

Разработка программы для редактирования текста с помощью библиотеки Kivy

Андрюенко Иван Сергеевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В данной статье исследуется процесс разработки приложения для редактирования текста с использованием библиотеки Kivy на языке программирования Python. Обсуждаются ключевые шаги создания графического интерфейса, включая использование библиотеки Kivy, взаимодействие с библиотекой Tkinter для диалогового окна выбора файлов, а также методы сохранения и загрузки текстовых данных. Описаны основные концепции разработки с использованием Kivy и умение создавать простые текстовые редакторы с удобным пользовательским интерфейсом.

Ключевые слова: Kivy, Python, графический интерфейс, текстовый редактор, Tkinter, разработка приложений.

Developing a text editing program using the Kivy library

Andrienko Ivan Sergeevich

Sholom-Aleichem Priamursky State University

Student

Abstract

This article explores the process of developing a text editing application using the Kivy library in the Python programming language. The key steps of creating a graphical interface are discussed, including using the Kivy library, interacting with the Tkinter library for the file selection dialog box, as well as methods for saving and loading text data. The basic concepts of development using Kivy and the ability to create simple text editors with a user-friendly interface are described.

Keywords: Kivy, Python, GUI, text editor, Tkinter, application development.

1 Введение

1.1 Актуальность

Разработка кроссплатформенных программы с использованием библиотеки Kivy важна в контексте современного программирования, где создание приложений с удобным графическим интерфейсом играет ключевую роль. Kivy предоставляет программистам Python эффективные средства для создания многоплатформенных приложений, ориентированных как на десктопные, так и на мобильные устройства. Разработка текстового редактора с применением данной библиотеки обеспечивает не только простоту в

реализации, но и возможность создания интуитивно понятного и эффективного пользовательского интерфейса. С учетом растущей популярности Python и необходимости создания универсальных приложений, использование Kivy для разработки текстового редактора становится актуальным и перспективным направлением в сфере разработки программного обеспечения.

1.2 Обзор исследований

В своей работе Е.А. Таравский, И.Ш. Сингатуллово рассмотрели подход к разработке с помощью фреймворка kivy, работающего как библиотека python. Знакомство с Kivy framework. Особенности Kivy. Чем хорош Kivy и причины выбора данного фреймворка [1]. В статье А.И. Федотова, Р.Г. Гильванов рассматриваются варианты разработки приложений, такие как Web-приложения, нативные и кроссплатформенные приложения. Рассматриваются достоинства и недостатки каждого варианта. Сравняется нативная и кроссплатформенная разработка. Подробнее рассматривается вариант кроссплатформенной разработки приложений, популярные фреймворки. В статье дается обзор языка программирования Python, приведены его достоинства и недостатки. Рассмотрена возможность кроссплатформенной разработки на языке программирования Python с помощью фреймворка Kivy [2]. В своей работе Е.В. Изукаев, Д.А. Шуклин рассмотрели основные возможности инструментов для разработки мобильных приложений с помощью языка Python [3]. В своей работе М.В. Болдырева рассмотрел подход к разработке с помощью фреймворка kivy, работающего как библиотека python [4]. В статье С.Н. Верзунов описаны программные средства, предназначенные для управления устройством определения основного и побочных параметров катушек индукционных магнитометров. Разработан метод, основанный на определении полного комплексного сопротивления индуктивного компонента магнитометра на различных частотах и подбор определяемых параметров исходя из наилучшего соответствия их наблюдаемым значениям с помощью генетического алгоритма. Программные средства, реализующие этот метод, разработаны на базе фреймворка Kivy и используют разработанную автором библиотеку-драйвер устройства сбора данных ЛКард E502 для языка Python [5].

1.3 Цель исследования

Цель исследования – разработать и представить процесс создания приложения для редактирования текста с использованием библиотеки Kivy на языке программирования Python.

2 Материалы и методы

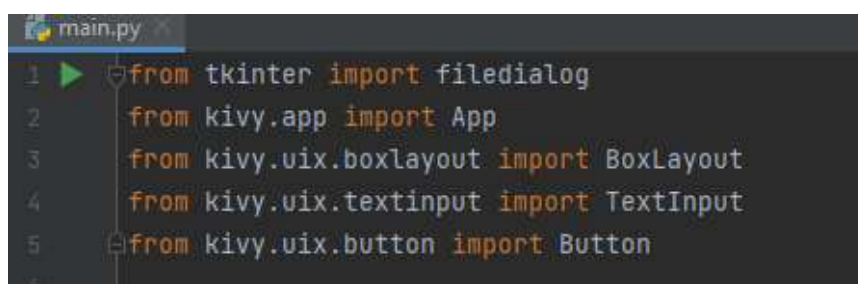
Разработка программы для редактирования текста выполнена на языке программирования Python в среде разработки PyCharm. Для построения графического интерфейса был использован набор виджетов Kivy, включая.

Для обеспечения функциональности выбора файлов и сохранения/загрузки текстовых данных использован модуль `filedialog` из библиотеки `Tkinter`.

3 Результаты и обсуждения

В библиотеке `Kivy` для построения пользовательского интерфейса можно написать код как в отдельном файле, так и непосредственно в коде `Python`. `Kivy` поддерживает декларативный язык разметки, называемый "Kv-языком", который может использоваться для определения структуры пользовательского интерфейса в отдельных файлах с расширением `".kv"`.

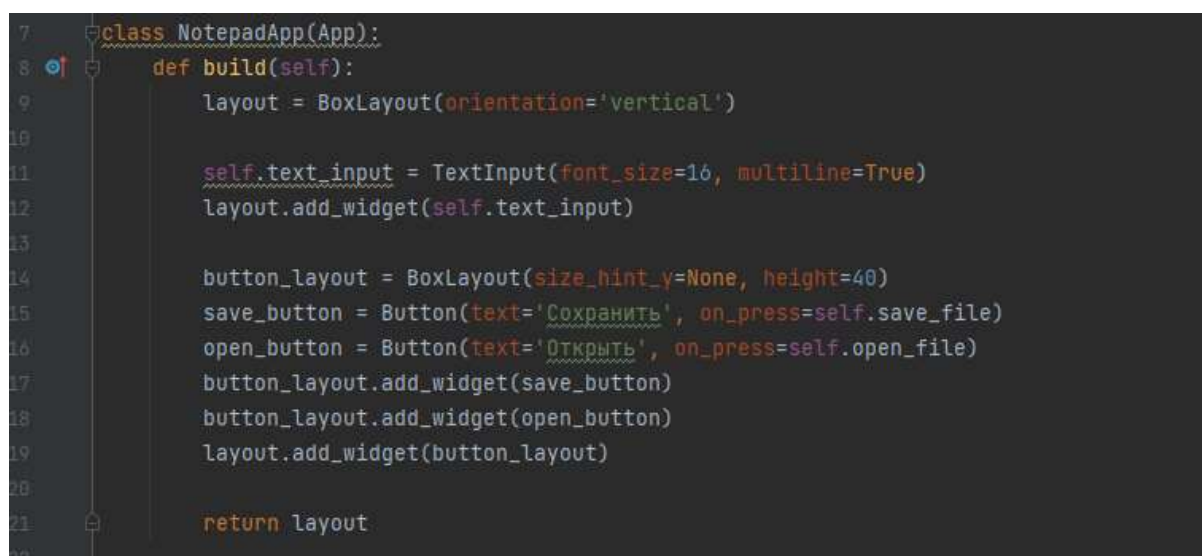
В данной работе будет рассмотрен метод написания пользовательского интерфейса в коде. Импортируем две библиотеки для создания графического приложения: `Tkinter` и `Kivy` (рис. 1).



```
main.py
1 from tkinter import filedialog
2 from kivy.app import App
3 from kivy.uix.boxlayout import BoxLayout
4 from kivy.uix.textinput import TextInput
5 from kivy.uix.button import Button
```

Рисунок 1 – Импорт необходимых библиотек

После импорта библиотек происходит определение класса `NotepadApp`, который является подклассом `App` из `Kivy`. Внутри этого класса определены методы для построения пользовательского интерфейса (`build`), сохранения текста в файл и загрузки текста из файла. Создается вертикальный контейнер `BoxLayout`, включающий виджет `TextInput` для ввода текста и две кнопки – сохранить и открыть, оформленные в горизонтальном контейнере `BoxLayout`. Каждая кнопка связана с методами `save_file` и `open_file`, которые выполняют сохранение и открытие файлов соответственно (рис. 2).



```
7 class NotepadApp(App):
8     def build(self):
9         layout = BoxLayout(orientation='vertical')
10
11         self.text_input = TextInput(font_size=16, multiline=True)
12         layout.add_widget(self.text_input)
13
14         button_layout = BoxLayout(size_hint_y=None, height=40)
15         save_button = Button(text='Сохранить', on_press=self.save_file)
16         open_button = Button(text='Открыть', on_press=self.open_file)
17         button_layout.add_widget(save_button)
18         button_layout.add_widget(open_button)
19         layout.add_widget(button_layout)
20
21         return layout
```

Рисунок 2 – Создание интерфейса

Пропишем метод `save_file`. Он предназначен для сохранения текста из виджета `TextInput` в файл. При вызове метода сначала извлекается текст из `TextInput`. Затем открывается диалоговое окно сохранения файла с использованием `filedialog.asksaveasfilename`, где пользователь может выбрать имя и местоположение файла. Если пользователь выбирает файл и нажимает "Сохранить", то введенный текст записывается в выбранный файл в формате `.txt`. Если файл не выбран, сохранение не выполняется (рис. 3).

```
29 def save_file(self, instance):
30     content = self.text_input.text
31
32     file_path = filedialog.asksaveasfilename(defaultextension=".txt", filetypes=[("Text files", "*.txt")])
33
34     if file_path:
35         with open(file_path, 'w') as file:
36             file.write(content)
```

Рисунок 3 – Метод `save_file`

Создадим метод `open_file` предназначенный для открытия текстового файла. При вызове метода открывается диалоговое окно выбора. Пользователь может выбрать файл с расширением `.txt`, и, если файл был выбран, его путь сохраняется в переменной `file_path`. Затем происходит чтение содержимого файла, и прочитанный текст устанавливается в виджет `TextInput`. Если файл не был выбран, открытие файла не выполняется. Завершающая часть кода создает экземпляр класса `NotepadApp` и запускает приложение с помощью метода `run` (рис. 4).

```
40
41 def open_file(self, instance):
42     file_path = filedialog.askopenfilename(filetypes=[("Text files", "*.txt")])
43
44     if file_path:
45         with open(file_path, 'r') as file:
46             content = file.read()
47             self.text_input.text = content
48
49 if __name__ == '__main__':
50     NotepadApp().run()
51
```

Рисунок 4 – Метод `open_file`

Тестируем приложение (рис. 5).



Рисунок 5 – Интерфейс приложения

При нажатии кнопки сохранить или открыть вызывается диалоговое окно, которое предоставляет пользователю возможность выбрать местоположение и имя файла для сохранения (рис. 6).

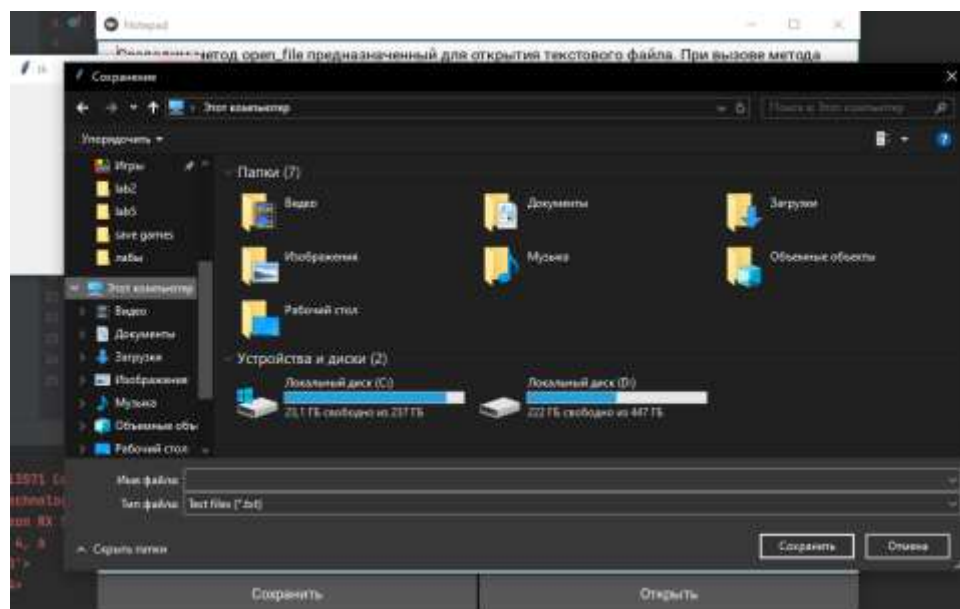


Рисунок 6 – Нажатие кнопки «Сохранить»

Выводы

В данной статье был рассмотрен процесс разработки программы для редактирования текста с применением библиотеки Kivy на языке программирования Python. Разработанное приложение предоставляет простой

текстовый редактор с графическим интерфейсом, позволяя пользователям вводить, сохранять и открывать текстовые данные. Использование библиотеки Kivy предоставляет возможность создания кроссплатформенных приложений, поддерживая как десктопные, так и мобильные платформы. Рассмотрены основные элементы Kivy и библиотека Tkinter. Эти возможности обогащают функционал приложения, делая его более универсальным и удобным для пользователей.

Библиографический список

1. Таравский Е.А., Сингатулло И.Ш. Использование kivy framework для создания мобильного приложения на языке программирования python // Концепция развития и эффективного использования научного потенциала общества. Сборник статей Международной научно-практической конференции. В 2-х частях. 2020. С. 99-101.
2. Федотова А.И., Гильванов Р.Г. Разработка кроссплатформенных приложений на языке python и фреймворке kivy Интеллектуальные технологии на транспорте. 2022. № 2 (30). С. 53-58.
3. Изукаев Е.В., Шуклин Д.А. Сравнительный анализ разработки мобильных приложений на Python. // Альманах научных работ молодых ученых Университета ИТМО. в 5 т. 2016. С. 229-230.
4. Болдырева М.В. Разработка кроссплатформенных приложений с помощью фреймворка kivy // Наука сегодня: факты, тенденции, прогнозы. Материалы международной научно-практической конференции. 2018. С. 11-12.
5. Верзунов С.Н. Разработка программной части устройства для измерения параметров индуктивных компонентов магнитометров. // Вестник Кыргызско-Российского Славянского университета. 2020. Т. 20. № 12. С. 136-144.