ГУАП

КАФЕДРА № 42

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| канд. техн. наук, доцент |  |  |  | Н. В. Богословская |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ № 9 |
| СОБЫТИЯ КЛАССОВ |
| по курсу: |
| ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | 4326 |  |  |  | Г. С. Томчук |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2024

СОДЕРЖАНИЕ

[1 Цель работы 3](#_Toc190257000)

[2 Задача 3](#_Toc190257001)

[3 Ключевые позиции 3](#_Toc190257002)

[4 Тестирование программы 5](#_Toc190257003)

[ВЫВОДЫ 6](#_Toc190257004)

1. Цель работы

Выполнение работы имело следующие цели: изучить механизмы работы событий в C# для реализации взаимодействия между объектами. Научиться добавлять события в классы, организовывать цепочку обработчиков и переопределять методы, вызывающие события, с использованием наследования.

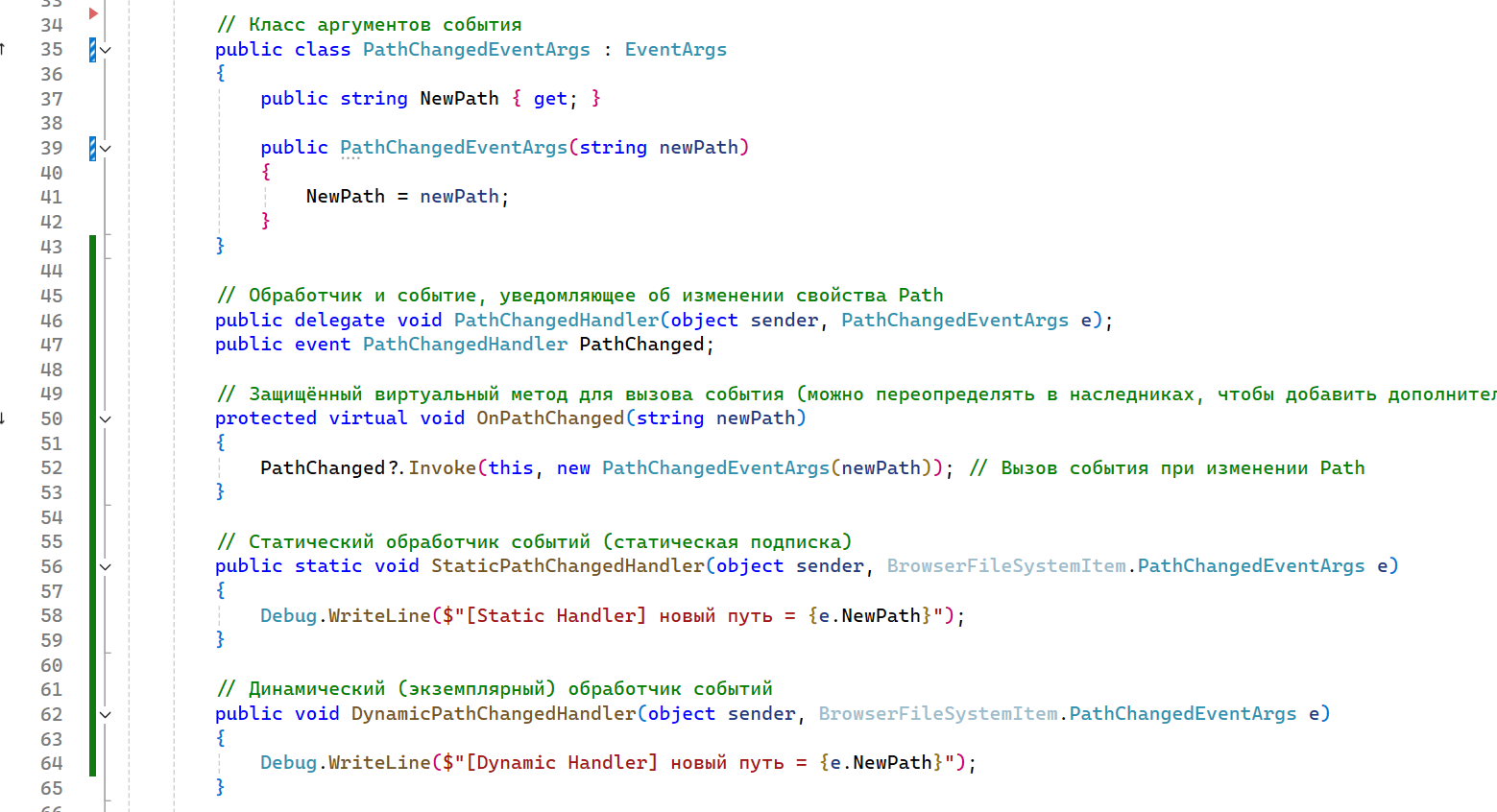
1. Задача

11. В приложении пользователь может создать объект класса Текстовый файл, используя классы Файл, Директория. Методы для работы с файлами должны полностью обеспечить пользователя возможностями создания, удаления, изменения, переименования файлов. Подзадачи лабораторной работы № 9 включают в себя:

* Реализовать события в собственных классах, используя стандартный шаблон событий (с делегатом EventHandler или обобщённым EventHandler<TEventArgs>).
* Продемонстрировать организацию цепочки обработчиков событий при их генерации.
* Исследовать возможности подписки на события статически и динамически, доказав возможность использования обоих подходов на практике.
* Использовать рекомендации по генерации событий базового класса в производных классах, протестировав их корректность.
* Подготовить программный код событий с подробными комментариями, демонстрирующий все возможности событий.
* Продемонстрировать примеры переопределения метода, вызывающего события, для расширения или модификации функциональности в производных классах.

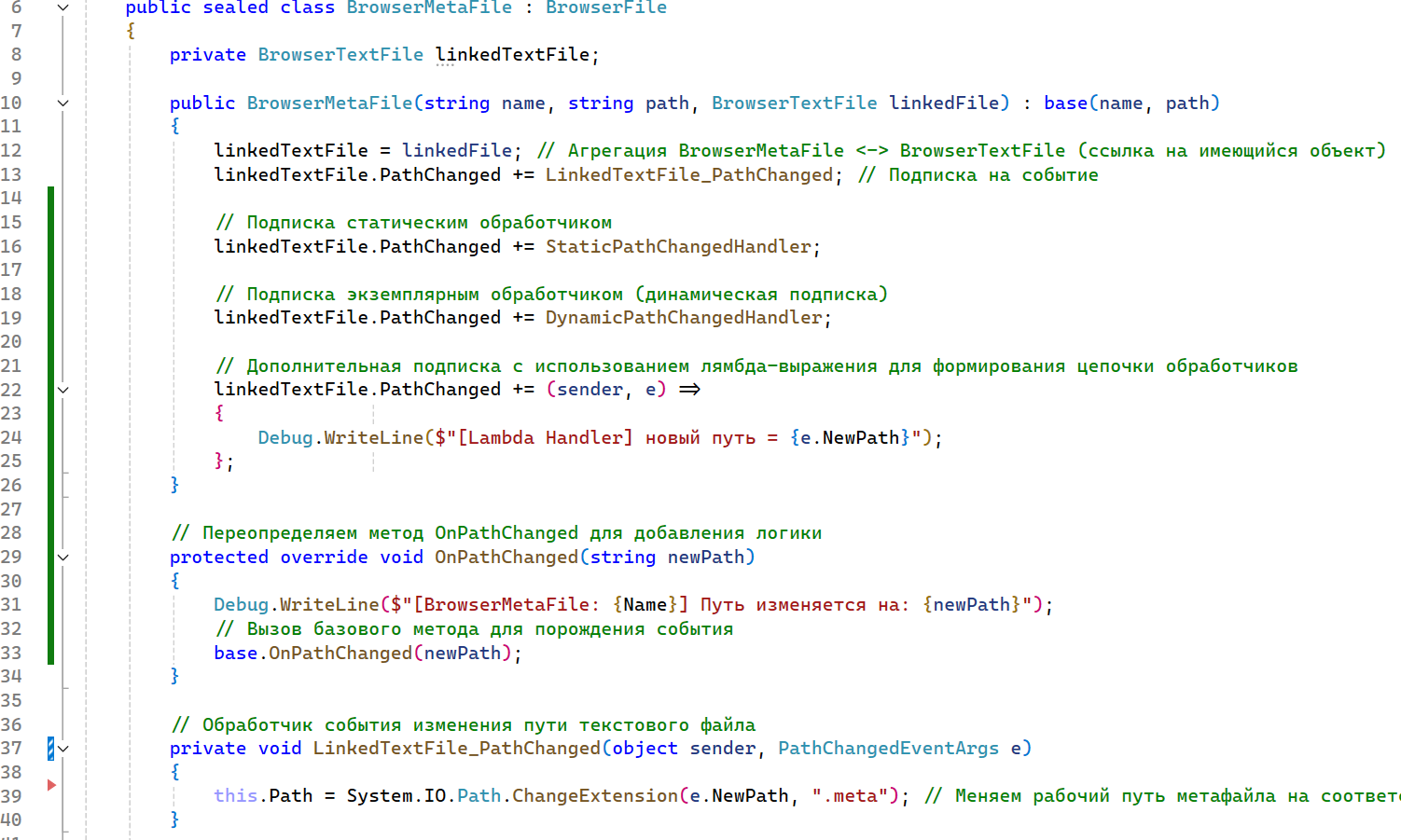
1. Ключевые позиции

На рис. 1 изображено определение события, обработчиков и метода вызова.



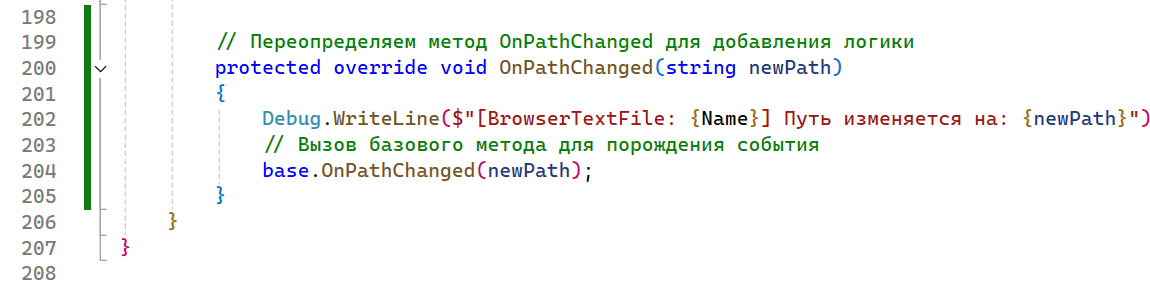
Рисунок

На рис. 2 показано переопределение метода, порождающего событие, в классе-наследнике, организация цепочки обработчиков событий. Также продемонстрирована возможность подписки на событие как статически, так и динамически.



Рисунок

На рис. 3 изображено переопределение метода, порождающего событие, в классе-наследнике.



Рисунок

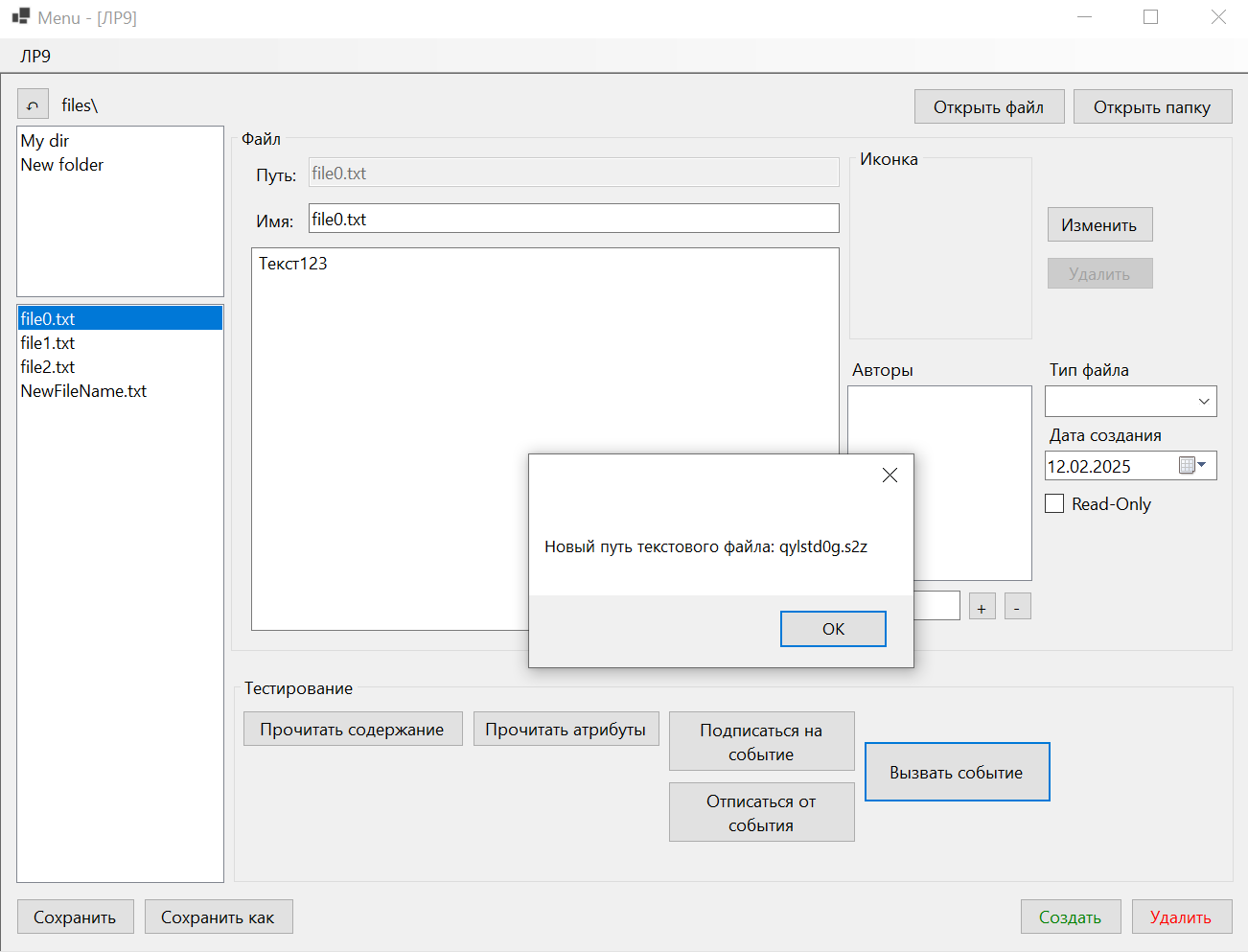
На рис. 4 показаны методы для тестирования работы события.



Рисунок

1. Тестирование программы

На рис. 5 представлены результаты тестирования программы.



Рисунок

ВЫВОДЫ

* Работа событий: События в C# являются удобным механизмом взаимодействия объектов, который позволяет подписчикам реагировать на определённые действия или изменения состояния.
* Цепочки обработчиков: Организация списка обработчиков позволяет вызывать несколько методов при возникновении одного события, что делает приложение более гибким.
* Подписка на события: Статическая подписка (компиляционно заданная) обеспечивает более строгую связь, тогда как динамическая (выполняемая во время выполнения программы) позволяет гибко изменять поведение приложения.
* События в наследовании: Использование базовых событий в производных классах расширяет возможности наследования, позволяя создавать более специализированные события.
* Практическая ценность: Изучение событий помогает создавать сложные системы с эффективным взаимодействием компонентов, что особенно важно в разработке интерфейсов пользователя и других приложений, основанных на событиях.
* Механизм переопределения: Возможность переопределять методы, вызывающие события, позволяет адаптировать поведение программы под задачи конкретного производного класса, сохраняя наследуемую структуру.