Отчёт по лабораторной работе №4

Отчет

Кичигина Полина Евгеньевна

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

Установка из коллекции репозиториев Copr (рис. 1)

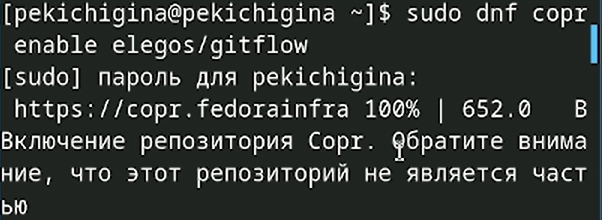


Рис. 1: Устанавливаем git-flow

1. Установка Node.js

На Node.js базируется программное обеспечение для семантического версионирования и общепринятых коммитов (рис. 2)

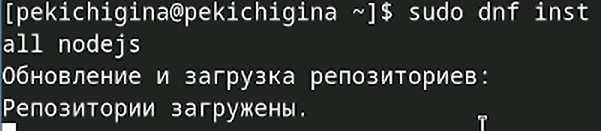


Рис. 2: Устанавливаем Node.js

1. Настройка Node.js

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



Рис. 3: Настраиваем Node.js

1. Общепринятые коммиты

Данная программа используется для помощи в форматировании коммитов (рис. 4)

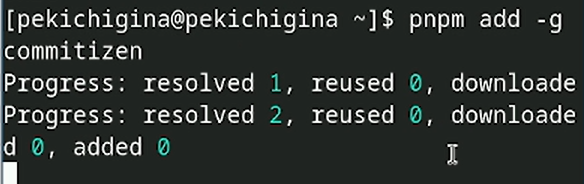


Рис. 4: commitizen

Данная программа используется для помощи в создании логов (рис. 5)

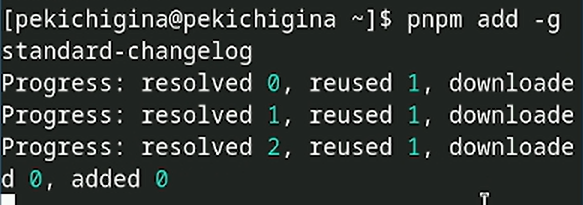


Рис. 5: standard-changelog

1. Практический сценарий использования git

Создайте репозиторий на GitHub. Для примера назовём его git-extended (рис. 6)

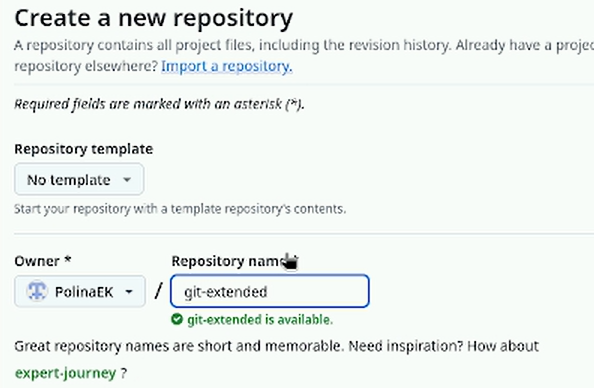


Рис. 6: Создаем репозиторий

Делаем первый коммит и выкладываем на github (рис. 7)

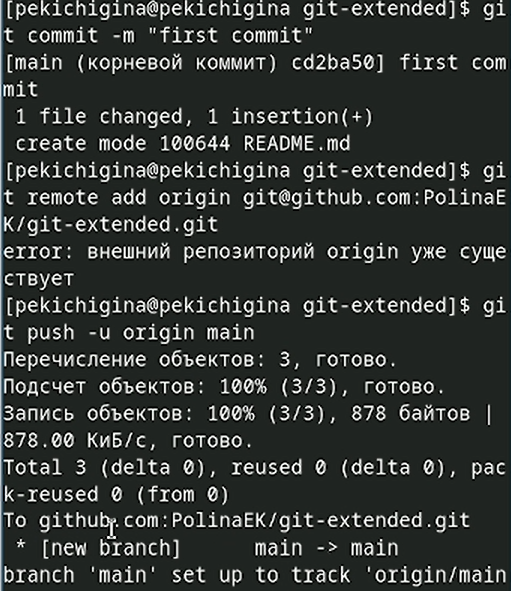


Рис. 7: Коммитим и выкладываем

1. Конфигурация общепринятых коммитов

Необходимо заполнить несколько параметров пакета.

Название пакета.  
Лицензия пакета. Список лицензий для npm: https://spdx.org/licenses/.   
Предлагается выбирать лицензию CC-BY-4.0.

Сконфигурим формат коммитов. Для этого добавим в файл package.json команду для формирования коммитов (рис. 8)

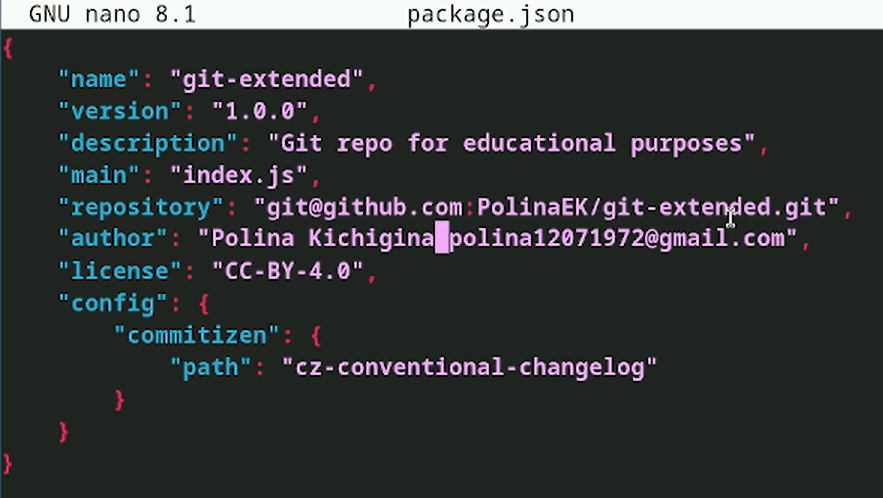


Рис. 8: Редактируем файл

Добавим новые файлы, выполним коммит и отправим на github (рис. 9)



Рис. 9: Редактируем файл, коммитим и топравляем

1. Конфигурация git-flow

Инициализируем git-flow. Префикс для ярлыков установим в v. Проверьте, что Вы на ветке develop(рис. 10)

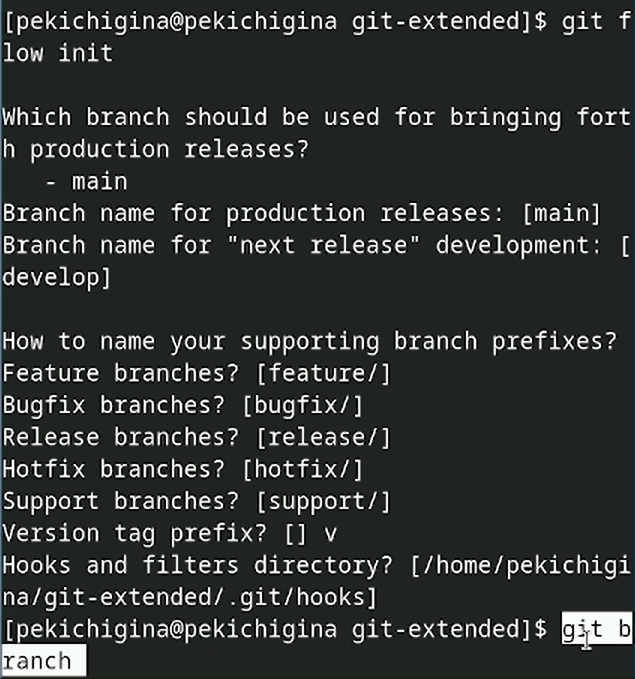


Рис. 10: Инициализируем

Загрузите весь репозиторий в хранилище и установите внешнюю ветку как вышестоящую для этой ветки(рис. 11)



Рис. 11: Загружаем и устанавливаем ветку

Создадим релиз с версией 1.0.0(рис. 12)



Рис. 12: Создаем релиз

Создадим журнал изменений и добавим журнал изменений в индекс(рис. 13)

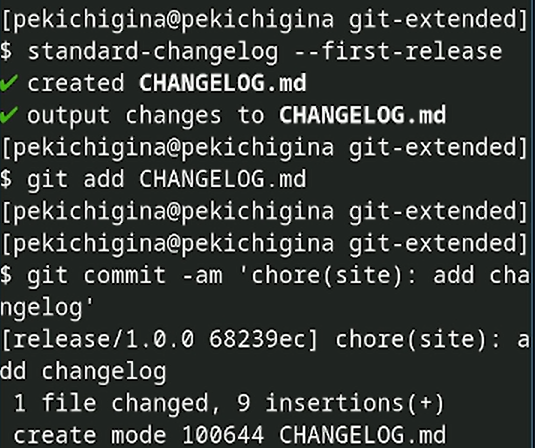


Рис. 13: Создаем журнал

Зальём релизную ветку в основную ветку, отправим данные на github и создадим релиз на github. Для этого будем использовать утилиты работы с github (рис. 14)

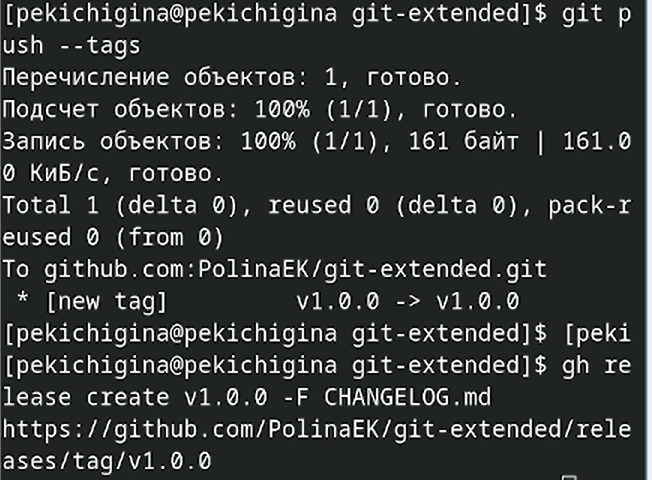


Рис. 14: Отправляем на github

1. Работа с репозиторием git

Создадим ветку для новой функциональности. Далее, продолжаем работу c git как обычно.По окончании разработки новой функциональности следующим шагом следует объединить ветку feature\_branch c develop (рис. 15)

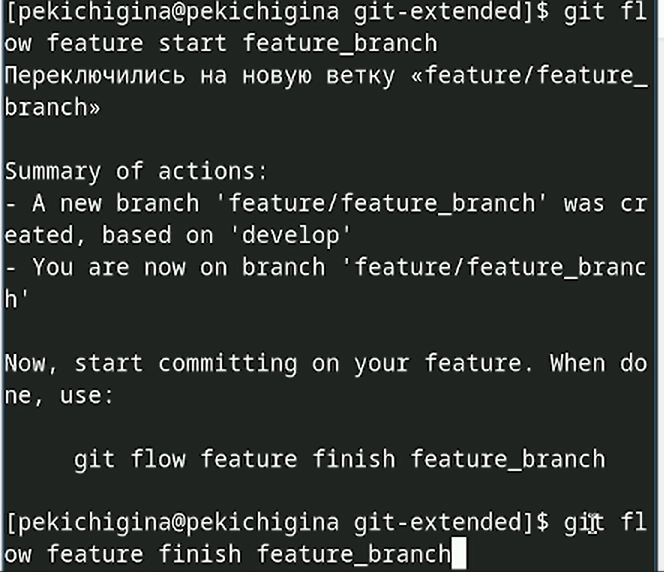


Рис. 15: Разрабатываем новую функциональность и объединяем ветки

1. Создание релиза git-flow

Создадим релиз с версией 1.2.3. Обновите номер версии в файле package.json. Установите её в 1.2.3 (рис. 16)

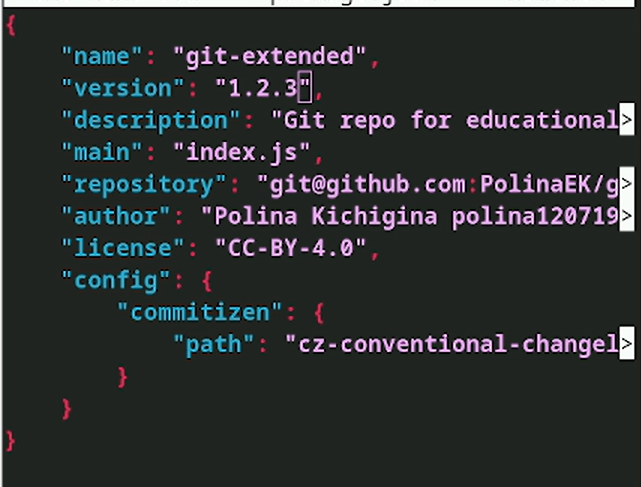


Рис. 16: Создаем релиз

Создадим журнал изменений и добавим журнал изменений в индекс (рис. 17)

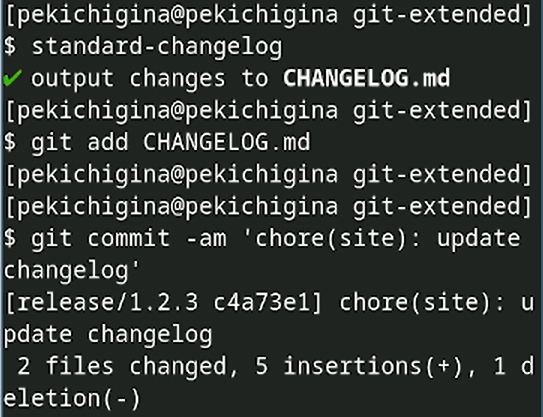


Рис. 17: Создаем и добавляем

Создадим релиз на github с комментарием из журнала изменений (рис. 18)



Рис. 18: Создаем

# 4 Выводы

Мы получили навыки правильной работы с репозиториями git.