Automatically cratch information from website

objects: Automatically cratch information about all the songs of a certain singer on a music website, including: singer name, song title, and lyrics.

1. Automatically open the login page of the website;
2. Automatically input mobile phone number and password;
3. Intercept and recognize the picture verification code, and then automatically fill in the recognized number into the verification code input box;
4. Automatically click the [Login] button;
5. Extract and store song information

Edit by David Mar 23, 2023

In [1]:
```
!pip install pytesseract
```

```
Requirement already satisfied: pytesseract in d:\anaconda3\lib\site-packages (0.3.10)
Requirement already satisfied: packaging>=21.3 in d:\anaconda3\lib\site-packages (from pytesseract) (23.0)
Requirement already satisfied: Pillow>=8.0.0 in d:\anaconda3\lib\site-packages (from pytesseract) (8.4.0)
```

import re, requests, and selenium

In [2]:
```
import re
import requests
import pytesseract
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.by import By
```

Automatically log in to the web page

In [3]:
```
# Get phone number and password
phone = input('Please insert the phone number: ')
password = input('Please insert the password: ')
```

```
Please insert the phone number: 13761736321
Please insert the password: 123456
```

In [4]:
```
# Get the singer's name
singer_name = input('Please insert the singer\' name: ')
```

```
Please insert the singer' name: 周杰伦
```

In [5]:
```
# Set browser silent mode
opts = Options()
opts.headless = True
```

In [6]:
```
# Initialize the Google Chrome driver and open the web page  (Chrome version 111.0.5563.111)
driver = webdriver.Chrome(options=opts)
driver.get('https://music.facode.cn/index.php/Home/Index/login.html')
```

In [7]:
```
# Navigate to the label of the mobile phone number input box and enter the mobile phone number (use driver.find_element_by_name)
user_tag = driver.find_element(by=By.NAME, value='phone')
user_tag.send_keys(phone)

# Navigate the label of the password input box and enter the password
password_tag = driver.find_element(by=By.NAME, value='pass')
password_tag.send_keys(password)
```

In [ ]:
```
# Navigate to the label where the image verification code is located
img_tag = driver.find_element(by=By.ID, value='graph_img')

# Define the saved screenshot name and Save the captured image as Verification code image.png
png_path = '../Verification_code_picture.png'

with open(png_path, 'wb') as f:
    f.write(img_tag.screenshot_as_png)

# Identify the content in the picture and remove redundant symbols
code = pytesseract.image_to_string(png_path)
code = code.strip()

# Locate the label of the verification code input box and enter the verification code
code_tag = driver.find_element(by=By.NAME, value='verify')
code_tag.send_keys(code)

# Locate the label of the [login] button and click the Login button
login_tag = driver.find_element(by=By.CLASS_NAME, value='login-btn')
login_tag.click()
```

In [ ]:
```
# Get cookie list and create a null string to store the cookie
cookie_list = driver.get_cookies()
cookies = ''

# Loop through the cookie list and take out the target cookie information
for cookie in cookie_list:
    cookies += '{}={};'.format(cookie['name'], cookie['value'])

# set request header
header = {'Cookie': cookies}

# close the browser
driver.quit()
```

Request singer song data and save to text files

In [ ]:
```
# Set request link, request header, request data
search_url = 'https://music.facode.cn//index.php/Home/Index/search_list.html'

data = {
    'value': singer_name,
    'info': '1',
    'page': 1,
}
```

In [ ]:
```
# get response data
search_res = requests.post(search_url, data=data, headers=header)
search_json = search_res.json()
```

In [ ]:
```
# Get the total number of search results and calculate the total number of pages
result_num = int(search_json['totalnum'])
page_num = result_num // 12
if result_num % 12 != 0:
    page_num += 1

# The number of times to loop through the total number of pages, starting from page 2
for page in range(1, page_num + 1):

    # Modify the number of pages in the request data dictionary
    data['page'] = page
    print('Begin to scratch {} page....'.format(page))

    # get response data
    search_res = requests.post(search_url, data=data, headers=header)
    search_json = search_res.json()

    # Loop through songs data
    for song in search_json['voice']:
        filename = '{}-{}'.format(song['name'], song['author'].replace('/', ''))

         # Request lyrics data
        lyrics_res = requests.post('https://music.facode.cn//index.php/Home/Index/lyrics.html', data={'id': song['id']}, headers=
        lyrics_json = lyrics_res.json()

        # Skip if no lyrics exist
        if lyrics_json['data'] is None:
            print(filename + ' there is no lyric!')
            continue

        # Cleansing Lyrics Using Regular Expressions
        match_result = re.sub('\[.*?]', '', lyrics_json['data'])

        # write lyrics to text file
        with open('../lyrics/' + filename + '.txt', 'w') as f:
            f.write(match_result)

        # Print the written songs information
        print(filename + ' The extraction and writing of lyrics has been completed!')
```