Университет ИТМО

Факультет программной инженерии и компьютерной техники

Кафедра вычислительной техники

Лабораторная работа № 5 по дисциплине ”Технологии программирования”

Выполнили:

Айтуганов Д. А.

Чебыкин И. Б.

Группа: P3401

Проверяющий: Оголюк А. А.

Санкт-Петербург, 2017

Задание

Вход: файл guess.txt содержащий имена для угадывания

Написать игру “Угадай по фото”

3 уровня сложности:

1) используются имена только 1-10
2) имена 1-50
3) имена 1-100

- из используемых имен случайно выбрать одно
- запустить поиск картинок в Google по выбранному
- получить ~30-50 первых ссылок на найденные по имени изображения
- выбрать случайно картинку и показать ее пользователю для угадывания
- после выбора сказать Правильно или Нет

п.с. желательно делать серверную часть, т.е. клиент играет в обычном браузере обращаясь к веб-серверу.

п.с. для поиска картинок желательно эмулировать обычный пользовательский запрос к Google т.е. можно использовать и Google image search API https://ajax.googleapis.com/ajax/services/search/images? или др. варианты НО в таком случае нужно предусмотреть существующие ограничения по кол-ву запросов т.е. кешировать информацию на случай исчерпания кол-ва разрешенных (бесплатных) запросов или другим образом обходить ограничение. Т.е. игра не должна прерываться после N запросов (ограничение API)

п.с. желательно “сбалансировать” параметры поиска (например искать только лица, использовать только первые 1-30 найденных и т.п.) для минимизации того что найденная картинка не соответствует имени

```
import http.server
import socketserver
import random
import os
from googleapiclient.discovery import build

PORT = 8080

class GameHandler(http.server.SimpleHTTPRequestHandler):
    service = build(”customsearch”, ”v1”, developerKey=”AIzaSyA0T5P96XjGMIsnJVU3H69XXY85XMjCu1o”)

    def write_str(self, msg):
        self.wfile.write(msg.encode('utf-8'))

    def do_GET(self):
        level = 1
```

```
    names = self.read_names(level)
    selected_name = names[random.randint(0, len(names) - 1)]
    links = self.get_links(selected_name)

    self.send_response(200)
    self.send_header('Content-type', 'text/html')
    self.end_headers()

    if not links:
        self.write_str("Error: search limit was achieved, no images in cache.")
    else:
        options = '\n'.join(["<option>{}</option>".format(n) for n in names])
        with open("index.html", "r") as template_html:
            data = template_html.read().format(links[random.randint(0, len(links) - 1)], selected_name, options)
        self.write_str(data)

def get_links(self, name):
    if self.can_search():
        res = self.service.cse().list(
            q=name,
            cx="013511285222309850888:zu1n-1yhcms",
            searchType="image",
            num=10,
        ).execute()
        if "items" not in res:
            print("No result!")
        else:
            links = [item["link"] for item in res["items"]]
            with open("{}.txt".format(name), "w+") as cache_file:
                cache_file.write("\n".join(links))
            return links
    else:
        if os.path.isfile("{}.txt".format(name)):
            with open("{}.txt".format(name)) as cache_file:
                return cache_file.read().split("\n")
        else:
            print("No cache!")

def can_search(self):
    with open("limit.txt", "r+") as limit_file:
        count = int(limit_file.read())
        can = count < 100
        if can:
            count += 1
            limit_file.seek(0, 0)
            limit_file.write("%d" % count)
        return can

def read_names(self, level):
    if level not in range(1, 4):
        raise ValueError("Incorrect level value")

    names_count = [10, 50, 100][level - 1]

    with open("guess.txt", "r") as names_file:
        names = names_file.read().split("\n")[:names_count - 1]
```

```
        return names


def main():
    httpd = socketserver.TCPServer((””, PORT), GameHandler)
    httpd.serve_forever()


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

```
<DOCTYPE html>
<html>
<head>
<style>
img {{
width: auto;
max-height: 480px;
}}
</style>
<meta charset=”utf-8”>
</head>
<body>
<img src=”{0}”></img>
<select>{2}</select>
<script>function showAnswer() {{ document.getElementById(”answer”).style.visibility = ”visible” }}</script>
<button onclick=”showAnswer()”>Проверить</button>
<div id=”answer” style=”visibility: hidden”>Ответ: {1}</div>
</body>
</html>
```