

Introductory Programming Using Python

Day 2

Republic Polytechnic

Tea Break

Vending Machine Code
851475



Morning:
9.30 am – 11.00am

Afternoon:
2.30pm – 4.30pm

Programme Day Two

Morning	Afternoon
<ul style="list-style-type: none">• Read and writing files• Copying, moving and deleting files and folders• Working with Excel	<ul style="list-style-type: none">• Image Processing• Connecting to the Web• Sending emails

Outline for the day

Time	Agenda
9.00am	Welcome and admin matters
9.15am – 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 – 4.30pm	
4.45pm – 5.00pm	Wrap up, Q&A

File Paths



Absolute file paths are noted 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.

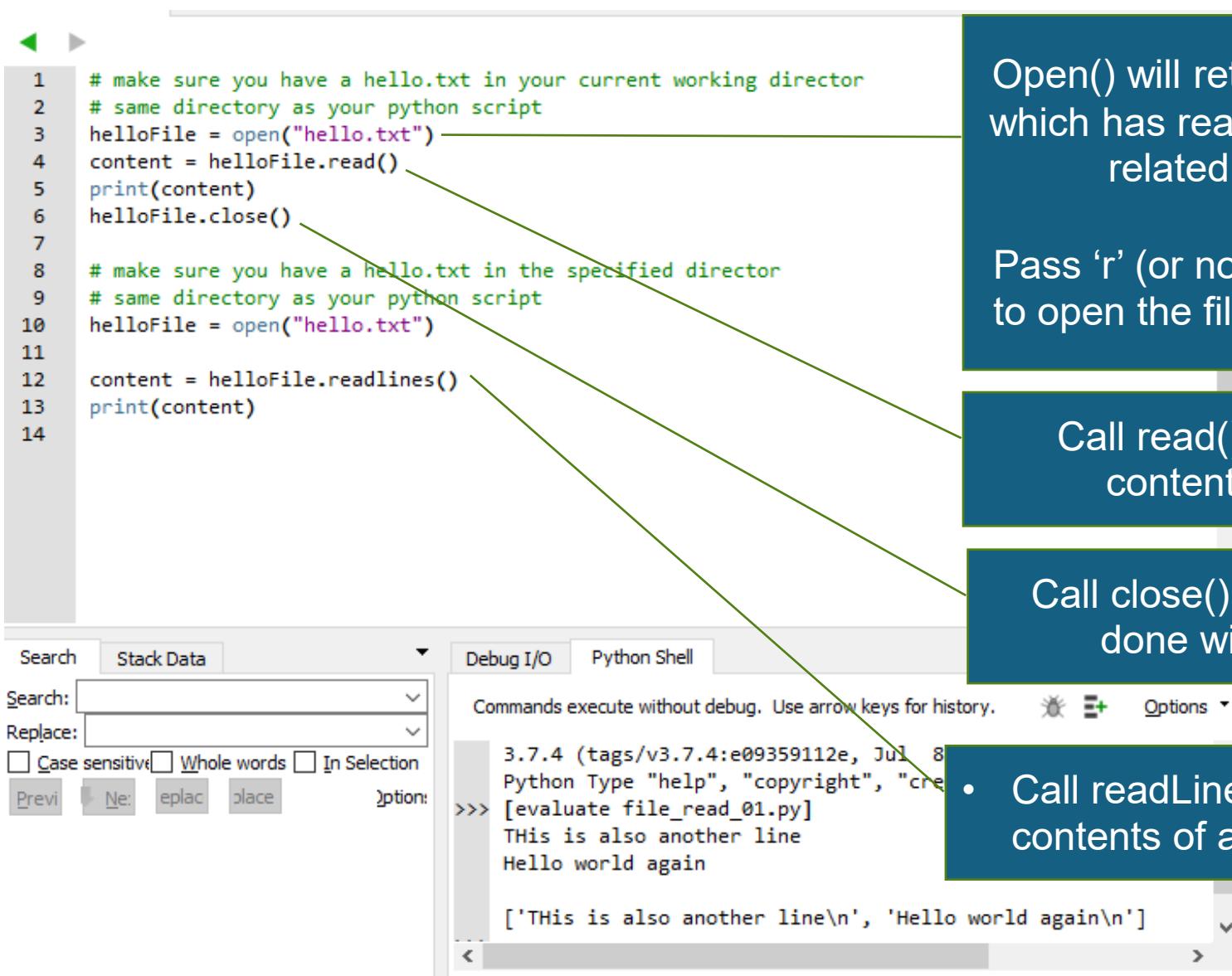
Relative file paths are noted by a **lack of a leading forward slash**.

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
```

Search Stack Data

Search: Replace: Case sensitive Whole words In Selection

Prev 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 2018, 04:14:37) [PyPy 6.1.0 (6.1.0+dev-0-gf98b)] Python Type "help", "copyright", "credits" or "license" for more information.
>>> [evaluate file_read_01.py]
THis is also another line
Hello world again
['THis is also another line\n', 'Hello world again\n']

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

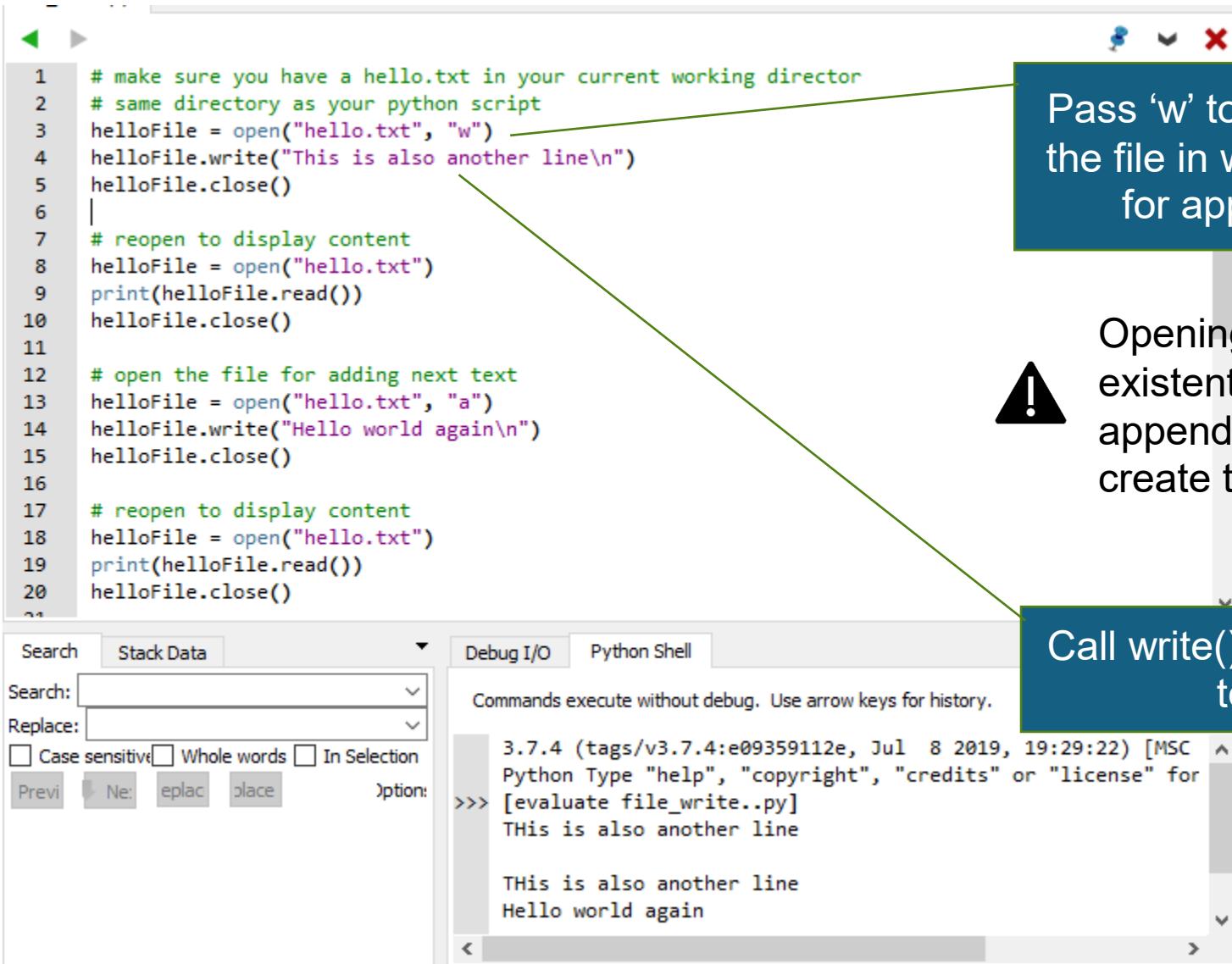
Pass 'r' (or nothing) to open() to open the file in read mode.

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

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()
```

Search Stack Data Debug I/O Python Shell

Search: Replace: Case sensitive Whole words In Selection

Prev Ne: replac place Option:

Commands execute without debug. Use arrow keys for history.

```
>>> [evaluate file_write.py]
THis is also another line

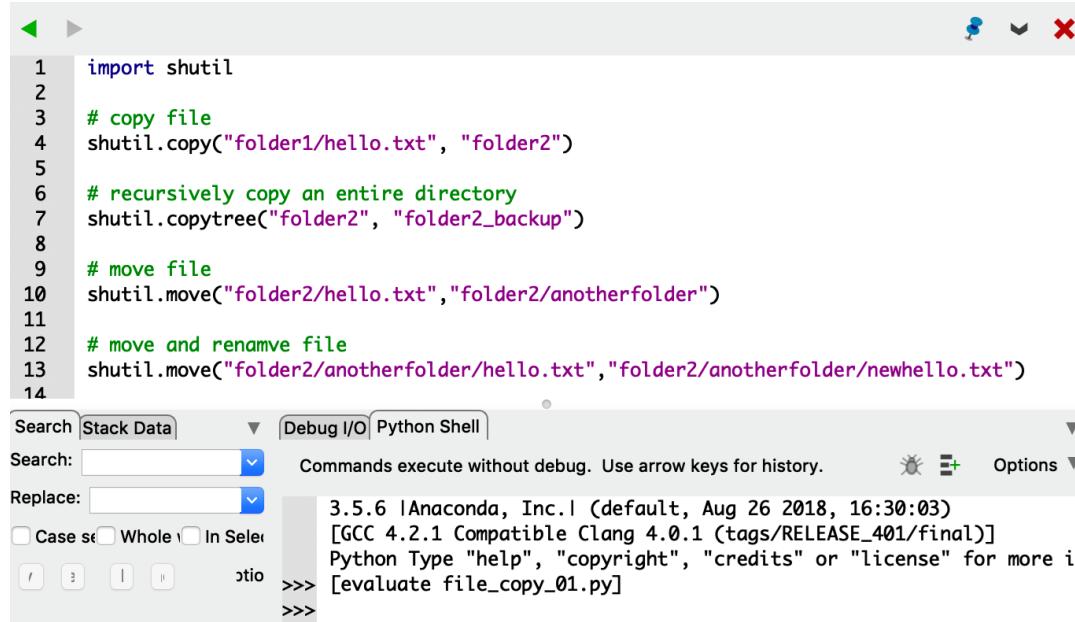
THis is also another line
Hello world again
```

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.

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 rename file
13 shutil.move("folder2/anotherfolder/hello.txt", "folder2/anotherfolder/newhello.txt")
14
```

Search Stack Data ▾ Debug I/O Python Shell
Search: Commands execute without debug. Use arrow keys for history.
Replace: Options ▾
Case se Whole In Select
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 information
>>> [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*).

Deleting files



```
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")
```

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

```
import os

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

- `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



Deleting can be dangerous, so do a dry run first

Use Case Sharing

- Organizing students' submissions into separate folder
 - Class of 25 students

25 folders,
one for
each
student

- Student 120101
- Student 120897
- Student 120904
- Student 121104
- Student 121243
- Student 121550
- Student 121804
- Student 121938
- Student 122061
- Student 122084
- Student 122152
- Student 122263
- Student 122431
- Student 122868
- Student 123295
- Student 123525
- Student 123534
- Student 123673
- Student 123864
- Student 123900
- Student 124059
- Student 124133
- Student 124990
- Student 128079



- Team 1
- Team 2
- Team 3
- Team 4
- Team 5

Student
submissio
n sorted by
teams

Exercise: Coding with ChatGPT

Use ChatGPT to generate the code to rename all the folders based on the requirement shown below

- 📁 18022827_JOEL TAN
- 📁 21019111_RACHEL LEE
- 📁 22120188_DAVIEN CHEAH
- 📁 22000128_LIM SIU FAN
- 📁 23087747_JOEL ONG YU FAN
- 📁 22009205_JOHNNY LOW
- 📁 22001400_SEOW WENG LONG



- 📁 JOEL TAN
- 📁 RACHEL LEE
- 📁 LIM SIU FAN
- 📁 SEOW WENG LONG
- 📁 JOHNNY LOW
- 📁 DAVIEN CHEAH
- 📁 JOEL ONG YU FAN

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:\<python312>\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 staff using company issued laptop with no Admin rights
 - Open command prompt
 - pip install --user <package_name>

Double-Dash

Using pip install

- Install 3rd party library using python code
 - This is the preferred approach if there is multiple installation of python in your laptop, and you are not sure which version of python you are currently working on.

```
import subprocess
import sys

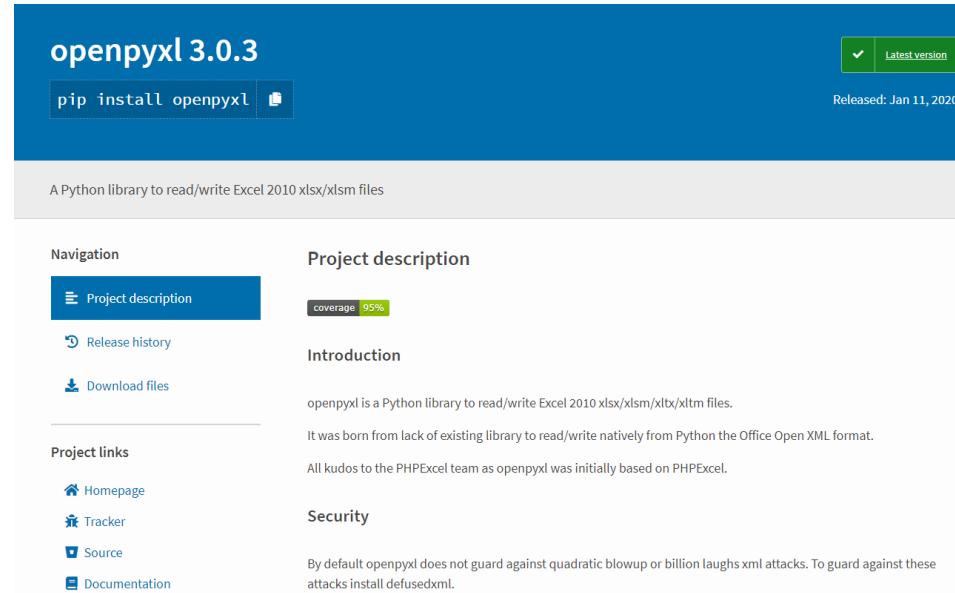
# replace "fpdf" with the library you plan to install
subprocess.check_call([sys.executable, "-m", "pip", "install", "fpdf"])
```



Excel Spreadsheet Manipulation with Python

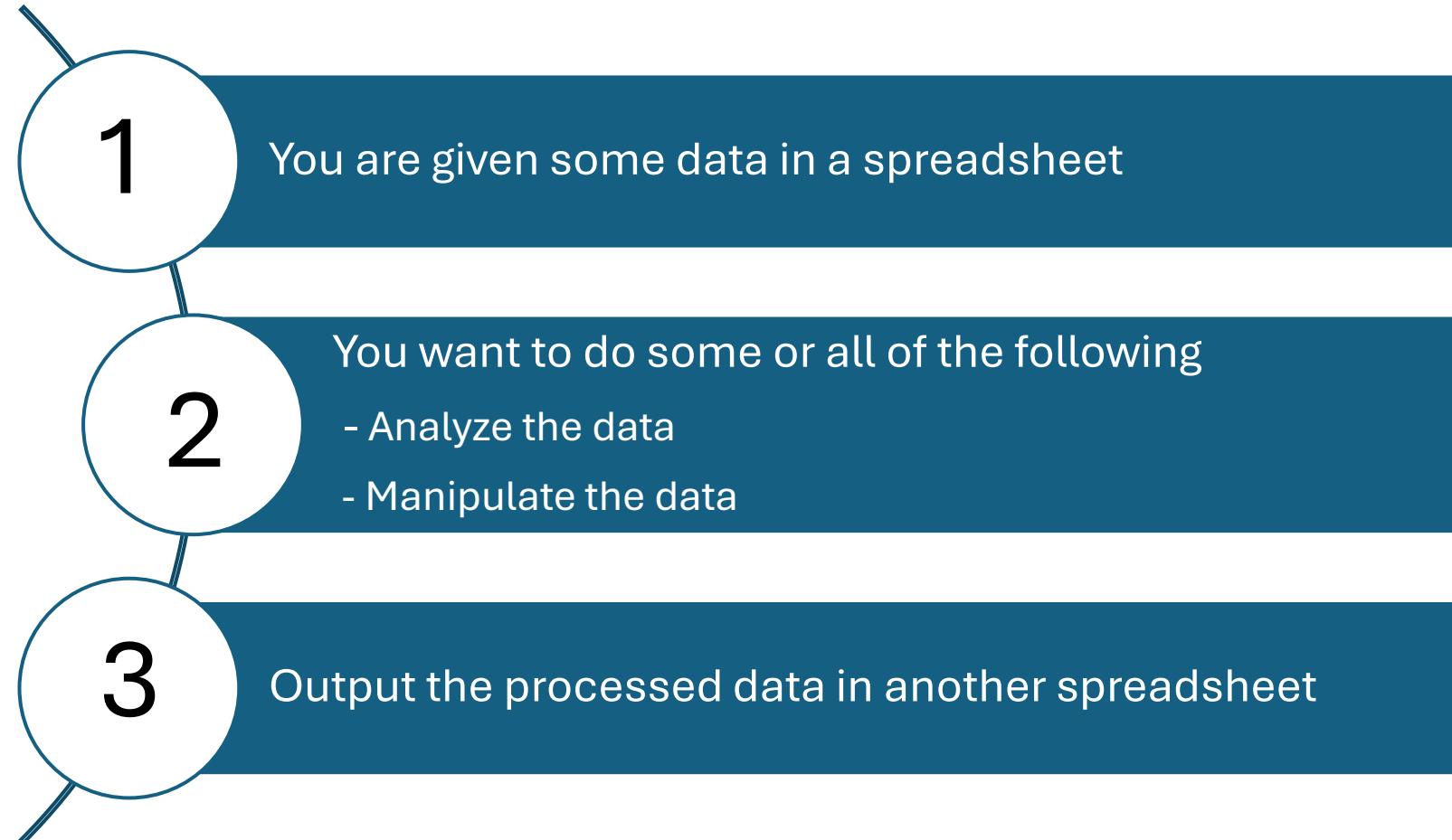
Working with Excel

- Install openpyxl module using “`pip install openpyxl`”
- Full openpyxl documentation:
<https://openpyxl.readthedocs.io/en/stable/index.html>



The screenshot shows the PyPI project page for openpyxl 3.0.3. The top header is blue with the text "openpyxl 3.0.3" and a green "Latest version" button. Below the header, there's a command-line interface section with "pip install openpyxl" and a package icon. To the right, it says "Released: Jan 11, 2020". The main content area has a grey background. It includes a "Project description" section with a "coverage 95%" badge, an "Introduction" section stating "openpyxl is a Python library to read/write Excel 2010 xlsx/xlsm files.", a "Navigation" sidebar with links to "Project description", "Release history", and "Download files", and a "Project links" sidebar with links to "Homepage", "Tracker", "Source", and "Documentation".

Typical Workflow for Excel Automation



Reading Excel file

1) Import openpyxl

```
import openpyxl
```

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

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

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

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

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

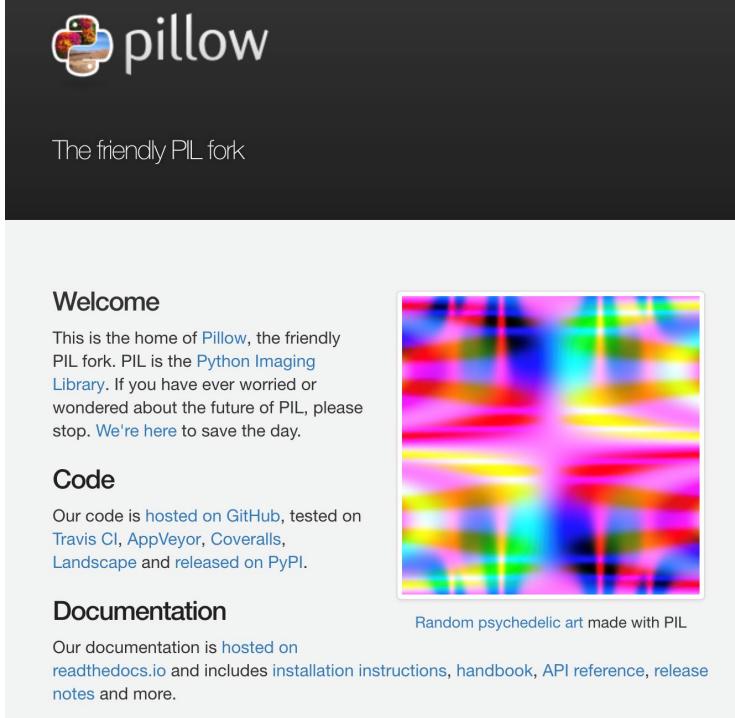
2) Add comments to cell

5) Save the spreadsheet



Image Processing with Python

Image Processing



The screenshot shows the official website for Pillow, the Python Imaging Library fork. The header features the Pillow logo (a colorful icon resembling a flower or a stylized letter 'P') and the text "pillow" in lowercase. Below the logo is the tagline "The friendly PIL fork". The main content area includes sections for "Welcome", "Code", and "Documentation". The "Welcome" section contains a paragraph about the project's purpose and history, mentioning the Python Imaging Library and its future. It also links to GitHub and other continuous integration services. The "Code" section provides links to GitHub and PyPI. The "Documentation" section links to the ReadTheDocs site. A small image of a colorful, abstract pattern is displayed with the caption "Random psychedelic art made with PIL".

Welcome
This is the home of [Pillow](#), the friendly PIL fork. PIL is the [Python Imaging Library](#). If you have ever worried or wondered about the future of PIL, please stop. [We're here](#) to save the day.

Code
Our code is hosted on [GitHub](#), tested on [Travis CI](#), [AppVeyor](#), [Coveralls](#), [Landscape](#) and [released on PyPI](#).

Documentation
Our documentation is [hosted on readthedocs.io](#) and includes [installation instructions](#), [handbook](#), [API reference](#), [release notes](#) and more.

Random psychedelic art made with PIL

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()
```

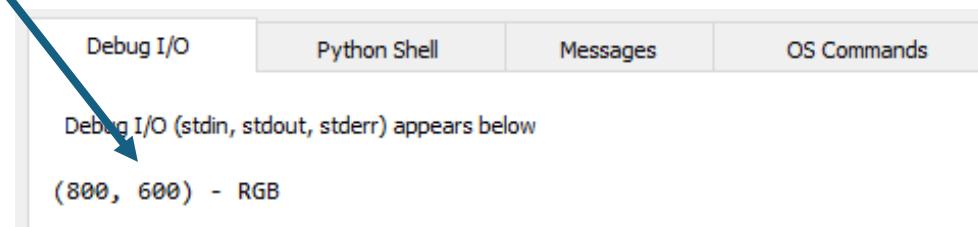


As a start we need to import it:
`import Image`

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`



The screenshot shows a Jupyter Notebook interface with four tabs at the top: Debug I/O, Python Shell, Messages, and OS Commands. A red box highlights the first cell's output, which displays the message "Debug I/O (stdin, stdout, stderr) appears below" followed by the text "(800, 600) - RGB".

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 Processing - Filters



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



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

```
image = ImageOps.grayscale(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 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 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
2. Resize all the file to 60 x 90.
3. 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")
```



Web Automation with Python

Connecting to the Web

- requests – download files and web pages from the Web

```
pip install requests
```

```
import requests
```

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

Get the required information from
the given URL



Economy



Education



Environment



Finance



Health



Infrastructure



Society



Technology



Transport

Connecting to the Web

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

<https://www.site24x7.com/tools/jsonpath-finder-validator.html>

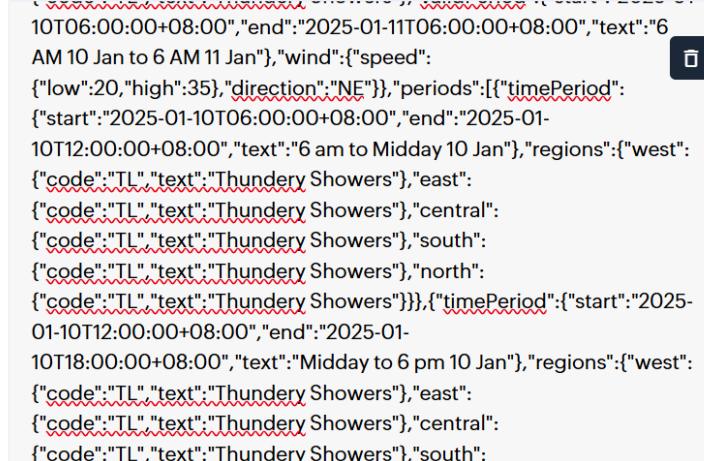
```
import json
import requests

url="https://api-open.data.gov.sg/v2/real-time/api/twenty-four-hr-forecast"

req=requests.get(url)

data = json.loads(req.text)
```

Paste in JSON, [browse](#) or load an [example](#) to begin.



The screenshot shows a code editor with Python code to fetch a JSON API. Below it is a preview area with placeholder text: "Paste in JSON, [browse](#) or load an [example](#) to begin." A large green arrow points from the raw JSON code to the JSON formatter on the right.



The screenshot shows the JSON data after being processed by the formatter. It is now a clean, hierarchical tree structure. The root object contains a "code" field (0), a "data" object with a "records" array, and a "general" object containing a "temperature" object with "low" and "high" fields. The "records" array has one item with a date of "2025-01-10" and an updated timestamp of "2025-01-10T09:00:56+08:00".

```
{
  "code": 0,
  "data": {
    "records": [
      {
        "date": "2025-01-10",
        "updatedTimestamp": "2025-01-10T09:00:56+08:00"
      }
    ]
  },
  "general": {
    "temperature": {
      "low": 23,
      "high": 35
    }
  }
}
```

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"
url="https://api-open.data.gov.sg/v2/real-time/api/twenty-four-hr-forecast"
```

```
req=requests.get(url)
```

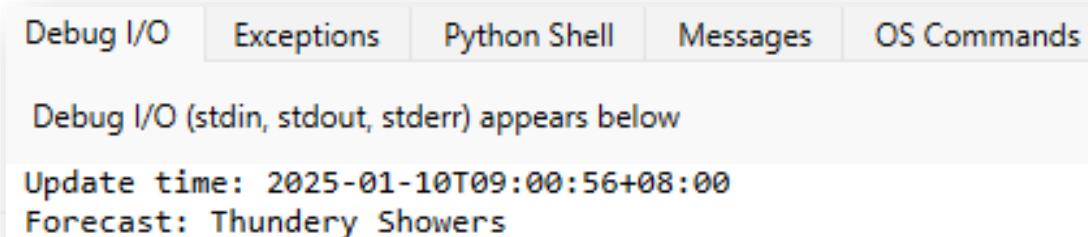
```
data = json.loads(req.text)
```

```
# print update timestamp
```

```
update_time = data["data"]["records"][0]["updatedTimestamp"]
print("Update time: " + update_time)
```

```
# print forecast
```

```
forecast = data["data"]["records"][0]["general"]["forecast"]["text"]
print("Forecast: " + forecast)
```



Path Notation

Dot Bracket

Keys options

Single Quotes Double Quotes Without Quotes

[\"data\"]\[\"records\"]\[0\]\[\"updatedTimestamp\"]

```
{
  code: 0,
  data: {
    records: [
      {
        date: "2025-05-02",
        updatedTimestamp: "2025-05-02T13:30:51+08:00"
      }
    ]
  }
}
```

Exercise

- Car Park Availability Data:
 - 1.url: <https://api.data.gov.sg/v1/transport/carpark-availability>
 - 2.Write the code to get the timestamp and the Carpark Number for the first set of carpark data.
 - 3.Print out the result as shown.

```
{  
  - items: [  
    - {  
      timestamp: "2021-08-19T13:23:28+08:00",  
      - carpark_data: [  
        - {  
          - carpark_info: [  
            - {  
              total_lots: "105",  
              lot_type: "C",  
              lots_available: "0"  
            }  
          ],  
          carpark_number: "HE12",  
          update_datetime: "2021-08-19T13:10:03"  
        },  
        - {  
          - carpark_info: [  
            - {  
              total_lots: "105",  
              lot_type: "C",  
              lots_available: "0"  
            }  
          ],  
          carpark_number: "HE12",  
          update_datetime: "2021-08-19T11:49:27+08:00"  
        }  
      ]  
    }  
  ]  
}
```

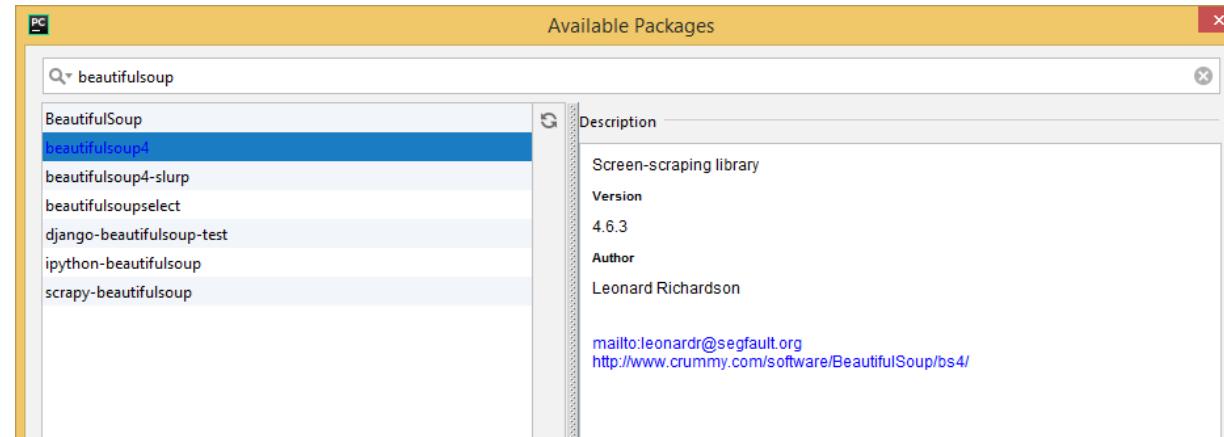
Update time: 2021-08-19T11:49:27+08:00
Carpark number: HE12

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/landon-long-dining-table-coffee.html>

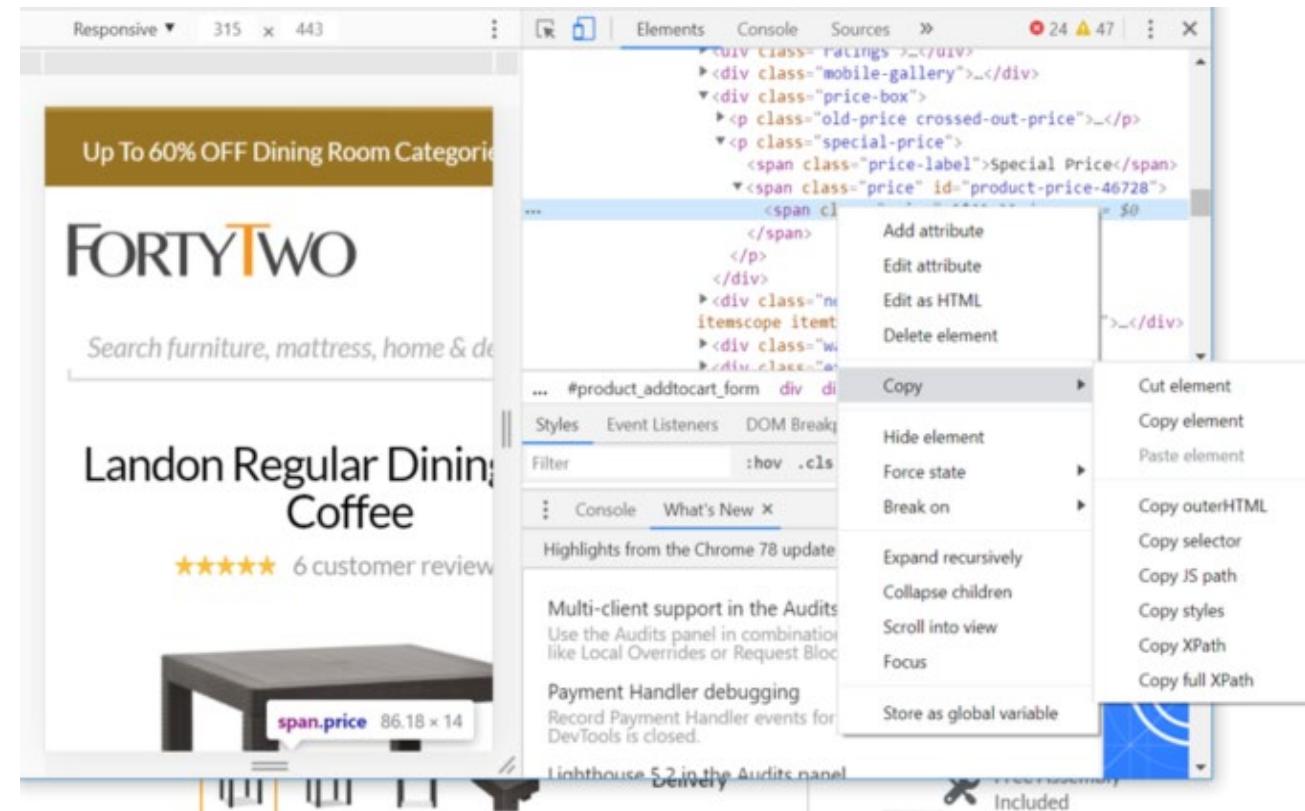
The screenshot shows the product page for the Landon Long Dining Table Coffee on the fortytwo website. The page includes a navigation bar with links for Furniture, Mattresses, Bed & Bath, Décor, Kitchen & Tabletop, Lighting & Fans, Kids, Home Essentials, Sale, and Trending. A search bar is at the top right. The main content features a large image of the dark brown dining table with a textured surface and four thick legs. To the left, there are smaller images of the table from different angles. The product title is "Landon Long Dining Table Coffee" with a 4.65 rating and 234 sold. It offers a 3-year warranty. The price is listed as S\$139.90, with a note that it was S\$247.90 previously. Payment options include 4 X S\$34.98 with Grab. A message indicates the item is temporarily out of stock. A "Notify me when its back" button is available. Below the main product information, there are sections for Product Description, Product Specifications, and Customers' Reviews. A live chat icon is visible in the bottom right corner.

Connecting to the Web

- Get the url

<https://www.fortytwo.sg/landon-long-dining-table-coffee.html>

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



Connecting to the Web



```
from urllib.request import Request, urlopen
from bs4 import BeautifulSoup

site = "https://www.fortytwo.sg/landon-long-dining-table-coffee.html"

hdr = {'User-Agent': 'Mozilla/5.0'}
req = Request(site, headers=hdr)
page = urlopen(req)
soup = BeautifulSoup(page, 'html.parser')

#new-price
new_price_element = soup.find(class_="new-price")

print("Current Price: " + new_price_element.text)

#old-price
old_price_element = soup.find(class_="old-price")

print("\nOld Price: " + old_price_element.text)
```

Debug I/O Exceptions Python Shell Messages

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

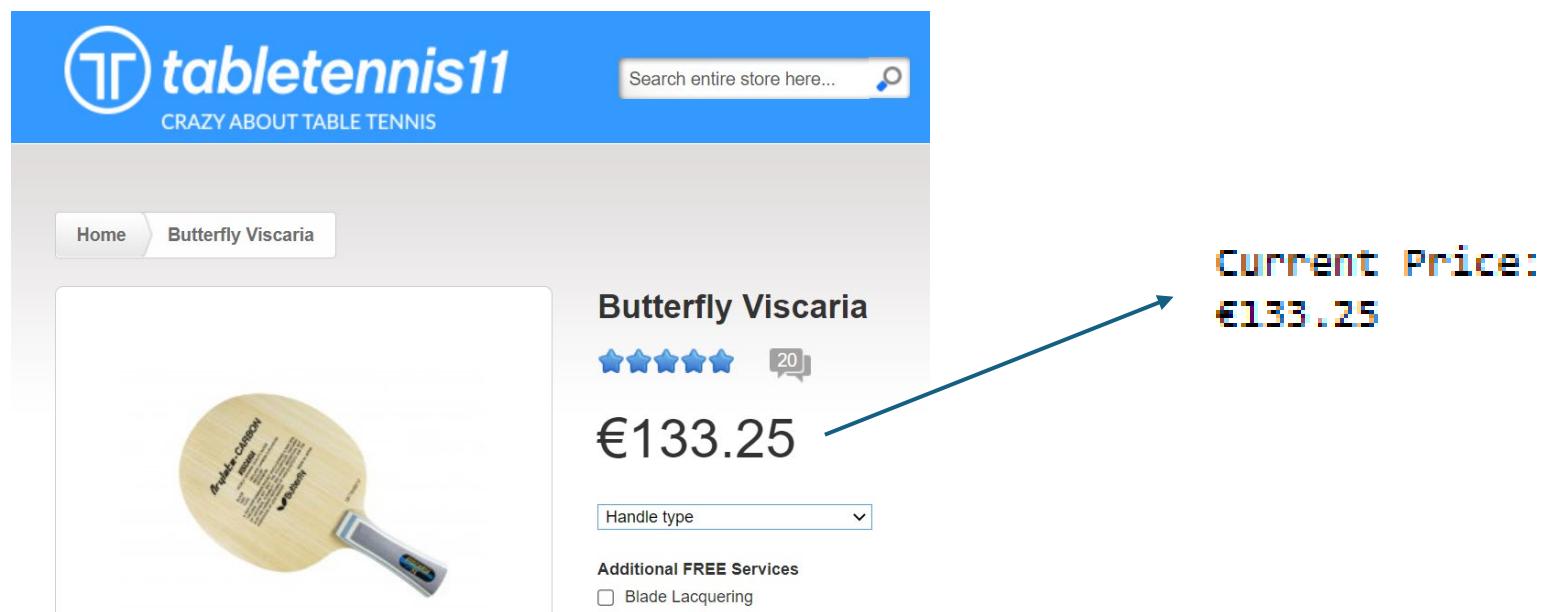
Current Price: SGD139.90

Old Price: SGD247.90

Exercise

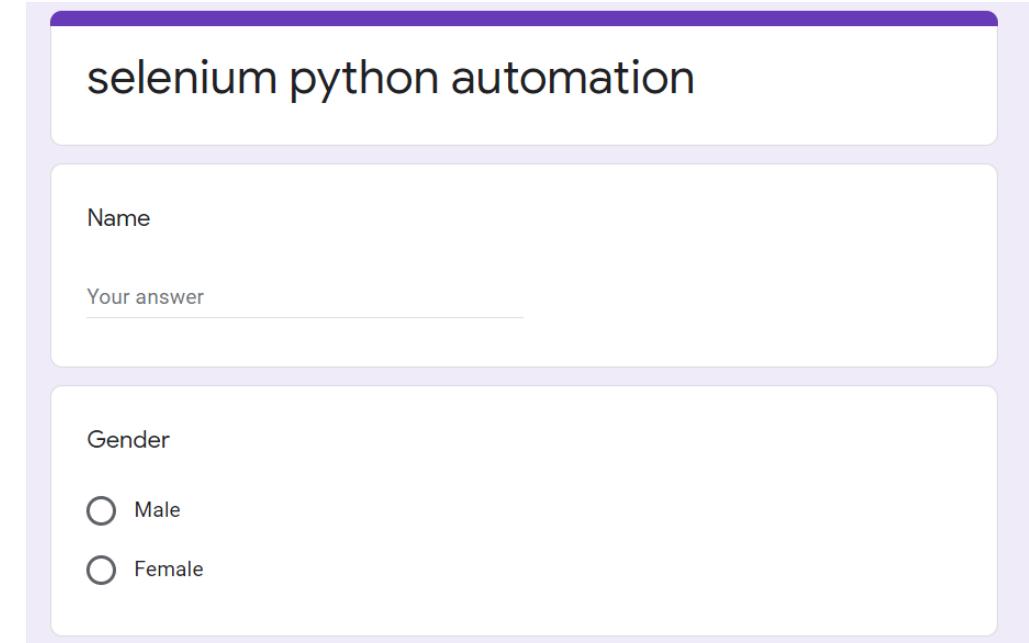
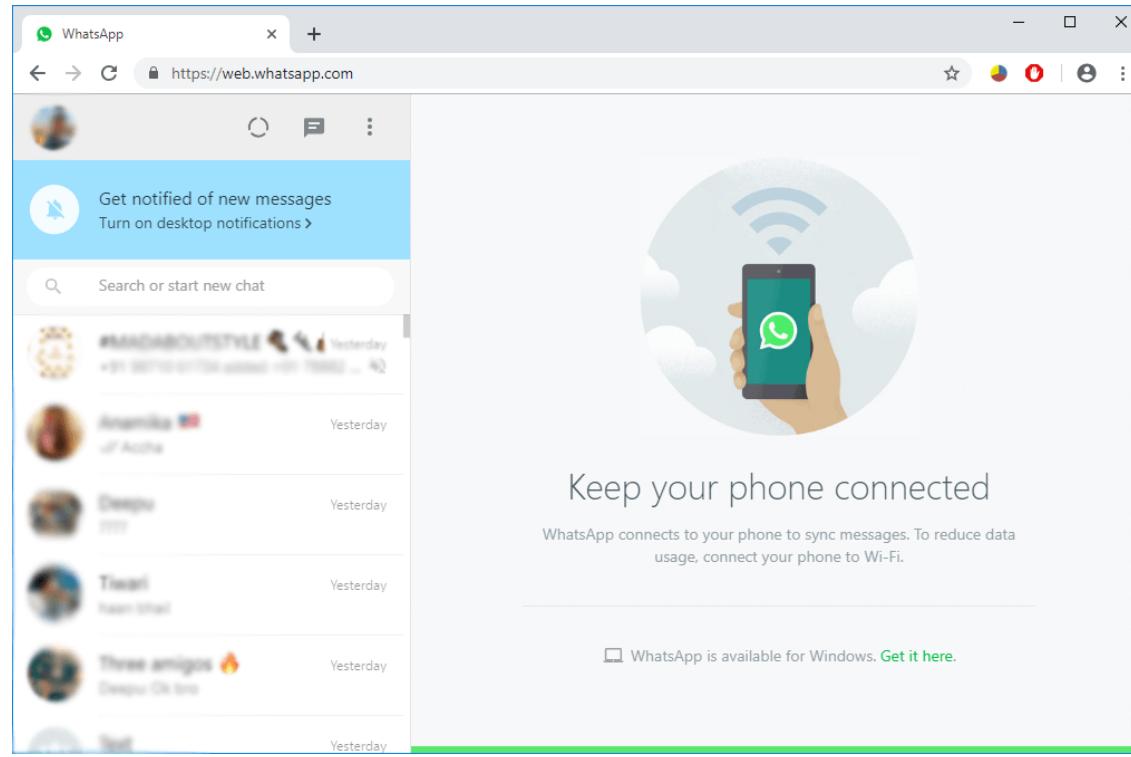
- Table Tennis Bat Price:

- 1.url: https://www.tabletennis11.com/other_eng/butterfly-viscaria
- 2.Write the code to get the price of the table tennis bat
- 3.Print out the result as shown.



Sharing other Use Cases

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



A screenshot of a Google Form titled "selenium python automation". The form consists of two main sections. The first section, labeled "Name", contains a text input field with the placeholder "Your answer". The second section, labeled "Gender", contains two radio button options: "Male" and "Female".

Other public APIs

- Open Weather API
 - openweathermap.org/api
- TMDB – Movie API
 - themoviedb.org/documentation/api
- News API
 - newsapi.org
- Jokes API
 - sv443.net/jokeapi/v2
- Quotes API
 - api.quotable.io

LTA DataMall



- LTA Datamall:
 - <https://datamall.lta.gov.sg/content/datamall/en/dynamic-data.html>
- Sample Dataset available
 - Bus Arrival
 - Bus Routes
- Get Bus Arrival Time
 - Get bus stop number:
 - https://www.transitlink.com.sg/eservice/eguide/service_route.php
 - Get buses and its arrival time

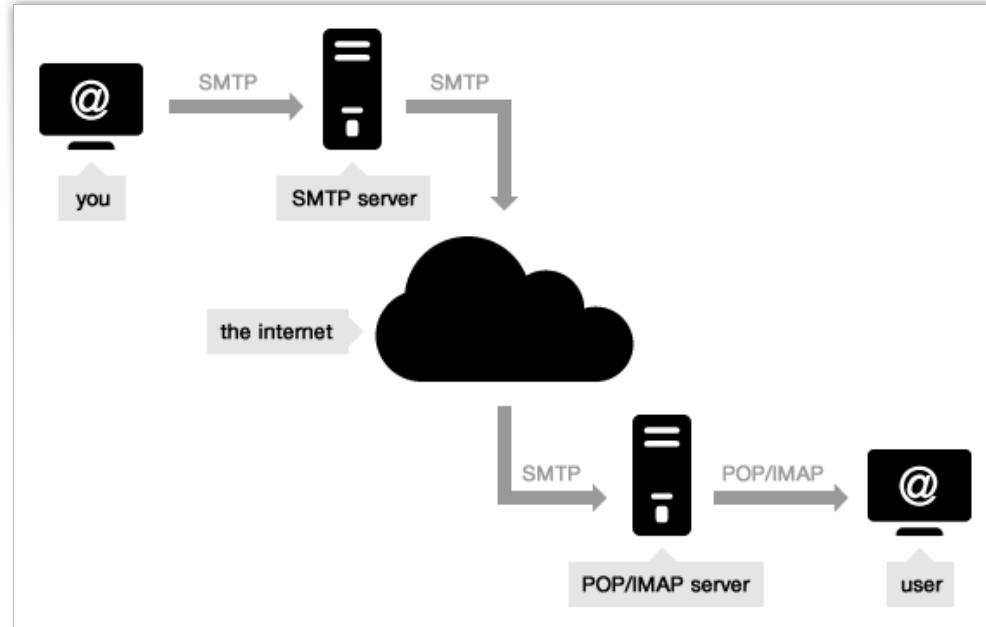


About Us Static Datasets Dynamic Datasets App Zone



Email Automation with Python

Send Email

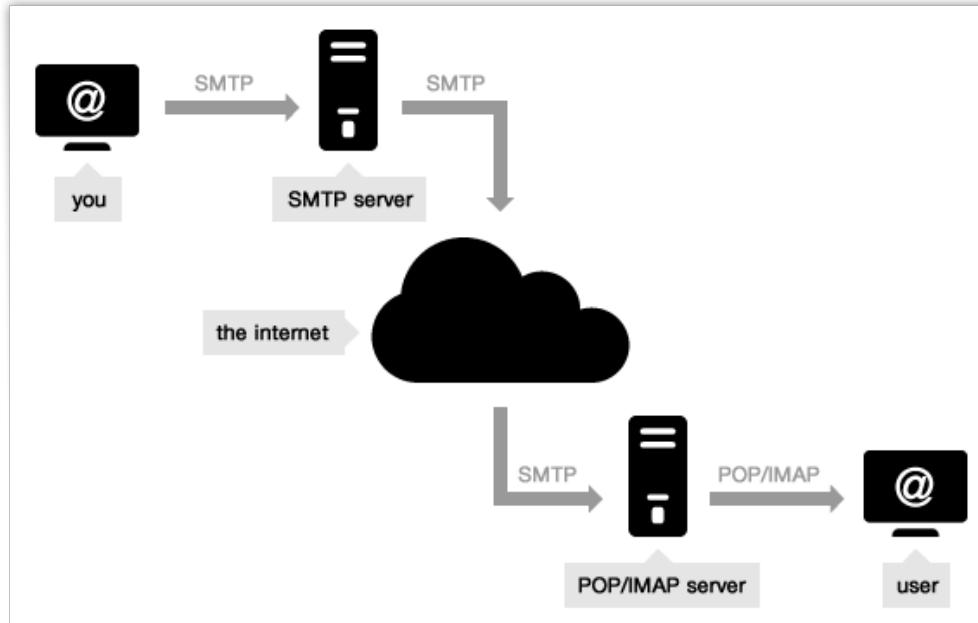


- SMTP (Simple Mail Transfer Protocol) is used for sending and delivering from a client to a server via port 25, 465 or 587: it's the **outgoing server**.
- IMAP and POP are two methods to access email. IMAP is the recommended method when you need to check your emails from several different devices,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

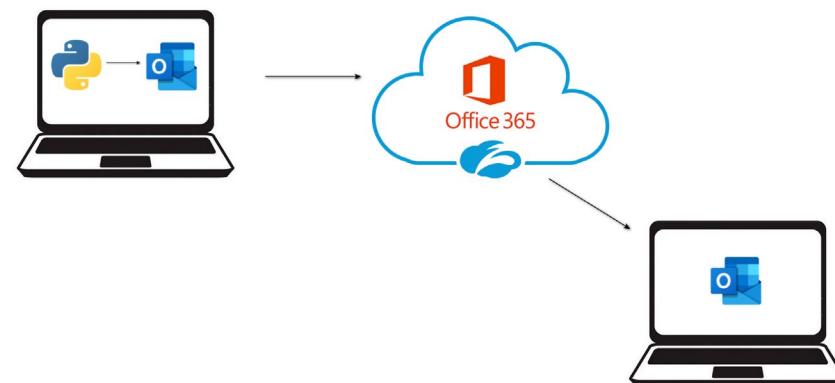
Send Email



- **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.

pywin32 package

- This assumes that we are using MS Outlook on the machine the python code is run
- pywin32 package is used to allow Python to access MS Outlook
- Python is not accessing the Outlook server directly, but it is using existing account set up in the MS Outlook



Send Email using Outlook

- The code above sends a simple HTML (meaning we can have tables, text formatting in it)
- No credentials are needed for the Python program as Python uses MS Outlook

```
## REMEMBER to perform a pip install pywin32 first.

import win32com.client

outlook = win32com.client.Dispatch('outlook.application')
mail = outlook.CreateItem(0)
mail.To = 'tan_kok_cheng@rp.edu.sg'
mail.Subject = 'Demo Email on 20 Nov 2024'
mail.HTMLBody = '<h3>This is an email sent from Outlook client on 20 Nov 2024...</h3>'
#mail.Body = "This is the normal body"
#mail.Attachments.Add('c:\\sample.xlsx')
#mail.Attachments.Add('c:\\sample2.xlsx')
#mail.CC = 'somebody@company.com'
mail.Send()
print("Sent")
```

Create appt using Outlook

- The code above creates a calendar appointment
- No credentials are needed for the Python program as Python uses MS Outlook

```
import win32com.client

outlook = win32com.client.Dispatch('outlook.application')
appt = outlook.CreateItem(1)
appt.Start = "2024-11-20 16:00"
appt.Subject = "Online Sales Starts!"
appt.Duration = 60
appt.Location = "Singapore"

#mail.Attachments.Add('c:\\sample.xlsx')
#mail.Attachments.Add('c:\\sample2.xlsx')
#mail.CC = 'somebody@company.com'
appt.MeetingStatus = 1

appt.Recipients.Add("tan_kok_cheng@rp.edu.sg")

appt.Save()
appt.Send()
print("Appt made")
```

End of Day 2

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

Thank you !

QUESTIONS ?

