Санкт-Петербургский Национальный Исследовательский Университет

Информационных Технологий, Механики и Оптики

Факультет Инфокоммуникационных Технологий

**Web-программирование**

**Лабораторная работа 2**

Выполнил

Носов А.Д.

Проверила

Марченко Е. В.

Санкт-Петербург, 2024

**Цель:** изучить основы работы c Git, научиться на практике использовать gulp и написать программу клиент согласно условию задания.

**Задачи:**

* Научиться работать с Git и GitHub;
* Изучить возможности gulp и создать task;
* Написать программу клиент, которая показывает web-страницы одна за другой из списка, причем в программе можно задавать адреса страниц и интервал показа.

**Ход работы**

**Задание 1**

Ссылка на репозиторий:

<https://github.com/Alex-Nosov-ITMO/WebDevelopment_2024-2025/tree/lab2>

Git уже установлен на компьютер, настроим его на работу с проектом лабораторной работы. Для этого мы чуть раньше склонировали его в локальный репозиторий с помощью команды git clone.

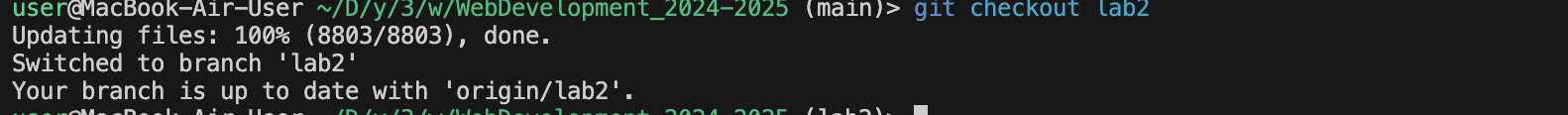
Переключимся на ветку lab2 с лабораторной работой.  


Рисунок 1 – Изменение ветки

Далее внесем изменения в файлы и создадим 3 коммита.

1. Внесли изменения в файл main.go, добавили его в область индексации с помощью команды git add и создали коммит с помощью команды git commit -m “”. То же самое проделали и для других коммитов



Рисунок 2 – Создание первого, второго и третьего коммита

C помощью команды git log проверим, что коммиты успешно созданы.

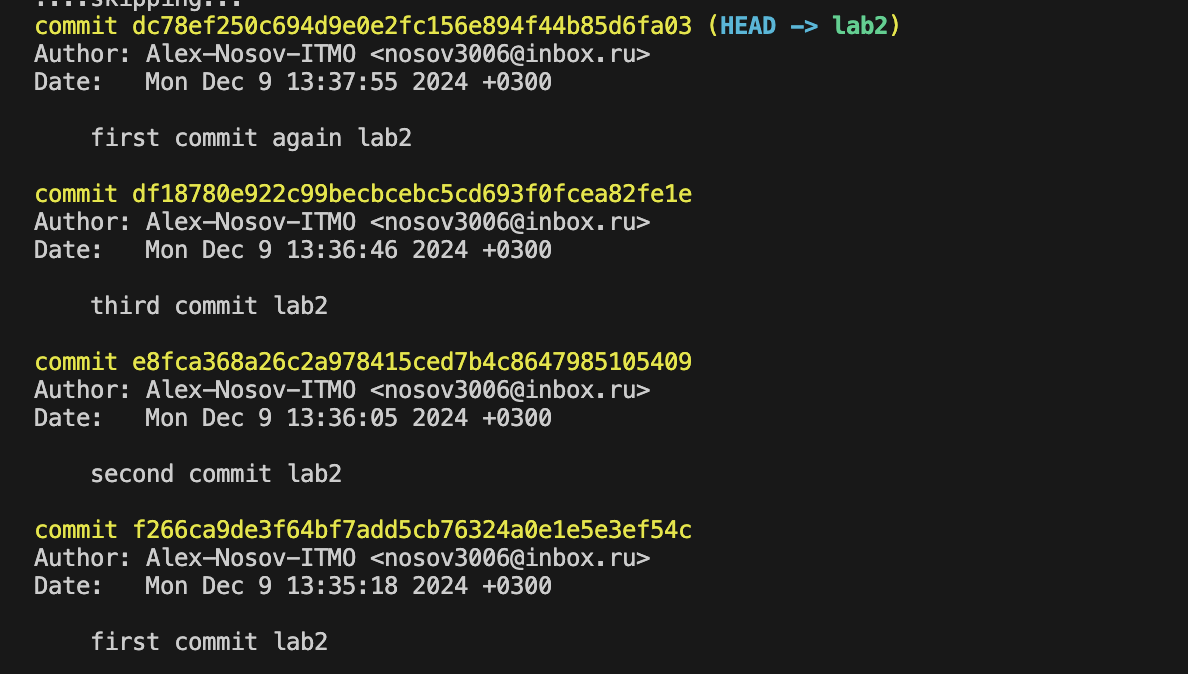


Рисунок 3 – История коммитов

Далее синхронизируем локальный репозиторий с удаленным на GitHub, тем самым отправляя наши изменения, с помощью команды git push origin lab\_1. Видим, что все 3 коммита успешно отобразились.

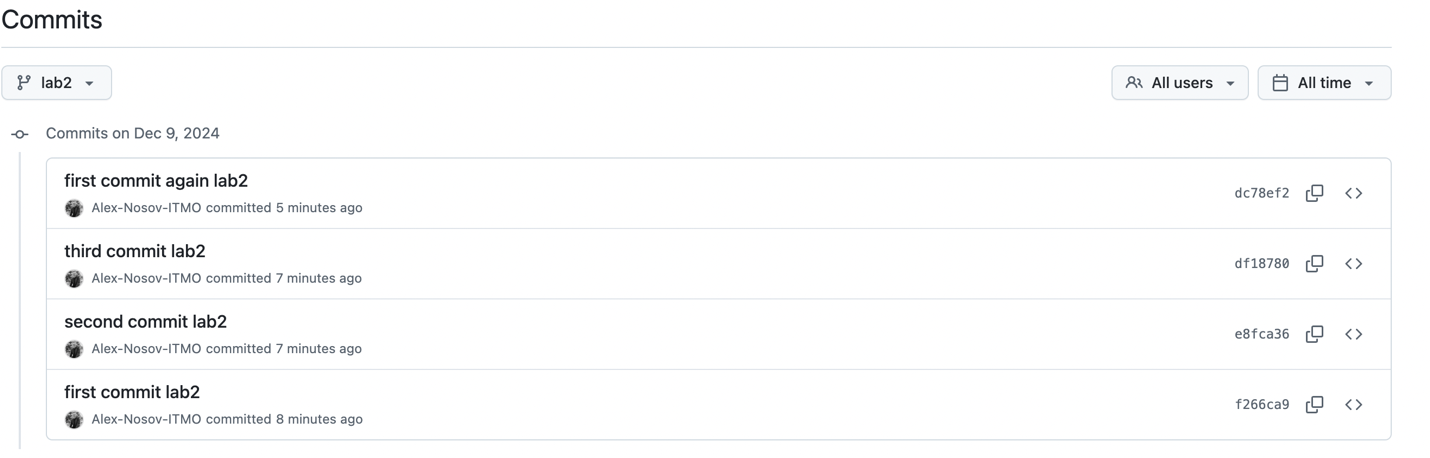


Рисунок 4 – Коммиты на GitHub

**Задание 2**

С помощью команды npm install --global gulp-cli установим Gulp глобально.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 5 - Установка Gulp глобально

Затем инициализируем проект и установим Gulp как зависимость.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6 – Команда npm init -y

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 7 – Команда npm install --save-dev gulp

Затем установим пакет browsesync через команду

npm install browser-sync gulp –save-dev.

Cоздадим файл gulpfile.js в корне проекта, он определяет задачи, которые будет выполнять gulp. Создадим 1 таск, который будет обновлять изменения в файле .html с помощью browsesync.

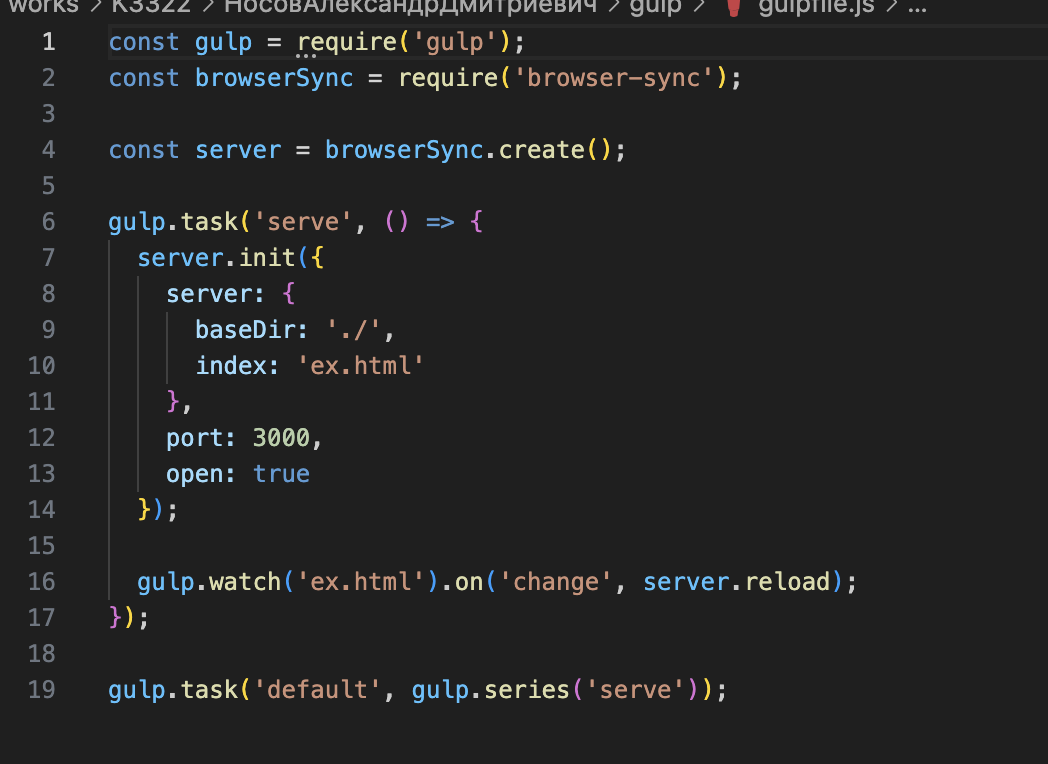


Рисунок 8 – Задачи для gulp

Проверим успешность выполнения скрипта.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 9 – Файл ex.html до изменения

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 10 – Запуск скрипта

На скриншоте видно, что при изменении файла страница перегружается.

**Задание 3**

Напишем программу клиент, которая показывает web-страницы одна за другой из списка, причем в программе можно задавать адреса страниц и интервал показа. Для этого создадим 3 файла:

index.html - основной файл веб-страницы, содержащий структуру HTML для интерфейса пользователя.

Изображение выглядит как текст, программное обеспечение, снимок экрана

Автоматически созданное описание

Рисунок 11 -Файл index.html

script.js - содержит JavaScript-код, который реализует функциональность страницы.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок – 12. Файл script.js

Посмотрим на результат.

Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 13 – Программа клиент, показывающая web-страницы по списку

Изображение выглядит как хоккей, текст, шлем, человек

Автоматически созданное описание

Рисунок14 – Отображение веб-страницы в окне показов

**Вывод**

В результате выполнения лабораторной работы были изучены git и gulp, созданы простые таски, а также написана программа-клиент, показывающая web-страницы по очереди из списка с возможностью добавления адресов страниц и установкой интервала показа