# Energy Air Selenium-Bot

Der folgende selenium-Bot wird verwenden, um das jährliche Energy Air Gewinnspiel voll-automatisiert zu gewinnen. Mithilfe einer App auf dem Android Phone wird das Authentication-Token (SMS) via Email weitergeleitet und vom Python Script eingelesen. Somit werden neue Sessions gleich nach Ablauf wieder erstellt.

Das Script ist in Python3 geschrieben und wurde mit Python 3.7.4 getestet.

**ACHTUNG**: BITTE BOT NICHT INS INTERNET HOCHLADEN ODER AN DRITTE WEITERGEBEN! - merci

## Das Betreiben des Botes

Als Beispiel Installation, wird das Verzeichnis `D:\EnergyAir-Bot-2019\` auf einem Windows 8.1 System verwendet. Zum erstmaligen einrichten, wird das Kapitel “Vorbereitung und Installation” gebraucht. Für jeden weiteren Start des Bots, kann anschliessend wie im Kapitel “Bot ausführen” beschrieben vorgegangen werden.

### Vorbereitung und Installation

Für das erstmalige Setup und das Ausführen des Energy Air Botes, sind initial vier Vorberitungschritte nötig.

#### Vorbereitung 1 - Gmail Account API aktivieren und Label vorbereiten

1. Klicke auf Enable The GMAIL API und “Download Client Config”
2. In der Gmail Inbox mail.google.com] ein neues `Label` namens “`Energy`” kreieren.
3. Unter Einstellungen (oben rechts), zu “Filter & blockierte Adressen gehen und “Neuen Filter erstellen” klicken.
4. Bei `von:` die Email Adresse `no-reply-smsforwarder@cofp.ru` eintragen
5. Filter erstellen und auf das neu erstellte Label “Energy” anwenden.

---

#### Vorbereitung 2 - Smartphone SMS to Email App einrichten

1. Android App Installieren SMS an Mail / Telefon - automatische Umleitung
2. Die App so konfigurieren, dass die Energy Nachrichten (SMS) an das vorher konfigurierte Gmail-Konto weitergeleitet werden.
3. **Achtung**: Die App muss im Hintergrund offen bleiben, demnach die App nicht aus den “offenen Apps” löschen. (**Evtl. Energieplan auf dem Handy anpassen.**)

***iPhone App (optional)***

- Falls kein Android Smartphone vorhanden ist, kann ein entsprechendes App aus dem Appstore

genutzt werden, welches SMS auf Mail umleitet. Die Regex um den Code im String zu finden muss dann jedoch entsprechend im gmail_nrg_code.py Script angepasst werden.\

- Mögliche App: SMS Weiterleiten

---

**Vorbereitung 3 - Einrichten des Bot Workspaces:**

1. Installation vom **Python 3.7.4** - `Während der Installation gilt es zu beachten, dass der Haken "`**`Add Python 3.7 to PATH`**`" gesetzt ist!`
2. Energy Air Bot herunterladen und in das Verzeichnis `D:\EnergyAir-Bot-2019\` entpacken.
3. Neues unprivilegiertes **CMD-Fenster** öffnen und in den Projekt Ordner navigieren: (Win + R || "CMD" eintippen || ENTER)

```
# d:
# cd EnergyAir-Bot-2019
```

4. Hier wird nun ein virtualenv für Python installiert:

```
# pip3 install virtualenv
# virtualenv venv
```

5. Nach dem erfolgreichen vorbereiten des virtualenvs, muss dieses für das aktuelle CMD-Fenster aktiviert werden, um dann auch gleich die noch benötigten Python requirements darin zu installieren.

```
# venv\Scripts\activate.bat
# pip install -r requirements.txt
```

6. `CMD-Fenster offen lassen!`

---

**Vorbereitung 4 - Erstmaliges Ausführen des Botes (Zur Berechtigung Gmail + setzen der Label ID)**

1. `credentials.json` (GMAIL API Auth) in den `Projektordner (D:\EnergyAir-Bot-2019\)` platzieren.
2. Im `gmail_nrg_code.py` gibt es zwei Stellen welche mit Uncomment markiert sind. Diese Blöcke müssen für die erste Ausführung komplett einkommentiert werden.
3. Das Skript mittels

```
# energy-sbot2019.py
```

in der CMD-Konsole ausführen.

4. Die `id` des Labels kopieren und an folgender Codestelle einfügen:

```
results = service.users().messages().list(userId='me',
```

```
labelIds=['UNREAD', 'Label_YOURLABELID'],
```

5. Beide Codeblöcke wieder auskommentieren.

---

#### Starten des Energy Air Botes

Zum starten des Botes im Bot Verzeichnis "`D:\EnergyAir-Bot-2019\`" und im Aktiven `venv` folgenden Befehl ausführen.

*Anschliessend warten auf den Gewinn.. ;)*

```
# energy-sbot2019.py
```

---

### Bot erneut ausführen (nach PC reboot oder für neue Nummer)

`Neues unprivilegiertes` **`CMD-Fenster`** `öffnen und nachfolgende Kommandos eintippen.`

```
d:
cd EnergyAir-Bot-2019
venv\Scripts\activate.bat
energy-sbot2019.py
```

**Beispiel:**

```
C:\Users\michael>d:
D:\>cd EnergyAir-Bot-2019
D:\EnergyAir-Bot-2019>venv\Scripts\activate.bat
(venv) D:\EnergyAir-Bot-2019>energy-sbot2019.py
```

## Sourcecode des selenium-Bots v2019

**`Komplett Download`**:

- energyair-bot-2019.zip

### Main Bot-code

`Filename:` **`energy-sbot2019.py`**

```
# -*- coding: utf-8 -*-
from selenium import webdriver
```

```
import time
import gmail_nrg_code as nrg_code

print("© 2019 Michael Reber .  ALL RIGHTS RESERVED.")
print("Created to win the endless energy game..")
print("\r")

print("May the Force be with you.")
print("Patience you must have, my young padawan.")

driver = webdriver.Chrome(executable_path=r"D:\EnergyAir-
Bot-2019\chromedriver.exe")
driver.get("https://game.energy.ch/")
# assert "Energy" in driver.title
answers = [
    "One Republic",
    "1300",
    "gewinnen",
    "XTRA-Circle",
    "Twitter",
    "E-Mail",
    "2014",
    "450 Tonnen",
    "70 Meter",
    "Die sechste",
    "Lo & Leduc",
    "im Radio, auf der Website und über Social Media",
    "40’000",
    "Energy Air findet trotzdem statt",
    "Im Privatjet",
    "Stade de Suisse, Bern",
    "Bastian Baker",
    "60",
    "Um 17 Uhr",
    "250",
    "Alvaro Soler",
    "14",
    "...der unter freiem Himmel stattfindet.",
    "Averdeck",
    "Sein Mami",
    "Eine komplett weisse Garderobe",
    "BSC Young Boys",
    "7. September 2019"
]

def press_answer(quest_nr):
    for answer in answers:
        try:
            labelname = "//label[@for='" + answer + "']"
```

```
            elem1 = driver.find_elements_by_xpath(labelname)[0]
            time.sleep(2)
            elem1.click()
            elem2 = driver.find_elements_by_xpath("//button[@id='next-
question']")[0]
            time.sleep(1)
            elem2.click()
            print(answer)
        except:
            pass
    return quest_nr

question_Nr = 0
counter = 0
input("Press to enter script")
print("Enter Phone Nr: (do not enter starting '0'! Example: 798765432")
tel_nr = int(input("+41"))

while True:
    while question_Nr < 10:
        press_answer(question_Nr)
        question_Nr += 1
    else:
        try:
            elem2 = driver.find_elements_by_xpath("//div[@class='circle col-
xs-4 col-sm-3 col-md-4 col-lg-3']")[6]
            elem2.click()
            try:
                elem1 = driver.find_elements_by_xpath("//button[@class='btn
btn-primary game-button btn-lg']")[0]
                elem1.click()
            except:
                pass
        except:
            try:
                try:
                    elem3 =
driver.find_elements_by_xpath("//input[@placeholder='Handynummer']")[0]
                    elem3.send_keys(tel_nr)
                    elem1 =
driver.find_elements_by_xpath("//button[@class='btn btn-primary game-button
btn-lg']")[0]
                    elem1.click()
                    found_mail = False
                    while found_mail is False:
                        sms_code = nrg_code.main()
                        if sms_code is not None:
                            found_mail = True
                            print(sms_code +
".................................................")
                            code_numb_list = []
```

```
                            for numb in sms_code:
                                code_numb_list.append(numb)
driver.find_elements_by_xpath("//input[@id='1']")[0].send_keys(code_numb_lis
t[0])
driver.find_elements_by_xpath("//input[@id='2']")[0].send_keys(code_numb_lis
t[1])
driver.find_elements_by_xpath("//input[@id='3']")[0].send_keys(code_numb_lis
t[2])
driver.find_elements_by_xpath("//input[@id='4']")[0].send_keys(code_numb_lis
t[3])
                            elem1 = \
                                driver.find_elements_by_xpath(
                                    "//button[@class='btn btn-primary game-
button btn-lg btn-declined']")[0]
                            elem1.click()
                        else:
                            time.sleep(5)
                except:
                    pass
                elem1 = driver.find_elements_by_xpath("//button[@class='btn
btn-primary game-button btn-lg']")[0]
                elem1.click()
            except:
                pass
            question_Nr = 0
    counter += 1
    print("COUNT:", counter)
    if counter == 115:
        driver.close()
        driver = webdriver.Chrome(executable_path=r"D:\EnergyAir-
Bot-2019\chromedriver.exe")
        driver.get("https://game.energy.ch/")
        assert "Energy" in driver.title
        counter = 0
```

## Email Sender-code

Filename: **gmail_nrg_code.py**

```
# -*- coding: utf-8 -*-
import pickle
import os.path
from googleapiclient.discovery import build
from google_auth_oauthlib.flow import InstalledAppFlow
from google.auth.transport.requests import Request
```

```
import re

# If modifying these scopes, delete the file token.pickle.
SCOPES = ['https://www.googleapis.com/auth/gmail.readonly',
'https://www.googleapis.com/auth/gmail.modify']


def main():
    """Shows basic usage of the Gmail API.
    Lists the user's Gmail labels.
    """
    creds = None
    # The file token.pickle stores the user's access and refresh tokens, and
is
    # created automatically when the authorization flow completes for the
first
    # time.
    if os.path.exists('token.pickle'):
        with open('token.pickle', 'rb') as token:
            creds = pickle.load(token)
    # If there are no (valid) credentials available, let the user log in.
    if not creds or not creds.valid:
        if creds and creds.expired and creds.refresh_token:
            creds.refresh(Request())
        else:
            flow = InstalledAppFlow.from_client_secrets_file(
                'credentials.json', SCOPES)
            creds = flow.run_local_server(port=0)
        # Save the credentials for the next run
        with open('token.pickle', 'wb') as token:
            pickle.dump(creds, token)

    service = build('gmail', 'v1', credentials=creds)

##########################################################################
###
    """Uncomment to print label_name + label_id
    is needed to determine Energy Label id.
    """
##########################################################################
###
#    results = service.users().labels().list(userId='me').execute()
#
#    labels = results.get('labels', [])
#
#    if not labels:
#        print('No labels found.')
#    else:
#        print('Labels:')
#        for label in labels:
#            print(label['name'] + " " + label['id'])
```

```
##########################################################################
###

    # Call the Gmail API to fetch INBOX Energy, only unread messages
    results = service.users().messages().list(userId='me',
labelIds=['UNREAD', 'Label_5204902350138961179'],
                                              maxResults=1).execute()
    messages = results.get('messages', [])

    if not messages:
        print("No messages found.")
        return None
    else:
        print("Message snippets:")
        for message in messages:
            msg = service.users().messages().get(userId='me', format='full',
id=message['id']).execute()
            # marks mail as read
            service.users().messages().modify(userId='me', id=message['id'],
                                              body={'removeLabelIds':
['UNREAD']}).execute()
            snippet = msg['snippet']
            print(snippet)
            verification_code = str(snippet).split("Dein Code für
game.energy.ch")[-1]
            verification_code =
verification_code.split(re.match('[0-3][0-9]/[0-1][0-9]/2019',
verification_code))[
                0].strip()
            print(verification_code)
            return verification_code
##########################################################################
###
    """uncomment to run once to fetch label id or for development"""
#if __name__ == '__main__':
#    main()
##########################################################################
###
```

## Python requirements

Filename: **requirements.txt**

```
cachetools==3.1.1
certifi==2019.6.16
chardet==3.0.4
```

```
google-api-python-client==1.7.11
google-auth==1.6.3
google-auth-httplib2==0.0.3
google-auth-oauthlib==0.4.0
httplib2==0.13.1
idna==2.8
oauthlib==3.1.0
pyasn1==0.4.6
pyasn1-modules==0.2.6
requests==2.22.0
requests-oauthlib==1.2.0
rsa==4.0
selenium==3.141.0
six==1.12.0
uritemplate==3.0.0
urllib3==1.25.3
```

Last update: **2019/08/27 14:40**