**Лабораторная работа №3**

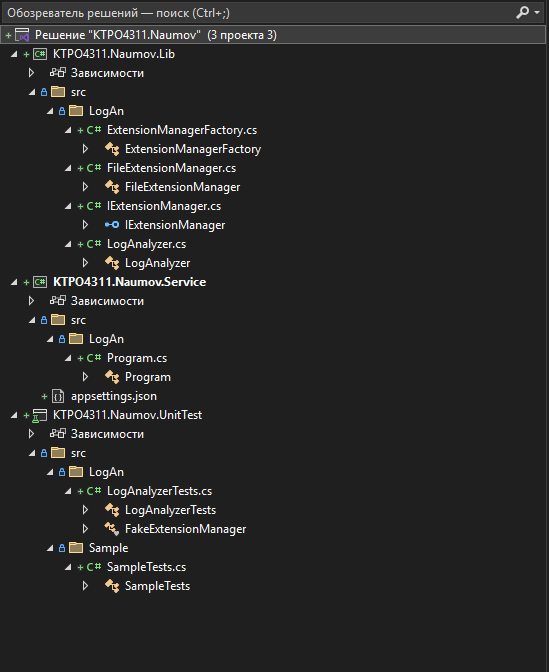
**Тестирование взаимодействия и подставные объекты**

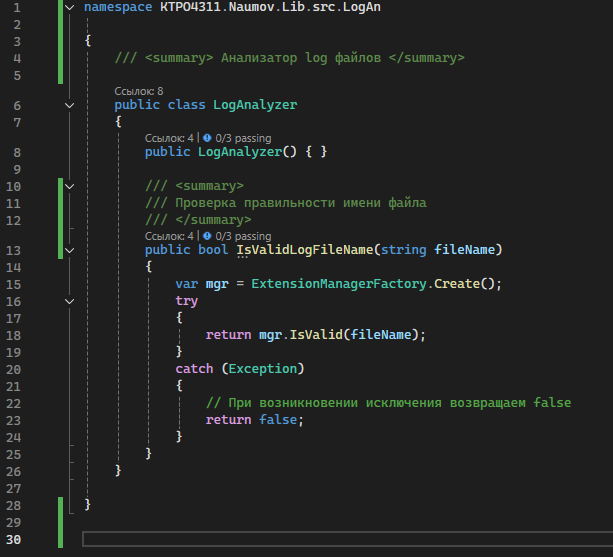
**Цель работы:**

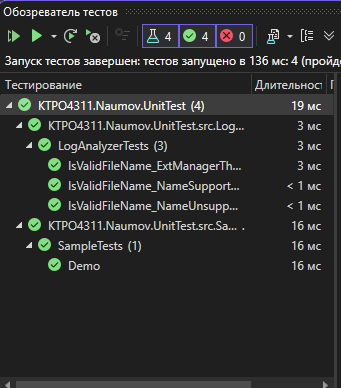
Приобретение практических навыков использования подставных объектов, для тестирования взаимодействия при автономном тестировании модулей, практика использования тестового каркаса NUnit, практика внедрения зависимости.

**Порядок выполнения работы**

**Подготовка проекта**

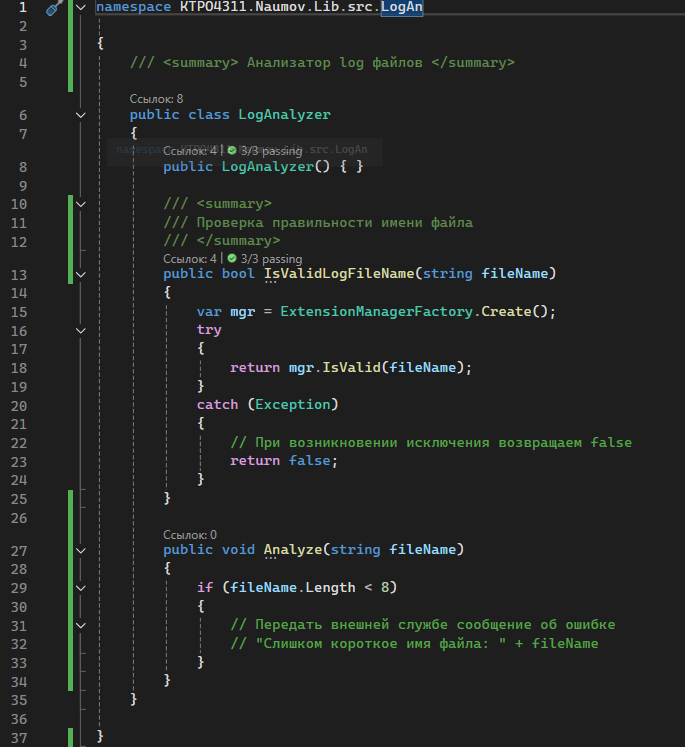




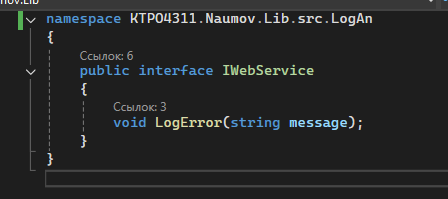


**Использование подставного объекта**

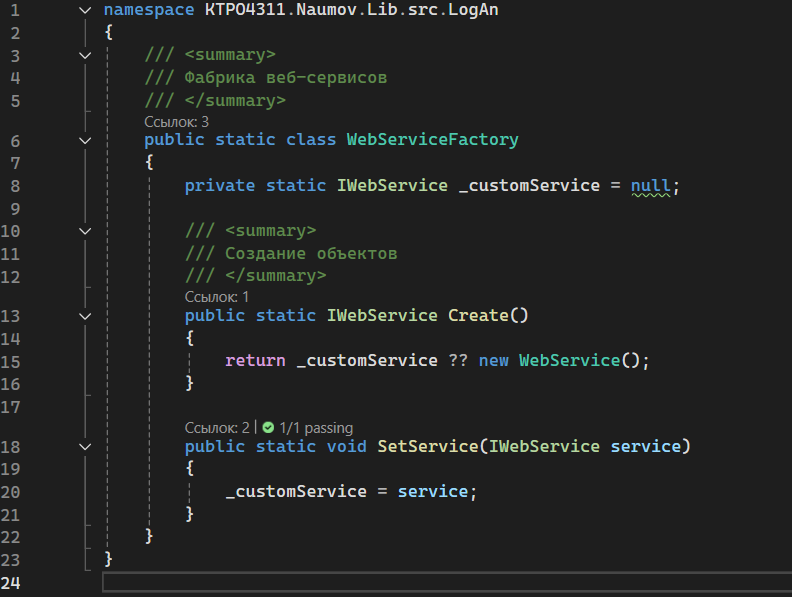
1. Добавляем в класс LogAnalyzer метод Analyze

****

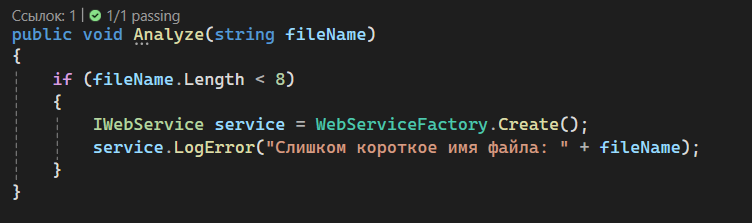
1. Создаём интерфейс IWebService.

****

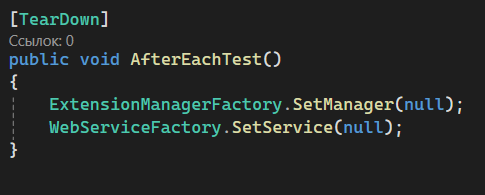
1. Создаём класс WebServiceFactory.



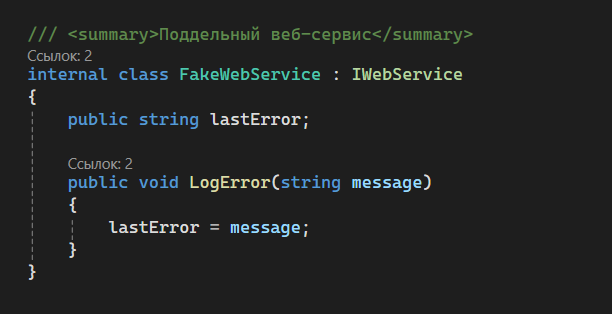
1. Обновляем метод Analyze у класса LogAnalyzer.



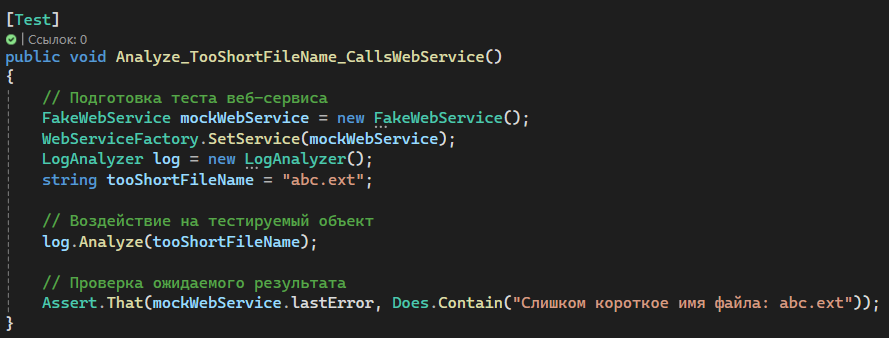
1. Обновляем метод AfterEachTest в классе LogAnalyzerTests.



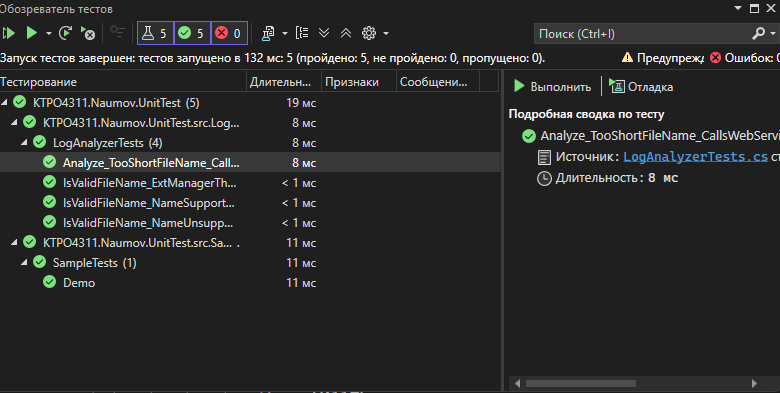
1. В LogAnalyzerTests добавляем класс поддельного веб-сервиса.

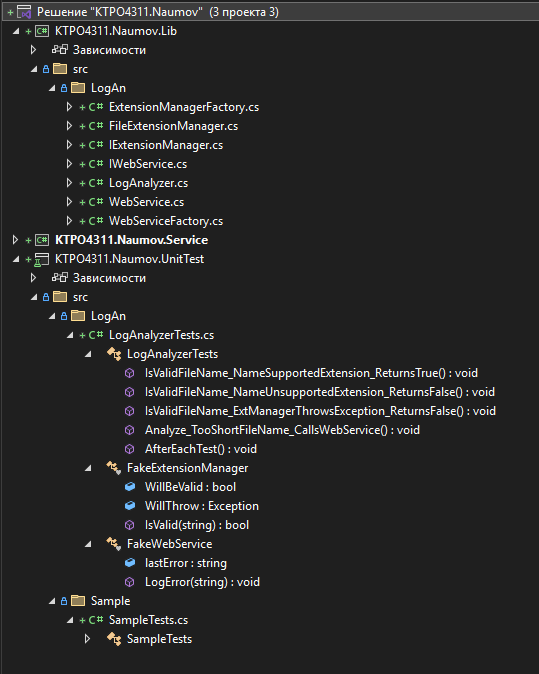


1. Создаём тестовый метод для сценария “Если имя слишком короткое, вызываем службу”.

****

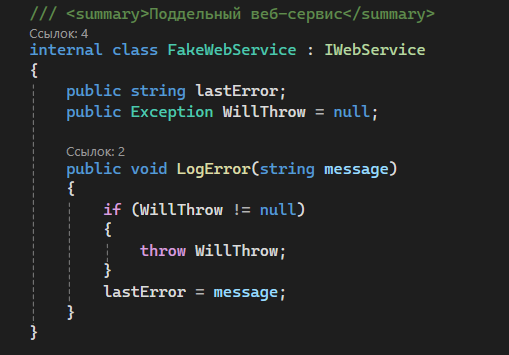
1. Выполним тесты



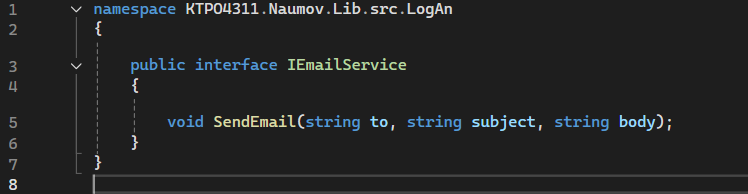


**Несколько поддельных объектов. Совместное использование поставки и заглушки**

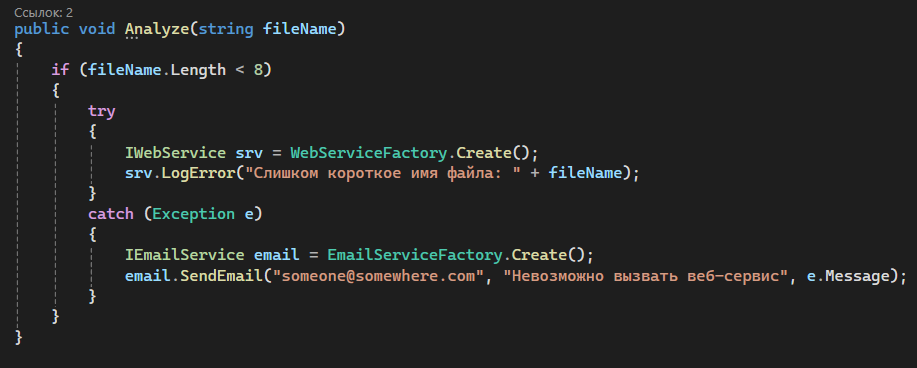
1. Обновляем FakeWebService для того, чтобы можно было управлять результатом вызова. Для этого добавляем поле WillThrow и проверку в метод LogError.



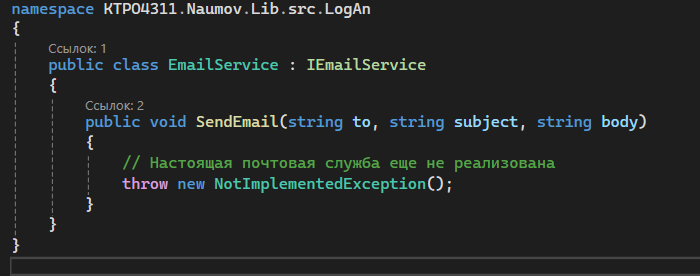
1. Выделяем интерфейс IEmailService.



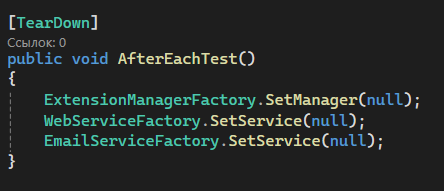
1. Заменяем псевдокод на обращение к службе



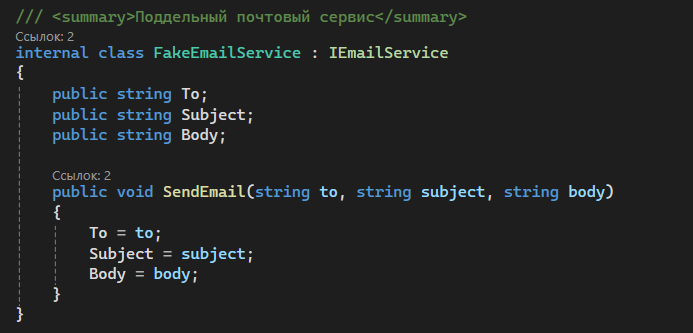
Создаём класс EmailService



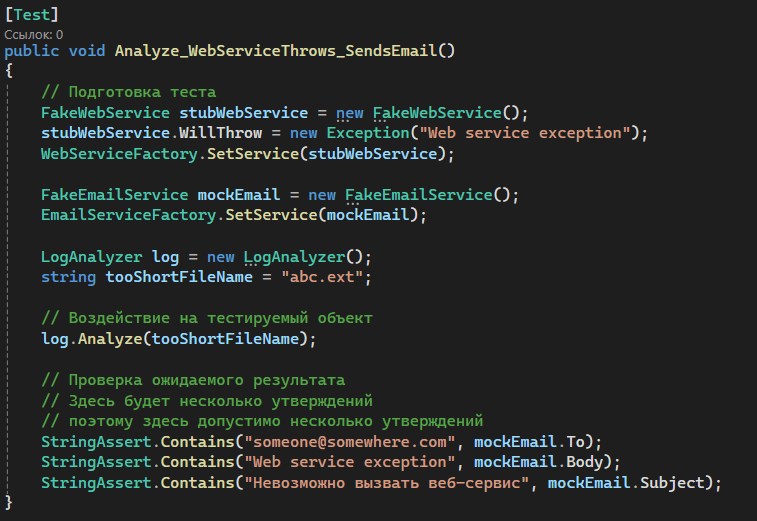
Обновляем метод AfterEachTest в LogAnalyzerTests для восстановления исходного состояния фабрики.



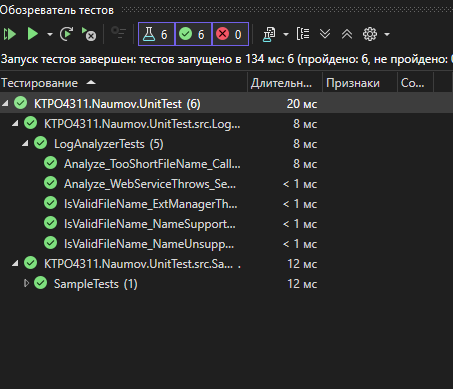
1. Создаём поддельный почтовый сервис



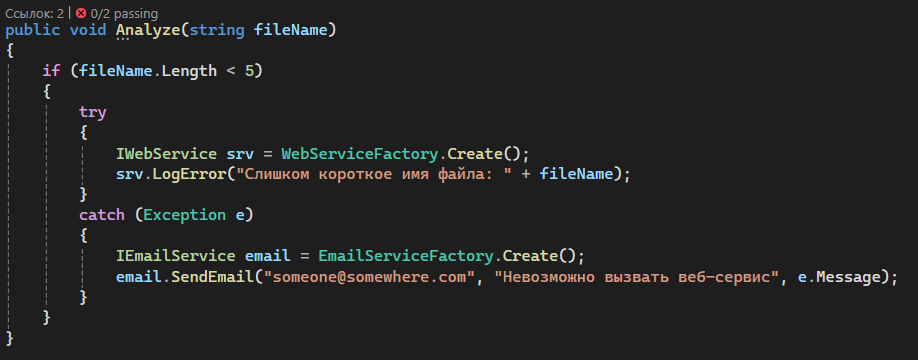
1. Создаём тестовый метод для сценария “Если веб-служба вызывает исключение, отправляем почтовое сообщение”.

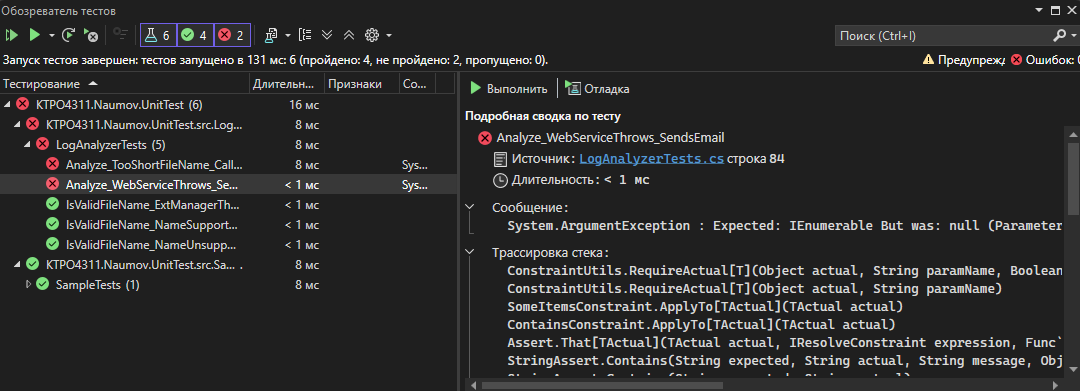


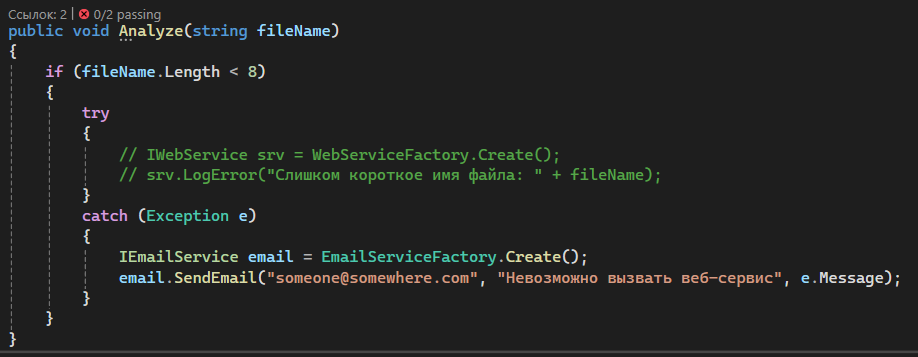
1. Выполним тесты.

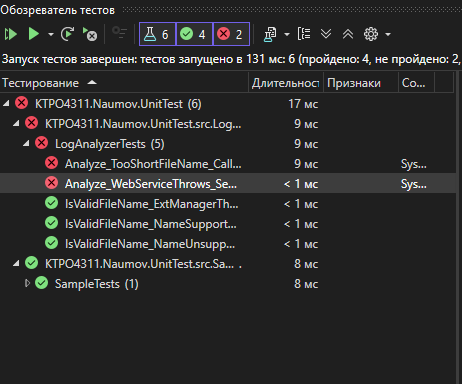


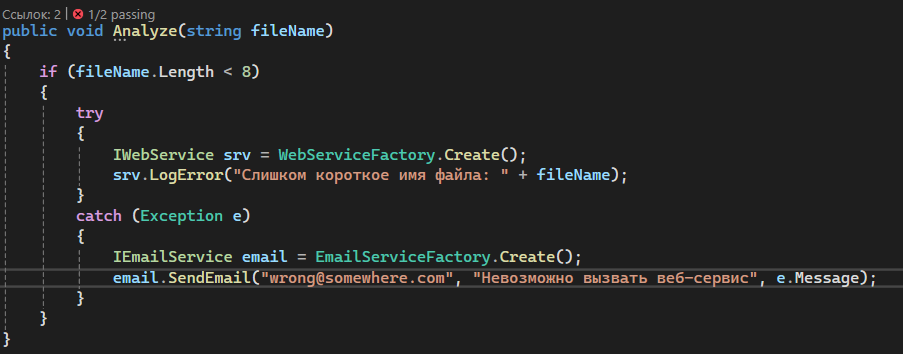
1. Проверим правильность тестового метода, внеся в тестируемые методы дефекты, ошибки, которые должен обнаружить тест.

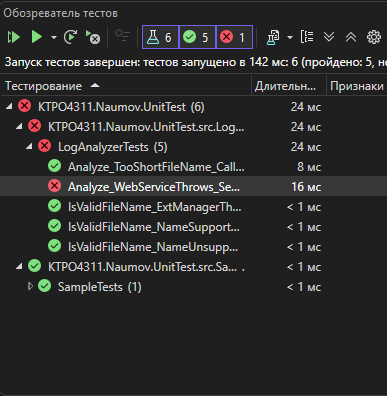












**Выводы**

В ходе лабораторной работы были освоены навыки использования подставных объектов для тестирования взаимодействия между компонентами программы. Были реализованы объекты для имитации ошибок и проверки реакции системы на эти ошибки.

Отдельно было рассмотрено применение сразу нескольких подставных объектов в одном тестовом сценарии. Такой подход позволил проверить ситуацию, когда сбой одного компонента вызывает автоматическую реакцию другого.