|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт кибербезопасности и цифровых технологий**

**Кафедра КБ-3 «Разработка программных решений и системное программирование»**

**ОТЧЁТ ПО ПРАКТИЧЕСКОМУ ЗАНЯТИЮ**

**по дисциплине «Интернет-разведка»**

|  |  |  |  |
| --- | --- | --- | --- |
| Отчет представлен к  рассмотрению:  Студент группы БСБО-02-23 | «22» апреля 2024г. | (подпись) | Юрков М.Ю. |
|  |  |  |  |
| Преподаватель | «22» апреля 2024 г. | (подпись) | Потапов С.О. |

Москва, 2024 г.

# Парсинг

## Теоритичсекая часть

Каталог товаров, спортивная статистика, цены на офферы… Что-то знакомое, правда? Эти и другие вещи собирают с помощью специальных софтов или вручную в документы. Там информация структурирована; нет необходимости разбираться что и где

Если вас заинтересовал такой метод, подумайте о веб-скрейпинге.

### Что такое веб-скрейпинг?

Web scraping – процесс сбора данных с помощью программы, то есть в автоматическом режиме. В русскоязычном пространстве этот процесс называют парсингом. А программу – парсером. Точно так же как за бугром говорят to scrape web page, у нас – парсить страницу. Так что если изучаете материал на английском, не переводите как “скрабить”, “скрабы” и так далее :

### Как работает веб-скрейпинг?

Запускаете программу и загружаете в нее адреса страниц. А еще наполняете софт ключевыми словами и фразами, блоками и числами, которые нужно собрать. Эта программка заходит на указанные сайты и копирует в файл все, что найдет. Это может быть файл CSV-формата или Excel-таблица.

Когда программа закончит работу, вы получите файл, в котором вся информация будет структурирована.

### Для чего он нужен?

С помощью веб-скрейпинга собирают нужные данные. Например, у вас новостное агентство и вы хотите проанализировать тексты своих конкурентов на конкретную тематику. Какую лексику они используют? Как подают информацию? Конечно, найти такие статьи можно вручную, но проще настроить программу и поручить эту задачу ей.

Или так: вы любитель литературы и сейчас страшно хотите найти информацию о болгарских поэтах. На болгарском. В болгарском интернете информации о болгарской литературе в принципе мало, и поэтому штудировать каждый сайт – долго. В таком случае есть смысл обратиться к парсеру. Загоняете в программу ключевые слова и фразы, по которым она будет искать материал о поэтах, – и ждете, пока софт завершит работу.

То есть парсить информацию могут все, кто захочет. В основном этим занимаются те, кому нужно проанализировать контент конкурентов.

## Практическая часть

from selenium import webdriver

from selenium.webdriver.chrome.service import Service

from selenium.webdriver.chrome.options import Options

from selenium.webdriver.common.by import By

from openpyxl import Workbook

from openpyxl import load\_workbook

import time

def parse(driver):

    scroll(driver, 30, 1)

    product\_cards = driver.find\_elements(By.CLASS\_NAME, 'product-card')

    items = []

    for card in product\_cards:

        link = card.find\_element(By.CSS\_SELECTOR, 'a.product-card\_\_link').get\_attribute('href')

        title = card.find\_element(By.CLASS\_NAME, 'product-card\_\_name').text

        price = card.find\_element(By.CLASS\_NAME, 'price\_\_lower-price').text

        brand = card.find\_element(By.CLASS\_NAME, 'product-card\_\_brand').text

        rating = card.find\_element(By.CLASS\_NAME, 'address-rate-mini').text

        img = card.find\_element(By.CLASS\_NAME, 'j-thumbnail').get\_attribute('img')

        items.append([link, title, price, brand, rating, img])

    return items

def scroll(driver, times, delay):

    for i in range(times):

        driver.execute\_script("window.scrollBy(0, 400);")

        time.sleep(delay)

def next\_page(driver):

    next\_page\_element = driver.find\_element(By.CLASS\_NAME, 'pagination-next')

    url = next\_page\_element.get\_attribute('href')

    driver.get(url)

def write\_to\_excel(items):

    try:

        wb = load\_workbook("creatine.xlsx")

        ws = wb.active

    except FileNotFoundError:

        wb = Workbook()

        ws = wb.active

    for row in items:

        ws.append(row)

    wb.save("creatine.xlsx")

def main():

    url = 'https://www.wildberries.ru/catalog/0/search.aspx?search=%D0%BA%D1%80%D0%B5%D0%B0%D1%82%D0%B8%D0%BD#c5544385'

    driver\_path = "C:\\Program Files\\ChromeDriver\\chromedriver.exe"

    chrome\_binary\_path = "C:\\Users\\user\\Downloads\\GoogleChromePortableBeta\\App\\Chrome-bin\\chrome.exe"

    options = Options()

    options.add\_experimental\_option("detach", True)

    options.binary\_location = chrome\_binary\_path

    service = Service(driver\_path)

    driver = webdriver.Chrome(service=service, options=options)

    driver.get(url)

    pages = 2

    header\_items = [['Ссылка', 'Название', 'Цена', 'Производитель', 'Рейтинг', 'Картинка']]

    write\_to\_excel(header\_items)

    for \_ in range(pages):

        items = parse(driver)

        write\_to\_excel(items)

        next\_page(driver)

if \_\_name\_\_ == "\_\_main\_\_":

    main()