Учитывая современное развитие Интернета, было бы кощунством не написать приложение, взаимодействующее со всемирной паутиной. Сегодня мы напишем простенький **html-парсер на Python**. Наше приложение будет читать код указанной страницы сайта и сохранять все ссылки в ней в отдельный файл. Это приложение может помочь SEO-аналитикам и веб-разработчикам.

Писать будем на **Python 3**, в котором есть встроенный класс для html-парсера из [модуля](http://pythonicway.com/python-modules) html.parser

[?](http://pythonicway.com/python-examples/python-terminal-examples/20-python-html-parser)

|  |  |
| --- | --- |
| 1 | from html.parser import HTMLParser |

Так же нам понадобится функция urlopen из модуля urllib

[?](http://pythonicway.com/python-examples/python-terminal-examples/20-python-html-parser)

|  |  |
| --- | --- |
| 1 | from urllib.request import urlopen |

Именно функция urlopen будет получать исходный код указанной странички.

Перегрузка класса HTMLParser

Наша задача таким образом перегрузить функционал существующего класса HTMLParser, чтобы он соответствовал нашей задаче.

[?](http://pythonicway.com/python-examples/python-terminal-examples/20-python-html-parser)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | class MyHTMLParser(HTMLParser):      def \_\_init\_\_(self, site\_name, \*args, \*\*kwargs):          # список ссылок          self.links = []          # имя сайта          self.site\_name = site\_name          # вызываем \_\_init\_\_ родителя          super().\_\_init\_\_(\*args, \*\*kwargs)          # при инициализации "скармливаем" парсеру содержимое страницы          self.feed(self.read\_site\_content())          # записываем список ссылок в файл          self.write\_to\_file() |

Базовый класс HTMLParser имеет несколько методов, нас в данном случае интересуют метод handle\_start\_tag. Этот метод вызывается каждый раз, когда наш парсер встречает в тексте октрывающий html-тэг.

[?](http://pythonicway.com/python-examples/python-terminal-examples/20-python-html-parser)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | def handle\_starttag(self, tag, attrs):       # проверяем является ли тэг тэгом ссылки       if tag == 'a':           # находим аттрибут адреса ссылки           for attr in attrs:               if attr[0] == 'href':                   # проверяем эту ссылку методом validate() (мы его еще напишем)                   if not self.validate(attr[0]):                       # вставляем адрес в список ссылок                            self.links.append(attr[1]) |

Напишем вспомогательный метод validate:

[?](http://pythonicway.com/python-examples/python-terminal-examples/20-python-html-parser)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | def validate(self, link):       """ Функция проверяет стоит ли добавлять ссылку в список адресов.       В список адресов стоит добавлять если ссылка:            1) Еще не в списке ссылок            2) Не вызывает javascript-код            3) Не ведет к какой-либо метке. (Не содержит #)        """        return link in self.links or '#' in link or 'javascript:' in link |

Создадим метод, который будет открывать указанную страницу и выдавать ее содержимое.

[?](http://pythonicway.com/python-examples/python-terminal-examples/20-python-html-parser)

|  |  |
| --- | --- |
| 1  2 | def read\_site\_content(self):      return str(urlopen(self.site\_name).read()) |

Осталось добавить возможность записи списка ссылок на диск в читабельном формате:

[?](http://pythonicway.com/python-examples/python-terminal-examples/20-python-html-parser)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | def write\_to\_file(self):       # открываем файл       f = open('links.txt', 'w')       # записываем отсортированный список ссылок, каждая с новой строки       f.write('\n'.join(sorted(self.links)))       # закрываем файл       f.close() |

Все готово, можем запускать парсер.

[?](http://pythonicway.com/python-examples/python-terminal-examples/20-python-html-parser)

|  |  |
| --- | --- |
| 1 | parser = MyHTMLParser("[http://python.org](http://python.org/)") |

После того как вы запустите данный скрипт в директории, где находится ваш файл появится текстовый документ links.txt, содержащий ссылки.



Конечно, данный пример достаточно примитивен, но на его основе вы можете попробовать написать, к примеру, веб-crawler, который будет анализировать весь сайт целиком, а не одну его страницу