

Парсер информации на Python

Всем привет Друзья ! Сегодня мы напишем игрушечный парсер , который будет заходить на сайт <http://kenesh.kg> и извлекать ФИО депутатов и телефонные номера .GOOD LUCK!!!

Первый шаг:

Для начала нам нужно создать виртуальное окружение.

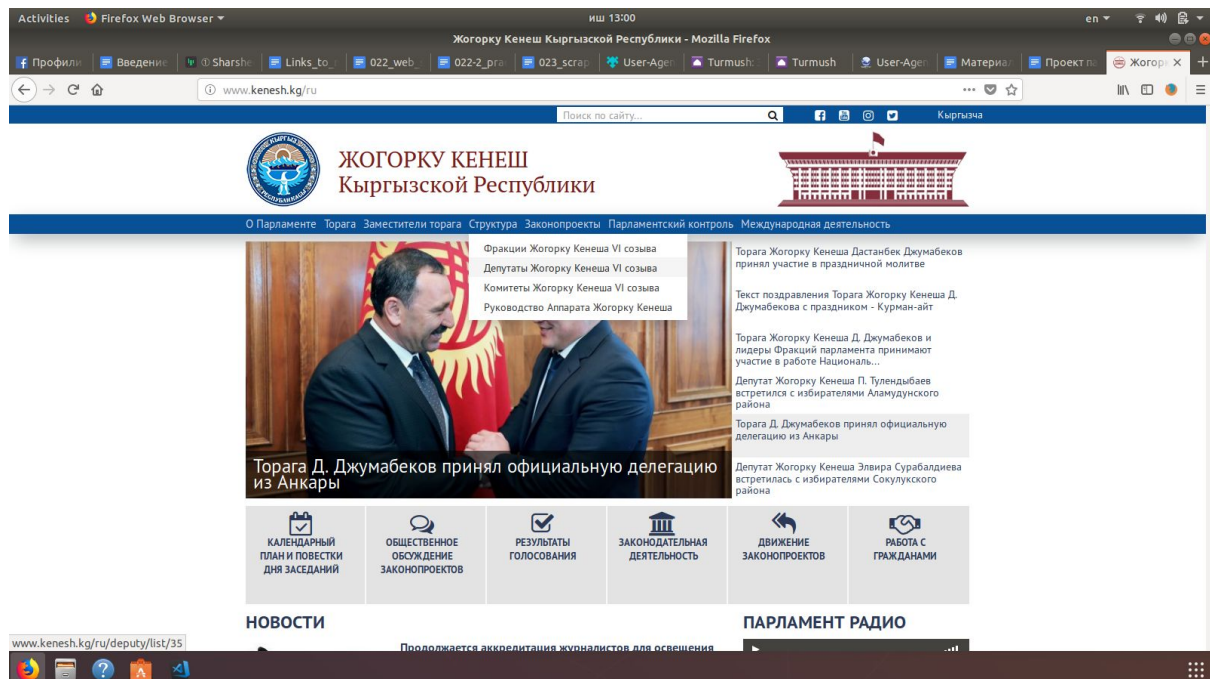
```
Python3 -m venv env
```

```
Source env/bin/activate
```

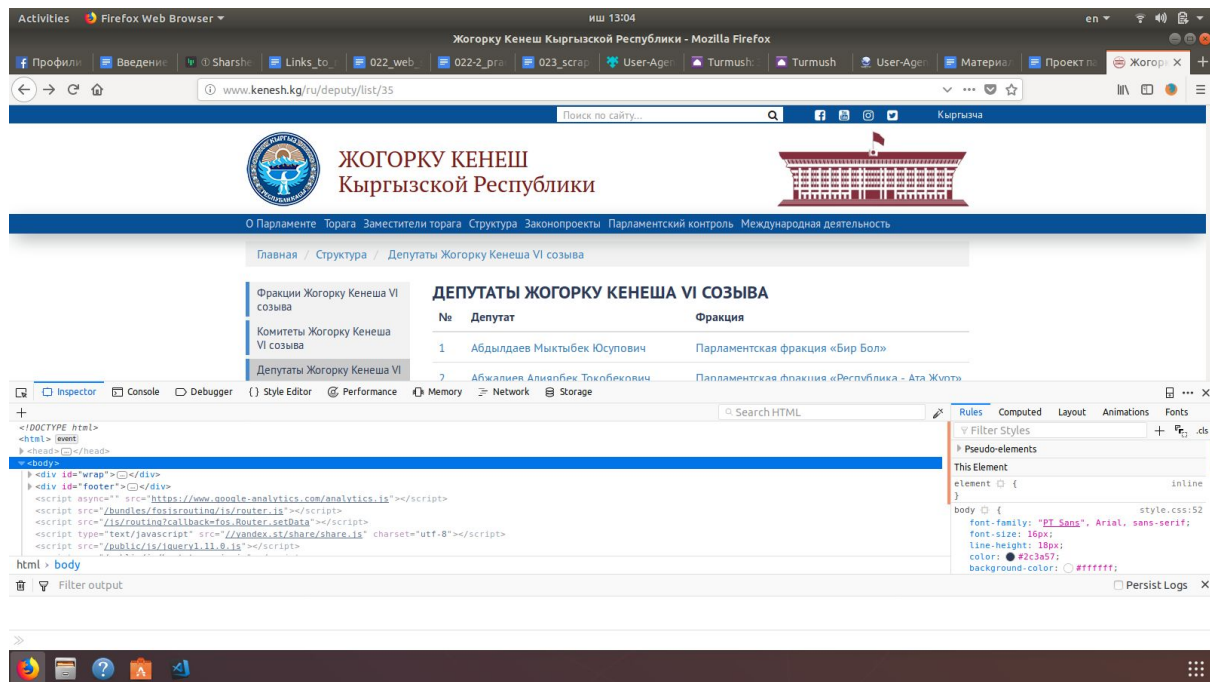
Создадим файл **main.py** (неважно как вы назовете, для удобства так назвали).

Для начала мы должны зайти на сайт и посмотреть в какой странице информация про депутатов.

Заходим в ссылку и посл



е открытия нажимаем F12



Далее смотрим в инспекторе , где именно храниться имена и фамилия

Начнём писать скриптик :

Сначала напишем точку входа

Дело в том, что в Питоне любой модуль является исполнимым - и если его запустить, он всегда получает имя `__main__`. Ну а `__name__` - это имя текущего модуля. Таким образом, проверка `if __name__ == "__main__"` проверяет что модуль был запущен из командной строки, а не был импортирован из другого модуля.

Иными словами, проверка `if __name__ == "__main__"` - это дополнительная **точка входа в программу**.

Пишем эту функцию и спускаем на самый вниз.

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

До точки входа создаем функцию `main()`

```
def main():
```

Давайте разделим задачу на куски ,

- 1) Мы создадим однопоточный парсер
- 2) Замерим время

- 3) После мы создадим многопоточный парсер в котором будем использовать библиотеку multiprocessing и будем работать с классом Pool
- 4) Дальше замерим время
- 5) И экспортируем полученные данные в формат csv .

Начнем с импорта библиотеки requests

Но чтобы импортировать , надо сначала установить с помощью команды

```
pip install requests
```

Открываем файл main.py в редакторе и

Прописываем следующий код

```
import requests
```

Далее создадим функцию get_html , которое будет принимать в качестве аргумента URL , далее внутри функции создадим переменную r от слова респонс и передадим значение requests.get(url) и дальше мы возвращаем у объекта респонс свойство текст

Наша первая функция готова !

```
import requests
```

```
def get_html(url):
```

```
    r = requests.get(url)
```

```
    return r.text
```

```
def main():
```

```
    all_links = []
```

```
if __name__ == '__main__':
```

```
    main()
```

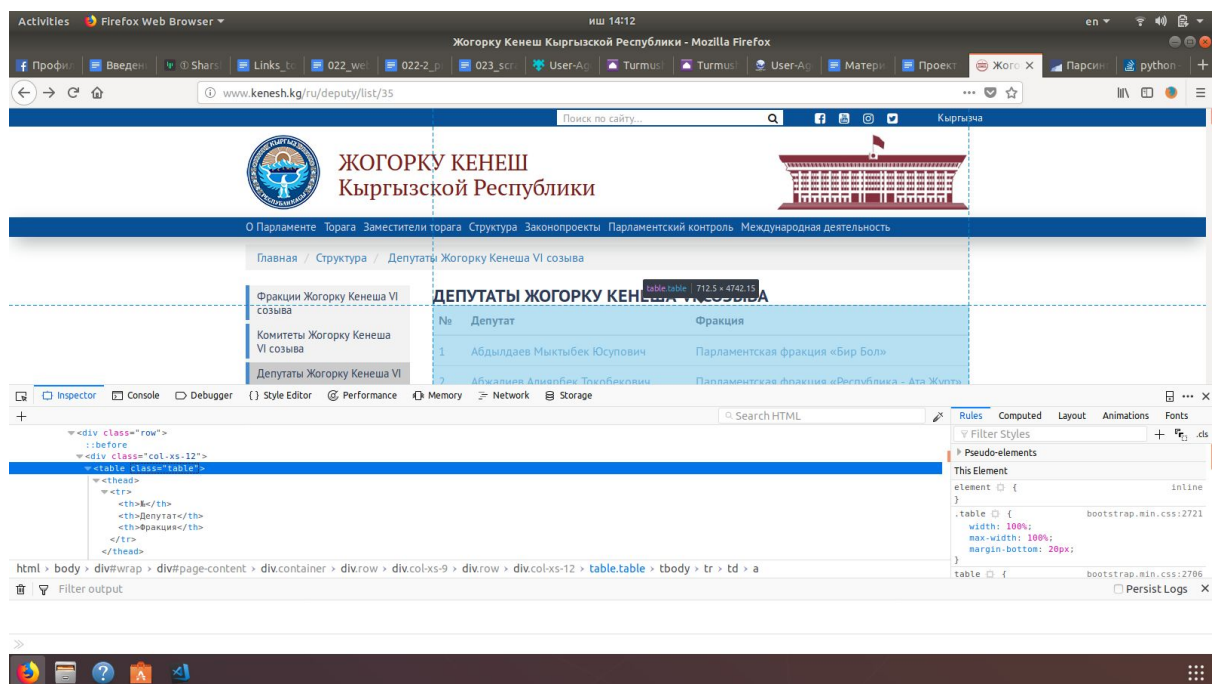
Следующая наша функция будет парсер ,мы будем парсить с помощью библиотеки beautiful soup ,для этого мы установим библиотеку с помощью команды `pip install bs4` и далее импортируем beautiful soup .

```
from bs4 import BeautifulSoup
```

Создадим функцию `get_all_links()`,которая примет html и создадим объект супа которая будет принимать два обязательных аргумента первый html и второй вид парсера `html.parser`

```
soup = BeautifulSoup(html,'html.parser')
```

Теперь мы заходим на сайт и в инспекторе ищем тег , где храниться все ссылки



Мы узнали , что все ссылки храниться в теге `<table class="table">`с классом `table` и также узнали названия храниться в теге `<td>`

Мы создадим переменную `tds` , в которую мы передадим объект супа и вызовем метод `find()`, в которую передадим

наш тег и класс и чтобы получить ссылки по одному вызовом метод `find_all()` и внутри пропишем тег `<td>`, наш код выглядит так

```
tds = soup.find('table',class_='table').find_all('td')
```

И создадим пустой список с названием `links = []`

Теперь для получения ссылок, мы пробежимся по циклу `for` `for td in tds:` мы в цикле `for` создали переменную `td` и пробегаемся по итерации внутри переменной `tds`, которую мы ранее создали

По правилам цикла вся итерация записалась в переменную `td`

Дальше мы создаем переменную `a`, так как мы будем получать ссылки с тега `a`.

И передаем значения

```
a = td.find('a').get('href')
```

Здесь мы к переменной `td` передаем метод `find` и внутри задаем тег `a`, далее мы получаем `href` с помощью метода `get()`

После мы создадим переменную `link` и к `link` мы даём домен нашего сайта и просто плюсуем наши полученные ссылки

```
link = 'http://kenesh.kg' + a
```

Следующий шаг, полученные ссылки мы апендим в переменную `links`, которую мы ранее создали

```
links.append(link)
```

и возвращаем

```
return links
```

Сейчас нам надо изменить функцию `main`

Создадим переменную `all_links` и к нему передадим нашу функцию `get_all_links()`, которая в качестве аргумента

принимает html ,то есть для этого внутри нашей функции мы передаем функцию get_html()

А наш get_html в аргументы принимает url

```
all_links = get_all_links(get_html(url))
```

Создадим переменную url и дадим в качестве значения наш

<http://www.kenesh.kg/ru/deputy/list/35>

Также пробежимся по циклу

```
for i in all_links:
```

```
    print(i)
```

Результат выглядит так :

```
import requests
```

```
from bs4 import BeautifulSoup
```

```
def get_html(url):
```

```
    r = requests.get(url)
```

```
    return r.text
```

```
def get_all_links(html):
```

```
    soup = BeautifulSoup(html,'html.parser')
```

```
    tds = soup.find('table',class_='table').find_all('td')
```

```
    links = []
```

```
    for td in tds:
```

```
        a = td.find('a').get('href')
```

```
        link = 'http://kenesh.kg' + a
```

```
        links.append(link)
```

```
    return links
```

```
def main():
```

```
    url = "http://www.kenesh.kg/ru/deputy/list/35"
```

```
    all_links = get_all_links(get_html(url))
```

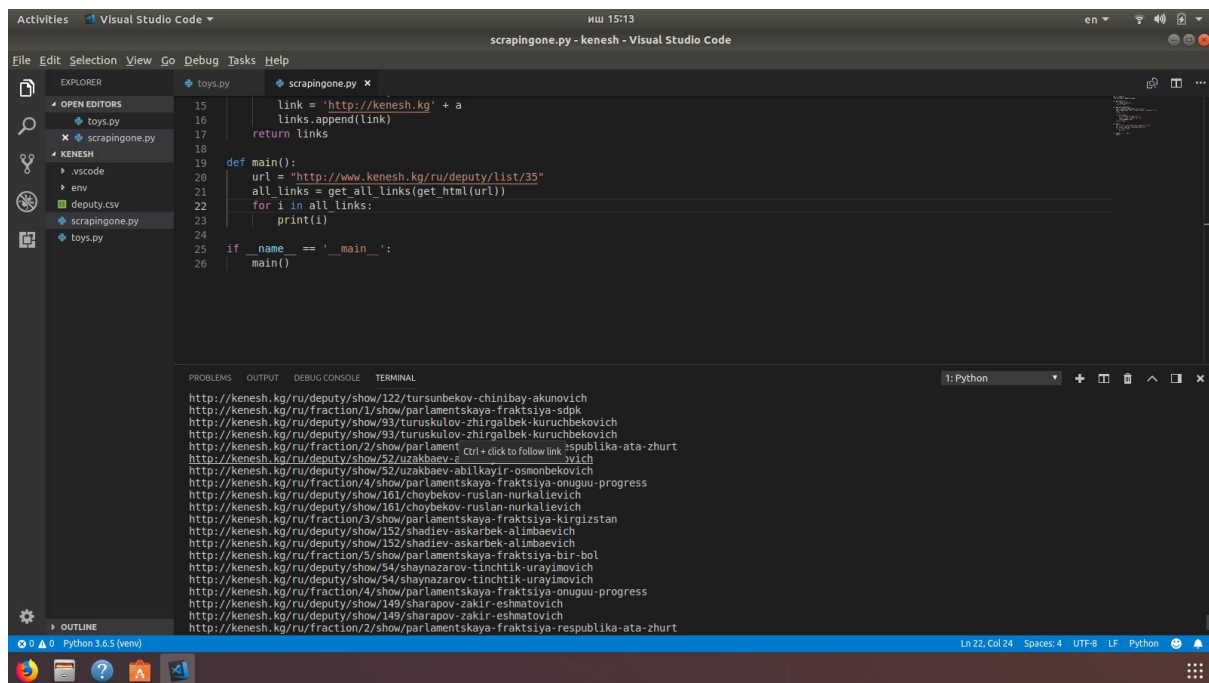
```
for i in all_links:
```

```
    print(i)
```

```
if __name__ == '__main__':
```

```
    main()
```

Запустим наш скрипт :



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a project named 'KENESH' with files like 'toys.py', 'scrapigone.py', 'deputy.csv', and 'toys.py'. The main editor displays the 'scrapigone.py' file with the following code:

```
15 link = 'http://kenesh.kg' + a
16 links.append(link)
17 return links
18
19 def main():
20     url = 'http://www.kenesh.kg/ru/deputy/list/35'
21     all_links = get_all_links(get_html(url))
22     for i in all_links:
23         print(i)
24
25 if __name__ == '__main__':
26     main()
```

The terminal at the bottom shows the output of the script, which is a list of URLs:

```
http://kenesh.kg/ru/deputy/show/122/tursunbekov-chinibay-akunovich
http://kenesh.kg/ru/fraction/1/show/parlamentskaya-fraktsiya-sdpk
http://kenesh.kg/ru/deputy/show/93/turuskulov-zhigalbek-kuruchbekovich
http://kenesh.kg/ru/deputy/show/93/turuskulov-zhigalbek-kuruchbekovich
http://kenesh.kg/ru/fraction/2/show/parlament-Cri+dicto+follow+link+sspublika-ata-zhurt
http://kenesh.kg/ru/deputy/show/52/uzakbaev-abilkayir-osmonbekovich
http://kenesh.kg/ru/fraction/4/show/parlamentskaya-fraktsiya-onuguu-progress
http://kenesh.kg/ru/deputy/show/161/choybekov-ruslan-nurkalievich
http://kenesh.kg/ru/fraction/3/show/parlamentskaya-fraktsiya-kingizstan
http://kenesh.kg/ru/deputy/show/152/shadiev-askarbek-alimbaevich
http://kenesh.kg/ru/deputy/show/152/shadiev-askarbek-alimbaevich
http://kenesh.kg/ru/fraction/5/show/parlamentskaya-fraktsiya-bir-bol
http://kenesh.kg/ru/deputy/show/54/shaynazarov-tinchik-urayimovich
http://kenesh.kg/ru/fraction/4/show/parlamentskaya-fraktsiya-onuguu-progress
http://kenesh.kg/ru/deputy/show/149/sharapov-zakir-esmatovich
http://kenesh.kg/ru/deputy/show/149/sharapov-zakir-esmatovich
http://kenesh.kg/ru/fraction/2/show/parlamentskaya-fraktsiya-respublika-ata-zhurt
```

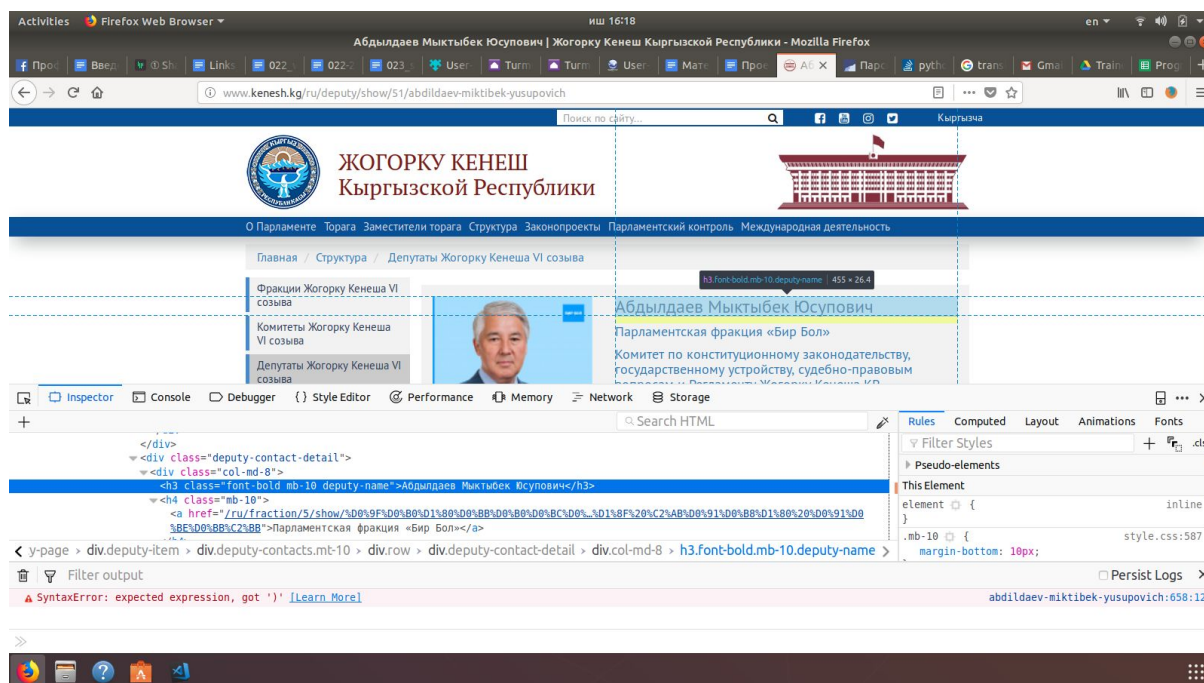
ВУАЛЯЯ получили все ссылки ,все сложное позади !

Осталось самое главное с парсить ФИО ,номер и биографию наших Депутатов)))

У нас есть готовый участок кода , где получаем ссылки .

Нам надо добавить функцию для парсинга данных , назовём `get_page_data()` , которая также будет принимать в качестве аргумента `html`, внутри нашей функции создадим объект супа и `beautiful soup` принимающий `html` и вид парсинга .

Далее переходим по ссылке и заходим в инспектор , там мы видим ,что имя депутатов внутри `<h3>` с классом `deputy-name`



Берём данные пропишем внутри try except ,просто если окажется пустым ,наш скрипт сломается ,чтобы такого не было мы пишем наши традиционные try except .Дальше вызовем метод text.strip , который удаляет символы, пробелы.

Таким образом вы получаете данные про номера и автобиографии .

Код выглядит так :

```
def get_page_data(html):
    soup = BeautifulSoup(html, 'html.parser')
    try:
        name = soup.find('h3',class_='deputy-name').text.strip()

    except:
        name = ""

    try:
        number = soup.find('p',class_='mb-10').text.strip()
```



```
except:
```

```
    number = "
```

```
try:
```

```
    bio = soup.find('div',id='biography').text.strip()
```

```
except:
```

```
    bio = "
```

```
data = {'name':name,'number':number,'bio':bio}
```

```
return data
```

В предпоследней строке мы создаем переменную data и скомпоуем в одну конструкцию в словарь , далее возвращаем .

Мы получили данные ,по задаче мы должны записать наши данные в файл с форматом csv

Для такой задачи мы должны импортировать встроенную библиотеку csv

```
import csv
```

Далее мы создадим функцию write_csv(),которая в качестве аргументы принимать наши полученные данные data

Для записи желательно использовать ,контекстный менеджер WITH as

```
with open('deputy.csv','a') as file:
```

С помощью open мы открываем файл deputy.csv, 'a' - это в расшифровке append ,то есть добавляет в конец списка . И далее as file - это означает , что мы создаем переменную под названием file и туда записываем .

Дальше создаем объект писателя назовем `writer` и пропишем модуль `csv` и вызовем метод `writer` в которую дадим аргумент файла `file` .

```
writer = csv.writer(file)
```

После также пропишем код который будем записывать в строку

```
writer.writerow((data['name'],data['number'],data['bio']))
```

Теперь в первую ячейку будет записан `name` во вторую `number` и в третью `bio`

Далее просто с принтуем в терминал

```
print(data['name'],data['number'],data['bio'],'parsed')
```

Полностью функция `write_csv` выглядит так :

```
def write_csv(data):  
    with open('deputy.csv','a') as file:  
        writer = csv.writer(file)  
        writer.writerow((data['name'],data['number'],data['bio']))  
        print(data['name'],data['number'],data['bio'],'parsed')
```

Почти всё готово , теперь осталось вызвать нашу функции в `main()`

Для этого нужно немножко изменить

```
def main():  
    start = datetime.now()  
    url = "http://www.kenesh.kg/ru/deputy/list/35"  
    all_links = get_all_links(get_html(url))  
    for url in all_links:  
        html = get_html(url)  
        data = get_page_data(html)  
        write_csv(data)
```

```
end = datetime.now()
result = end - start
print(str(result))
```

Что у нас здесь произошло ?

В первой строке мы замеряем время с помощью модуля `datetime.now()`

из библиотеки `datetime` , которую мы импортируем с стандартной библиотеки Python
`from datetime import datetime !`

Далее все оставляем как было ,только изменим наш цикл .

Как видно , мы в цикле создали переменную `url` и провели итерацию внутри переменной `all_links`

после мы записали по отдельности и вызвали функцию `write_csv` с аргументом `data`

После мы заканчиваем измерять время и принтуем время выполнения !

Полный код выглядит так :

```
import csv
import requests
from datetime import datetime
from bs4 import BeautifulSoup
```

```
def get_html(url):
    r = requests.get(url)
    return r.text
```

```
def get_all_links(html):
    soup = BeautifulSoup(html,'html.parser')
    tds = soup.find('table',class_='table').find_all('td')
```

```
links = []
```

```
for td in tds:
```

```
    a = td.find('a').get('href')
```

```
    link = 'http://kenesh.kg' + a
```

```
    links.append(link)
```

```
return links
```

```
def get_page_data(html):
```

```
    soup = BeautifulSoup(html, 'html.parser')
```

```
    try:
```

```
        name = soup.find('h3',class_='deputy-name').text.strip()
```

```
    except:
```

```
        name = "
```

```
    try:
```

```
        number = soup.find('p',class_='mb-10').text.strip()
```

```
    except:
```

```
        number = "
```

```
    try:
```

```
        bio = soup.find('div',id='biography').text.strip()
```

```
    except:
```

```
        bio = "
```

```
    data = {'name':name,'number':number,'bio':bio}
```

```
return data
```

```
def write_csv(data):
```

```
    with open('deputy.csv','a') as file:
```

```
        writer = csv.writer(file)
```

```
writer.writerow((data['name'],data['number'],data['bio']))
print(data['name'],data['number'],data['bio'],'parsed')
```

```
def main():
    start = datetime.now()
    url = "http://www.kenesh.kg/ru/deputy/list/35"
    all_links = get_all_links(get_html(url))
    for url in all_links:
        html = get_html(url)
        data = get_page_data(html)
        write_csv(data)
    end = datetime.now()
    result = end - start
    print(str(result))
```

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

УРА!!!!!!!!!!!! у нас получилось ,у меня на парсинг данных ушло времени 1,50 с

The screenshot shows the Visual Studio Code interface with a Python script named `scrapingone.py` open. The script imports `csv`, `requests`, `datetime`, and `BeautifulSoup`. It defines functions `get_html(url)` and `get_all_links(html)`. The `main` function is called at the bottom. The terminal window at the bottom displays the output of the script, showing the start and end times of the execution and the resulting data for several deputies, including their names, IDs, and biographies. A notification at the bottom right indicates that the Marketplace has extensions that can help with '.csv' files.

```
1 import csv
2 import requests
3 from datetime import datetime
4 from bs4 import BeautifulSoup
5
6 def get_html(url):
7     r = requests.get(url)
8     return r.text
9
10 def get_all_links(html):
11     soup = BeautifulSoup(html, 'html.parser')
12     tds = soup.find('table', class_='table').find_all('td')
13     links = []
14
15 С ноября 2016 года по списку политической партии «Кыргызстан» избран депутатом Жогорку Кенеша Кыргызской Республики VI созыва.
16 Женат, воспитывает троих детей. parsed
17 parsed
18 Исуров Абдумажит Лелезович 0312 63-86-48 Биография
19 Родился 12 апреля 1964 года в селе Мырдык Жети-Өгүзского района Иссык-Кульской области.
20 Образование высшее. В 1985 году окончил Кыргызский Национальный Университет имени Жусупа Баласагына по специальности «География», в 2008 году - Кыргызский Национальный университет имени Жу
21 сула Баласагына по специальности «Юриспруденция», в 2014 году - Дипломатическую Академию МИД КР им.Казы Дикамоева по специальности «Международные отношения».
22 Трудовую деятельность начал в 1986 году преподавателем кафедры ботаники и химии в Пржевальском институте.
23 С 1987-1992 годы работал инструктором в Иссык-Кульском ОБЛКСМ Киргизии.
24 С 1992 -2005 годы работал коммерческим директором СП «Айым-Галф», в 2005-2008 годы Генеральным директором ОсОО «Трэвел Альянс групп».
25 В 2007 - 2010 годы - депутат Жогорку Кенеша Кыргызской Республики IV созыва.
26 В 2015 году был избран депутатом Жогорку Кенеша от ПП «Өнүгүү-Прогресс» VI созыва.
27 Женат, воспитывает пятерых детей. parsed
28 Исуров Абдумажит Лелезович 0312 63-86-48 Биография
29 Родился 12 апреля 1964 года в селе Мырдык Жети-Өгүзского района Иссык-Кульской области.
30 Образование высшее. В 1985 году окончил Кыргызский Национальный Университет имени Жусупа Баласагына по специальности «География», в 2008 году - Кыргызский Национальный университет имени Жу
31 сула Баласагына по специальности «Юриспруденция», в 2014 году - Дипломатическую Академию МИД КР им.Казы Дикамоева по специальности «Международные отношения».
32 Трудовую деятельность начал в 1986 году преподавателем кафедры ботаники и химии в Пржевальском институте.
33 С 1987-1992 годы работал инструктором в Иссык-Кульском ОБЛКСМ Киргизии.
34 С 1992 -2005 годы работал коммерческим директором СП «Айым-Галф», в 2005-2008 годы Генеральным директором ОсОО «Трэвел Альянс групп».
35 В 2007 - 2010 годы - депутат Жогорку Кенеша Кыргызской Республики IV созыва.
36 В 2015 году был избран депутатом Жогорку Кенеша от ПП «Өнүгүү-Прогресс» VI созыва.
37 Женат, воспитывает пятерых детей. parsed
38 parsed
39 03:50:089976
40 (env) dastan@dastan:~/projects/kenesh$
```

Всё сложное позади ,только нас не устраивает время парсера 2 минуты это очень долго , но за нас создатели питона придумали Multiprocessing - многопоточность .

Давайте ведём изменения

В модуле multiprocessing импортируем класс Pool

```
from multiprocessing import Pool
```

Что такое Pool?

Наверняка вы пили с диспенсера воду,смысл пула в этом .

То есть в офисе есть диспенсер и все работники подходят и пьют воду ,а пул также работает только с задачами , каждой задаче дает по одному процессу (стакану воды) .

Наша задача заключается в том , что нужно парсить данные очень быстро ,для этого нужно поделить наши задачи на несколько процессов !

Пожалуй начнём)))

В main мы пишем контекстный менеджер

```
with Pool(40) as p:
```

```
    p.map(make_all,all_links)
```

передаем 40 процессов и в переменной объекта вызовим функцию map()

Что такое функция map - это универсальная функция , которая в себе имеет цикл и принимает в качестве аргумента функцию и переменную ,далее все проходимые итерации записывает в функцию .

Я до этого создал волшебную функцию make_all - которое делает все)))

В make_all в качестве аргумента передаем url ,далее некоторые данные с main перенесем в make_all выглядит это так

```
def make_all(url):
```

```
html = get_html(url)
data = get_page_data(html)
write_csv(data)
```

Теперь наша функция будет в map принимать итерируемые данные с all_links

Полный код нашего многопоточного парсера выглядит так :

```
import requests
import csv
from bs4 import BeautifulSoup
from datetime import datetime
from multiprocessing import Pool

def get_html(url):
    r = requests.get(url)
    return r.text

def get_all_links(html):
    soup = BeautifulSoup(html, 'html.parser')
    tds = soup.find('table', class_='table').find_all('td')
    links = []

    for td in tds:
        a = td.find('a').get('href')
        link = 'http://kenesh.kg' + a
        links.append(link)
    return links

def get_page_data(html):
    soup = BeautifulSoup(html, 'html.parser')
    try:
        name = soup.find('h3', class_='deputy-name').text.strip()
```



```
except:
```

```
    name = "
```

```
try:
```

```
    number = soup.find('p',class_='mb-10').text.strip()
```

```
except:
```

```
    number = "
```

```
try:
```

```
    bio = soup.find('div',id='biography').text.strip()
```

```
except:
```

```
    bio = "
```

```
data = {'name':name,'number':number,'bio':bio}
```

```
return data
```

```
def write_csv(data):
```

```
    with open('deputy.csv','a') as file:
```

```
        writer = csv.writer(file)
```

```
        writer.writerow((data['name'],data['number'],data['bio']))
```

```
        print(data['name'],data['number'],data['bio'],'parsed')
```

```
def make_all(url):
```

```
    html = get_html(url)
```

```
    data = get_page_data(html)
```

```
    write_csv(data)
```

```
def main():
```

```
    start = datetime.now()
```

```
    url = "http://kenesh.kg/ru/deputy/list/35"
```

```
    all_links = get_all_links(get_html(url))
```

```
    with Pool(40) as p:
```

```
        p.map(make_all,all_links)
```

```
    end = datetime.now()
```

```
total = end - start
```

```
print(str(total))
```

```
if __name__ == '__main__':
```

```
    main()
```

The screenshot shows a Linux desktop environment. On the left, a terminal window displays a list of processes running under the user 'dastan'. The processes are all named 'python' and have various IDs and memory usage. On the right, a code editor window shows a Python script. The script is a simple program that prints the total time taken to execute a task. The script is as follows:

```
import time
start = time.time()
# ... (some code) ...
end = time.time()
total = end - start
print(str(total))
```

The terminal window also shows the output of the script, which is '15.000000'.

ДАВАЙТЕ РАДОВАТЬСЯ , ВУАЛЯЯ мы закончили у меня
спарсило за 15 секунд

The screenshot shows the Visual Studio Code interface with a Python script named `toys.py` open in the editor. The script uses `requests` to fetch data from a website and `concurrent.futures.ThreadPoolExecutor` to process the data in parallel. The terminal window displays the output of the script, which is a list of 10 'parsed' entries. The status bar at the bottom indicates the file encoding is UTF-8 and the language is Python.

```
47 data = get_page_data(html)
48 write_csv(data)
49 def main():
50     start = datetime.now()
51     url = "http://kenesh.kg/ru/deputy/list/35"
52     all_links = get_all_links(get_html(url))
53     with Pool(40) as p:
54         p.map(make_all, all_links)
55     end = datetime.now()
56     total = end - start
57     print(str(total))
58
59 if __name__ == '__main__':
60     main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

1: Python

Образование высшее. Окончил Кыргызский государственный институт физической культуры по специальности «тренер-преподаватель». Трудовую деятельность начал в 1991 году тренером в Жалал-Абадской Специализированной детско-юношеской школе олимпийского резерва (СДЮШОР), где проработал до 1996 года. В 1996-2004 годы занимался частным предпринимательством; 2004-2009 годы – депутат городского кенеша города Жалал-Абад; 2011-2012 годы – директор СДЮШОР КЗ имени Р.Санатбаева; С 2012 по настоящее время – депутат городского кенеша города Жалал-Абад. С 2009 года президент Федерации греко-римской борьбы Жалал-Абадской области; Мастер спорта по греко-римской борьбе СССР 1991 года, Заслуженный тренер КР, Отличник образования КР, Отличник Физической культуры и спорта КР. В 2011 году за вклад в развитие образования удостоен Почетной грамоты Министерства образования и науки Кыргызской Республики, В 2013 году награжден олагодарственным письмом от премьер-министра Кыргызстана. Женат, воспитывает 6 детей.

parsed
parsed
parsed
parsed
parsed
parsed
parsed
parsed
parsed
parsed

0:00:15.084459
(env) dastan@dastan:~/projects/kenesh\$

The Marketplace has extensions that can help with '.csv' files

Search Marketplace

Ln 53, Col 23 Spaces: 4 UTF-8 LF Python

Надеюсь эта информация поможет вам в дальнейших проектах .

