Оспанова А. Б., Туякбаева А. С.

Евразийский национальный университет им. Л. Н. Гумилева, г. Астана

o.ademi111@gmail.com, aidanaddmin.mks@mail.ru

Автоматизация работы с электронными документами

*В работе исследованы возможности модулей для языка Python для автоматизации работы с электронными документами. С этой целью изучен необходимый программный инструментарий, выполнена установка и настройка. В качестве примера реализована автоматическая генерация документов MS Word по шаблону.*

Скриптовый язык Python обладает многим достоинствами, предоставляющими возможности эффективного решения различных задач программирования. Это простота, лаконичность, гибкость в смысле широких возможностей применения, наличие множества модулей и библиотек, кроссплатформенность, а также мощная поддержка ввиду популярности языка. Язык эффективно используется для создания Desktop-приложений, в Web-программировании, в научно-исследовательских целях для работы с большими данными, в криптографии.

Существует множество модулей и библиотек, предоставляющих интересные возможности оптимизации некоторых задач в системах электронного документооборота. Так, в таблице 1 приведены некоторые пакеты, предназначенные для автоматизации и оптимизации работ с pdf-файлами, Excel-таблицами, документами Word, Web-содержимым [1-16].

Таблица 1. Пакеты и модули для Python для обработки документов и текстов

|  |  |  |
| --- | --- | --- |
| Форматы | Модули и пакеты | Краткое описание |
| Работа с Pdf-файлами | ReportLab | Создание pdf-документов |
| rst2pdf | Перевод текстовых документов в pdf-формат с сохранением стилей |
| PyPDF2 | Предоставляет обширные возможности для работы с pdf-файлами |
| pisa | Конвертирование документов форматов XHTML/HTML/CSS в pdf |
| Работа с Excel-таблицами | openpyxl | Библиотека для чтения и записи таблиц Excel |
| xlrd | Чтение файлов Excel |
| xlwt | Создание и заполнение файлов Excel |
| xlutils | Набор утилит для расширения возможностей xlrd и xlwt |
| pyExcelerator | Библиотека для создания файлов Excel |
| Работа с документами Word | uno | Работа с OpenOffice.org/LibreOffice |
| python-docx | Создание и редактирование файлов MS Word (.docx) |
| Работа с Web-содержимым | Scrapy | Извлечение и экспортирование данных с Web-ресурсов |
| Urllib, urllib2 (Python 2.x), urllib.\* (Python 3.0) | Работа с Web-данными посредством обработки URL |
| mechanize | Навигация по Web-формам |
| Beautiful Soup | Библиотека для извлечения данных из HTML и XML файлов |

Размеры потоков цифровой информации, с которой имеют дело практически все современные организации, обуславливают необходимость искать и разрабатывать средства автоматической обработки электронных документов.

В данной работе изучены возможности пакета Python-docx. Модуль позволяет генерировать документы MS Word по заданному шаблону, редактировать документы и прочее. В качестве примера с помощью этого инструмента сгенерированы файлы тезисов на Традиционную IX Международную научно-практическую конференцию молодых ученых, докторантов, магистрантов и студентов «Образование, наука, инновации – вклад молодых исследователей», посвященную 20-летию столицы Республики Казахстан и университета «Туран-Астана» некоторых участников из ЕНУ им. Л. Н. Гумилева (кафедра «Информатики и информационной безопасности»); в частности, данный файл.

Так, на языке Python с использованием модуля Python-docx в текстовом редакторе посредством списков (структура данных в python) и циклов были подготовлены:

- база участников конференции – авторов статей с необходимой информацией для заполнения заявки,

- база с названиями тезисов,

- база данных со списками использованных источников – рис. 1.

|  |
| --- |
| authors = [[u'Оспанова А. Б.', 'o.ademi111@gmail'], [u'Туякбаева А. С.', 'aidanaddmin.mks@mail.ru'], [u'Уалиханов Г. С.', 'vip.galimkhan@mail.ru']]  thesis\_name = [u'Автоматизация работы с электронными документами', u'Управление утилитами Kali Linux'] |

Рисунок 1. Подготовленная информация для автоматической обработки

Далее с помощью функции, позволяющей управлять стилями абзацев и разделов в документах Word ([17]), создан стиль, описанный в требованиях к формату тезисов, перечисленных в соответствующем информационном письме данной конференции – рис. 2.

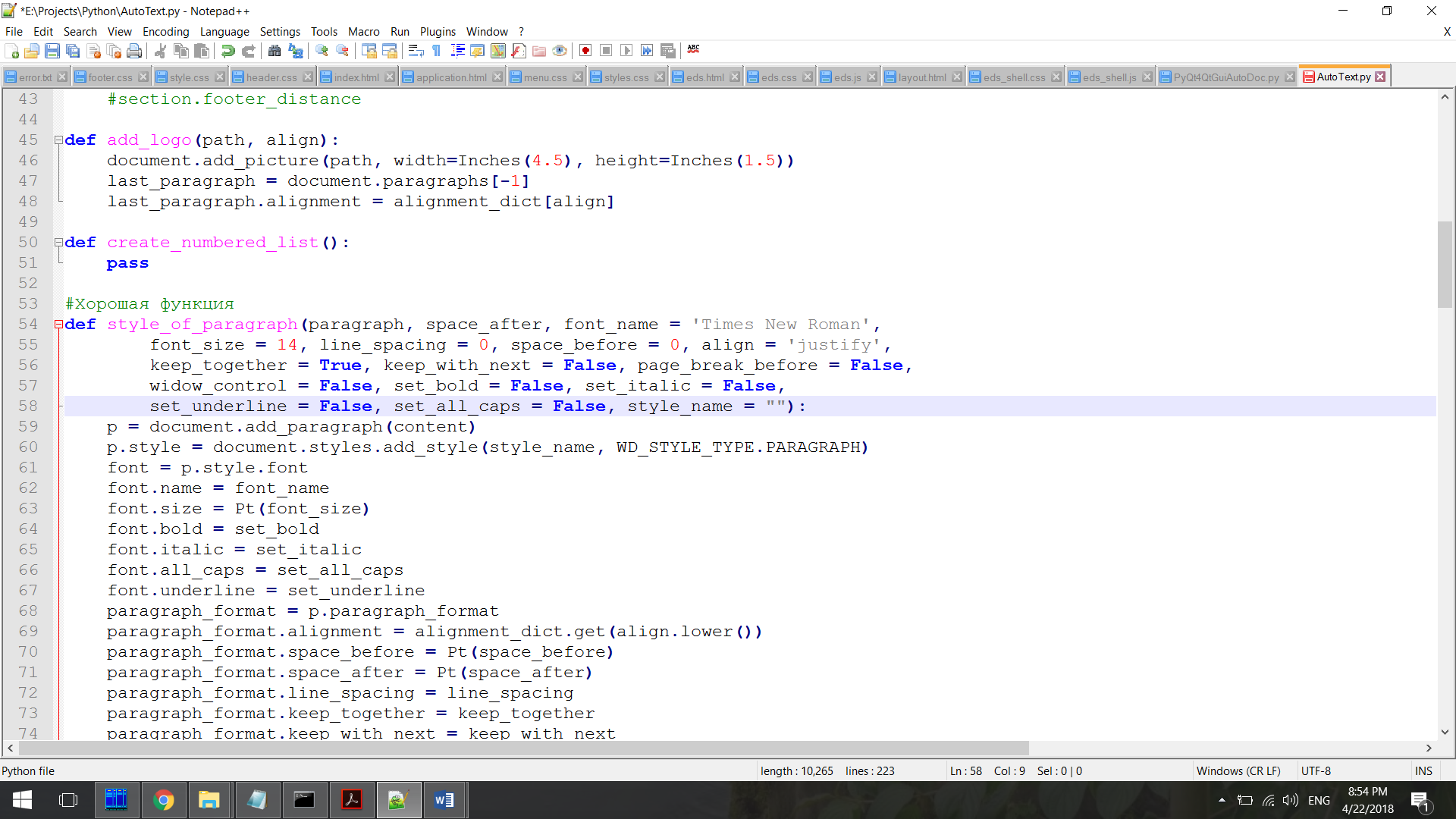


Рисунок 2. Параметры функции управления стилями

С помощью типа данных «Словари» для Python разработан парсер для автоматического формирования фамилий соавторов и информации:

- соответствующих электронных адресов,

- выбора соответствующих файлов с основным текстом статьи (неформатированным!),

- соответствующих списков использованных источников,

а также для автоматического формирования названий генерируемых файлов готовых статей, отформатированных по нужным требованиям – рис. 3-4.

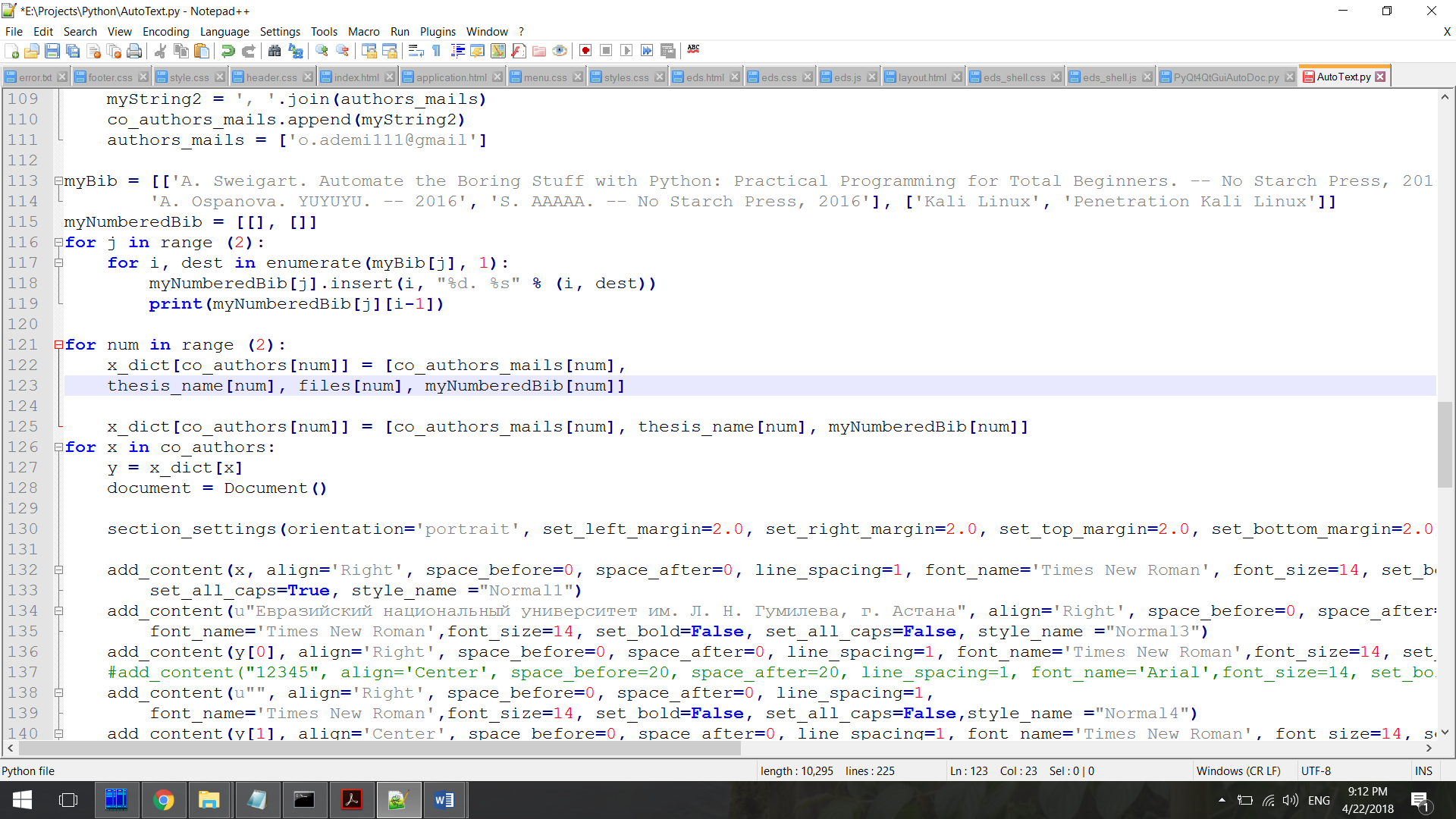


Рисунок 3. Автоматическое формирование файлов с соответствующим контентом

|  |
| --- |
| for x in co\_authors:  …  addTo = x + u"\_Конф\_Туран-Астана\_2018"  document.save(addTo+'.docx') |

Рисунок 4. Автоматическое формирование файлов с соответствующими контентом и названием

Список использованных источников

1. Th. House, Th. Road. ReportLab PDF Library. User Guide. – Wimbledon.: London, UK (<https://www.reportlab.com/docs/reportlab-userguide.pdf>).

2. R. Alsina. How to use rst2pdf – (<https://ralsina.me/static/manual.pdf>).

3. PyPDF2 Documentation – (<https://pythonhosted.org/PyPDF2/>).

4. D. Holtwick. pisa 3.0.30. XHTML/HTML/CSS to PDF converter. – (<https://trac-hacks.org/raw-attachment/wiki/TracWikiPrintPlugin/pisa-en.pdf>).

5. Openpyxl. A Python library to read/write Excel 2010 xlsx/xlsm files – (<https://openpyxl.readthedocs.io/en/stable/)/>.

6. Reading Excel with Python (xlrd) – (<https://blogs.harvard.edu/rprasad/2014/06/16/reading-excel-with-python-xlrd/>).

7. Creating Microsoft Excel Spreadsheets with Python and xlwt – (<https://www.blog.pythonlibrary.org/2014/03/24/creating-microsoft-excel-spreadsheets-with-python-and-xlwt/>).

8. xlutils Documentation – (<https://media.readthedocs.org/pdf/xlutils/stable/xlutils.pdf>).

9. R. van Hattem. pyExcelerator Documentation. – (<https://media.readthedocs.org/pdf/pyexcelerator/latest/pyexcelerator.pdf>).

10. Interacting with OpenOffice.org/LibreOffice using UNO – (<https://pypi.org/project/unotools/>).

11. S. Canny. Python-docx Documentation. – (<https://media.readthedocs.org/pdf/python-docx/latest/python-docx.pdf>).

12. Scrapy Tutorial – (<https://doc.scrapy.org/en/latest/intro/tutorial.html>).

13. M. Foord. HOWTO Fetch Internet Resources Using urllib2 – (https://docs.python.org/2/howto/urllib2.html); HOWTO Fetch Internet Resources Using The urllib Package – (<https://docs.python.org/3.8/howto/urllib2.html>).

14. Browsing in Python with Mechanize – (<http://www.pythonforbeginners.com/mechanize/browsing-in-python-with-mechanize/>).

15. Beautiful Soup Documentation – (<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>).

16. A. Sweigart. Automate the Boring Stuff with Python: Practical Programming for Total Beginners. – No Starch Press, 2016.

17. Changing Paragraph formatting in python-docx – j<https://stackoverflow.com/questions/48150222/changing-paragraph-formatting-in-python-docx?utm_medium=organic&utm_source=google_rich_qa&utm_campaign>= google\_rich\_qa.