Нуранов Айдар 931920

1. Постановка задачи

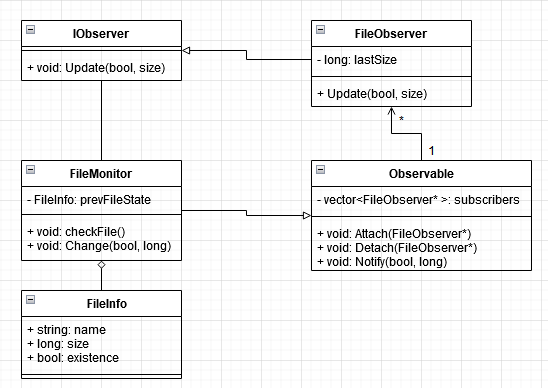
На основе паттерна Наблюдателя, написать программу для слежения за состоянием выбранного файла.   
  
Ограничимся  двумя характеристиками:  
1. Существует файл или нет;  
2. Каков размер файла.  
  
Программа будет выводить на консоль уведомление о произошедших изменениях в файле.  
Существует несколько ситуаций для наблюдаемого файла  
1. Файл существует , файл не  пустой - на экран выводится факт существования файла и его  размер.  
2. Файл существует, файл был изменен - на экран выводится факт существования файла, сообщение о том что файл был изменен и его размер.    
3. Файл не существует - на экран выводится информация о том что файл не существует.

2. Предлагаемое решение

**Наблюдатель** — это поведенческий паттерн проектирования, который создаёт механизм подписки, позволяющий одним объектам следить и реагировать на события, происходящие в других объектах.

Паттерн Наблюдатель предлагает хранить внутри объекта издателя список ссылок на объекты подписчиков, причём издатель не должен вести список подписки самостоятельно. Он предоставит методы, с помощью которых подписчики могли бы добавлять или убирать себя из списка. Когда в издателе будет происходить важное событие, он будет проходиться по списку подписчиков и оповещать их об этом, вызывая определённый метод объектов-подписчиков.

Все подписчики должны следовать общему интерфейсу и иметь единый метод оповещения.

Uml диаграмма: 

Для начала создаем шаблонный класс наблюдателя, от которого будет реализован конкретный класс FileObserver. Создадим так же класс, хранящий информацию об файле, хранящий состояние файла такие как размер и существование. Наконец реализация класса FileMonitor который будет оповещать всех наблюдателей об изменении файла.

3. Коды программ

fileinfo.h

fileinfo.cpp

observable.h

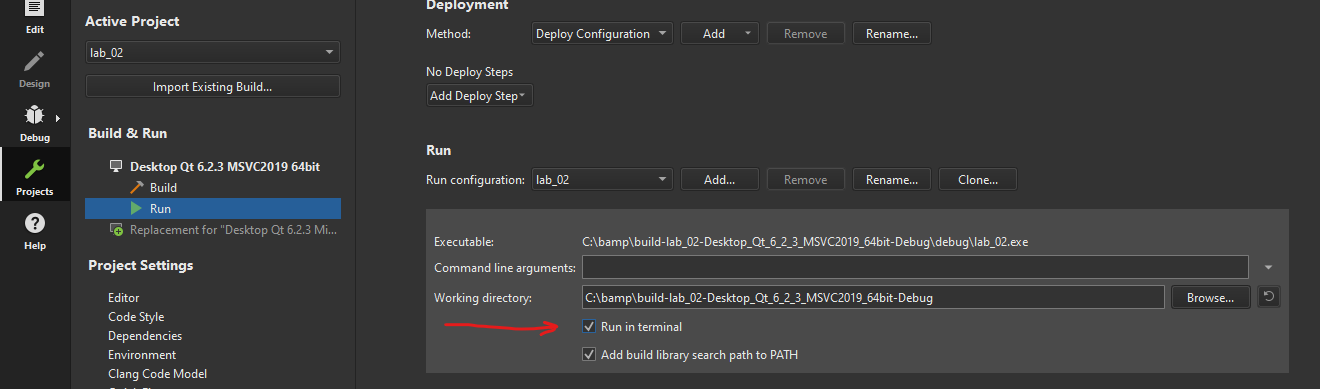
observable.cpp

observer.h

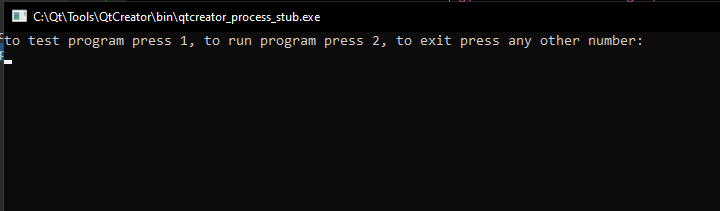
observer.cpp

4. Инструкция пользователя

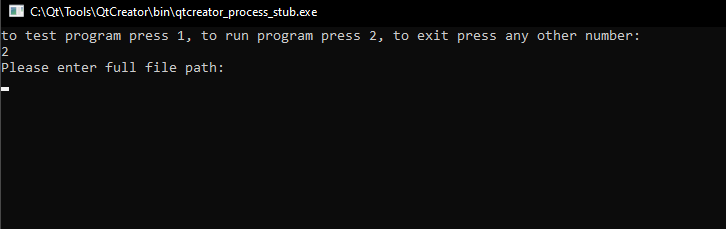
Для корректного запуска приложения нужно убедиться, что запуск в программе будет происходить в терминале, это необходимо для корректного чтения данных.

Для этого нужно во вкладке Projects > Build & Run > Run должна быть отмечена галочка внутри чекбокса “Run in terminal”. Теперь проект будет запускаться во внешнем терминале

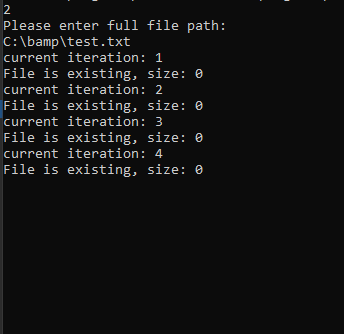
При запуске программы на встречает сообщение с работой данной программы что бы запустить тесты программы необходимо ввести 1, для проверки работы Наблюдателя в бесконечном цикле необходимо ввести 2



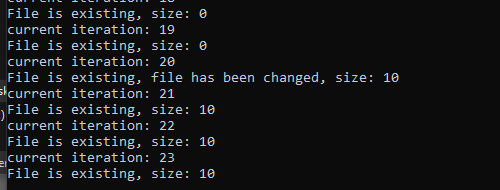
Далее нас попросят ввести полный путь до файла



После ввода пути до файла программа начнет наблюдать за этим файлом

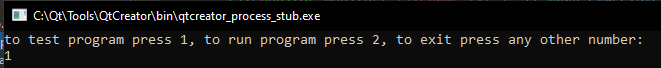


Теперь мы можем изменять этот файл



5. Тестирование

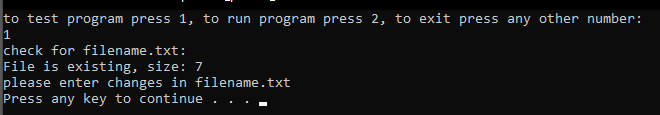
Для входа в режим тестирования после запуска программы вводим 1



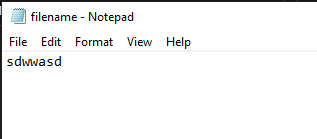
Первый тест на стандартное изменение файла

Программа попросит изменить содержимое файла filename.txt

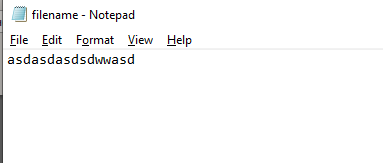
После изменения файла в консоль будет выведено что файл был изменен



До:

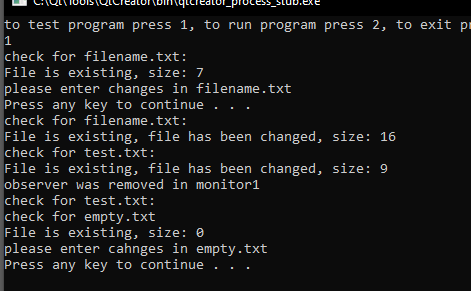


После:

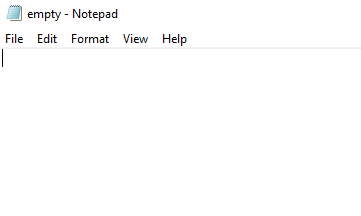


Второй тест на подписки разных файлов и удаление подписки

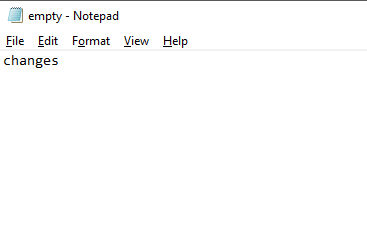
Третий тест изначальный фалй пустой, программа запросит внести изменения, выведется что файл был изменен и его размер



До:

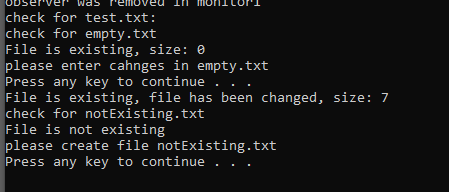


После:



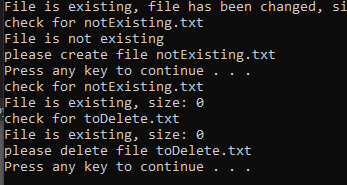
Четвертый тест. Первоначальный файл не существует, программа запросит добавить его

Выведется что файл не существует  
После создания файла в консоль выведется сообщение что файл существует

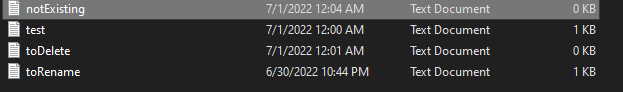


Пятый тест. Изначально файл существует, необходимо уалить его

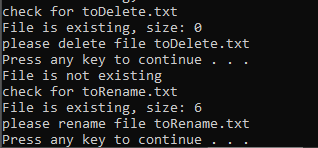
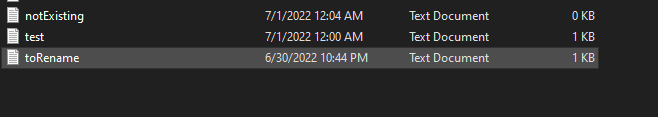
Сначала выодится на консоль что файл существует и его размер  
после удаления выведется на консоль что файл отсутствует



До удаления



После удаления



Последний тест на переименование файла, программа запросит переименовать файл

Сначала на консоль выводится что файл существует после переименования выводится что файл не существует

