Отчет по лабораторной работе №4

Операционные системы

Гашимова Эсма Эльшан кызы

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 Теоретическое введение

## 3.1 Рабочий процесс Gitflow

Рабочий процесс Gitflow Workflow. Будем описывать его с использованием пакета git-flow. 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. Последовательность действий при работе по модели Gitflow: Из ветки master создаётся ветка develop. Из ветки develop создаётся ветка release. Из ветки develop создаются ветки feature. Когда работа над веткой feature завершена, она сливается с веткой develop. Когда работа над веткой релиза release завершена, она сливается в ветки develop и master. Если в master обнаружена проблема, из master создаётся ветка hotfix. Когда работа над веткой исправления hotfix завершена, она сливается в ветки develop и master.

# 4 Выполнение лабораторной работы

## 4.1 Установка программного обеспечения.

### 4.1.1 Установка git-flow

1. Открываем терминал и входим в режим суперпользователя, устанавливаем gitflow. (рис. 1).

Установка gitflow в режиме суперпольщователя

Рис. 1: Установка gitflow в режиме суперпольщователя

### 4.1.2 Установка и настройка node.js

1. Устанавливаем pnpm и nodejs. (рис. 2).

Установка приложений

Рис. 2: Установка приложений

1. Настраиваем nodejs. (рис. 3).

Настройка node.js

Рис. 3: Настройка node.js

### 4.1.3 Общепринятые коммиты.

1. Настраиваем commitizen и standard-changelog (рис. 4).

Настройка программ

Рис. 4: Настройка программ

## 4.2 Практический сценарий использования git.

### 4.2.1 Создание репозитория git.

1. Создаем репозиторий git, настраиваем его и делаем в него первый коммит. (рис. 5).

Создание репозитория, первый коммит

Рис. 5: Создание репозитория, первый коммит

1. Настраиваем пакет файлов nodejs (рис. 6). В файле package.json меняем необходимые данные. (рис. 7).

Настройка пакета

Рис. 6: Настройка пакета

Изменения файла

Рис. 7: Изменения файла

1. Выполняем коммит (рис. 8). Выкладываем на github. (рис. 9).

Выполнение коммита

Рис. 8: Выполнение коммита

Команда push

Рис. 9: Команда push

1. Инициализируем gitflow, проверяем, на какой ветке мы находимся в данный момент, после чего загружаем весь репозиторий в хранилище. (рис. 10).

Инициализация gitflow

Рис. 10: Инициализация gitflow

1. Устанавливаем внешнюю ветку как вышестоящую для этой ветки (рис. 11). Создаем релиз с версией 1.0.0 и журнал изменений. (рис. 12).

Установка внешней ветки

Рис. 11: Установка внешней ветки

Создание релиза и журнала изменений

Рис. 12: Создание релиза и журнала изменений

1. Заливаем релизную ветку в основную, добавляем журнал изменений в индекс, после чего заливаем релизную ветку в основную.
2. Отправляем данные и теги на гитхаб. (рис. 13).

Команды push –all и push –tags

Рис. 13: Команды push –all и push –tags

### 4.2.2 Работа с репозиторием git.

1. Создаем релиз на гитхабе. Создаем ветку для новой функциональности. (рис. 14).

Создание новой ветки

Рис. 14: Создание новой ветки

1. Объединяем новую ветку с develop. Создаём релиз с версией 1.2.3. (рис. 15).

Объединение веток, создание релиза с более новой версией

Рис. 15: Объединение веток, создание релиза с более новой версией

1. Изменяем номер версии в файле package.json.
2. Заливаем релизную ветку в основную . Отправляем данные на гитхаб.
3. Создаём релиз на гитхабе с комментарием из журнала изменений.

# 5 Выводы

В процессе выполнения лабораторной работы я приобрел навыки правильной работы с репозиториями git.

# Список литературы