УДК 004

Реализация jQuery select на языке python

Радионов Сергей Владимирович
Приамурский государственный университет им. Шолом-Алейхема
Студент

Аннотация
Целью данного исследования является реализация функции jQuery select. В статье показан процесс разработки программы на языке программирования python. Результатом исследования является функция, которая может по введённому паттерну вернуть объект тэга HTML.
Ключевые слова: python, html, синтаксический анализ, программа.

Python jQuery select implementation

Radionov Sergey Vladimirovich
Sholom-Aleichem Priamursky State University
Student

Abstract
The purpose of this study is to implement the jQuery select function. The article shows the process of developing a program in the python programming language. The result of the study is a function that can return the object of an HTML tag according to the entered pattern.
Keywords: python, html, parsing, program.

В мире широко распространен интернет. Основным средством просмотра информации в интернете являются web страницы. Эти страницы написаны на гипертекстовом языке HTML. Чтобы получить данные из HTML текста, такие как цена продукта в интернет-магазине или данные о погоде метеорологического сервиса, необходимо провести синтаксический анализ (парсинг) кода страницы. Для получения данных из тэгов нужно получить этот тэг, а для этого удобно использовать поиск по паттерну, такой как у jQuery select.

В статье И.Кутепова рассматривается пример построения вычислителя на основе нечеткой логики и применение языка Python для написания интегрированной среды разработки [2]. И.Е.Бронштейн в своей статье описал вывод типов для программного кода на языке Python. Сначала производится обзор описанных в научной литературе алгоритмов вывода типов для языков с параметрическим полиморфизмом. Затем даётся описание нового алгоритма, являющегося модификацией одного из предыдущих: алгоритма декартова произведения. Показывается, как модуль вывода типов, использующий новый алгоритм, анализирует различные конструкции языка

Python. Представляются результаты работы над прототипом [3]. В работе В.В.Найденова протестирована производительность пары аналогичных приложений, реализующих CRUD логику с помощью прослойки ORM. Сравнивается SQLAlchemy – де-факто стандартный ORM для Python с динамическим ORM для C++ собственной разработки – YB.ORM. Сравнивается производительность при использовании CPython и PyPy. Проверяется влияние отключения логов на производительность [4]. В статье И.Е.Бронштейн рассматриваются виды дефектов, которые обычно встречаются в программном коде на языке Python. Показывается, что возможные дефекты для Python не похожи на те, что часто встречаются в коде на Си/Си++ и, следовательно, необходимо исследование дефектов в крупных проектах с открытым исходным кодом. Даётся классификация найденных дефектов на основе того, нужен ли для нахождения ошибки вывод типов. Показывается, что существует небольшая доля "простых" дефектов, но для обнаружения большинства дефектов вывод типов необходим. Рассматривается вопрос, какие конструкции языка Python должны поддерживаться при выводе типов для нахождения реальных дефектов [5]. В работе Д.А.Кузнецов рассматривается структура интерфейса программы «Фармацевтическая экономическая безопасность» для фармацевтических организаций. Описывается функциональное предназначение и возможности пунктов основного меню прикладной программы [6]. Ю.А.Котов, А.В.Шаповалов в своей статье рассмотрели интерфейс программной реализации экспертной системы для восстановления простой замены букв текста. Описаны базовые элементы интерфейса, включающие выбор функциональных методов и базовых операций (замена, сдвиг, перебор). Не менее значимы иностранные исследования в данной сфере [8-9].

Данная статья является продолжением статьи синтаксического анализа HTML [10]. Поэтому сначала необходимо создать скрипты как в той статье.

Сначала создадим функцию, которая будет возвращать список значений атрибута из паттерна. Назовем эту функцию get_group_list (Рис.1). На вход подается символ атрибута, для класса это точка, а для id – решетка. Эта функция лежит внутри класса HtmlTag. Подав в эту функцию паттерн «div.class1.class2#id» и символ точки, она вернет список из 2ух классов. Если подать символ решетки вернет id.

Для того чтобы можно было выбрать нужный тэг в дереве тэгов, добавим в класс HtmlTag функцию select (Рис.2). Для нахождения нужного тэга будет использоваться алгоритм поиска в ширину (BFS) [11].

```python
@staticmethod
def get_group_list(char, selector):
    i = 0
    group = []
    tmp = ''
    is_group = False
    while i < len(selector):
        if selector[i] == char:
            if len(tmp) > 0:
                group.append(tmp)
            tmp = ''
            is_group = True
        elif selector[i] == '.' or selector[i] == '#':
            if len(tmp) > 0:
                group.append(tmp)
            tmp = ''
            is_group = False
        elif is_group:
            tmp += selector[i]
        i += 1
    if is_group:
        group.append(tmp)
    return group
```

Рис.1 Функция get_group_list

```python
def select(self, cmd):
    selectors = cmd.split(' ')
    q = Queue()
    q.put({'element': self, 'selectors': selectors})
    results = []
```

Рис.2. Функция select

В функции задаем начало поиска от тэга, с которого она вызвана и осуществляем поиск в ширину (Рис.3). Символ пробела в паттерне обозначает, что следующие за ним набор значений тэга относится к дочерним. Таким образом есть 3 варианта результат проверки каждого тэга в алгоритме:

1. Тэг не подходит по критериям поиска – добавляются все дочерние тэги в поиск и паттерн остается без изменений.
2. Тэг подходит под первую часть паттерна (до первого пробела в паттерне), существует вторая часть паттерна – добавляются все дочерние тэги, а из паттерна убирается первая часть.
3. Тэг подходит под первую часть паттерна (до первого пробела в паттерне), второй части паттерна не существует – тэг добавляется в результаты.

```
while not q.empty():
    data = q.get()
    elem = data['element']
    selectors = data['selectors']

    for child in elem.childrens:
        q.put({'element': child, 'selectors': selectors})

    s_classes = None
    if '.' in selectors[0]:
        s_classes = self.get_group_list('.', selectors[0])     #list of classes
    s_ids = None
    if '#' in selectors[0]:
        s_ids = self.get_group_list('#', selectors[0])[0]      #id

    tag = copy.copy(selectors[0])                       #clean name
    if '.' in tag:
        tag = tag[:tag.find('.')]
    if '#' in tag:
        tag = tag[:tag.find('#')]

    tag_check = True
    classes_check = True
    ids_check = True

    if tag != '':
        tag_check = elem.tag == tag
    if s_classes:
        for s_class in s_classes:
            if ('class' not in elem.attrs) or (s_class not in elem.attrs['class']):
                classes_check = False
                break
    if s_ids:
        ids_check = 'id' in elem.attrs and elem.attrs['id'] == s_ids

    if tag_check and classes_check and ids_check:
        if len(selectors) == 1:
            results.append(elem)
        else:
            s = copy.deepcopy(selectors)
            del s[0]
            for child in elem.childrens:
                q.put({'element': child, 'selectors': s})
return results
```

Рис.3. Поиск в ширину

Таким образом была реализована функция jQuery select для ранее синтаксически проанализированного HTML текста. Для использования функции достаточно вызвать select() у тэга от которого необходимо начать поиск.

Библиографический список

1. Лутц М. Изучаем python. М., 2009.
2. Кутепов И. Применение языка python при проектировании нечеткого контроллера // Компоненты и технологии. 2013. № 8 (145). С. 148-154.
3. Бронштейн И.Е. Вывод типов для языка python // Труды Института системного программирования РАН. 2013. Т. 24. С. 161-190.
4. Найденов В.В. Тестирование производительности orm в языках python и c++ // RSDN Magazine. 2014. № 1. С. 05-08.
5. Бронштейн И.Е. Исследование дефектов в коде программ на языке python // Программирование. 2013. Т. 39. № 6. С. 25-32.
6. Кузнецов Д.А. Интерфейс программы "фармацевтическая экономическая безопасность" // Российский медико-биологический вестник им. академика И.П. Павлова. 2009. № 2. С. 162-165.
7. Котов Ю.А., Шаповалов А.В. Интерфейс программы восстановления простой замены букв текста // Современные тенденции развития науки и технологий. 2016. № 4-4. С. 57-59.
8. Smith A. W. et al. Application program interface that enables communication for a network software platform : пат. 7117504 США. – 2006.
9. Parikh V., Moore R., Cheng H. Application program interface for a graphics system : пат. 6456290 США. – 2002.
10. Создание скрипта для синтаксического анализа HTML // Постулат URL: http://e-postulat.ru/index.php/Postulat/article/view/3045/3091 (дата обращения: 23.01.2020).
11. Реализация алгоритма поиска в ширину на языке python // Постулат URL: http://e-postulat.ru/index.php/Postulat/article/view/3055/3101 (дата обращения: 23.01.2020).