МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДНР

ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра ПИ им. Л.П. Фельдмана

Лабораторная работа №2

Тема: Работа с ветками

Курс: Профессиональная практика программной инженерии

Выполнил

Сырых А.С.

Проверил

Ищенко А. П.

Донецк 2022

Задание к лабораторной работе

Разработать архитектуру в репозитории для разработки выбранного по заданию из 1 лабораторной работы проекта. Реализовать каждый из модулей в отдельных ветках, сделать не менее пяти фиксаций касательно разработки программного кода и после влить в ветку master. Зафиксировать лог ревизии для каждой из веток. Удалить все созданные ветки и предоставить вывод команды git log для trunk после вливания всех веток.

Порядок выполнения работы

1. Создать ветку для первого модуля программы.

2. Последовательно переключить рабочую копию на созданную ветку.

3. Сделать не менее пяти фиксаций, касающихся добавления файлов и редактирования кода.

4. Проверить лог ревизии git log --pretty=format:\"%h %ad | %s%d [%an]\" --graph —date=short

5. Переключить рабочую копию на master. git checkout master

6. Повторить пункт 1 для оставшихся модулей.

7. Влить все ветки git merge

8. Получить конфликт слияния.

9. Разрешить конфликт и закоммитить результирующий код.

10.Посмотреть лог ревизии git log --pretty=format:\"%h %ad | %s%d [%an]\" --graph —date=short

11.Удалить все ветки, кроме master git branch –D

Архитектура проекта

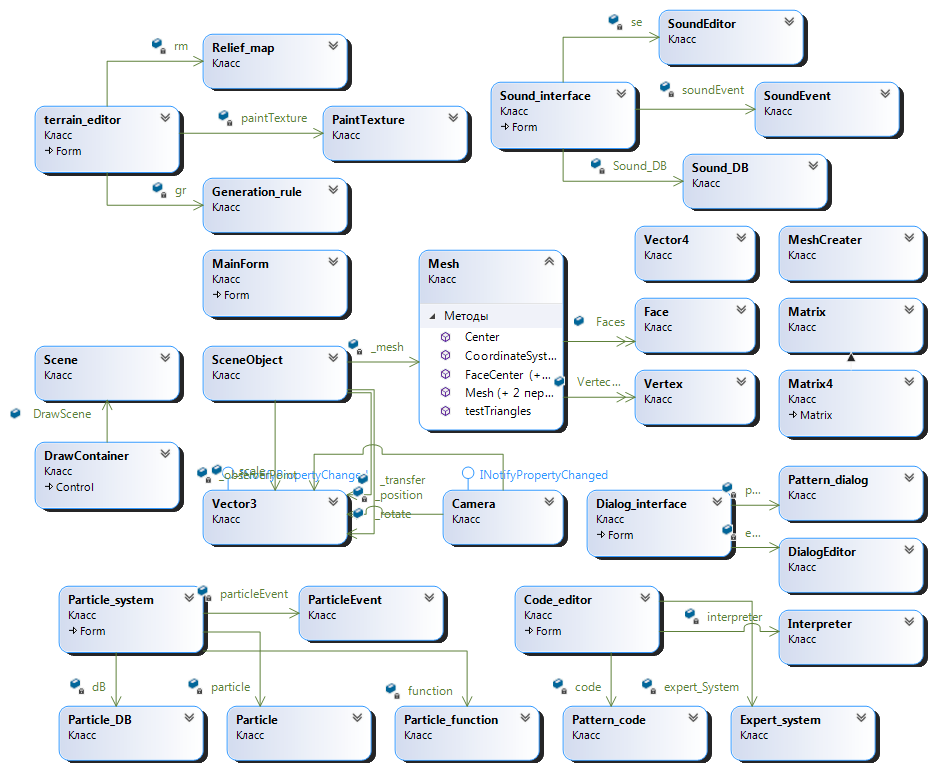


Рисунок 1 – Архитектура проекта

Всего нужно создать 7 веток для каждого из модулей, описанных в 1 лабораторной работе.

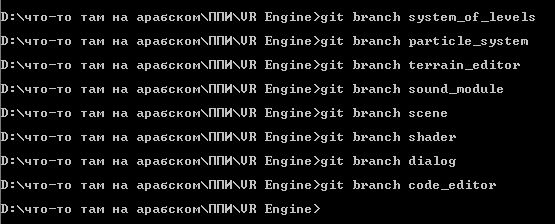


Рисунок 2 – Создание веток

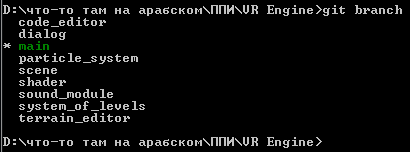


Рисунок 3 – Отображение результата создания веток

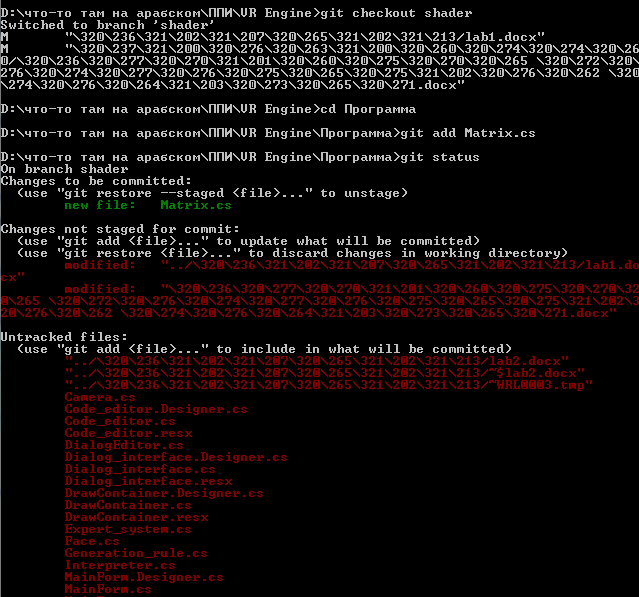


Рисунок 4 – Переход на ветку shader и добавление класса Matrix

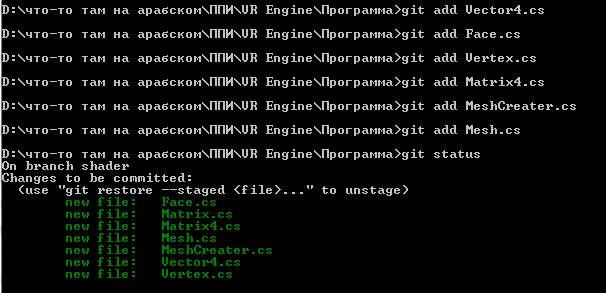


Рисунок 5 – Аналогично добавляются другие классы в ветку shader

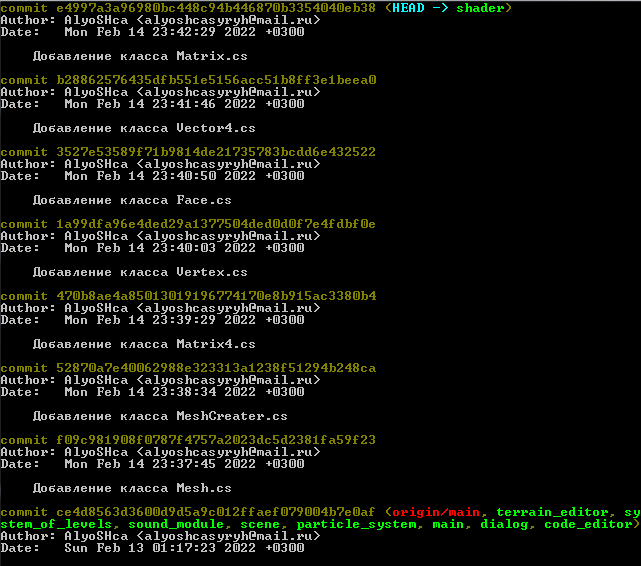


Рисунок 6 – Git log для ветки shader

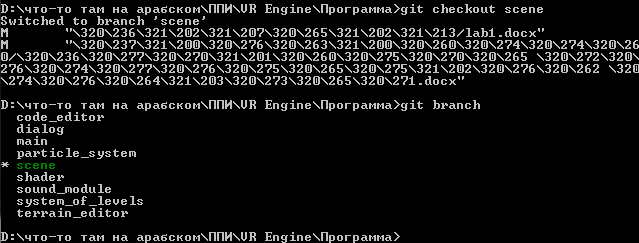


Рисунок 7 – Переход на ветку scene

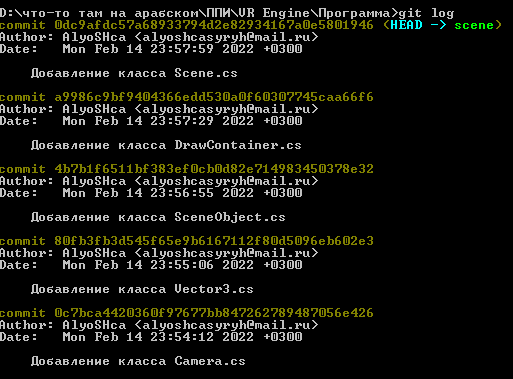


Рисунок 8 – Git log ветки Scene

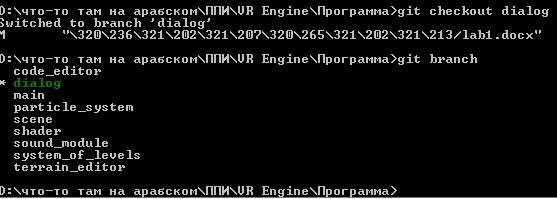


Рисунок 9 – Переход на ветку dialog

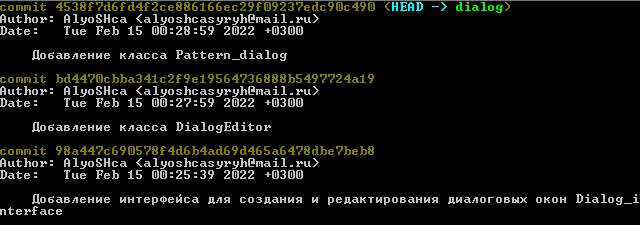


Рисунок 10 – Git log для ветки dialog

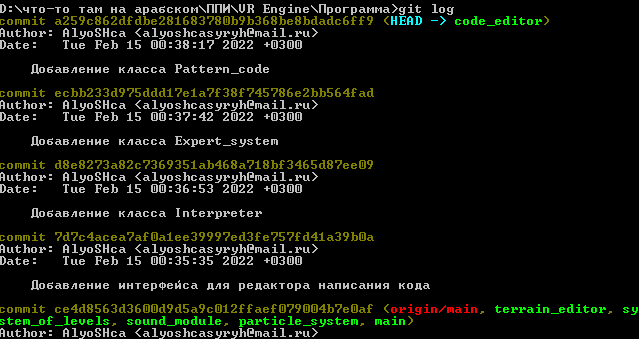


Рисунок 11 – Git log для ветки code\_editor

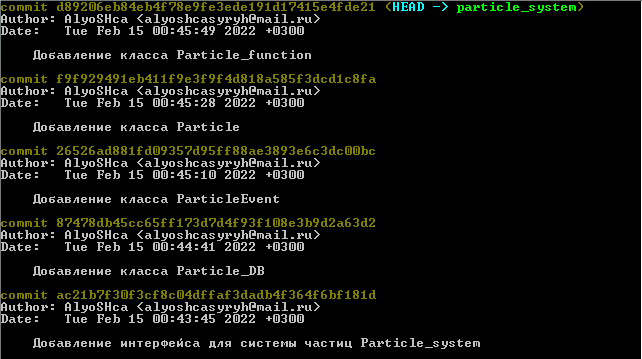


Рисунок 12 – Git log для ветки particle\_system

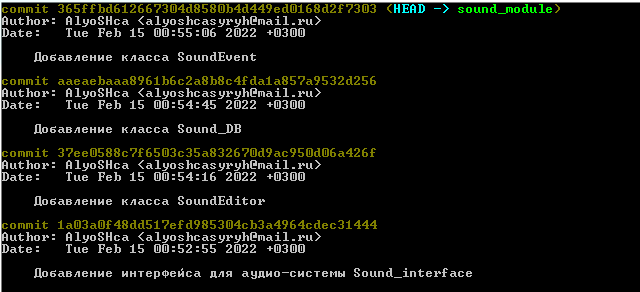


Рисунок 13 – Git log для sound\_module

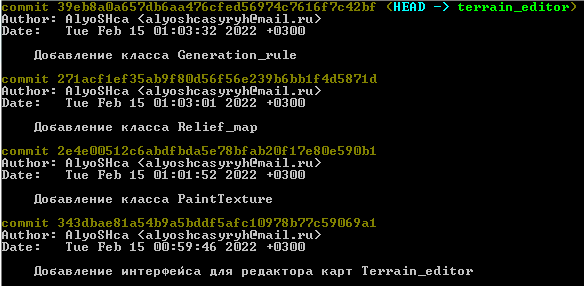


Рисунок 14 – Git log для Terrain\_editor

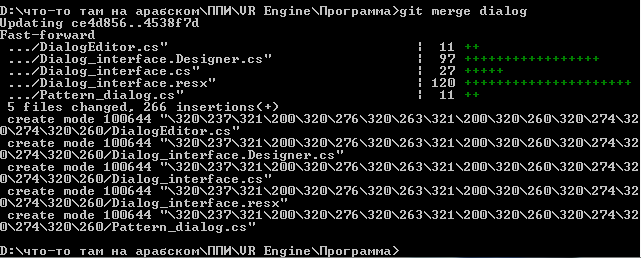


Рисунок 15 – Слияние главной ветки с dialog

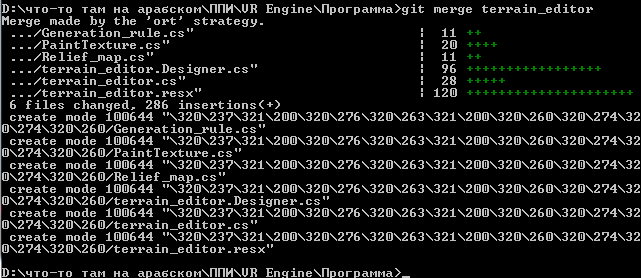


Рисунок 16 – Слияние главной ветки c terrain\_editor

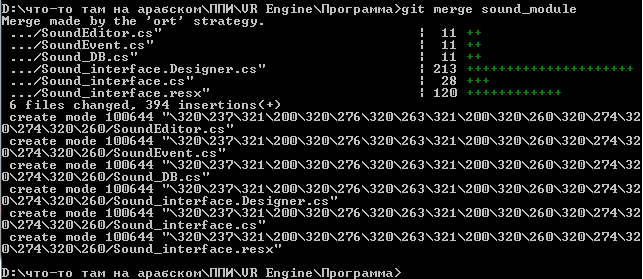


Рисунок 17 – Слияние главной ветки с sound\_module

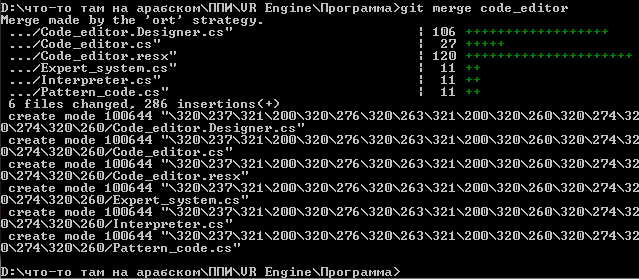


Рисунок 18 – Слияние главной ветки с code\_editor

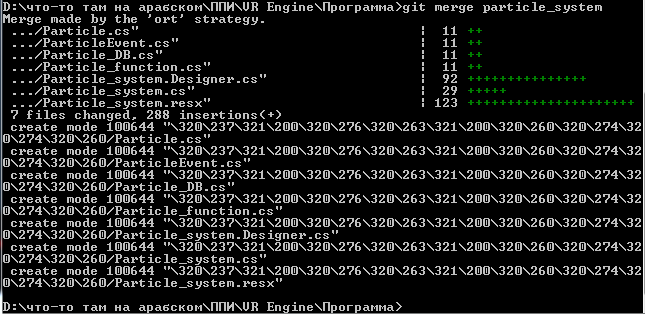


Рисунок 19 – Слияние главной ветки с particle\_system

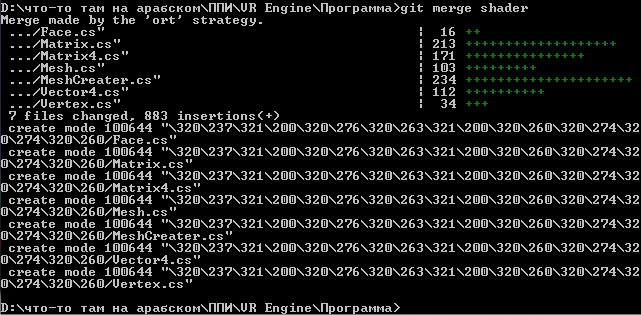


Рисунок 20 – Слияние главной ветки с shader

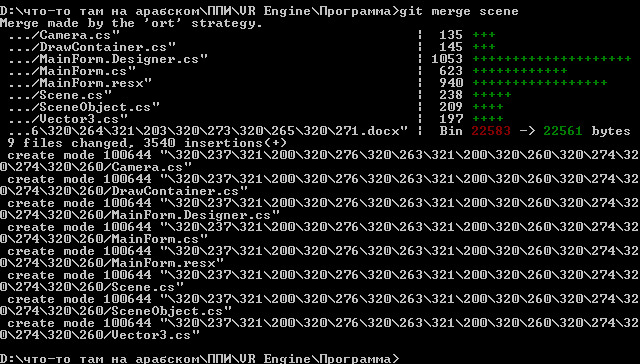


Рисунок 21 – Слияние главной ветки с scene

А теперь попытаемся получить конфликт слияния. Редактируем Описание компонентов модулей, добавляем в дочернюю ветку scene и пытаемся слить ее с главной. Обязательно делаем коммиты в ветке main и scene по данному файлу.

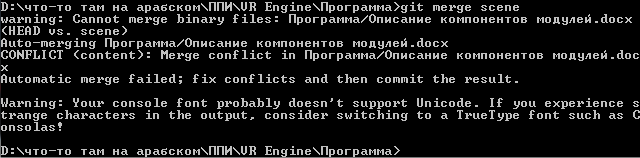


Рисунок 22 – Конфликт слияния главной ветки с scene

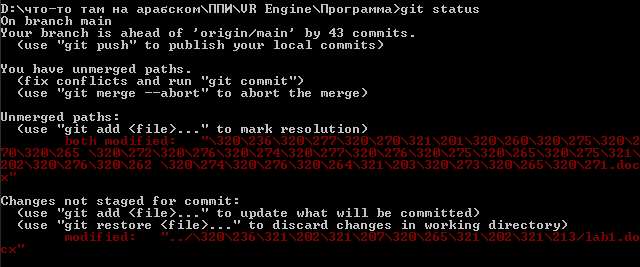


Рисунок 23 – При выводе статуса отображаются две версии файла

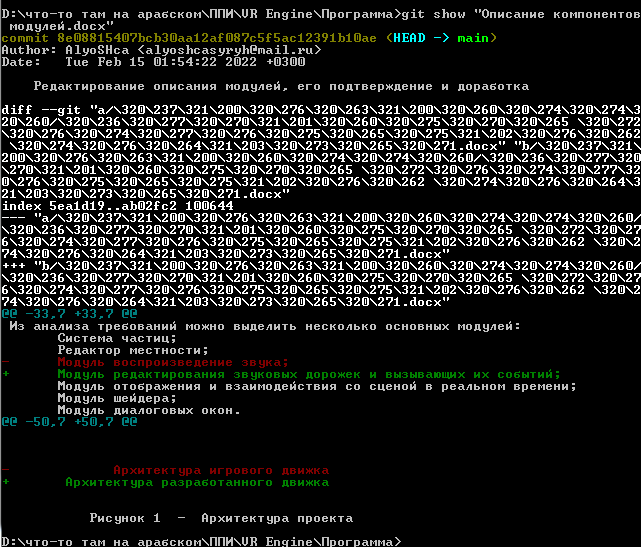


Рисунок 24 – Просмотр и решение конфликтных строк, добавление окончательного варианта

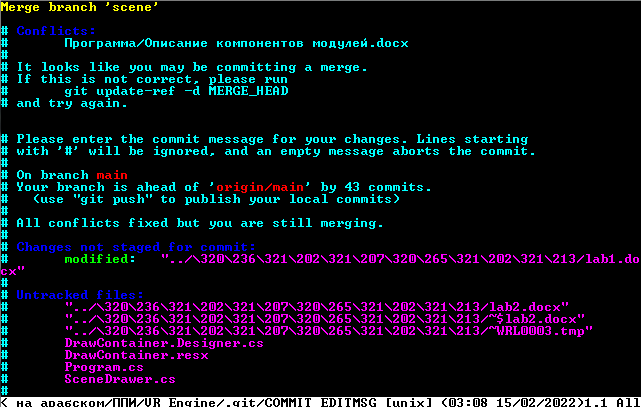


Рисунок 25 – Просмотр автоматического коммита



Рисунок 26 – Получение сообщения о конечном слиянии веток main и scene

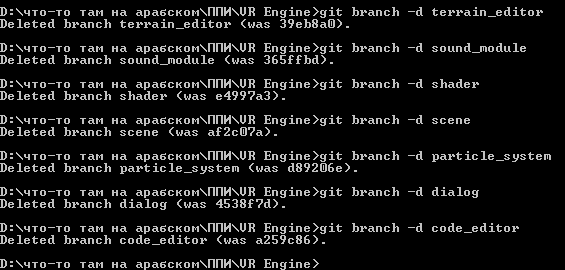


Рисунок 27 – Удаление веток

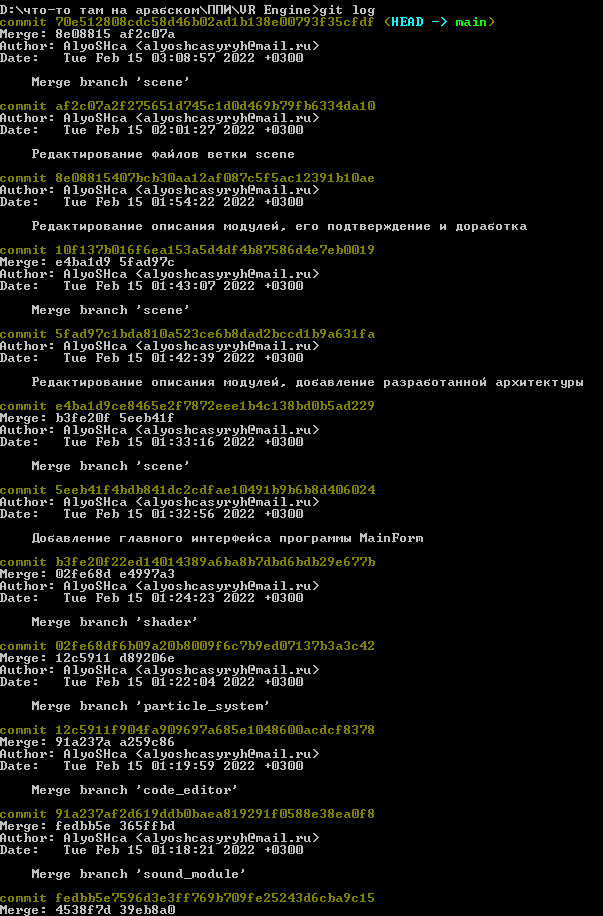


Рисунок 28 – Последние коммиты