✨ Member-only story

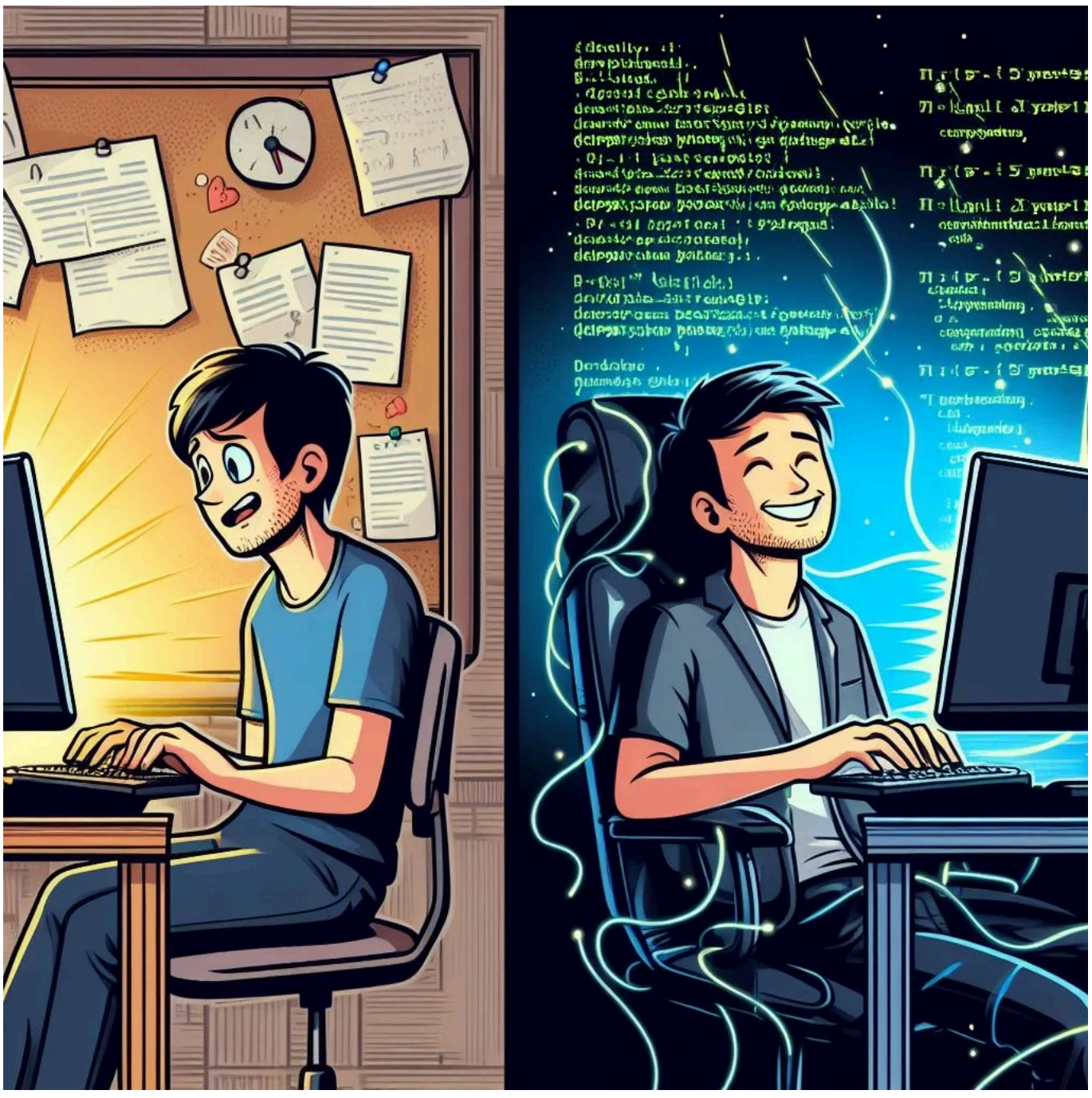
AUTOMATION SCRIPTS YOU NEED TO TRY (COLLECTION)

17 Mindblowing Python Automation Scripts I Use Everyday

Scripts That Increased My Productivity and Performance

Abhay Parashar · Follow

Published in The Pythoneers · 14 min read · Jul 4, 2024



Me Before and After using Automation Scripts — AI Generate Images using Microsoft Copilot

I've been using Python for almost 5 years now and the one thing that still attracts me and gives me motivation for more research is its ability to Automate things. For the past 1 year, I have been exploring the Automation side of Python and have discovered some amazing Python packages, facts, and interesting scripts. In this blog, I will be sharing a bunch of Automation scripts that I use daily and have Increased My productivity and performance.

1. Speakify

I love books but hate reading them on my own, rather I love listening to them. This automation script is a lifesaver for me and I use it a lot to listen to PDFs and convert them into AudioBooks to listen to later.

Yup!!! That's me enjoying my AudioBook 😎

```
import PyPDF2
import pyttsx3

# Open the PDF file (Enter Path To Your PDF)
file = open('story.pdf', 'rb')
readpdf = PyPDF2.PdfReader(file)

# Initialize text-to-speech engine
speaker = pyttsx3.init()
rate = speaker.getProperty('rate')   # Get current speaking rate
speaker.setProperty('rate', 200)

volume = speaker.getProperty('volume')
speaker.setProperty('volume', 1)  # Set volume level (0.0 to 1.0)

# Get and set a different voice
voices = speaker.getProperty('voices')
for voice in voices:
    if "english" in voice.name.lower() and "us" in voice.name.lower():
        speaker.setProperty('voice', voice.id)
        break
# Iterate over each page in the PDF
for pagenumber in range(len(readpdf.pages)):
    # Extract text from the page
    page = readpdf.pages[pagenumber]
    text = page.extract_text()

    # Use the speaker to read the text
    # speaker.say(text)
    # speaker.runAndWait()

# Save the last extracted text to an audio file (if needed)
speaker.save_to_file(text, 'story.mp3')
speaker.runAndWait()

# Stop the speaker
speaker.stop()

# Close the PDF file
file.close()
```

Real-Life Applications

- **Accessibility for the Visually Impaired**: Providing audio versions of written content to help visually impaired individuals access information easily.
- **On-the-Go Learning**: Allowing users to listen to articles or textbooks while commuting or exercising.
- **Language Learning**: Helping language learners improve their listening skills by providing audio versions of texts.
- **Education**: Providing students with audio versions of their reading materials for more flexible study options.

2. TabTornado

Before writing this script I used to bookmark things that I've found interesting to read next day, however after few weeks I realized that my bookmark shelf is getting bigger day by day and Everyday I used to had a hard time finding my new bookmarks. So I figured out a pythonic way to tackle this problem. With this automation script, I can just copy and paste all the links and then open them with a single click.

One Click Is All It Takes!!!

```
import webbrowser
with open('links.txt') as file:
    links = file.readlines()
    for link in links:
        webbrowser.open('link')
```

Application

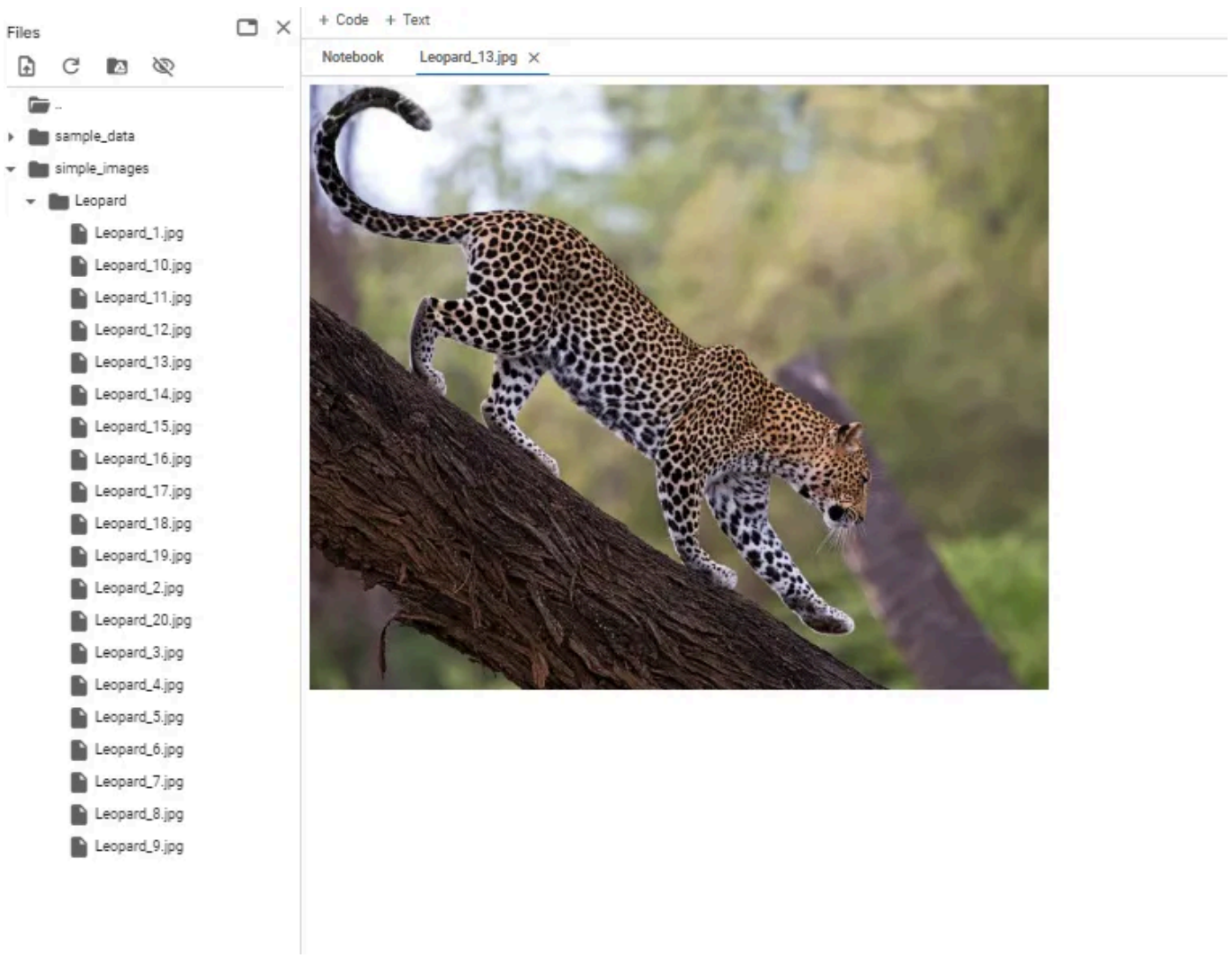
Increased Work Efficiency: Professionals who need to check multiple work-related sites can streamline their routine, focusing on content rather than the process of opening links.
Learning and Development: Online learners can open all course materials, articles, and resources at once, making their study sessions more efficient.

3. PicFetcher

Gathering a large amount of image data is a key challenge in computer vision projects. As Andrew Ng points out, having a big dataset can be more important than the specific algorithm used. High-quality data is essential for improving the performance and accuracy of machine learning models. This automation script makes the process easier by letting you download a set number of images from the web in just minutes with minimal coding effort.

```
# Importing the necessary module and function
from simple_image_download import simple_image_download as simp

# Creating a response object
response = simp.simple_image_download

## Keyword
keyword = "Dog"
```

```
# Downloading images
try:
    response().download(keyword, 20)
    print("Images downloaded successfully.")
except Exception as e:
    print("An error occurred:", e)
```



Script Output — Screenshot Taken By Author

Applications

- Building Computer Vision Datasets, Banner image Content Creation, Marketing Campaigns, Academic Research, and many more.

4. PyInspector

Every developer knows the frustration of tracking down bugs in Python code, often getting caught in a web of errors. Writing clean and efficient code is crucial, but manually analyzing code quality can be daunting. This automation script uses the Pylint and Flake8 packages to thoroughly review your code, compare it against coding standards, and pinpoint logical errors. It ensures your code follows industry best practices and remains error-free.

```
import os
import subprocess

def analyze_code(directory):
    # List Python files in the directory
    python_files = [file for file in os.listdir(directory) if file.endswith('.py

    if not python_files:
        print("No Python files found in the specified directory.")
        return

    # Analyze each Python file using pylint and flake8
    for file in python_files:
```

```
        print(f"Analyzing file: {file}")
        file_path = os.path.join(directory, file)

        # Run pylint
        print("\nRunning pylint...")
        pylint_command = f"pylint {file_path}"
        subprocess.run(pylint_command, shell=True)

        # Run flake8
        print("\nRunning flake8...")
        flake8_command = f"flake8 {file_path}"
        subprocess.run(flake8_command, shell=True)

if __name__ == "__main__":
    directory = r"C:\Users\abhay\OneDrive\Desktop\Part7"
    analyze_code(directory)
```

```
Analyzing file: autocomplete.py

Running pylint...
************* Module autocomplete
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:2:0: C0301: Line too long (232/100) (line-too-long)
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:14:0: C0304: Final newline missing (missing-final-newline)
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:7:0: C0116: Missing function or method docstring (missing-function-docstring)
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:7:23: W0621: Redefining name 'query' from outer scope (line 13) (redefined-outer-name)
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:9:4: C0103: Variable name "r" doesn't conform to snake_case naming style (invalid-name)
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:9:8: W3101: Missing timeout argument for method 'requests.get' can cause your program to hang indefini
tely (missing-timeout)
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:13:0: C0103: Constant name "query" doesn't conform to UPPER_CASE naming style (invalid-name)

-------------------------------------------------------------------
Your code has been rated at 1.25/10 (previous run: 1.25/10, +0.00)


Running flake8...
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:2:80: E501 line too long (232 > 79 characters)
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:7:1: E302 expected 2 blank lines, found 0
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:8:80: E501 line too long (86 > 79 characters)
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:13:1: E305 expected 2 blank lines after class or function definition, found 0
C:\Users\abhay\OneDrive\Desktop\Part7\autocomplete.py:14:26: W292 no newline at end of file
```

Script Output — Analyzing Quality of one of my old script

5. DataDummy

Whether you're a data scientist needing sample data to test models or just looking to fill out an unnecessary form with random information, this Python automation script proves incredibly useful. It generates realistic-looking but entirely artificial datasets, perfect for testing, development, and simulation purposes. This tool can quickly create names, emails, phone numbers, and more, providing a versatile solution for various data generation needs.

```
import pandas as pd
from faker import Faker
import random

fake = Faker()

def generate_fake_data(num_entries=10):
    data = []

    for _ in range(num_entries):
        entry = {
            "Name": fake.name(),
            "Address": fake.address(),
            "Email": fake.email(),
            "Phone Number": fake.phone_number(),
            "Date of Birth": fake.date_of_birth(minimum_age=18, maximum_age=65).
            "Random Number": random.randint(1, 100),
            "Job Title": fake.job(),
            "Company": fake.company(),
            "Lorem Ipsum Text": fake.text(),
        }
```

```
        data.append(entry)

    return pd.DataFrame(data)

if __name__ == "__main__":
    num_entries = 10  # You can adjust the number of entries you want to generat
    fake_data_df = generate_fake_data(num_entries)



## Dataframe with Fake Data
fake_data_df
```

	Name	Address	Email	Phone Number	Date of Birth	Random Number	Job Title	Company	Lorem Ipsum Text
0	Denise Gonzalez	945 Thomas Isle\nCalvintown, NV 07743	courtneymelton@example.com	(922)933-8709	1984-08-17	81	Teacher, English as a foreign language	Zuniga-Conley	Group century blood tree. Himself bit common v...
1	Sierra Vaughan	PSC 8558, Box 7830\nAPO AP 56144	lacey20@example.net	(816)253-5701	1991-09-27	23	Industrial buyer	James, Porter and Holt	Watch popular join today. Impact region genera...
2	Jamie Martin	8049 Kari Turnpike\nLaurietown, AL 17515	gwiggins@example.net	(788)605-9204	1967-02-03	50	Warden/ranger	Martinez, Haas and Glover	Paper mother nor he heavy write loss. Final fo...
3	Austin White	PSC 9672, Box 7607\nAPO AE 28084	christina85@example.net	001-657-238-8745x84597	1989-11-08	80	Historic buildings inspector/conservation officer	Hines, Shaw and Harper	Including sound size try religious again. Boar...
4	Chris Chandler	USNS Miles\nFPO AA 33739	tsmith@example.net	+1-338-453-5130x10043	2003-01-24	38	Operations geologist	Cole, Stewart and Miller	Country wide dark east line. Lay later speech ...
5	Mary Lopez	372 Jacob Wall Apt. 759\nPort Andreamouth, VA ...	russelldonna@example.com	(578)202-0625	1996-12-09	16	Engineer, agricultural	Burns, Wilson and Harris	Throw team money hard. Through admit national ...
6	Jordan Nash	PSC 4103, Box 5938\nAPO AA 94980	holmesmichelle@example.net	8339935122	1988-08-29	82	Best boy	Bautista, Ross and Escobar	Education camera cut. Way tax really internati...
7	Jeanette Blair	1447 Jeff Plains\nWest Jessica, LA 38200	sallen@example.net	462.656.4264	1975-05-02	67	Technical author	Gibbs, Lawson and Lawson	Series black probably class writer before spen...
8	Sheila Moore	94780 Sanchez Row\nWilsonland, MD 73657	jonesmichael@example.com	+1-842-559-7005	1994-03-21	14	Insurance account manager	Hamilton Group	Kitchen short new ball operation best. Perhaps...
9	Linda Washington	45416 Donald Isle\nGriffinfort, LA 56864	anna75@example.net	949-895-3758x9384	1973-04-11	75	Forensic scientist	Turner and Sons	Drop section look today. Itself vote turn grou...

Output from the Above Script

6. BgBuster

This automation script has become an essential part of my daily toolkit. As a writer, I frequently work with images and often need them without backgrounds. While the re are many online tools available for this task, I have concerns about the privacy and security of my images on the internet. This Python script utilizes the rembg package to remove image backgrounds locally, ensuring that my pictures remain secure and private.

My Honest Reaction When I First Discovered This Python Script — Tenot GIF

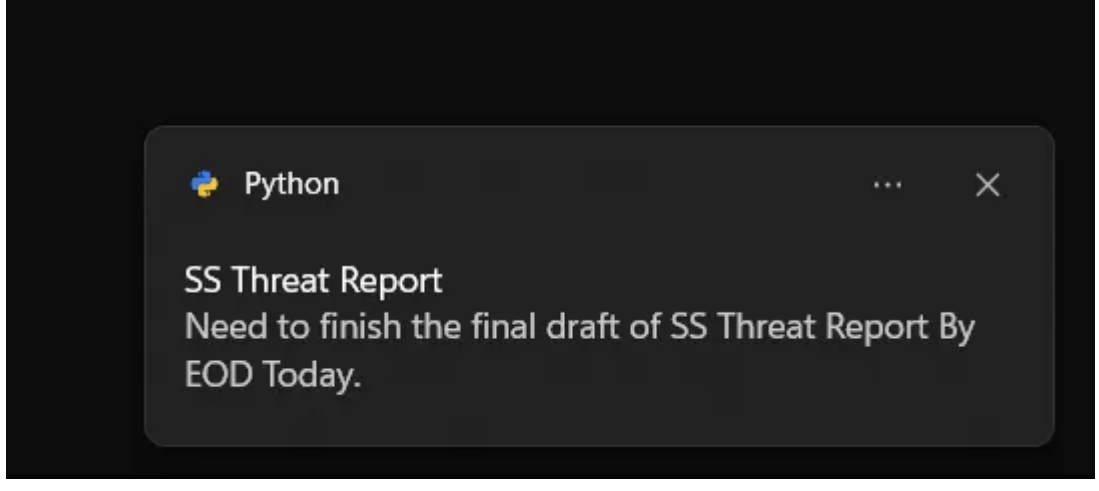
```
from rembg import remove
from PIL import Image

## Path for input and output image
input_img = 'monkey.jpg'
output_img = 'monkey_rmbg.png'

## loading and removing background
inp = Image.open(input_img)
output = remove(inp)

## Saving background removed image to same location as input image
output.save(output_img)
```

Before vs After — BgBuster In Action — Output Screenshot Taken By Author

7. MemoryMate

I often need to remember important tasks while immersed in my work. To address this and boost my productivity, I developed MemoryMate using Python, which serves as my digital memory aid. It sends me reminders with custom messages after a specified period to ensure I complete my tasks on time. MemoryMate has significantly improved my productivity and helped me consistently meet deadlines. The best part is that it's simple, easy to replicate, and incredibly useful.

```
from win10toast import ToastNotifier
import time

toaster = ToastNotifier()

def set_reminder():
    reminder_header = input("What would you like me to remember?\n")
    related_message = input("Related Message:\n")
    time_minutes = float(input("In how many minutes?\n"))

    time_seconds = time_minutes * 60

    print("Setting up reminder...")
    time.sleep(2)
    print("All set!")

    time.sleep(time_seconds)

    toaster.show_toast(
        title=f"{reminder_header}",
        msg=f"{related_message}",
        duration=10,
        threaded=True
    )

    while toaster.notification_active():
        time.sleep(0.005)

if __name__ == "__main__":
    set_reminder()
```



Script Output

> Do You Know that In Python, you can perform calculations with extremely large integers like `999**999` without overflow errors because Python automatically uses a “big integer” type for large numbers.

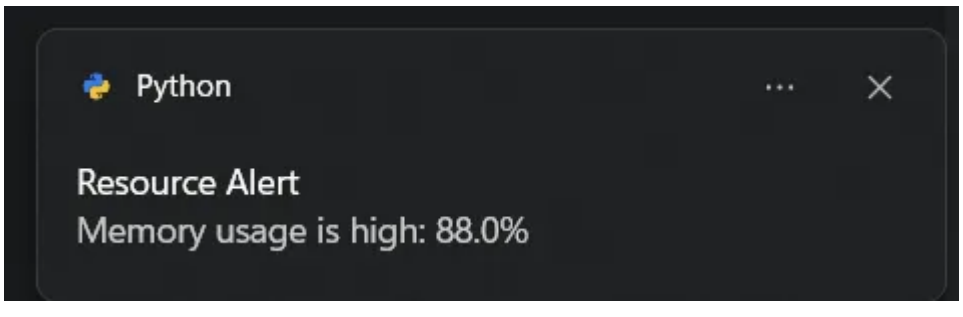
8. MonitorMax 😎

System resource monitoring is crucial for displaying real-time utilization of various system resources. It’s an invaluable tool for users, system administrators, and developers who need to keep track of system performance, identify bottlenecks, and ensure efficient resource management. This Python automation script helps monitor CPU, GPU, battery, and memory usage, generating alerts if any resource usage exceeds safe thresholds.

Don’t worry Our Script Keeping An Eye On Everything

```
import psutil
import time
from win10toast import ToastNotifier

# Initialize the ToastNotifier object
toaster = ToastNotifier()

# Set the threshold values for CPU usage, memory usage, GPU usage, and battery l
cpu_threshold = 40  # Percentage
memory_threshold = 40  # Percentage
gpu_threshold = 40  # Percentage
battery_threshold = 100  # Percentage

# Infinite loop to continuously monitor system resources
while True:
    try:
```

```
# Get system resource information
cpu_usage = psutil.cpu_percent(interval=1)
memory_usage = psutil.virtual_memory().percent
gpu_usage = psutil.virtual_memory().percent
battery = psutil.sensors_battery()

        toaster.show_toast("Resource Alert", message, duration=10)

    # Check memory usage
    if memory_usage >= memory_threshold:
        message = f"Memory usage is high: {memory_usage}%"
        toaster.show_toast("Resource Alert", message, duration=10)

    # Check GPU usage
    if gpu_usage >= gpu_threshold:
        message = f"GPU usage is high: {gpu_usage}%"
        toaster.show_toast("Resource Alert", message, duration=10)

    # Check battery level
    if battery is not None and battery.percent <= battery_threshold and not
        message = f"Battery level is low: {battery.percent}%"
        toaster.show_toast("Battery Alert", message, duration=10)

    # Wait for 5 minutes before checking the resources again
        time.sleep(300)

except Exception as e:
    print("An error occurred:", str(e))
    break
```



Output — Resource Alerts — Screenshot Taken By Author

Applications:

This script can be used in day-to-day scenarios like playing games, running a local server, training a DL model locally, etc. By tracking all resources you can make sure your scripts or task use optimal memory and if not you can optimize it accordingly. Resource monitor dashboard (You can create a dashboard using Tkinter to get a live resource graph similar to a taskbar with the addition of notifications, and high memory usages sound alerts).

9. EmailBlitz

Handling bulk email communication can be challenging, whether it's for marketing campaigns, newsletters, or organizational updates. This Python automation script makes the task easier by enabling you to send emails in bulk effortlessly. It streamlines the communication process, allowing you to reach a large group of recipients simultaneously and ensuring timely and efficient message delivery. Ideal for marketers, administrators, or anyone who needs to send numerous emails, this script enhances productivity, saves time, and helps maintain a personal touch in your communications.

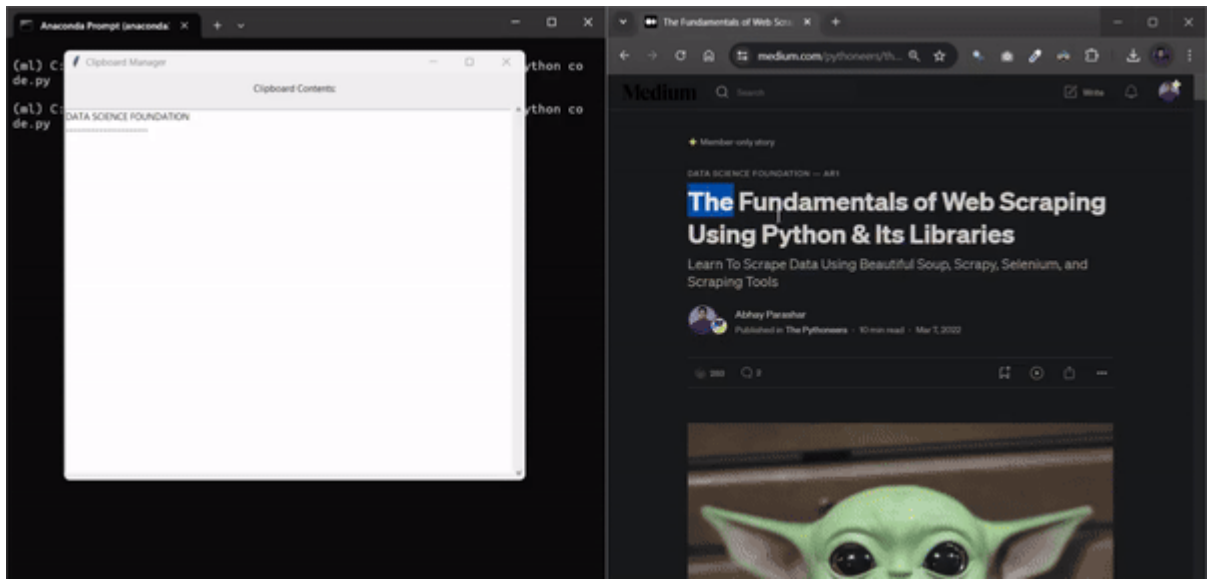
```
import smtplib
from email.message import EmailMessage
import pandas as pd

def send_email(remail, rsubject, rcontent):
    email = EmailMessage()                          ## Creating a object for Ema
    email['from'] = 'The Pythoneer Here'            ## Person who is sending
    email['to'] = remail                            ## Whom we are sending
    email['subject'] = rsubject                     ## Subject of email
    email.set_content(rcontent)                     ## content of email
    with smtplib.SMTP(host='smtp.gmail.com',port=587)as smtp:
        smtp.ehlo()                                 ## server object
        smtp.starttls()                             ## used to send data between
        smtp.login(SENDER_EMAIL,SENDER_PSWRD)       ## login id and password of
        smtp.send_message(email)                    ## Sending email
        print("email send to ",remail)              ## Printing success message



if __name__ == '__main__':
    df = pd.read_excel('list.xlsx')
    length = len(df)+1

    for index, item in df.iterrows():
        email = item[0]
        subject = item[1]
        content = item[2]

        send_email(email,subject,content)
```

10. ClipSaver

Have you ever found yourself juggling multiple text snippets, only to lose track of what you've copied? Imagine having a tool that can keep track of everything you copy throughout the day. This Python automation script does just that. It monitors everything you copy and seamlessly stores each text snippet in a sleek graphical interface. No more searching through endless tabs and risking the loss of valuable information — this script keeps everything organized and easily accessible.

```
import tkinter as tk
from tkinter import ttk
import pyperclip

def update_listbox():
    new_item = pyperclip.paste()
    if new_item not in X:
        X.append(new_item)
        listbox.insert(tk.END, new_item)
        listbox.insert(tk.END, "----------------------")
    listbox.yview(tk.END)
    root.after(1000, update_listbox)

def copy_to_clipboard(event):
    selected_item = listbox.get(listbox.curselection())
    if selected_item:
        pyperclip.copy(selected_item)

X = []

root = tk.Tk()
root.title("Clipboard Manager")
```

```
root.geometry("500x500")
root.configure(bg="#f0f0f0")

frame = tk.Frame(root, bg="#f0f0f0")
frame.pack(padx=10, pady=10)

label = tk.Label(frame, text="Clipboard Contents:", bg="#f0f0f0")
label.grid(row=0, column=0)

scrollbar = tk.Scrollbar(root)
scrollbar.pack(side=tk.RIGHT, fill=tk.Y)

listbox = tk.Listbox(root, width=150, height=150, yscrollcommand=scrollbar.set)
listbox.pack(pady=10)
scrollbar.config(command=listbox.yview)

update_listbox()

listbox.bind("<Double-Button-1>", copy_to_clipboard)

root.mainloop()
```



Script Output — ClipSaver —GIF Created By Author

Did you knew that the .py extension doesn't matter much when saving code files ??

.....

You can use any type of extension to save your Python file, whether its `.cow` , `.cat` or `.mango` if your script is valid it will run and give you the desired output that you would expect.

11. BriefBot

I love reading articles, research papers, and news publications daily, and I know many people share this habit. However, finding the time to read full articles can be challenging. This Python automation script addresses that by

using Neural Networks to generate quick summaries. It utilizes web scraping to extract the article's content and feeds it into a pre-trained model that produces an abstract summary, saving time and making it easier to stay informed.

```
from transformers import BartForConditionalGeneration, BartTokenizer
import requests
from bs4 import BeautifulSoup

## Function to summarize article
def summarize_article(article_text, max_length=150):
    model_name = "facebook/bart-large-cnn"
    tokenizer = BartTokenizer.from_pretrained(model_name)
    model = BartForConditionalGeneration.from_pretrained(model_name)

    inputs = tokenizer.encode("summarize: " + article_text, return_tensors="pt",
    summary_ids = model.generate(inputs, max_length=max_length, min_length=50, l

    summary = tokenizer.decode(summary_ids[0], skip_special_tokens=True)
    return summary

## Function to scrape content of the aricle
def scrape_webpage(url):
    try:
        headers = { "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) App
        response = requests.get(url, headers=headers)
        response.raise_for_status()
        soup = BeautifulSoup(response.text, 'html.parser')
        all_text = soup.get_text(separator='\n', strip=True)
        return all_text
    except requests.exceptions.RequestException as e:
        print(f"Error: {e}")
        return None

if __name__ == "__main__":
    webpage_url = "https://thehackernews.com/2024/07/new-ransomware-group-exploi
    webpage_text = scrape_webpage(webpage_url)

    if webpage_text:
        summary = summarize_article(webpage_text)
        print("\nSummarized Article:")
        print(summary)
    else:
        print("Webpage scraping failed.")
```

Summarized Article:

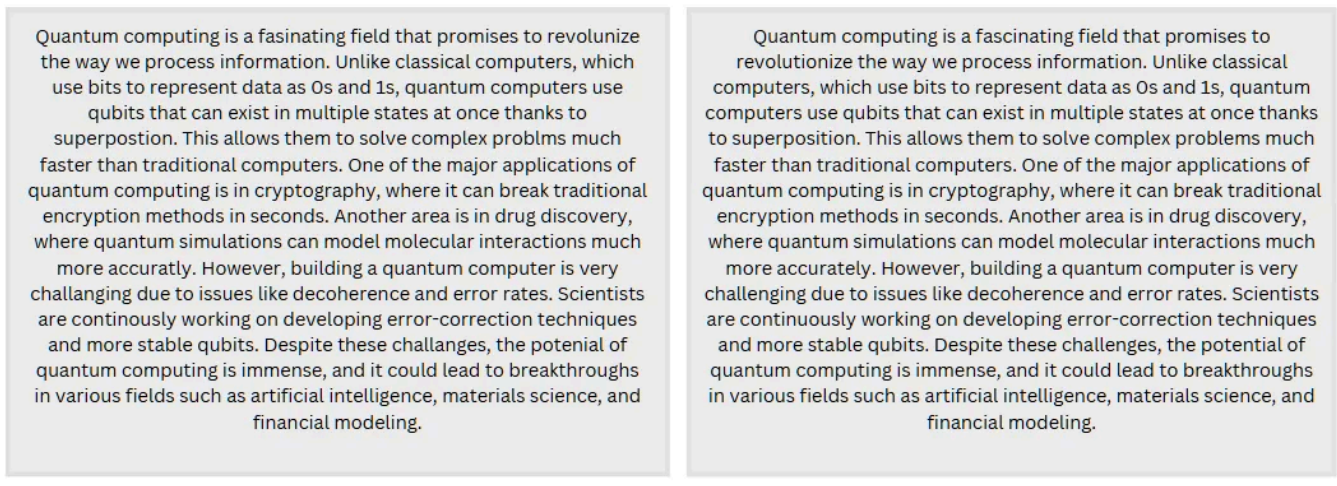
A now-patched security flaw in Veeam Backup & Replication software is being exploited by a nascent ransomware operation known as EstateRansomware. Singapore-headquartered Group-IB, which discovered the threat actor in early April 2024, said the modus operandi involved the exploitation of CVE-2023-27532(CVSS score: 7.5)

Summarized Article — BriefBot Output — Screenshot By Author

12. SpellGuard

No matter how proficient we are in English, we all make spelling and grammar mistakes when writing lengthy reports, articles, or research papers. In the age of AI, numerous powerful Python packages can help rectify these errors and provide a polished proofread to your work. This Python script leverages AI to detect and correct spelling and grammar mistakes, ensuring your writing is clear, accurate, and professional.

```
## Installing Library
!pip install lmproof

## Importing Library
import lmproof as lm

## Function for proofreading
def Proofread(text):
    proof = lm.load("en")
    error_free_text = proof.proofread(text)
    return error_free_text

## Sample Text
TEXT = '' ## Place sample Text here

## Function Call
Print(Proofread(TEXT))
```



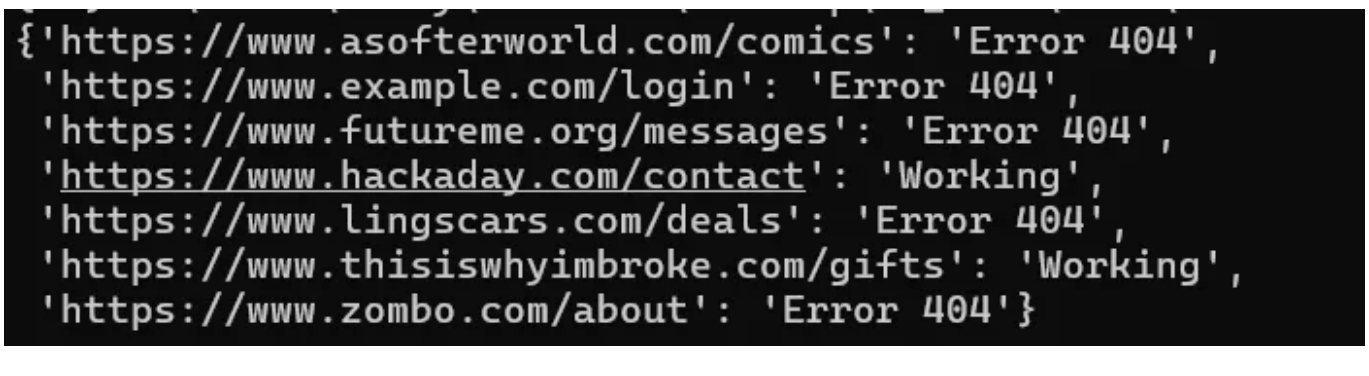
Script Output

13. LinkStatus

Owning a blog site is still a dream for many writers. Ensuring that all your links are functioning properly is crucial for maintaining a professional and user-friendly blog. Broken links can frustrate readers and hurt your site's credibility. This Python automation script allows you to check the web connectivity of multiple URLs effortlessly. By regularly monitoring your URLs, this script ensures that your links are always live and functional, enhancing your site's reliability and user experience.

```
import csv
import requests
import pprint

def get_status(website):
    try:
        status = requests.get(website).status_code
        return "Working" if status == 200 else "Error 404"
    except:
        return "Connection Failed!!"

def main():
    with open("sites.txt", "r") as fr:
        websites = [line.strip() for line in fr]

    web_status_dict = {website: get_status(website) for website in websites}
```

```
    pprint.pprint(web_status_dict)

    # Write results to CSV file
    with open("web_status.csv", "w", newline='') as csvfile:
        fieldnames = ["Website", "Status"]
        writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

        writer.writeheader()
        for website, status in web_status_dict.items():
            writer.writerow({"Website": website, "Status": status})

        print("Data Uploaded to CSV File!!")

if __name__ == "__main__":
    main()
```



Script Output

You can keep Chinese Strings As Var Names In Python.

Eg: 金竹戈女日 = “Hello World”

14. DailyDigest

Staying informed about the latest happenings in your city, state, country, or the world is important, but our busy schedules often prevent us from dedicating time to reading the news. This automation script solves that problem by scraping trending news from Google News and reading it out loud to you. Whether you’re starting your day or on the go, this script ensures you stay updated with the top news stories effortlessly.

```
import pyttsx3
import requests

engine = pyttsx3.init('sapi5')
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[0].id)

def speak(audio):
    engine.say(audio)
    engine.runAndWait()

def trndnews():
    url = " http://newsapi.org/v2/top-headlines?country=us&apiKey=GET_YOUR_OWN"
```

```
    page = requests.get(url).json()
    article = page["articles"]
    results = []
    for ar in article:
        results.append(ar["title"])
    for i in range(len(results)):
        print(i + 1, results[i])
    speak(results)

trndnews()

''' Run The Script To Get The Top Headlines From USA'''
```

15. QRGenie

The popularity of QR Codes has skyrocketed once people started using them for sending and receiving payments. Nowadays they are used to share social links, secret messages, coupon codes, and more. This Python automation script helps you create customized QR codes with your chosen data, allowing you to share information effortlessly and impress your audience.

```
import qrcode

def generate_qr_code(link, filename):
    """Generates a QR code for the given link and saves it as filename."""

    qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_L,
        box_size=10,
        border=4,
    )
    qr.add_data(link)
    qr.make(fit=True)

    img = qr.make_image(fill_color="black", back_color="white")
    img.save("profile.png")

if __name__ == "__main__":
    generate_qr_code("https://abhayparashar31.medium.com/", "Profile.png")
    print(f"QR code saved successfully!!")
```

Generated QR Code — Try it out 😉

16. ShrinkLink

I deal with numerous daily links, some I store, and some I share with my readers. The part I dislike most about URLs is their length, which can be annoying and hard to read. This automation script efficiently addresses that problem by using an external API to convert long URLs into short, manageable links.

```
import pyshorteners
def generate_short_url(long_url):
  s = pyshorteners.Shortener()
  return s.tinyurl.short(long_url)

long_url = input('Paste Long URl \n')
short_url = generate_short_url(long_url)
print(short_url)
```

```
Paste Long URl
https://medium.com/@abhayparashar31
https://tinyurl.com/2zsb85wx
```

17. CaptureIt

Whether you're a gamer, influencer, artist, or developer, screen recording software is essential for capturing your activities. However, many existing solutions are costly or impose limitations such as watermarks and time constraints. This Python automation script offers a straightforward solution for screen recording without watermarks, and time limits, and with customizable screen window options.

```
import cv2
import numpy as np
import pyautogui

SCREEN_SIZE = tuple(pyautogui.size())
fourcc = cv2.VideoWriter_fourcc('M','J','P','G')
fps = 12.0
record_seconds = 20
out = cv2.VideoWriter("video.mp4", fourcc, fps, SCREEN_SIZE)

for _ in range(int(record_seconds * fps)):
    img = pyautogui.screenshot(region=(0, 0, 500, 900))
    frame = np.array(img)
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    out.write(frame)
    cv2.imshow("video frame", frame)
    if cv2.waitKey(1) == ord("q"):
        break

cv2.destroyAllWindows()
out.release()
```

Bonus

H2OReminder

This automation script is a simple yet powerful program designed to remind users to stay hydrated by drinking water at regular intervals. It's an invaluable tool for individuals who spend extended hours in front of computers or have hectic schedules that can lead to forgetting to drink enough water. This script promotes healthy habits by encouraging regular water intake, essential for maintaining overall health and well-being.

```
import time
from plyer import notification

if __name__ == "__main__":
    while True:
        notification.notify(
            title="Please Drink Water",
            message="The U.S. National Academies of Sciences, Engineering, and M
            app_icon="./Desktop-drinkWater-Notification/icon.ico",
            timeout=12
        )
        time.sleep(1800)  ## Change it according to your water breaks!!!
```

Thanks For Reading Till Here, If You Like My Content and Want To Support Me The Best Way is —

1. Leave a **Clap** 👋 and your thoughts 💬 below.
2. Follow Me On ***Medium***.
3. Connect With Me On *LinkedIn*.
4. Attach yourself to *My Email List* to never miss reading another article of mine
5. Do Follow **The Pythoneers** Publication For more similar stories.