Отчёт по лабораторной работе №4

Дисциплина: Операционные Системы

Азарцова Вероника Валерьевна

Содержание

# 1 Цель работы

Цель данной лабораторной работы - получение практических 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 Теоретическое введение

## 3.1 Общая информация

Gitflow Workflow опубликована и популяризована Винсентом Дриссеном. Gitflow Workflow предполагает выстраивание строгой модели ветвления с учётом выпуска проекта. Данная модель отлично подходит для организации рабочего процесса на основе релизов. Работа по модели Gitflow включает создание отдельной ветки для исправлений ошибок в рабочей среде.

## 3.2 Последовательность действий при работе по модели Gitflow

1. Из ветки master создаётся ветка develop.
2. Из ветки develop создаётся ветка release.
3. Из ветки develop создаются ветки feature.
4. Когда работа над веткой feature завершена, она сливается с веткой develop.
5. Когда работа над веткой релиза release завершена, она сливается в ветки develop и master.
6. Если в master обнаружена проблема, из master создаётся ветка hotfix.
7. Когда работа над веткой исправления hotfix завершена, она сливается в ветки develop и master.

## 3.3 Краткое описание семантического версионирования

Семантическое версионирование описывается в манифесте семантического версионирования.

Кратко его можно описать следующим образом:

Версия задаётся в виде кортежа МАЖОРНАЯ\_ВЕРСИЯ.МИНОРНАЯ\_ВЕРСИЯ.ПАТЧ. Номер версии следует увеличивать:  
МАЖОРНУЮ версию, когда сделаны обратно несовместимые изменения API.  
МИНОРНУЮ версию, когда вы добавляете новую функциональность, не нарушая обратной совместимости.  
ПАТЧ-версию, когда вы делаете обратно совместимые исправления.  
Дополнительные обозначения для предрелизных и билд-метаданных возможны как дополнения к МАЖОРНАЯ.МИНОРНАЯ.ПАТЧ формату.

# 4 Выполнение лабораторной работы

Устанавливаю git-flow из коллекции репозиториев copr (рис. 1).

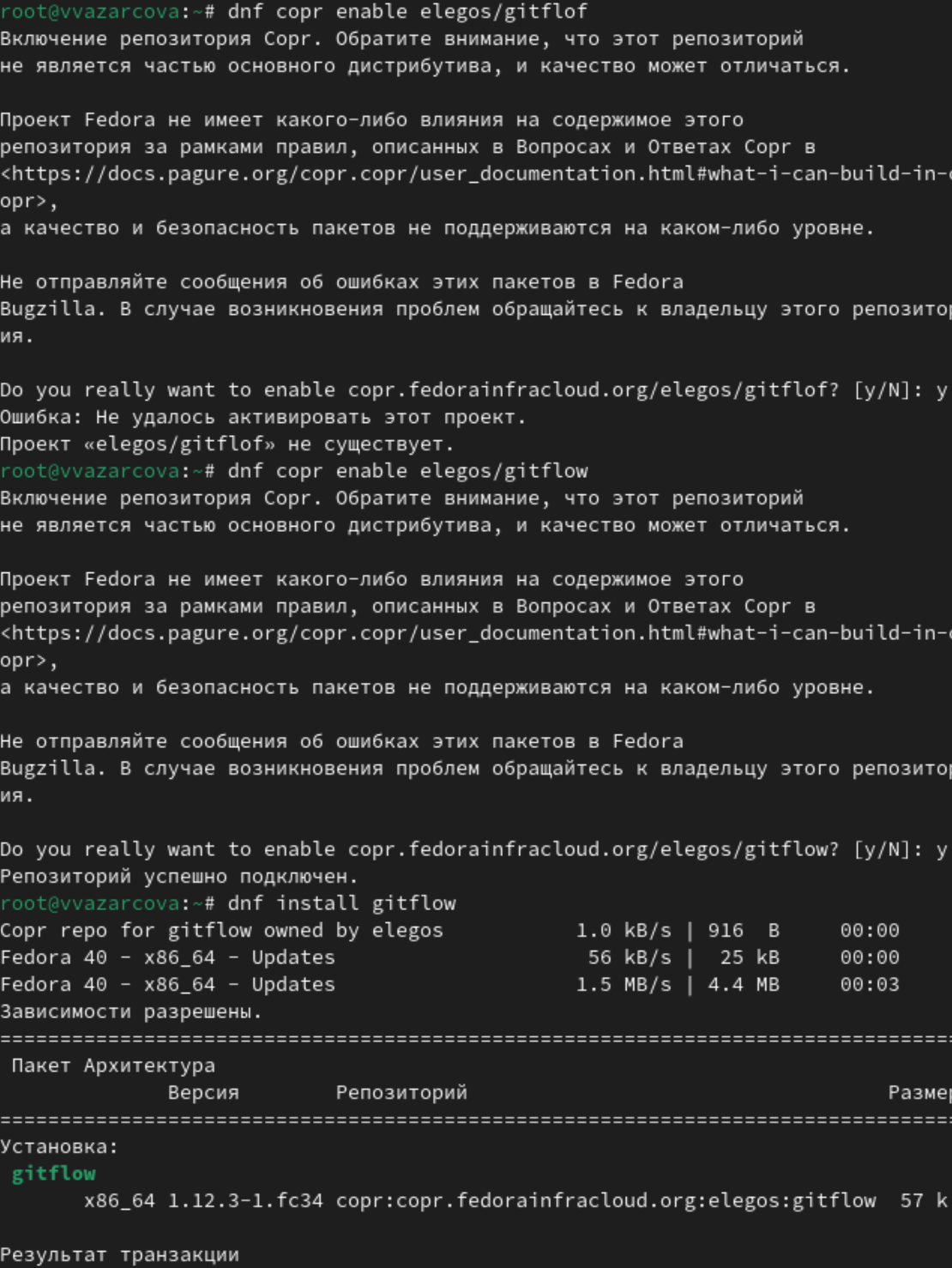


Рис. 1: Установка git-flow

Устанавливаю Node.js для семантического версионирования и общепринятых коммитов (рис. 2).

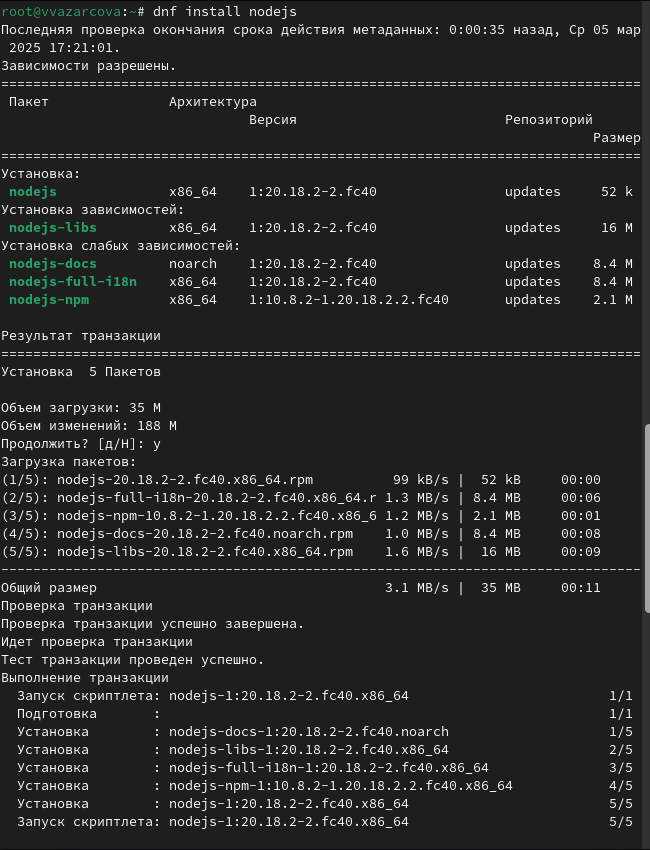


Рис. 2: Установка Node.js

Настраиваю Node.js, добавив каталог с исполняемыми файлами, устанавливаемыми yarn, в переменную PATH (рис. 3).

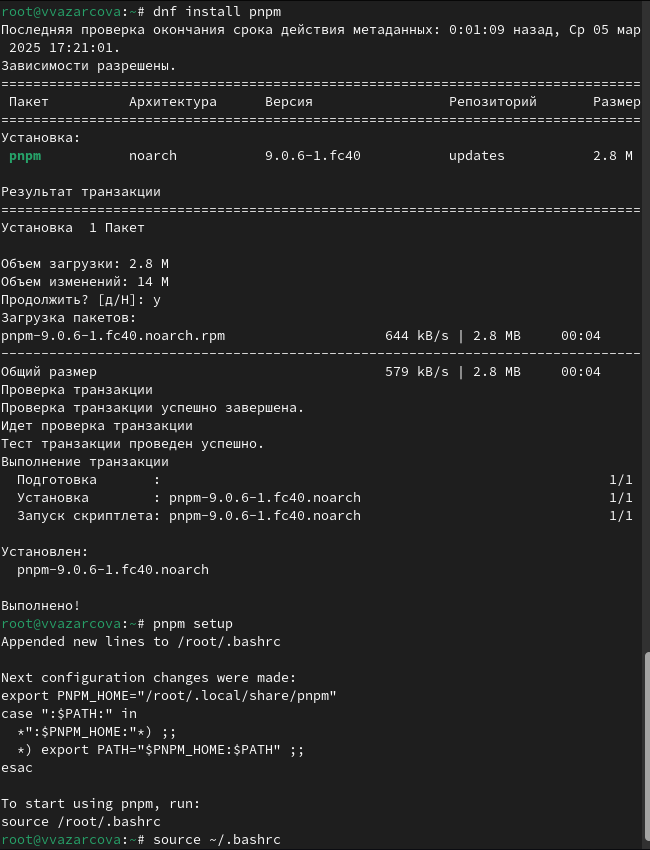


Рис. 3: Настройка Node.js

Устанавливаю commitizen для помощи в форматировании коммитов (рис. 4).

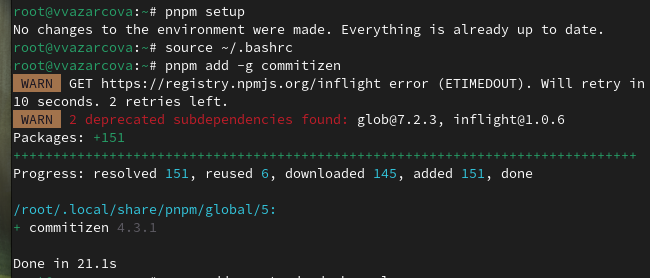


Рис. 4: Установка commitizen

Устанавливаю standard-changelog для помощи в создании логов (рис. 5).

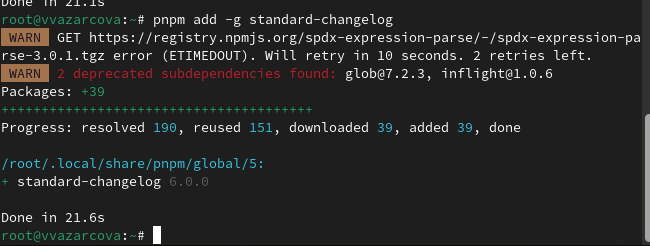


Рис. 5: Установка standard-changelog

Создаю репозиторий git-extended на github и клонирую его на устройство (рис. 6).

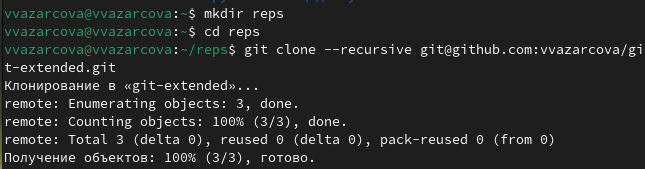


Рис. 6: Создание репозитория git-extended

Создаю пустой файл 123.txt, делаю первый коммит (рис. 7).

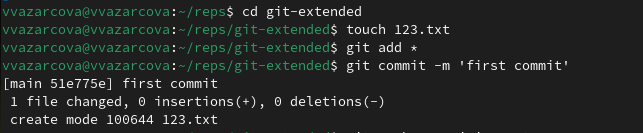


Рис. 7: Первый коммит

Отправляю коммит на сервер github (рис. 8).

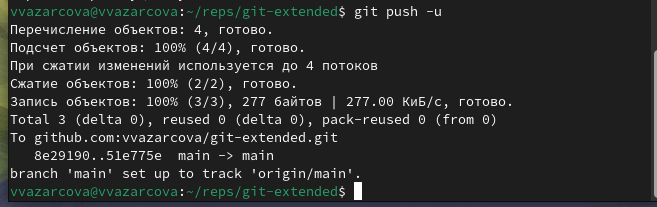


Рис. 8: Отправка коммита

Создаю файл конфигурации для пакетов Node.js, открываю и редактирую его соответственно примеру в задании лабораторной работы (рис. 9).

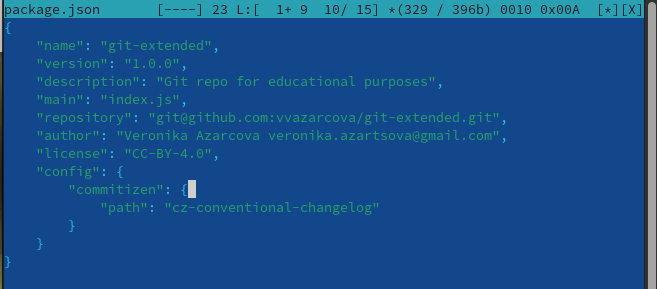


Рис. 9: Конфигурация Node.js

Добавляю новые файлы, выполняю коммит и отправляю его на сервер github (рис. 10).

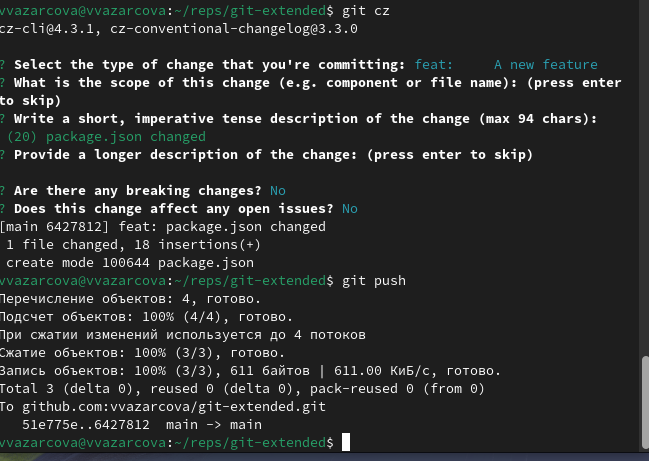


Рис. 10: Второй коммит

Инициализирую git-flow и проверяю, что я на ветке develop (рис. 11).

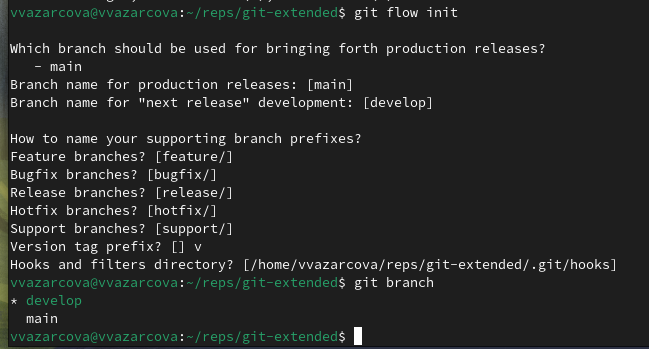


Рис. 11: Инициализация git-flow

Создаю релиз с версией 1.0.0, журнал изменений. Добавляю журнал изменений в индекс и заливаю релизную ветку в основную ветку (рис. 12).

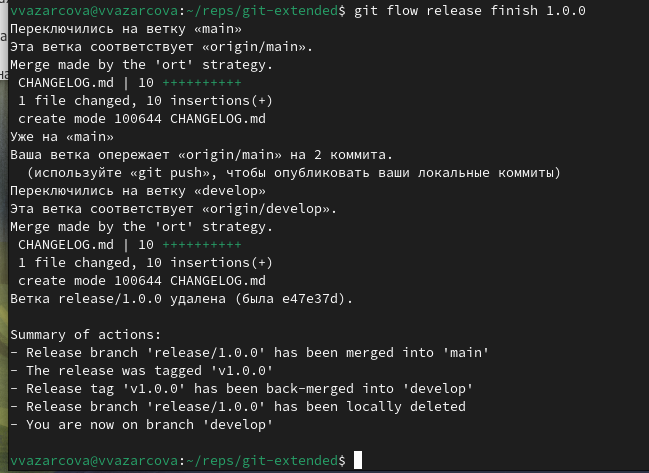


Рис. 12: Релиз 1.0.0

Создаю ветку для новой функциональности, далее, продолжаю работу c git как обычно. По окончании разработки новой функциональности следующим шагом объединяю ветку feature\_branch c develop (рис. 13).

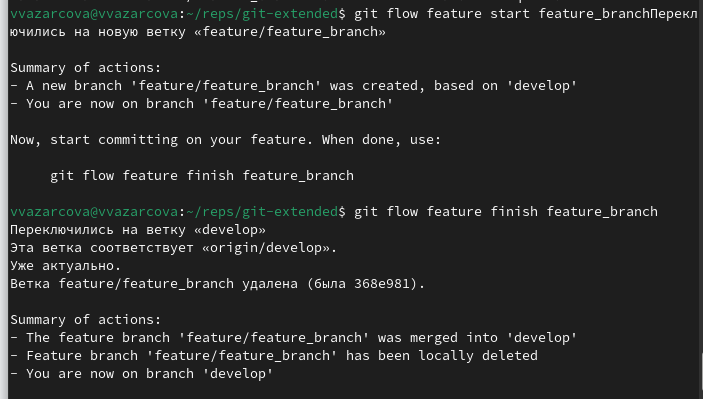


Рис. 13: Объединение ветки feature\_branch c develop

Обновляю номер версии в файле package.json до 1.2.3. Создаю журнал изменений и добавляю его в индекс. Заливаю релизную ветку в основную ветку и отправляю данные на github. Создаю релиз на github с комментарием из журнала изменений (рис. 14).

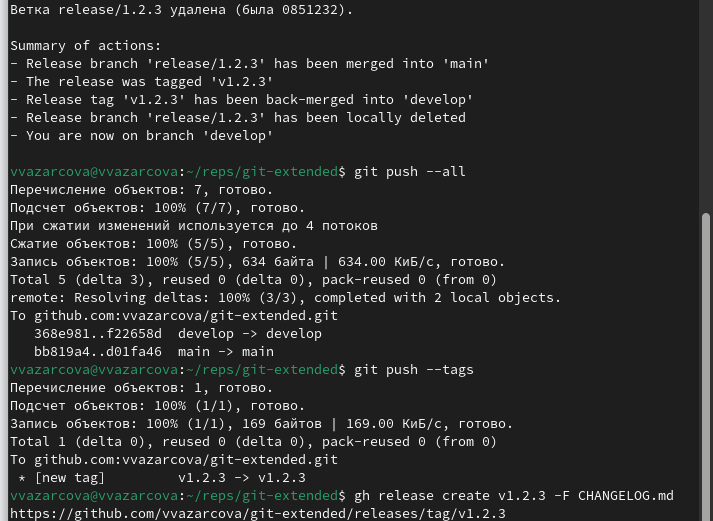


Рис. 14: Релиз 1.2.3

# 5 Выводы

Подводя итоги проведенной лабораторной работе, я получила практические навыки правильной работы с репозиториями git, выполнила работу для тестового репозитория и преобразовала рабочий репозиторий в репозиторий с git-flow и conventional commits.

# Список литературы

1. GDB: The GNU Project Debugger. — URL: https://www.gnu.org/software/gdb/.
2. GNU Bash Manual. — 2016. — URL: https://www.gnu.org/software/bash/manual/.
3. Midnight Commander Development Center. — 2021. — URL: https://midnight-commander.org/.
4. NASM Assembly Language Tutorials. — 2021. — URL: https://asmtutor.com/.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O’Reilly Media, 2005. —354 с. — (In a Nutshell). — ISBN 0596009658. — URL: http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658.
6. Robbins A. Bash Pocket Reference. — O’Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: https://www.nasm.us/docs.php.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.