МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО «ДонНТУ»

Факультет Интеллектуальных систем и программирования

Кафедра «Программной инженерии» им. Л. П. Фельдмана

Лабораторная работа №4

по курсу: «Профессиональная практика программной инженерии»

по теме: «Создание самодокументирующегося кода»

Выполнил:

ст. гр. ПИ-21б

Васильченко О. Е.

Проверил:

асс. каф. ПИ

Филипишин Д. А.

Донецк – 2025

**Цель работы:** научиться добавлять в программный код специальным образом оформление докблок-комментарии, для последующей автоматической генерации API reference, а также познакомиться с форматом оформления документации DocBook.

**Задание к лабораторной работе:**

Внедрить в ранее разработанный программный код докблоки, соответствующие выбранному языку программирования.

Разработать DocBook и проверить корректность созданного DocBook файла с помощью средств редактора или одного из онлайн валидаторов.

Автоматически сформировать справочное руководство для программистов и выслать вместе с отчётом. В руководстве должны присутствовать все модули, описанные в лабораторной работе №1.

**Ход работы**

Листинг кода с докблоками:  
math\_calculations.py

def sum\_range(values):  
 *"""  
 Вычисляет сумму значений в списке.  
  
 Args:  
 values (list): Список чисел.  
  
 Returns:  
 float: Сумма значений.  
 """* return sum(values)  
  
  
def average\_range(values):  
 *"""  
 Вычисляет среднее значение в списке.  
  
 Args:  
 values (list): Список чисел.  
  
 Returns:  
 float: Среднее значение.  
 """* return sum(values) / len(values)

table\_creation.py

class Table:  
 *"""  
 Класс для создания и управления таблицами.  
  
 Attributes:  
 cells (dict): Словарь для хранения значений ячеек таблицы.  
  
 Methods:  
 set\_cell(address, value): Устанавливает значение в указанную ячейку.  
 get\_cell(address): Возвращает значение из указанной ячейки.  
 clear\_table(): Очищает все данные в таблице.  
 """* def \_\_init\_\_(self):  
 *"""Инициализирует пустую таблицу."""* self.cells = {}  
  
 def set\_cell(self, address, value):  
 *"""  
 Устанавливает значение в указанную ячейку.  
  
 Args:  
 address (str): Адрес ячейки (например, "A1").  
 value (any): Значение для записи в ячейку.  
 """* self.cells[address] = value  
  
 def get\_cell(self, address):  
 *"""  
 Возвращает значение из указанной ячейки.  
  
 Args:  
 address (str): Адрес ячейки (например, "A1").  
  
 Returns:  
 any: Значение ячейки или None, если ячейка не существует.  
 """* return self.cells.get(address, None)  
  
 def clear\_table(self):  
 *"""  
 Очищает все данные в таблице.  
  
 Clears the entire table by resetting the cells dictionary.  
 """* self.cells.clear()

chart\_builder.py

import matplotlib.pyplot as plt  
  
def build\_bar\_chart(data):  
 *"""  
 Строит столбчатую диаграмму на основе данных.  
  
 Args:  
 data (dict): Словарь, где ключи — категории, значения — числа.  
 """* categories = list(data.keys())  
 values = list(data.values())  
 plt.bar(categories, values)  
 plt.show()

Сформированная xml-документация:

<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML V4.5//EN"  
"http://www.oasis-open.org/docbook/xml/4.5/docbookx.dtd">  
<book>  
 <title>Справочное руководство для программистов</title>  
  
 <chapter>  
 <title>Модуль table\_creation</title>  
 <section>  
 <title>Класс Table</title>  
 <para>  
 Класс для создания и управления таблицами.  
 </para>  
 <method>  
 <title>set\_cell(address, value)</title>  
 <para>  
 Устанавливает значение в указанную ячейку.  
 </para>  
 <parameter>address (str): Адрес ячейки.</parameter>  
 <parameter>value (any): Значение для записи.</parameter>  
 </method>  
 <method>  
 <title>get\_cell(address)</title>  
 <para>  
 Возвращает значение из указанной ячейки.  
 </para>  
 <parameter>address (str): Адрес ячейки.</parameter>  
 <returns>any: Значение ячейки или None.</returns>  
 </method>  
 </section>  
 </chapter>  
  
 <chapter>  
 <title>Модуль math\_calculations</title>  
 <section>  
 <title>Функция sum\_range(values)</title>  
 <para>  
 Вычисляет сумму значений в списке.  
 </para>  
 <parameter>values (list): Список чисел.</parameter>  
 <returns>float: Сумма значений.</returns>  
 </section>  
 </chapter>  
</book>

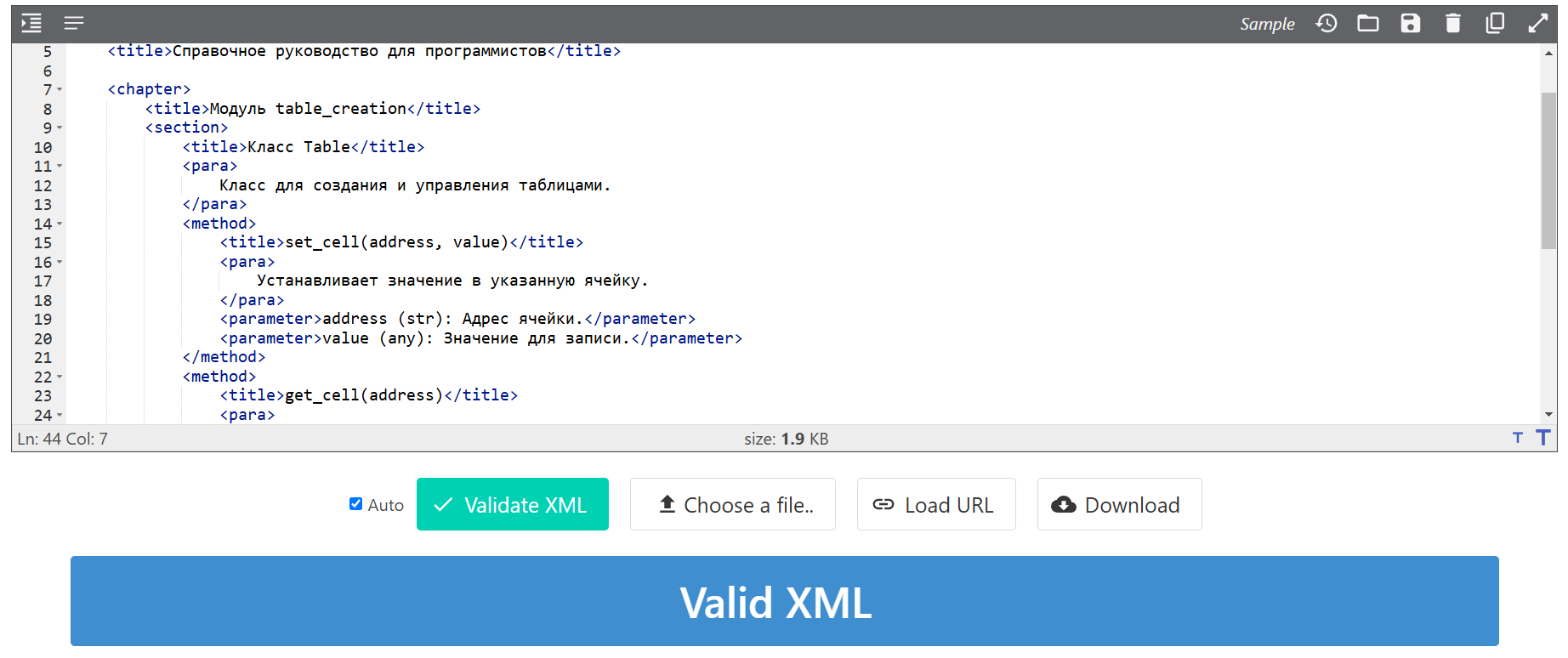


Рисунок 1 – применение валидатора для проверки xml-документации

Для валидации xml был использован валидатор: [https://codebeautify.org/xmlvalidator#](https://codebeautify.org/xmlvalidator%23)