document.ln()

#write a long paragraph
document.set_font("Times", "", 11)
document.set_text_color(0)
document.multi_cell(0,10, "This is an example of a long paragraph. \n" * 10)
document.ln()

#save the document
document.output("pdf_report.pdf")
```

- Import fpdf
- Create a new pdf document
- Add page
- Add text
- Save file

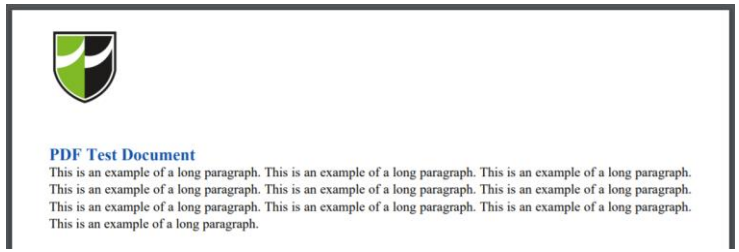


pdf_text.py

PDF – adding images

```
import fpdf

#create a new pdf
document = fpdf.FPDF()

#define font and color for title and add the first page
document.set_font("Times","B", 14)
document.set_text_color(19,83,173)
document.add_page()

#add a image
document.image("rp_logo.png", x=10, y=5, w=23)

document.set_y(40);

#write the title of the document
document.cell(0,5,"PDF Test Document")
document.ln()

#write a long paragraph
document.set_font("Times", "", 11)
document.set_text_color(0)
document.multi_cell(0,5, "This is an example of a long paragraph. " * 10)
document.ln()

#save the document
document.output("pdf_report.pdf")
```

- Import fpdf
- Create a new pdf document
- Add page
- Add text, logo
- Save file



https://pyfpdf.readthedocs.io/en/latest/reference/image/index.html

pdf_image.py

PDF – Adding password

```
import fpdf
import PyPDF2

#create a new pdf
document = fpdf.FPDF()

#define font and color for title and add the first page
document.set_font("Times","B", 14)
document.set_text_color(19,83,173)
document.add_page()

#write the title of the document
document.cell(0,5,"PDF Test Document")
document.ln()

#save the document
document.output("pdf_report_before_pw.pdf")

#save the document into a new password protected/encrypted pdf
pdffile = open(r"pdf_report_before_pw.pdf", "rb")
pdfReader = PyPDF2.PdfFileReader(pdffile)
pdfWriter = PyPDF2.PdfFileWriter()
for pageNum in range(pdfReader.numPages):
    pdfWriter.addPage(pdfReader.getPage(pageNum))

pdfWriter.encrypt('123')
resultPDF = open(r"pdf_report_after_pw.pdf", "wb")
pdfWriter.write(resultPDF)
resultPDF.close()
pdffile.close()
```

- pip install PyPDF2

Password is 123

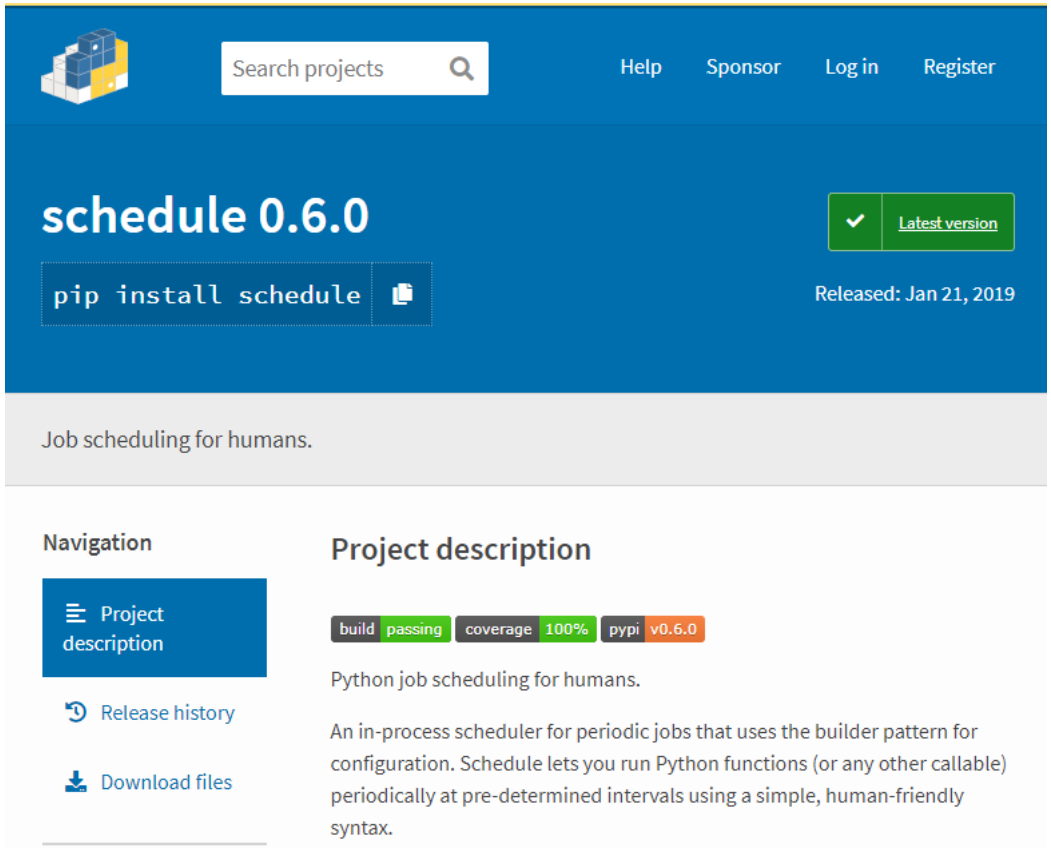
https://pythonhosted.org/PyPDF2/

pdf_password.py

Use Cases

- Automation:
  - Generation of reports with data from spreadsheet or database
  - Generation of Course Certificates in PDF format
  - Password protection of banking statement in PDF file

Schedule



- Install schedule
  - pip install schedule

https://pypi.org/project/schedule/

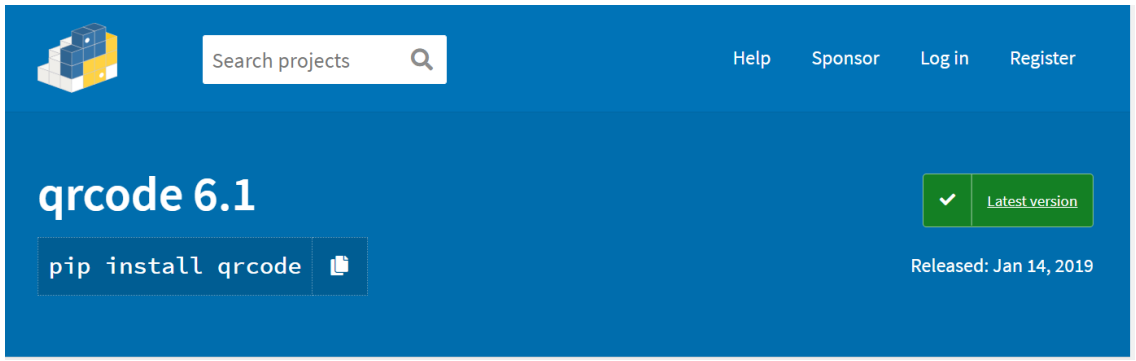
Schedule

- Schedule python script to run at regular interval

```
import schedule
import time

def task1():
    print("Doing task1... ...")

def task2():
    print("Doing task 2... ...")

schedule.every(1).minutes.do(task1)

schedule.every(7).seconds.do(task2)

#schedule.every(5).hour.do(job)
#schedule.every().day.at("10:30").do(job)
#schedule.every().monday.do(job)
#schedule.every().wednesday.at("13:15").do(job)
#schedule.every().minute.at(":17").do(job)

while True:
    schedule.run_pending()
    time.sleep(1)
```

test_scheduler.py

QR Code



- Install qrcode
  - pip install qrcode

```
import qrcode
img = qrcode.make('http://www.google.com')
img.save("test.png")
```

https://pypi.org/project/qrcode/

qr_code.py

End of Day 2

This concludes the Introduction to Python,
I hope you enjoyed it.

Thank you !

QUESTIONS ?

Where to go from here ?

Getting started step by step
http://www.python.org/about/gettingstarted/

Run through the python tutorials:
http://docs.python.org/tutorial/index.html

Keep the API doc under your pillow:
http://docs.python.org/library/index.html

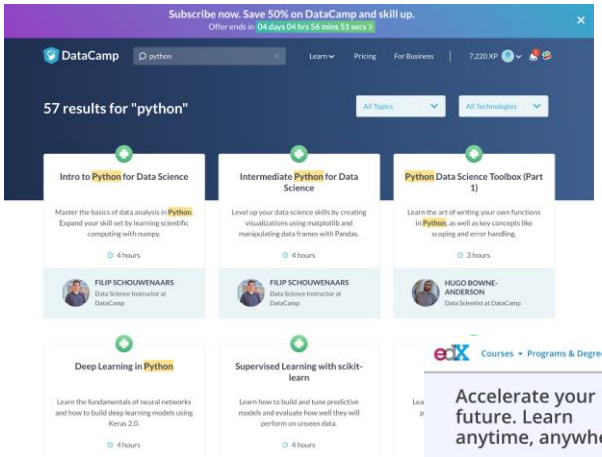
Advanced examples:
http://www.diveintopython.org/toc/index.html

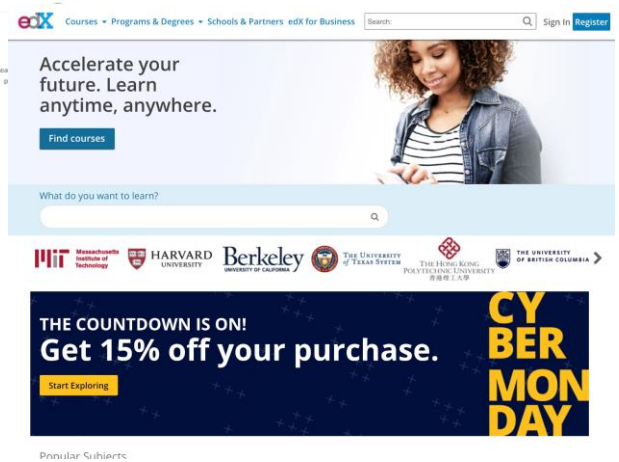
Where to go from here ?

MOOC:
DataCamp
https://www.datacamp.com/

Edx
https://www.edx.org/

Udemy (freemium course)
https://t.me/freecourse





Where to go from here ?

Think Python is an introduction to Python programming for beginners. It starts with basic concepts of programming, and is carefully designed to define all terms when they are first used and to develop each new concept in a logical progression. Larger pieces, like recursion and object-oriented programming are divided into a sequence of smaller steps and introduced over the course of several chapters.

Think Python is a Free Book. It is available under the Creative Commons Attribution-NonCommercial 3.0 Unported License, which means that you are free to copy, distribute, and modify it, as long as you attribute the work and don't use it for commercial purposes. http://greenteapress.com/thinkpython/thinkpython.pdf

Lifelong Learning

Scan me

- https://www.rp.edu.sg/soi/lifelong-learning

Short Courses

SOI offers an extensive variety of short, industry-relevant courses for ICT skills upgrading and skills acquisition. Our courses are categorized under different areas, ranging from Artificial Intelligence (AI), Business Intelligence/ Business Analytics (BI/BA), Business Processes (BP), Unmanned Aerial Vehicle (UAV), IT Security, New/ Digital Media, Software Development to the Internet of Things (IoT). To view our short course offerings, click on the relevant tab below.

AI | Data Analytics | IT Security | DevOps | Software Development | New/ Digital Media | UAV | RPA

+ Artificial Intelligence for Everyone - A Practical Experience (1 day Beginner)

+ Artificial Intelligence for Techies - A Hands-On Approach (1 day Beginner)

+ An Introduction to Code-Free Machine Learning (1 day Beginner)

Low-code Mobile App Development

- Using outsystems.com platform
- Create Mobile App with minimal coding
- https://bit.ly/rp-lowcode
- Next run: Jan 2021