

Introduction

hello welcome to outlook notebook this notebook aims to explain how to run this.

firstly we have requirement this code will work from python 3.6.5 and up you'll need the following libraries:

- tkinter
- win32com

to install these in command prompt (windows power shell or anaconda prompt)

for reference look at the [mircosoft API](#)

▼ Library Installation Guide

tkinter

To install the `tkinter` library for Python's GUI development, you can use the following command:


```
pip install tk
```

win32com

```
pip install pywin32
```

this is a simple install libraries

```
import re
import tkinter as tk
from tkinter import filedialog
import pandas as pd
import win32com.client
import os
from datetime import datetime, timedelta
```

for those of you who are familiar with Tkinter this is very familiar this will make a popup like the following:  this is the part that make it possible and what its doing is basically calling a

dialog box (if your old enough youll remember what these where)basic it a way of prompting a user to select something:

```
class OutlookWindow:
    def __init__(self):
        self.root = tk.Tk()
        self.root.withdraw()

    def ask_directory(self):
        selected_directory = filedialog.askdirectory()
        return selected_directory

    def mainloop(self):
        self.root.mainloop()
```

below is for more email like application havent made anything on it but it there just sitting there 🙄



```
class Outlook_mange_api:
    def __init__(self) -> None:
        pass
```

now for the main meat of the project lets start this with how to class is called so every object has a output_directory ,api (so you can try different apis), days received to control where to download, a not allowed extension's

```
class Outlook:
    def __init__(self, output_directory, api, days_received=1, not_allowed=False):
        self.dir = output_directory
        self.api = api
        self.day_receive = days_received
        self.not_allowed = not_allowed
```

the following code is simply a check of each variables put into the object

the filename is there as an option to read from so when its not the first you run this it wont ask you how long ect this option isn't working but the foundations are made to change it

```

def check_variables(self, filename=None):
    if filename is None:
        if self.api != "Mapi":
            self.api = "Mapi"

        self.day_receive = int(input("Please enter the number of how many days: "))
        directory_pattern = r'^[A-Za-z0-9_\-. /]+$'

        if re.match(directory_pattern, self.dir):
            outlook_window = OutlookWindow()
            selected_directory = outlook_window.ask_directory()
            self.dir = selected_directory

        if self.not_allowed:
            self.not_allowed = [".png", ".jpg", ".gif", ".ARTask"]

        attributes = {
            'dir': [self.dir],
            'api': [self.api],
            'day_receive': [self.day_receive],
            'not_allowed': [self.not_allowed]
        }

        df = pd.DataFrame.from_dict(attributes)
        return df
    else:
        attributes = pd.read_csv(filename + r".csv", index_col=None)
        return attributes

```

save attribute is where the save to file is happening maybe a json or txt would do better here

```

def save_attributes(self, filename):
    attributes = vars(self) # Get the instance's attributes as a dictionary
    df = pd.DataFrame.from_dict(attributes)
    df.to_csv(filename + r".csv", index=False)

def download_attachments(Sef,dataframe):
    not_allowed_extensions=dataframe.iloc[0,3]
    api=dataframe.iloc[0,1]
    output_dir=dataframe.iloc[0,0]
    day_recived=dataframe.iloc[0,2]

```

once the data has been loaded we look take a look at the mircosoft api closely if you want to download off certain folders youll have to look this up but this can be added to the class as manage

folder inbox

```

Outlook =win32com.client.Dispatch("Outlook.Application").GetNamespace(api)
inbox = Outlook.GetDefaultFolder(6)
received_dt = datetime.now() -timedelta(days=int(day_recived))
received_dt = received_dt.strftime('%d/%m/%Y %H:%M %p')
messages = inbox.Items
messages = messages.Restrict("[ReceivedTime] >= '" + received_dt + "'")
# add in the furture to let the user add file strings to restrict

```

sometimes when runing code we dont want errors but in this we expect there to be errors for example when the extentions are restricted or simply because the meessage is empty anyway we want our program to not stop due to an error

```

try:
    for message in list(messages):
        try:
            s= message.sender
            for attachment in message.Attachments:
                if not (any(attachment.FileName.lower().endswith(ext) for ext in not_
                    attachment.SaveASFile(os.path.join(output_dir,attachment.FileName

                print(f"attachment {attachment.FileName} from {s} saved")

            except Exception as e: print("error when saving the attachment:" + str(e))
        except Exception as e: print("error when processing emails messages:" + str(e))

```

now its a simple case of using this object with 6 lines of code

```

# Example usage
obj = Outlook("output", "API key", 1, True)
file_path = "attributes"
if not os.path.isfile(file_path):
    file_path= None
data =obj.check_variables(file_path)
obj.save_attributes(r"attributes")
obj.download_attachments(data)

```

