Отчёт по лабораторной работе №4

*Дисциплина: Операционные системы*

Долгаев Евгений НММбд-01-24

Содержание

# 1 Цель работы

Получение навыков правильной работы с репозиториями git.

# 2 Задание

1. Выполнить работу для тестового репозитория.
2. Преобразовать рабочий репозиторий в репозиторий с git-flow и conventional commits.

# 3 Выполнение лабораторной работы

## 3.1 Установка программного обеспечения

Для начала установим Gitflow (рис. 1).

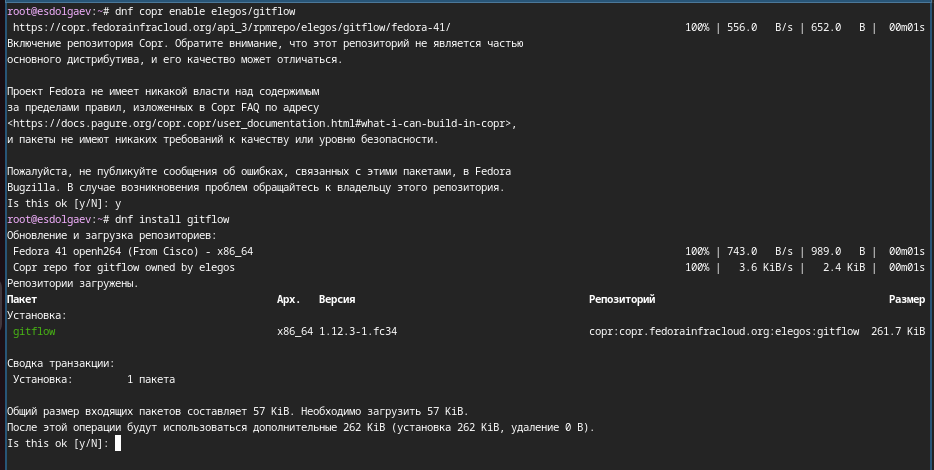


Рис. 1: Установка Gitflow

Далее установим Node.js и настроим его (рис. 2, 3, 4)

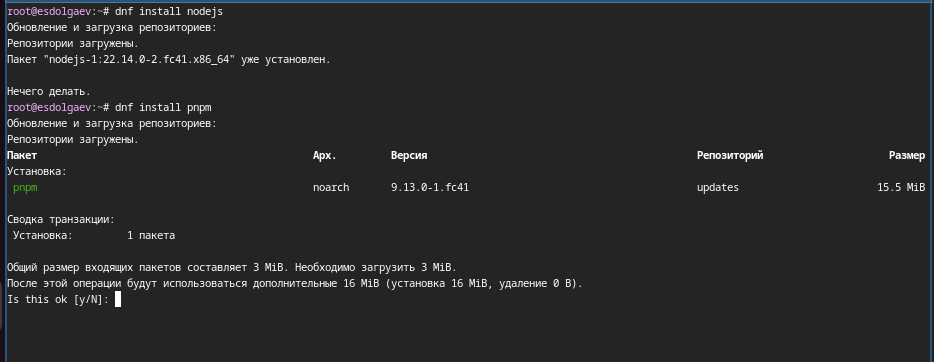


Рис. 2: Установка Node.js

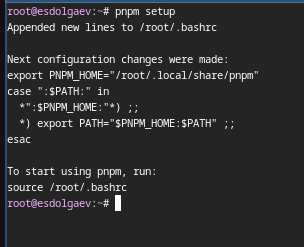


Рис. 3: Настройка Node.js

Рис. 4: Настройка Node.js

Рис. 4: Настройка Node.js

Настроим общепринятые коммиты (рис. 5, 6):

1. commitizen

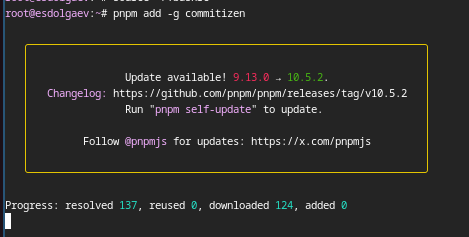


Рис. 5: commitizen

1. standard-changelog



Рис. 6: standard-changelog

## 3.2 Практический сценарий использования git

Создим репозиторий на GitHub. Для примера назовём его git-extended (рис. 7).

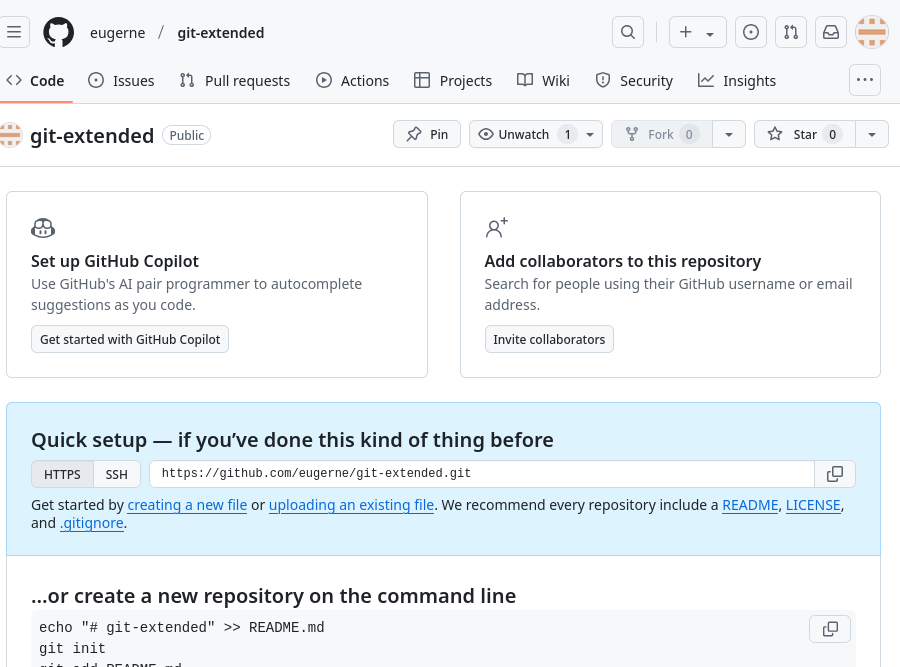


Рис. 7: Репозиторий на Github

Сделаем первый коммит (рис. 8, 9).

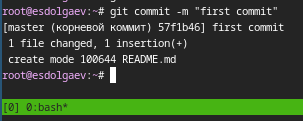


Рис. 8: Первый коммит

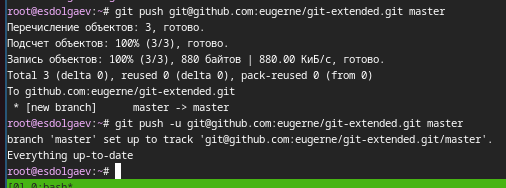


Рис. 9: Выполнение команды git push

Далее произведем конфигурацию для пакетов Node.js с помощью команды:

pnpm init

Сконфигурим формат коммитов. Для этого добавим в файл package.json команду для формирования коммитов:

"config": {  
 "commitizen": {  
 "path": "cz-conventional-changelog"  
 }  
 }

Добавим новые файлы (рис. 10):

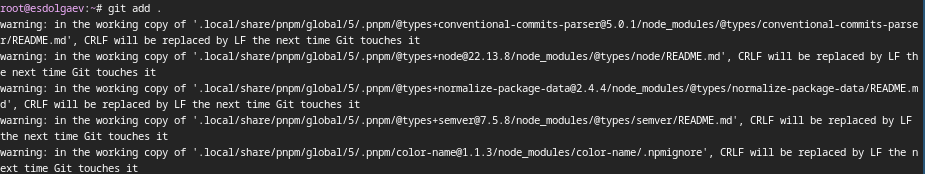


Рис. 10: Добавление новых файлов

Выполним коммит и отправим на github (рис. 11):



Рис. 11: Работа команд git cz и git push

Инициализируем git-flow (рис. 12):

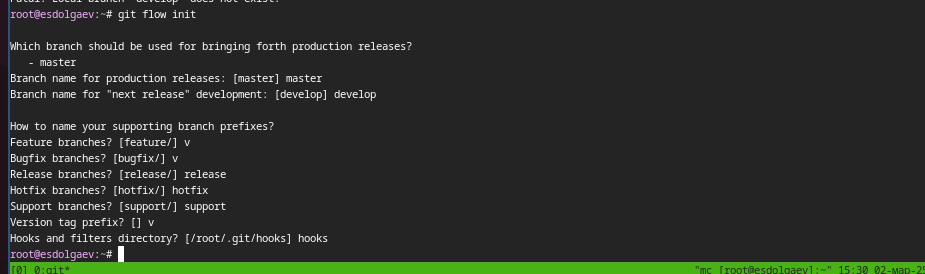


Рис. 12: Инициализация git-flow

Проверим, что мы на ветке develop: (рис. 13):



Рис. 13: Проверка

Загрузим весь репозиторий в хранилище: (рис. 14):

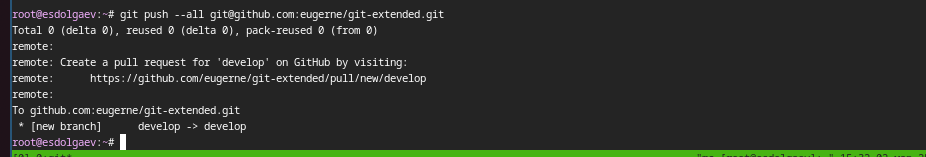


Рис. 14: Загрузка репозитория

Установим внешнюю ветку как вышестоящую для этой ветки (рис. 15):

Рис. 15: Установка внешней ветки

Рис. 15: Установка внешней ветки

Создадим релиз с версией 1.0.0 (рис. 16):

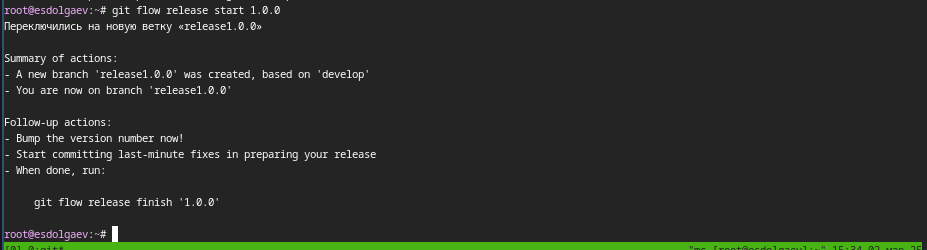


Рис. 16: Создание релиза

Создадим журнал изменений (рис. 17):



Рис. 17: Создание журнала изменений

Добавим журнал изменений в индекс (рис. 18):



Рис. 18: Добавление журнала изменений в индекс

Зальём релизную ветку в основную ветку (рис. 19):



Рис. 19: Релизная ветка сливается в основную

Отправим данные на github (рис. 20, 21):

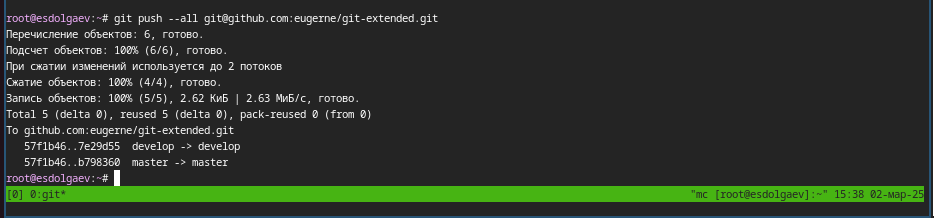


Рис. 20: Отправка на Github

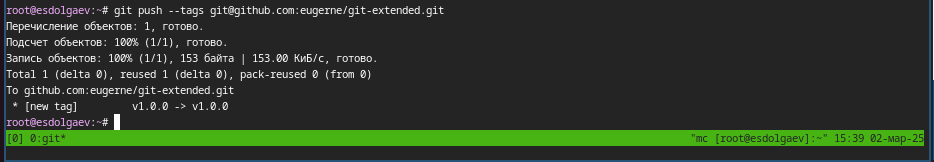


Рис. 21: Отправка на Github

Создадим ветку для новой функциональности (рис. 22):

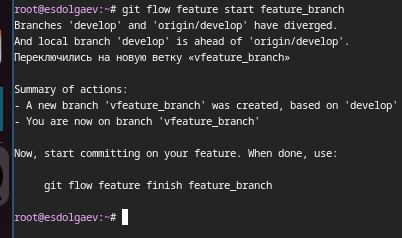


Рис. 22: Новая ветка

Далее, продолжаем работу c git как обычно. По окончании разработки новой функциональности следующим шагом следует объединить ветку feature\_branch c develop (рис. 23):



Рис. 23: Объединение веток

Создадим релиз с версией 1.2.3 (рис. 24):

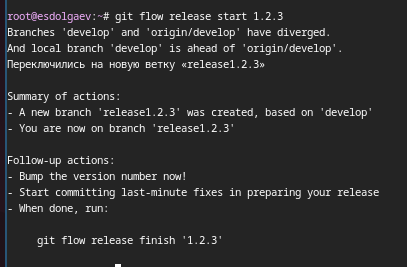


Рис. 24: Создание релиза

Обновим номер версии в файле package.json (рис. 25).



Рис. 25: Новая ветка

Создадим журнал изменений (рис. 26):

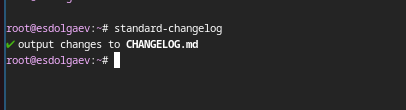


Рис. 26: Журнал изменений

Добавим журнал изменений в индекс (рис. 27):

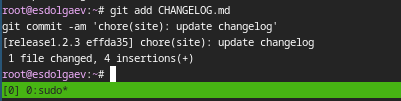


Рис. 27: Добавление в индукс

Создадим релиз на github с комментарием из журнала изменений. Для этого будем использовать утилиты работы с github: (рис. 28):

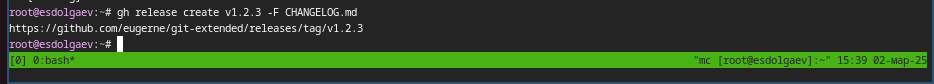


Рис. 28: Создание релиза

# 4 Выводы

В ходе выполнения лаборатоной работы я получил навыки управления репозиториями git.

# Список литературы