Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Мегафакультет компьютерных технологий и управления

Факультет программной инженерии и компьютерной техники

Информатика

Лабораторная работа №4

Исследование протоколов, форматов обмена информацией и языков разметки документов

Вариант 11

Студент:

Бутов Иван Алексеевич

Группа: P3117

Преподаватель:

Машина Екатерина Алексеевна

Санкт-Петербург

2022

Оглавление

Задание

1. Изучить особенности языков разметки/форматов JSON, YAML, XML.
2. Понять устройство страницы с расписанием для своей группы.
3. Сформировать файл с расписанием в формате XML.
4. Написать программу на языке Python 3.x, которая бы осуществляла парсинг и конвертацию исходного файла в новый (без библиотек и регулярных выражений).
5. Выполнить задание с использованием сторонних библиотек.
6. Переписать исходный код, добавив в него использование регулярных выражений.
7. Сравнить полученные результаты и объяснить их сходство/различие.
8. Используя свою исходную программу из обязательного задания, программу из дополнительного задания No1 и программу из дополнительного задания No2, сравнить стократное время выполнения парсинга + конвертации в цикле.
9. Переписать исходную программу, чтобы она осуществляла парсинг и конвертацию исходного файла в любой другой формат (кроме JSON, YAML, XML, HTML): PROTOBUF, TSV, CSV, WML и т.п.

Основные этапы выполнения

Устройство расписания на сайте

Расписание <table>

День <day>

Занятие <lesson>

Неделя <week_type>

Время <time>

Группы <groups>

Корпус <corpus>

Аудитория <room>

Предмет <subject>

Преподаватель <teacher>

Формат <lesson_format>

Файл с расписанием в формате XML

```
<?xml version = "1.0" encoding = "UTF-8"?>

<table>
    <day day_of_week="Вт">
        <lesson>
            <week_type>нечётная неделя</week_type>
            <time_start>08:20</time_start>
            <time_end>09:50</time_end>
            <groups>3, 7, 9, 11, 13, 15, 17</groups>
            <corpus>ул.Ломоносова, д.9, лит. М</corpus>
            <room></room>
            <subject>Информатика(Лек): Актовый зал</subject>
            <teacher>Балакшин Павел Валерьевич</teacher>
            <lesson_format>Очно - дистанционный </lesson_format>
        </lesson>
        <lesson>
            <week_type>чётная неделя</week_type>
            <time_start>10:00</time_start>
            <time_end>11:30</time_end>
            <groups>2, 4, 6, 8, 10, 12, 14, 16</groups>
            <corpus>Кронверкский пр., д.49, лит.А</corpus>
            <room>2332 (бывш. 324) ауд.</room>
            <subject>Основы профессиональной деятельности(Лаб)</subject>
            <teacher>Перцев Тимофей Сергеевич</teacher>
            <lesson_format>Очно - дистанционный </lesson_format>
        </lesson>
        <lesson>
            <week_type>нечётная неделя</week_type>
            <time_start>10:00</time_start>
            <time_end>11:30</time_end>
            <groups>3, 7, 9, 11, 13, 15, 17</groups>
            <corpus>ул.Ломоносова, д.9, лит. М</corpus>
            <room></room>
            <subject>Основы профессиональной деятельности(Лек): Актовый
зал</subject>
            <teacher>Клименков Сергей Викторович</teacher>
            <lesson_format>Очно - дистанционный </lesson_format>
        </lesson>
        <lesson>
            <week_type>нечётная неделя</week_type>
            <time_start>11:40</time_start>
            <time_end>13:10</time_end>
            <groups>3, 7, 9, 11, 13, 15, 17</groups>
            <corpus>ул.Ломоносова, д.9, лит. М</corpus>
            <room>1223 ауд.</room>
            <subject>Программирование(Лек)</subject>
            <teacher>Письмак Алексей Евгеньевич</teacher>
            <lesson_format>Очно - дистанционный </lesson_format>
        </lesson>
        <lesson>
            <week_type>чётная неделя</week_type>
            <time_start>11:40</time_start>
            <time_end>13:10</time_end>
            <groups>2, 4, 6, 8, 10, 12, 14, 16</groups>
            <corpus>Кронверкский пр., д.49, лит.А</corpus>
            <room>2332 (бывш. 324) ауд.</room>
            <subject>Основы профессиональной деятельности(Лаб)</subject>
            <teacher>Перцев Тимофей Сергеевич</teacher>
            <lesson_format>Очно - дистанционный </lesson_format>
        </lesson>
    </day>
</table>
```

Основная программа

```
FILE = open('../Вторник.xml', 'r', encoding="utf-8")
file = FILE.read()

CursorPosition = 0
objectStack = []
curNesting = 0
content = ""

while CursorPosition < len(file):
      curSymb = file[CursorPosition]
      if curSymb == '<':
            tagHandler()
      else:
            content += curSymb
        CursorPosition += 1

#fileOut()
#simpleOut()
YAMLOut()
```

Вспомогательные функции

```
class Object:

    def __init__(self, name, nestingLevel):
        self.name = name
        self.nestingLevel = nestingLevel
        self.content = ""
        self.isMeaning = True

def stackFindFirstFromEnd(nesting):
    global objectStack
    for i in range (len(objectStack)-1, 0-1, -1):
        obj = objectStack[i]
        if obj.nestingLevel == nesting:
            return obj
    print("NOT_FIND_SUCH_OBJECT_EXCEPTION")

def getTagNameWithAttribute(str):
    name = ""
    for i in range(len(str)):
        if str[i] == '<':
            return "NEW_OPEN_TAG_EXCEPTION"
        if str[i] == '>':
            return name
        name += str[i]

def attributeHandler(tagContent):
    global objectStack
    global curNesting
    for i in range(1, len(tagContent)):
        attList = tagContent[i].split("=")
        obj = Object(attList[0], curNesting)
        obj.content = attList[1]
        objectStack.append(obj)

def openTagHadler(tagName, tagContent):
    global objectStack
    global curNesting
    global content

    obj = Object(tagName, curNesting)
    objectStack.append(obj)
    if curNesting != 0:
        parentObj = stackFindFirstFromEnd(curNesting - 1)
        parentObj.isMeaning = False
    curNesting += 1
    content = ""
    attributeHandler(tagContent)
```

```
def closeTagHandler(tagName, tagContent):
    global curNesting
    global content

    curNesting -= 1
    obj = stackFindFirstFromEnd(curNesting)
    if tagName == '/' + obj.name:
        if obj.isMeaning and len(tagContent) == 1:
            obj.content = content
        elif obj.isMeaning and len(tagContent) > 1:
            pass
        elif not (obj.isMeaning) and len(tagContent) > 1:
            pass
        content = ""
    else:
        print("INVALID_XML_TAG_EXCEPTION")

def tagHandler():
    global CursorPosition

    tagNameWithAttribute = getTagNameWithAttribute(file[CursorPosition + 1:])
    tagContent = tagNameWithAttribute.split(' ')
    tagName = tagContent[0]
    if tagName[0] == '/':
        closeTagHandler(tagName, tagContent)
    elif tagName.__contains__("?xml"):
        pass
    else:
        openTagHadler(tagName, tagContent)
    CursorPosition += len(tagNameWithAttribute) + 1

def fileOut():
    global file
    print()
    print("FILE:")
    print(file)
    print()

def simpleOut():
    global objectStack
    print()
    print("SIMPLE OUT:")
    for obj in objectStack:
        print(obj.nestingLevel, obj.name, ":", obj.content)
    print()
```

Функция для вывода в формате YAML

```
def YAMLOut():
    global objectStack
    writeSpaces = True
    print()
    print("YAML OUT:")
    l = [0] * 10
    isWriteListing = [False]*10
    for i in range(len(objectStack)):
        obj = objectStack[i]

        for j in range(i+1, len(objectStack)):
            nextObj = objectStack[j]
            if nextObj.nestingLevel < obj.nestingLevel:
                break
            if nextObj.nestingLevel > obj.nestingLevel:
                continue
            if (nextObj.name == obj.name) and not(isWriteListing[obj.nestingLevel]):
                l[obj.nestingLevel] += 1
            else:
                break
        #spaces
        newLine = True
        str = ""
        if writeSpaces:
            str += "  " * (obj.nestingLevel)
        writeSpaces = True
        #name or -
        if l[obj.nestingLevel] == 0:
            str += obj.name + ": "
        else:
            if not(isWriteListing[obj.nestingLevel]):
                str += obj.name + ": "
                isWriteListing[obj.nestingLevel] = True
                l[obj.nestingLevel] += 1
                if obj.content != "":
                    str += "\n" + ("  " * (obj.nestingLevel)) + "  - "
                if obj.content == "":
                    str += "\n" + ("  " * (obj.nestingLevel)) + "- "
                    newLine = False
                    writeSpaces = False

            else:
                if obj.content != "":
                    str += "  - "
                else:
                    if (i+1<len(objectStack)) and objectStack[i+1].nestingLevel >
obj.nestingLevel:
                        str += "- " #переход на новую строку (кроме после заголовка)
                        newLine = False
                        writeSpaces = False
                    else:
                        str += "- " #обработка пустых значений тэгов

            l[obj.nestingLevel] -= 1
            if l[obj.nestingLevel] == 0:
                isWriteListing[obj.nestingLevel] = False

        #content
        str += obj.content

        #print it
        if newLine:
            print(str)
        else:
            print(str, end='')

    print()
```

Измененные функции с добавлением регулярных выражений

```
#add reg
def getTagNameWithAttribute(str):
    match = re.findall(r"(.+?)\>", str)
    #print(match[0])
    return match[0]

#add reg
def attributeHandler(tagContent):
    global objectStack
    global curNesting
    for i in range(1, len(tagContent)):
        print(tagContent[i])
        attList = re.findall(r"[\w\"]+[^=]", tagContent[i])
        #print('reg:', attList)
        obj = Object(attList[0], curNesting)
        obj.content = attList[1]
        objectStack.append(obj)

#add reg
def tagHandler():
    global CursorPosition

    tagNameWithAttribute = getTagNameWithAttribute(file[CursorPosition + 1:])
    tagContent = re.findall(r"[\w=.\"/]+", tagNameWithAttribute)
    #print(tagNameWithAttribute)
    #print(tagContent)
    tagName = tagContent[0]
    if re.match(r"/", tagNameWithAttribute):
        closeTagHandler(tagName, tagContent)
    elif re.match(r"\?xml", tagNameWithAttribute):
        pass
    else:
        openTagHadler(tagName, tagContent)
    CursorPosition += len(tagNameWithAttribute) + 1
```

С использованием библиотек

```
import yaml
import xmltodict
#import xmldict_translate

FILE = open('../Вторник.xml', 'r', encoding="utf-8")
file = FILE.read()

pyObj = xmltodict.parse(file)
#pyObj = xmldict_translate.xml2dict(file)
print(pyObj)
yamlDoc = yaml.dump(pyObj, encoding=None, allow_unicode=True)
print(yamlDoc)
```

Анализ времени выполнения

Среднее время выполнения конвертации файла из XML в YAML рассчитывалось на основе времени выполнения операции 100 раз.

Собственный конвертер:	0.00193288
Собственный конвертер + регулярные выражения:	0.00522456
Конвертер на основе библиотек:	0.00409738

Собственный конвертер оказался самым быстрым, так как он проектировался под конкретную, а не общую задачу. Добавление регулярных выражений замедлило его работу, так как подключение библиотеки занимает некоторое время, но не дает большого выигрыша во времени исполнения. Конвертер на основе готовых библиотек справился быстрее регулярных выражений, но все еще медленнее собственного конвертера.

Вывод

В ходе лабораторной работы познакомился с основными языками разметками и принципами их работы. Создание собственного парсера и конвертера позволили мне значительно лучше понять языки разметки и методы их практического применения.

Список литературы

1. Лямин А.В., Череповская Е.Н. Объектно-ориентированное программирование. Компьютерный практикум. – СПб: Университет ИТМО, 2017. – 143 с. – Режим доступа: https://books.ifmo.ru/file/pdf/2256.pdf.
2. Алексеев Е.Г., Богатырев С.Д. Информатика. Мультимедийный электронный учебник. – Режим доступа: http://inf.e-alekseev.ru/text/toc.html