Лабораторная работа №4

Продвинутое использование git

Полякова Юлия Александровна

Содержание

# 1 Цель работы

Получение навыков правильной работы с репозиториями git

# 2 Задание

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

# 3 Выполнение лабораторной работы

1. Установка git-flow командами dnf copr enable elegos/gitflow и dnf install gitflow под суперпользователем (рис. 1).

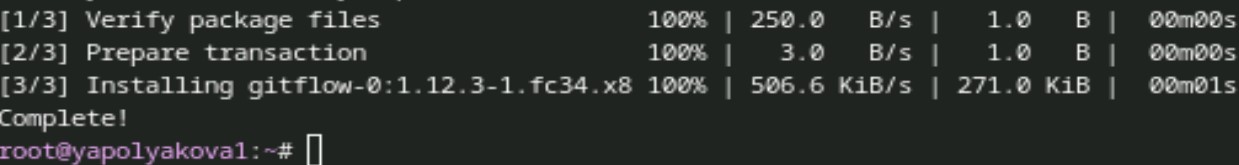


Рис. 1: Установка git-flow

1. Установка Node.js (dnf install nodejs) и pnpm (рис. 2)

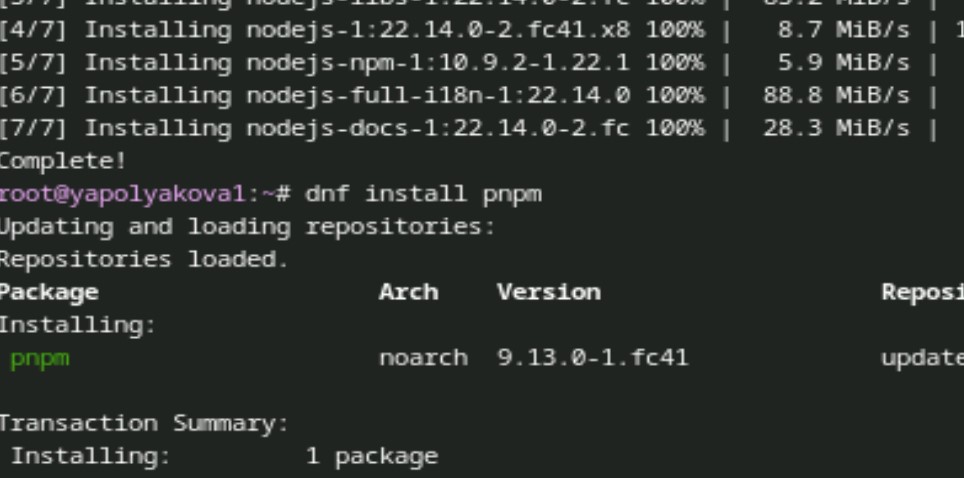


Рис. 2: Установка Node.js

1. Настройка Node.js. Для работы с Node.js добавим каталог с исполняемыми файлами, устанавливаемыми yarn, в переменную PATH. (рис. 3)

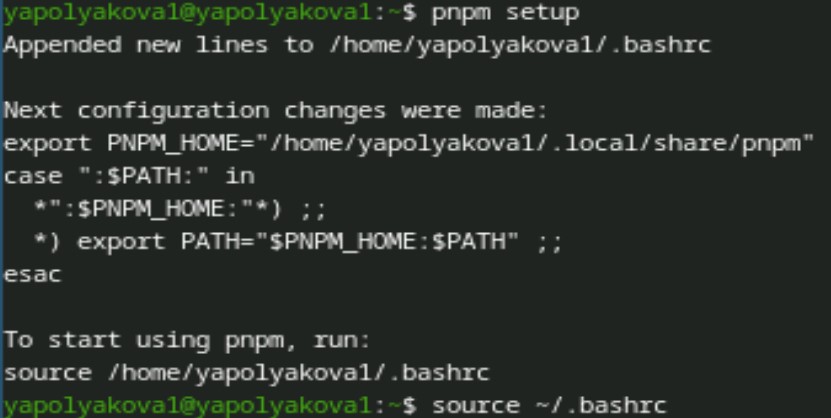


Рис. 3: Настройка Node.js

1. Устанавливаем программу для создания общепринятых коммитов (рис. 4)

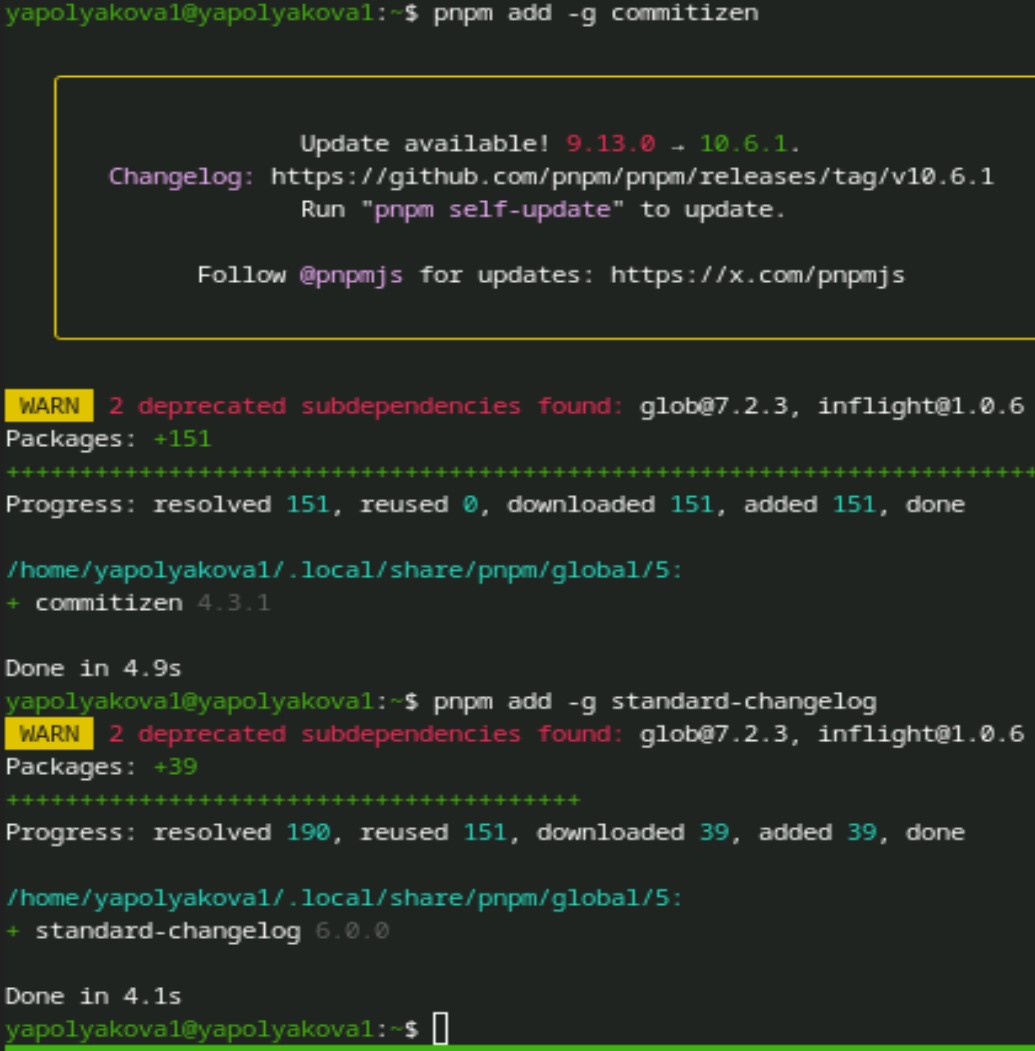


Рис. 4: Общепринятые коммиты

1. Создаем репозиторий с именем git-extended на GitHub, клонируем его в папку с таким же именем, переходим в эту папку (рис. 5)

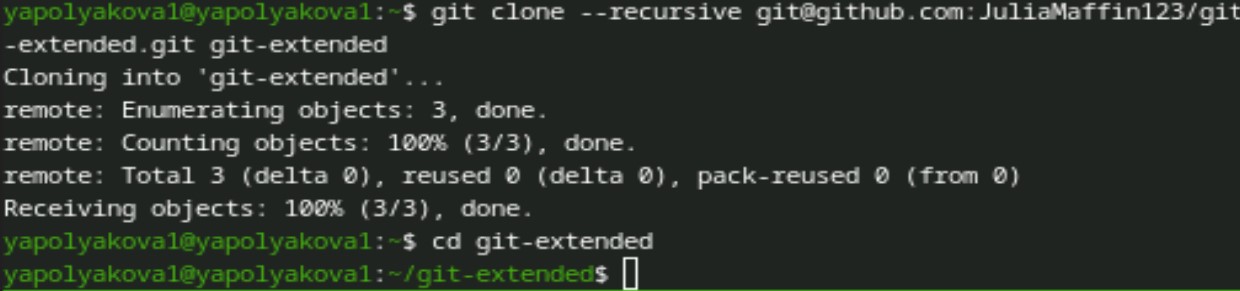


Рис. 5: Создание репозитория git

1. Делаем первый коммит и выкладывем на github (рис. 6)

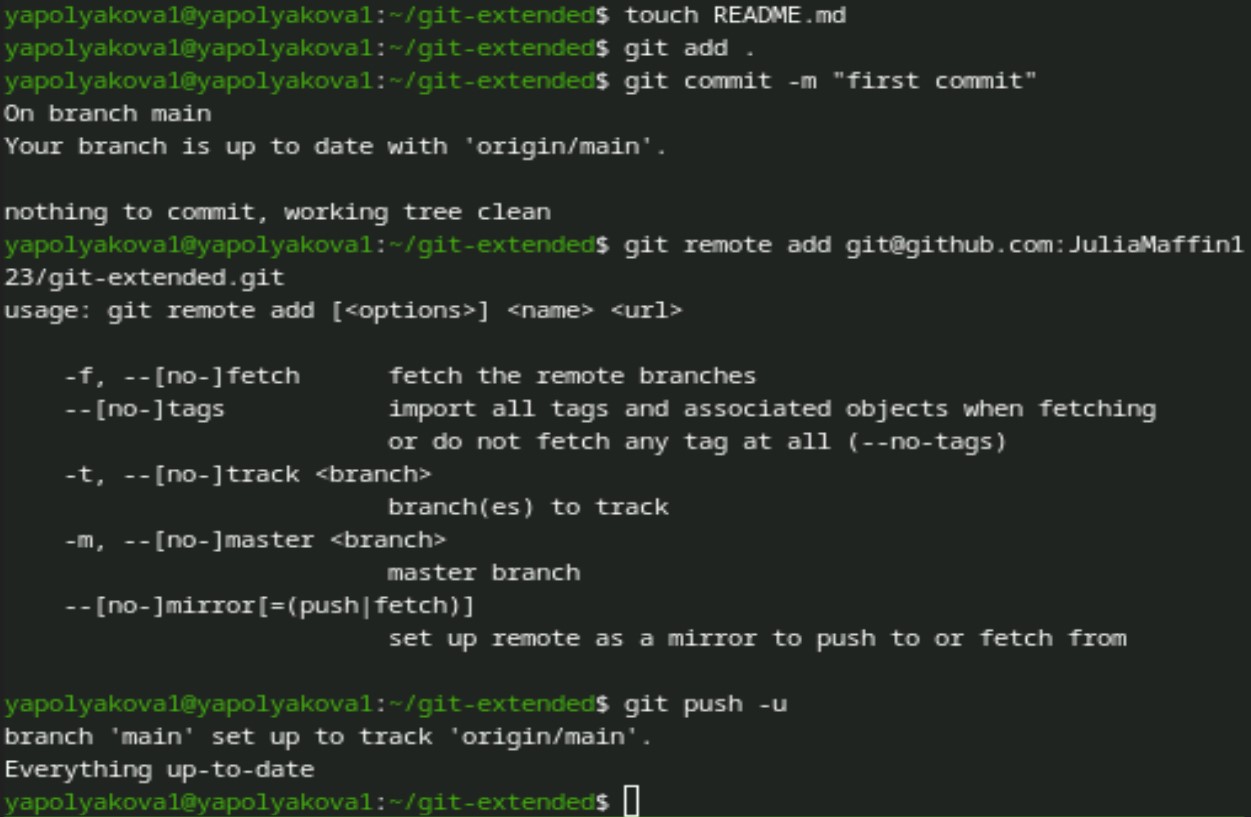


Рис. 6: Первый коммит, загрузка файлов

1. Задаем конфигурацию для пакетов Node.js (рис. 7)

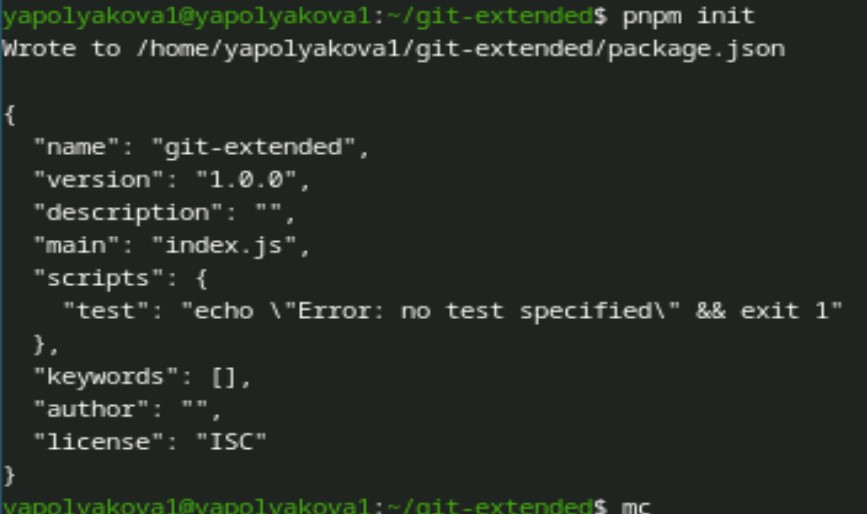


Рис. 7: Задаем конфигурацию

1. Добавим в файл package.json команду для формирования коммитов (рис. 8)

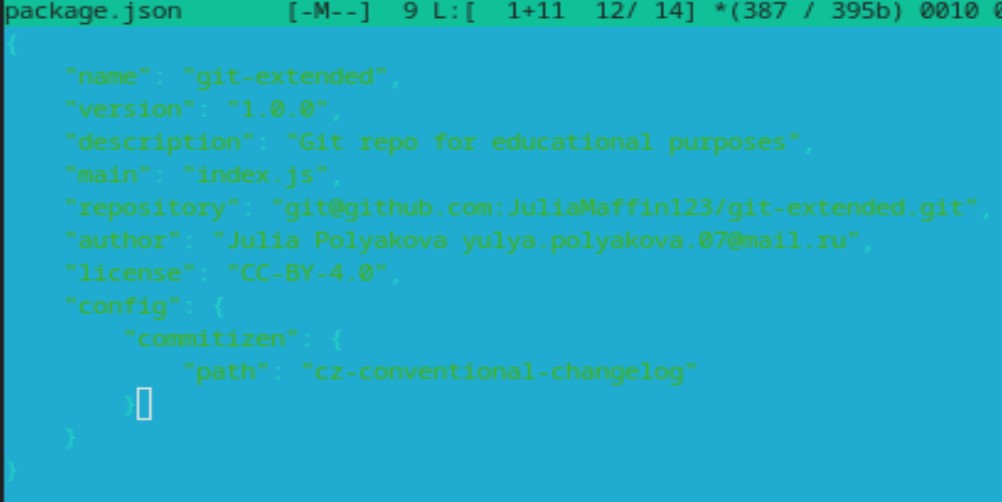


Рис. 8: Меняем package.json

1. Добавим новые файлы на сервер (рис. 9)

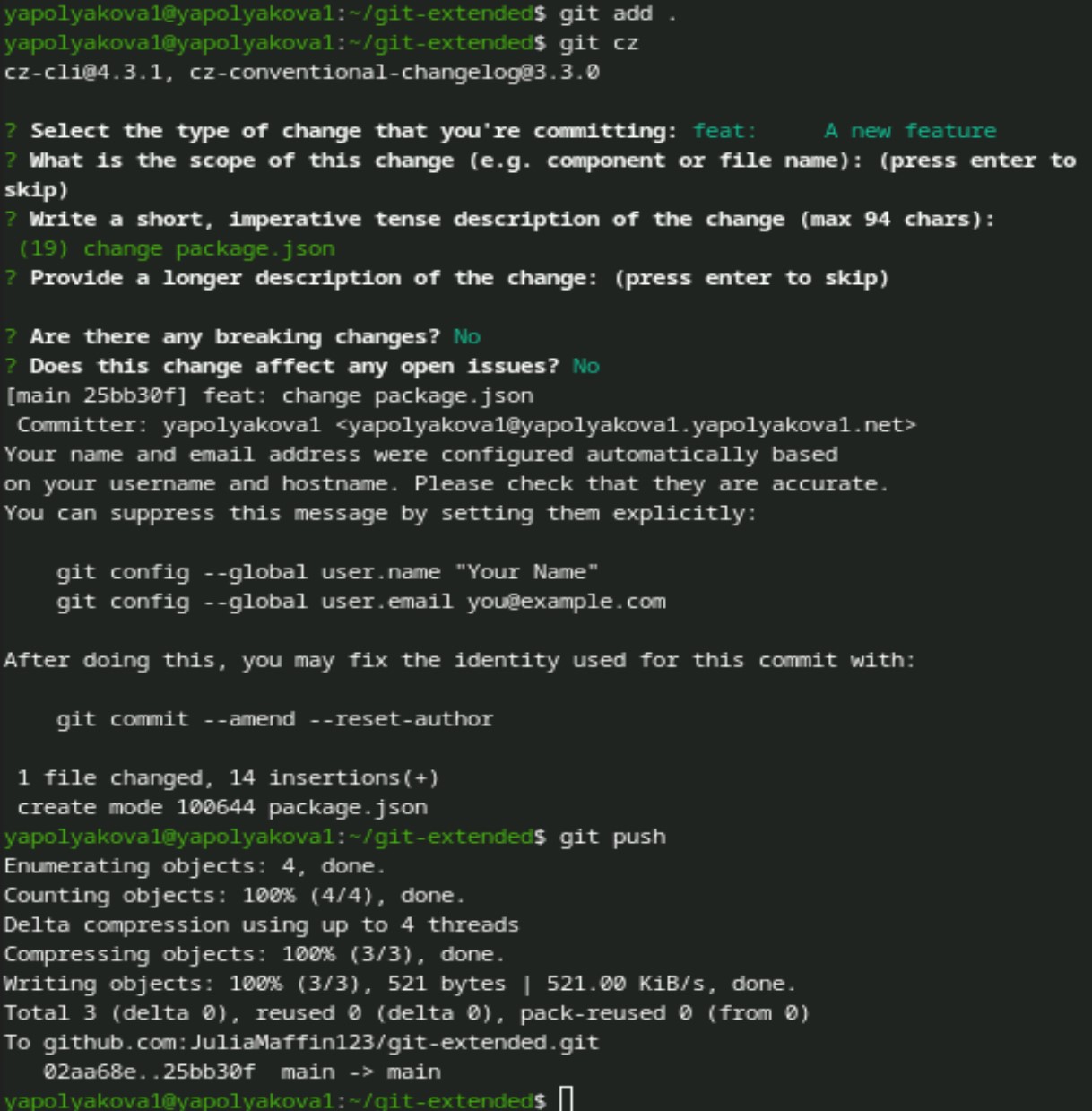


Рис. 9: Отправление файлов на сервер

1. Инициализируем git-flow, проверяем, что мы на ветке develop, загружаем весь репозиторий на git (рис. 10)

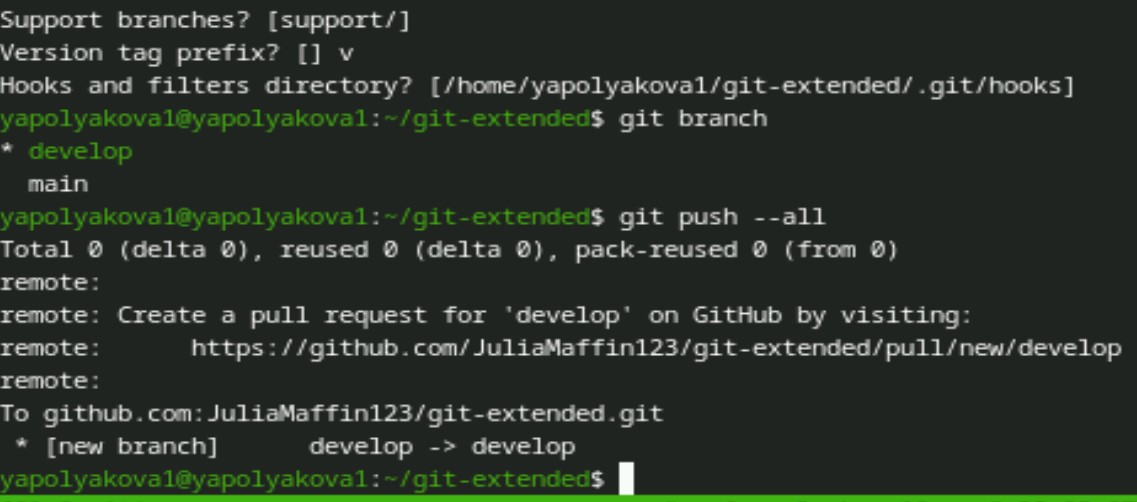


Рис. 10: Конфигурация git-flow 1-ая ч.

1. Устанавливаем внешнюю ветку как вышестоящую, создаем релиз с версией 1.0.0 (рис. 11)

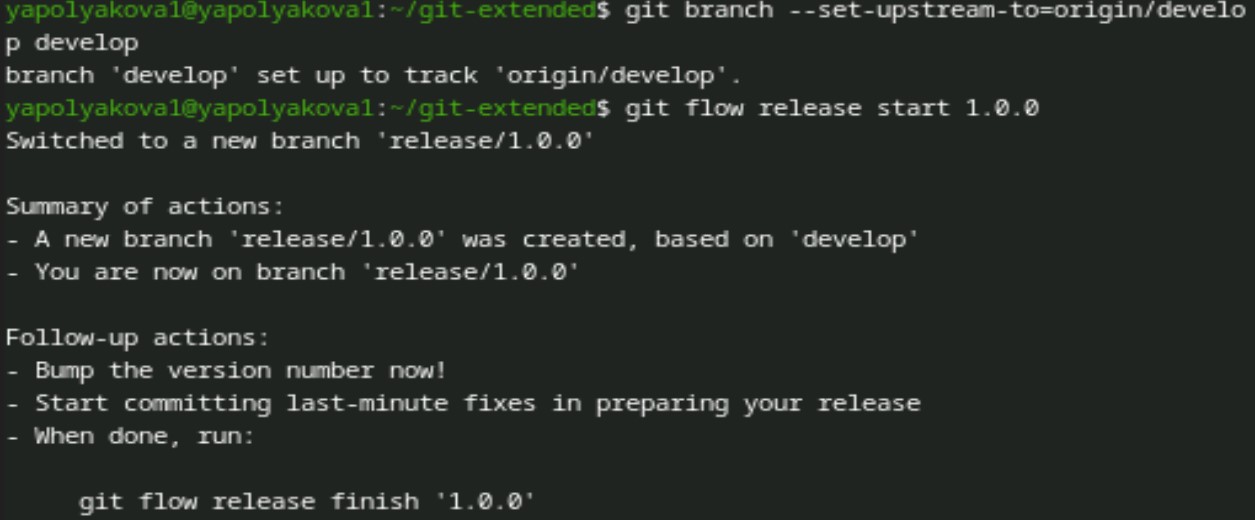


Рис. 11: Конфигурация git-flow 2-ая ч.

1. Создаем журнал изменений, добавим его в индекс (рис. 12)

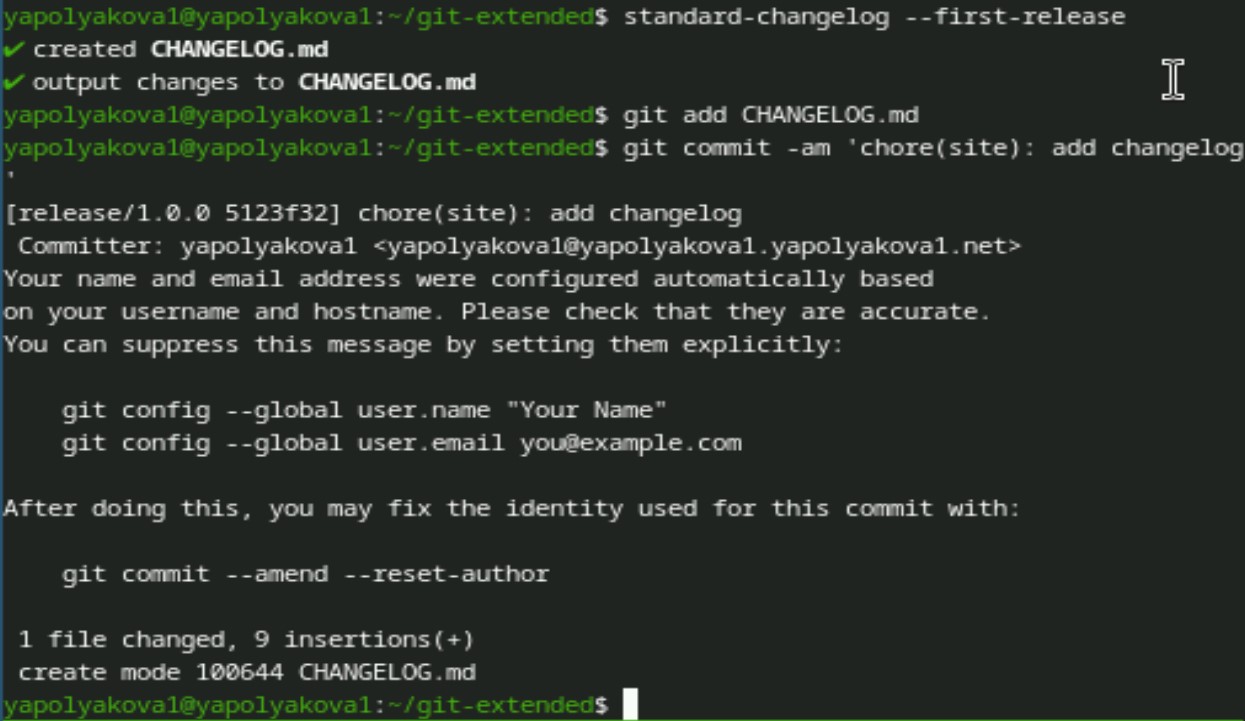


Рис. 12: Конфигурация git-flow 3-я ч.

1. Зальём релизную ветку в основую (рис. 13)

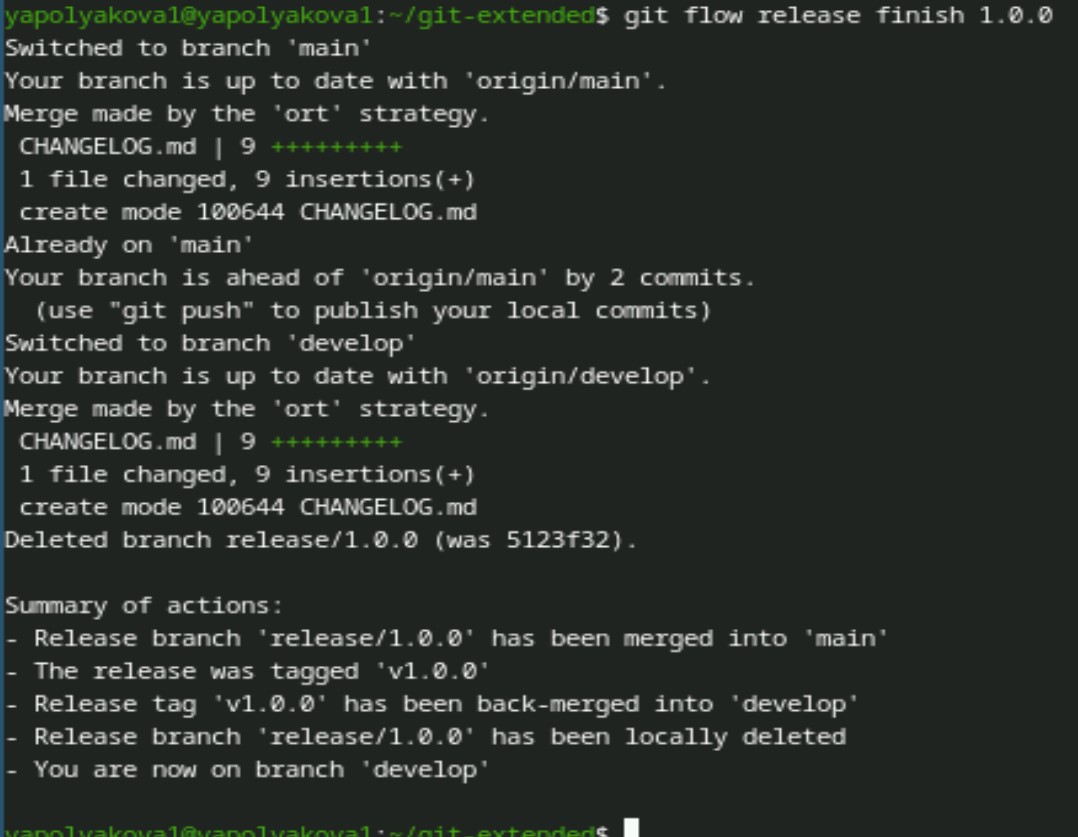


Рис. 13: Конфигурация git-flow 4-ая ч.

1. Отправляем данные о репозитории и ярлыках на github. Создаем релиз с помощью gh (рис. 14)

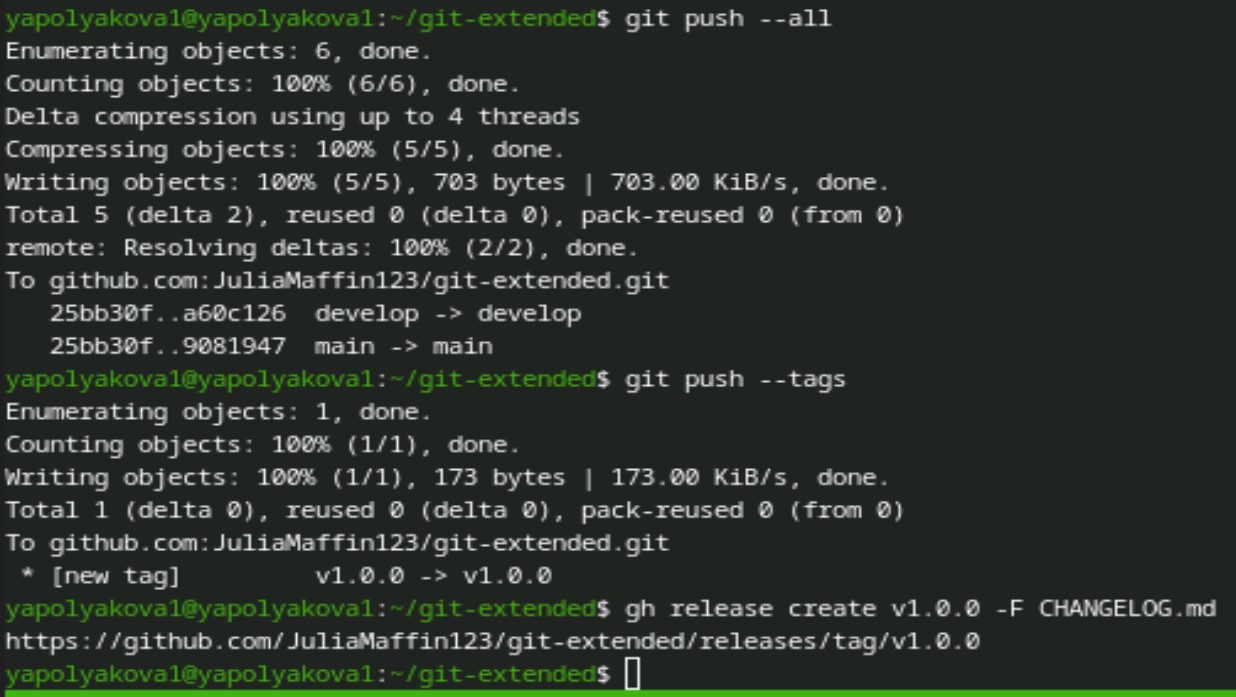


Рис. 14: Конфигурация git-flow 5-ая ч.

1. При разработке новой функциональности мы создаем ветку, работаем с git, а затем объединяем ветку feature\_branch c develop (рис. 15)

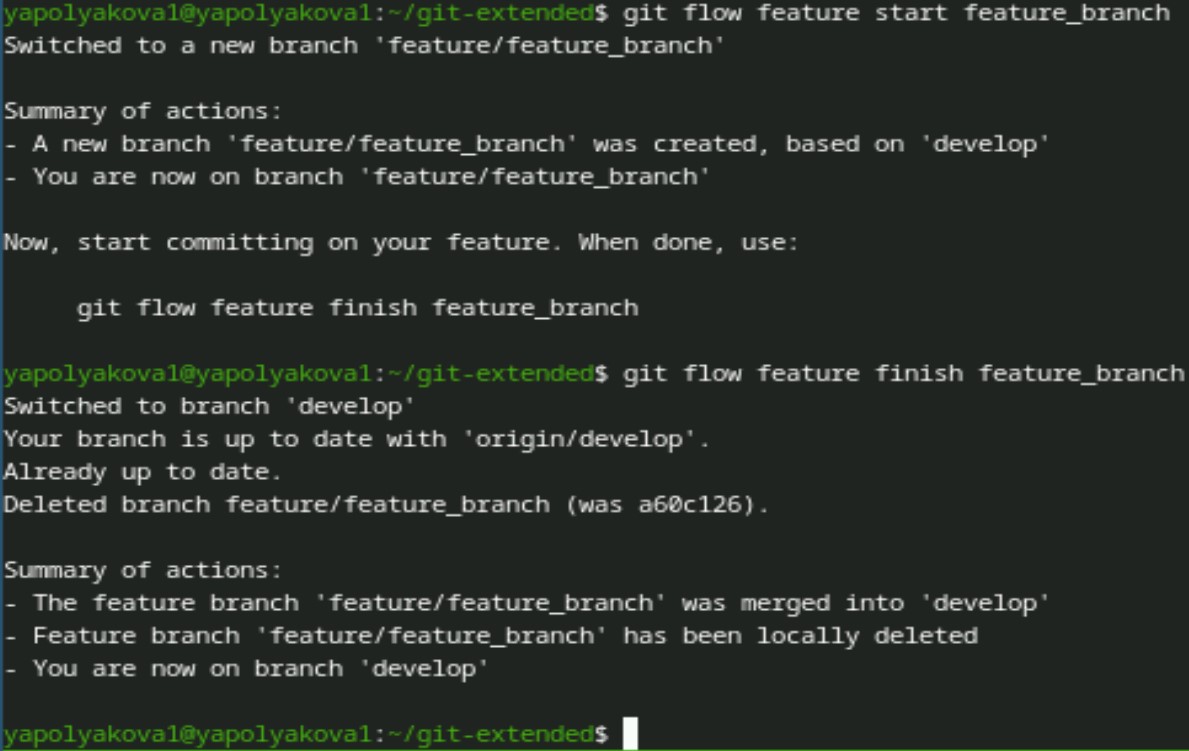


Рис. 15: Разработка новой функциональности

1. Создаем релиз с версией 1.2.3, обновляем номер версии в файле package.json, создаем журнал изменений (рис. 16)

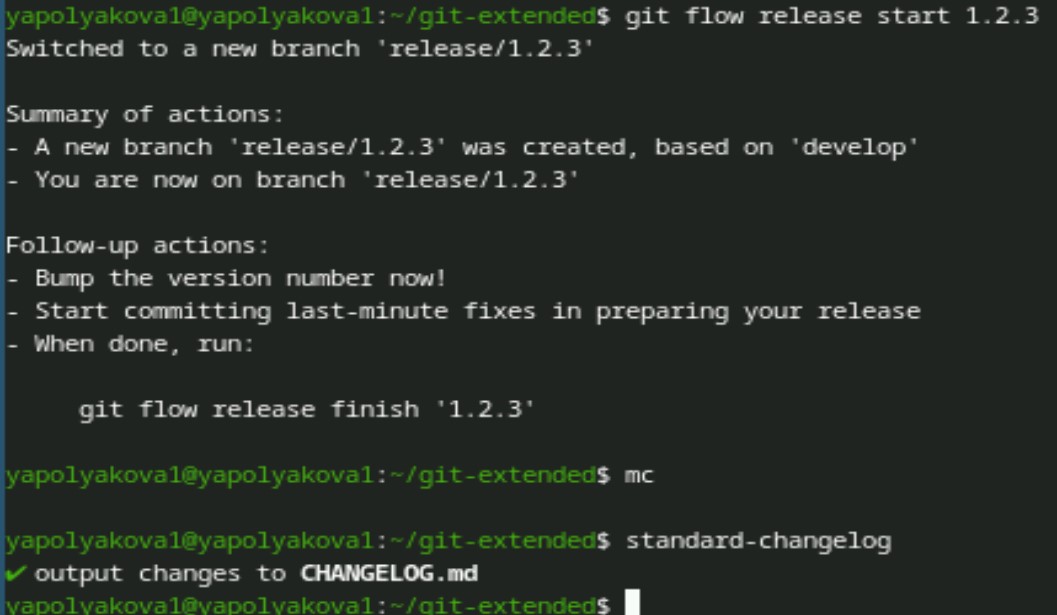


Рис. 16: Создание релиза git-flow 1-ая ч.

1. Добавим журнал изменений в индекс (рис. 17)

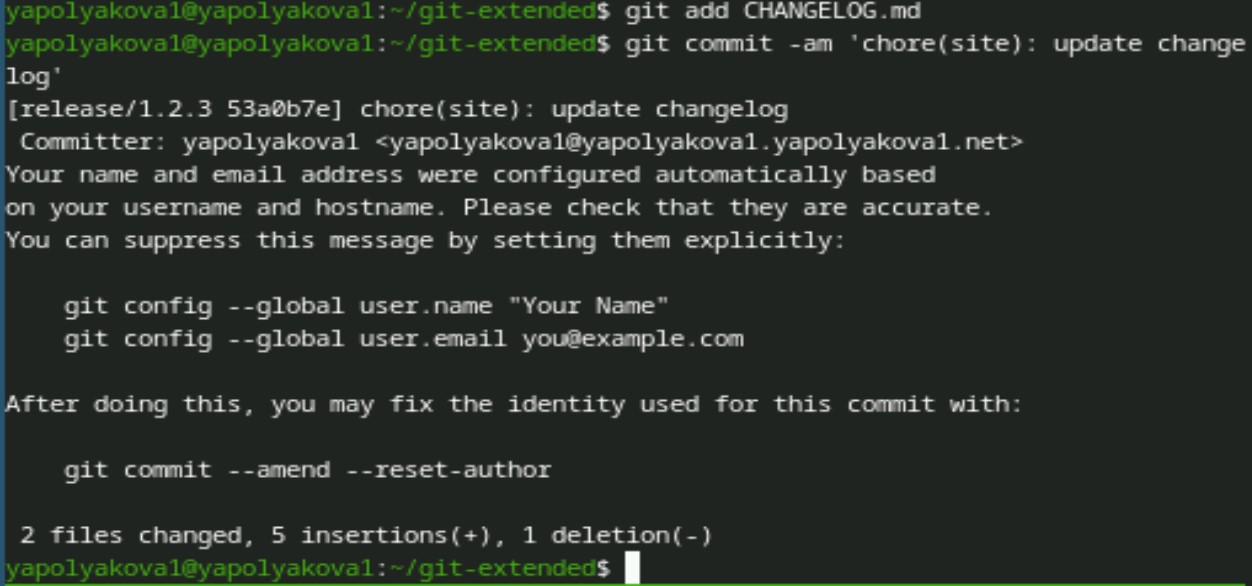


Рис. 17: Создание релиза git-flow 2-ая ч.

1. Зальём релизную ветку в основную ветку (рис. 18)

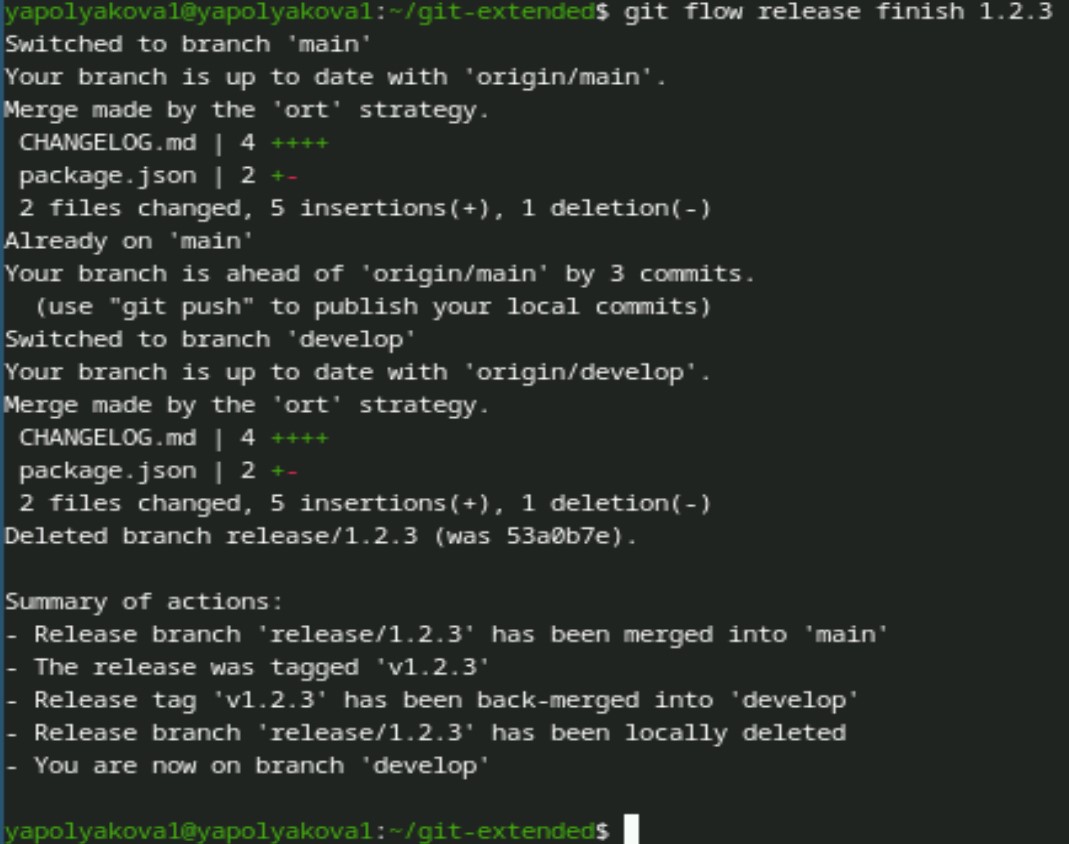


Рис. 18: Создание релиза git-flow 3-я ч.

1. Отправим данные на github и создадим релиз на github с комментарием из журнала изменений (рис. 19)

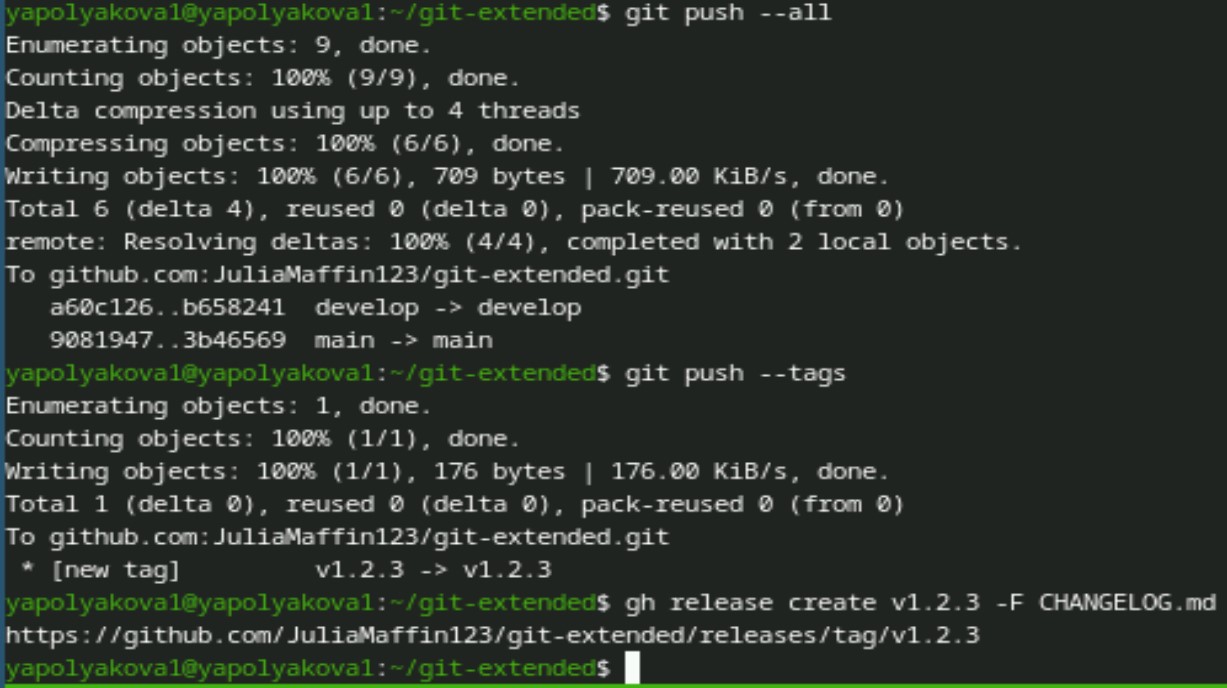


Рис. 19: Создание релиза git-flow 4-ая ч.

# 4 Вывод

Были получены навыки правильной работы с репозиториями git