**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Математическая Кибернетика и Информационные технологии»

Лабораторная работа по ИТП

«Основы работы с системой контроля версий»

Выполнил: Студент группы

БВТ2203

Попов Александр

Москва

2023

**Цель работы:** изучение базовых функций и подходов к командной разработке в системе контроля версий Git.

**Задание:** определить проект (состоящий из нескольких файлов), который будет добавлен под версионный контроль, и произвести ряд операций в СКВ, включающих:

1. Создание репозитория или клонирование существующего
2. Внесение ряда изменений в файл(ы) и выполнение коммитов (кол-во не меньше 2)
3. Откат изменений до одного из предыдущих коммитов двумя разными способами
4. Создание, переименование и удаление ветки
5. Слияние ветки с веткой мастер с решением конфликта содержания файлов
6. Настройка игнорирования git’ом файлов определенного расширения с использованием glob-шаблонов
7. Вывод лога коммитов в измененном пользовательском формате
8. Настройка удалённого репозитория: получение и отправка изменений в репозиторий
9. Организация совместной работы над проектом: клонирование чужого репозитория, выполнение коммитов, отправка изменений

**Ход работы**



Рис. 1. Инициализируем систему Git.

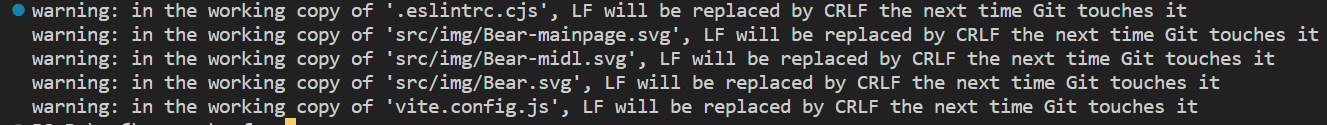


Рис. 2. Git add.

Команда git add добавляет в индекс файлы.

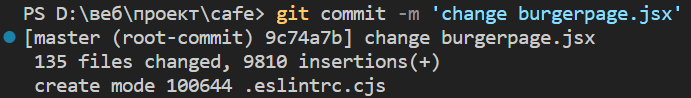


Рис. 3 Коммитим изменения.

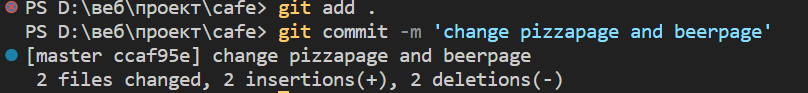


Рис. 4. Коммитим еще раз.

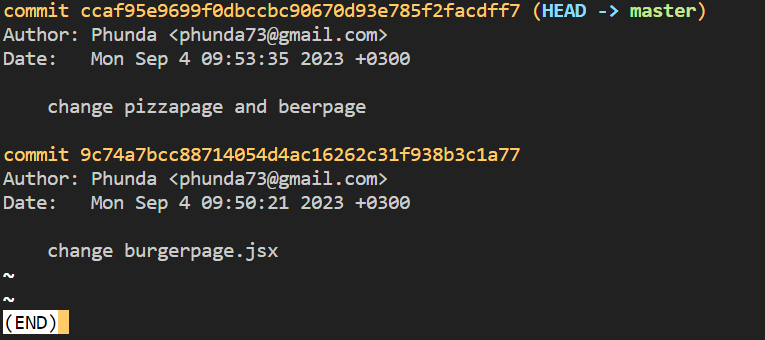


Рис. 5. Выводим логи наших коммитов

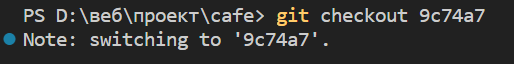


Рис. 6. Переключаемся на другой коммит

Достаточно команды git checkout, чтобы просто переключиться на другой коммит.

Если мы хотите продолжить работу с другого коммита, нам понадобится новая ветка. Можно переключиться и создать ее одной командой:

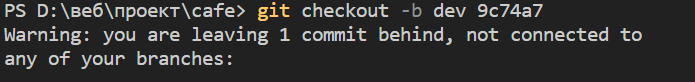


Рис. 7. Создание новой ветки на определенном коммите.



Рис. 8. Команда rebase.

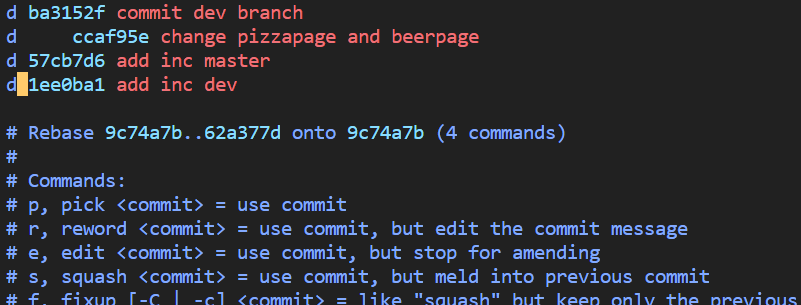


Рис. 9. Удаляем все предыдущие коммиты.

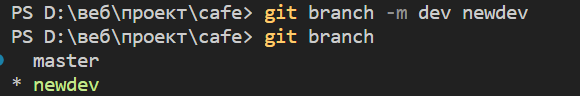


Рис. 10. Переименование ветки.



Рис. 11. Удаление ветки

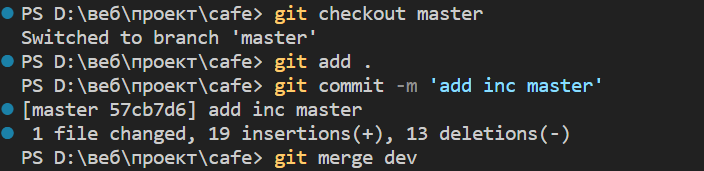


Рис. 12. Merge ветки dev

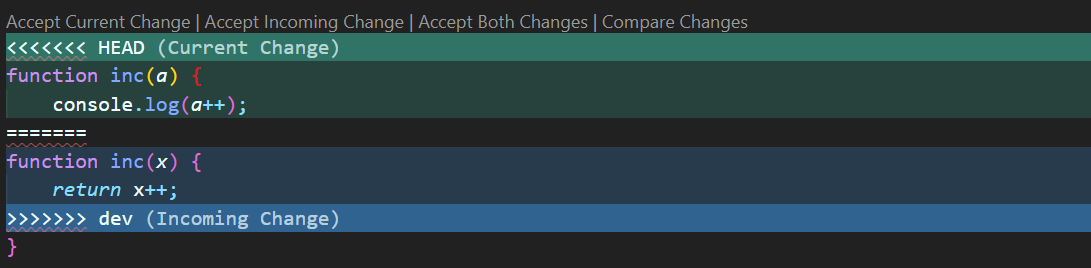


Рис. 13. Решение конфликтов.

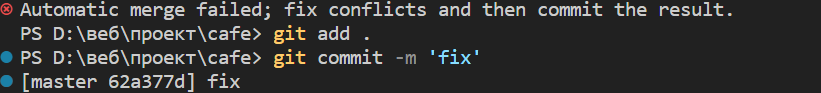


Рис. 14. Коммит merge

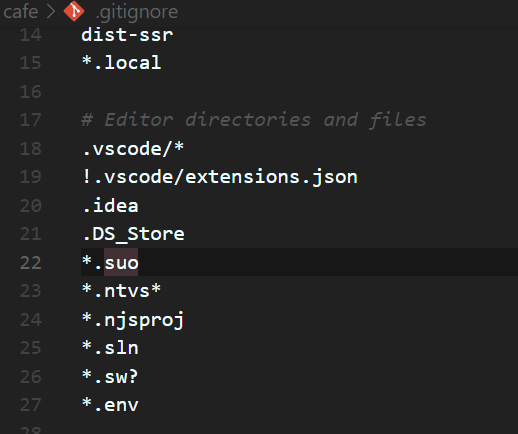


Рис. 15. Файл .gitignore.

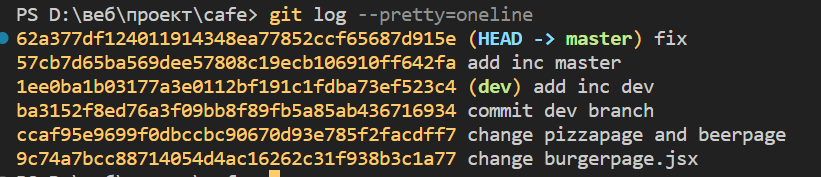


Рис. 16. Вывод log’ов в пользовательском формате

Следующей действительно полезной опцией является --pretty. Эта опция меняет формат вывода. Существует несколько встроенных вариантов отображения. Опция oneline выводит каждый коммит в одну строку.



Рис. 17. Привязка удаленного репозитория к локальному.

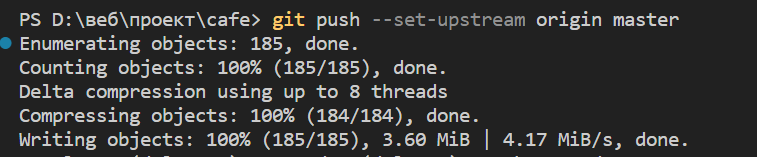


Рис. 18. Пуш новой ветки в удаленный репозиторий.

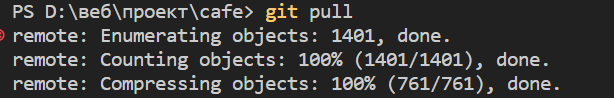


Рис. 19. Загрузка изменений с удаленного репозитория.

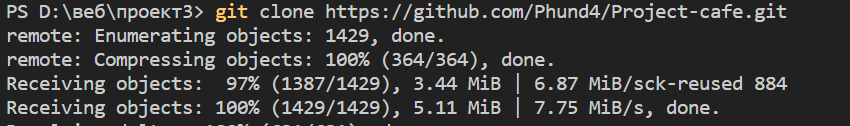


Рис. 20. Клонирование репозитория.

**Выводы**

1. Я научился создавать репозиторий командой git init и клонировать существующий командой git clone.

2. Я научился коммитить изменения git commit и добавлять файлы в индекс git add.

3. Я научился откатываться до определенного коммита несколькими способами (git checkout, git rebase).

4. Я научился создавать ветки git branch, переключаться между ветками git checkout, переименовывать ветки и удалять ветки git checkout –D.

5. Я научился мерджить две ветки git merge и фиксить конфликты.

6. Я научился игнорировать файлы для системы git при помощи файла .gitingore и glob-шаблонов.

7. Я научился выводить логи коммитов git log, а также менять их в пользовательском формате (пример: --pretty-oneline).

8. Я научился загружать изменения на удаленный репозиторий git push и получать изменения git pull/fetch.

9. Я научился загружать удаленный репозиторий git clone.