Лабораторная работа №2

Дисциплина: Операционные системы

Савостин Олег

Содержание

Список иллюстраций

Список таблиц

# 1 Цель работы

Получить навыки правильной работы с репозиториями git.

# 2 Задание

1. Установка программного обеспечения
2. Практический сценарий использования git

# 3 Теоретическое введение

Основные ветки (master) и ветки разработки (develop)  
 Для фиксации истории проекта в рамках этого процесса вместо одной ветки master используются две ветки. В ветке master хранится официальная история релиза, а ветка develop предназначена для объединения всех функций. Кроме того, для удобства рекомендуется присваивать всем коммитам в ветке master номер версии.  
  
 При использовании библиотеки расширений git-flow нужно инициализировать структуру в существующем репозитории:  
  
 git flow init  
  
 Для github параметр Version tag prefix следует установить в v.  
  
 После этого проверьте, на какой ветке Вы находитесь:  
  
 git branch  
  
Функциональные ветки (feature)  
 Под каждую новую функцию должна быть отведена собственная ветка, которую можно отправлять в центральный репозиторий для создания резервной копии или совместной работы команды. Ветки feature создаются не на основе master, а на основе develop. Когда работа над функцией завершается, соответствующая ветка сливается обратно с веткой develop. Функции не следует отправлять напрямую в ветку master.  
 Как правило, ветки feature создаются на основе последней ветки develop.  
  
 Создание функциональной ветки  
  
 Создадим новую функциональную ветку:  
  
 git flow feature start feature\_branch  
  
 Далее работаем как обычно.  
  
 Окончание работы с функциональной веткой  
  
 По завершении работы над функцией следует объединить ветку feature\_branch с develop:  
  
 git flow feature finish feature\_branch  
  
Ветки выпуска (release)  
 Когда в ветке develop оказывается достаточно функций для выпуска, из ветки develop создаётся ветка release. Создание этой ветки запускает следующий цикл выпуска, и с этого момента новые функции добавить больше нельзя — допускается лишь отладка, создание документации и решение других задач. Когда подготовка релиза завершается, ветка release сливается с master и ей присваивается номер версии. После нужно выполнить слияние с веткой develop, в которой с момента создания ветки релиза могли возникнуть изменения.  
 Благодаря тому, что для подготовки выпусков используется специальная ветка, одна команда может дорабатывать текущий выпуск, в то время как другая команда продолжает работу над функциями для следующего.  
  
 Создать новую ветку release можно с помощью следующей команды:  
  
 git flow release start 1.0.0  
  
 Для завершения работы на ветке release используются следующие команды:  
  
 git flow release finish 1.0.0  
  
Ветки исправления (hotfix)  
 Ветки поддержки или ветки hotfix используются для быстрого внесения исправлений в рабочие релизы. Они создаются от ветки master. Это единственная ветка, которая должна быть создана непосредственно от master. Как только исправление завершено, ветку следует объединить с master и develop. Ветка master должна быть помечена обновлённым номером версии.  
 Наличие специальной ветки для исправления ошибок позволяет команде решать проблемы, не прерывая остальную часть рабочего процесса и не ожидая следующего цикла релиза.  
  
 Ветку hotfix можно создать с помощью следующих команд:  
  
 git flow hotfix start hotfix\_branch  
  
 По завершении работы ветка hotfix объединяется с master и develop:  
  
 git flow hotfix finish hotfix\_branch

# 4 Выполнение лабораторной работы

## 4.1 Установка программного обеспечения

Сперва устанавливаю git-flow из коллекции репозиториев Copr (рис. 1).



Рис. 1: Установка git-flow

Устанавливаю Node.js (рис. 2).

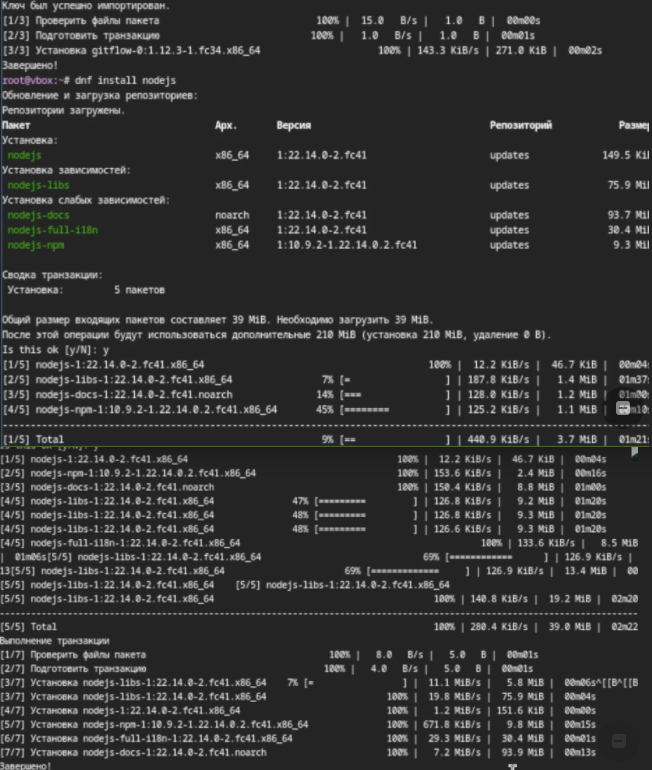


Рис. 2: Установка Node.js

После установки начинаю настраивать программы. Настраиваю Node.js ввожу команду pnpm setup и добавляю каталог с исполняемыми файлами (рис. 3).

Рис. 3: pnpm setup

Рис. 3: pnpm setup

Перелогинусь. (рис. 4).

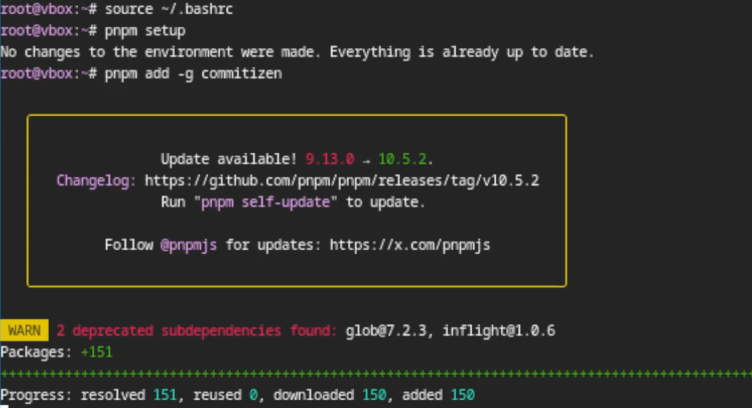


Рис. 4: source ~/.bashrc

Испольщзую общепринятые коммиты (рис. 5).

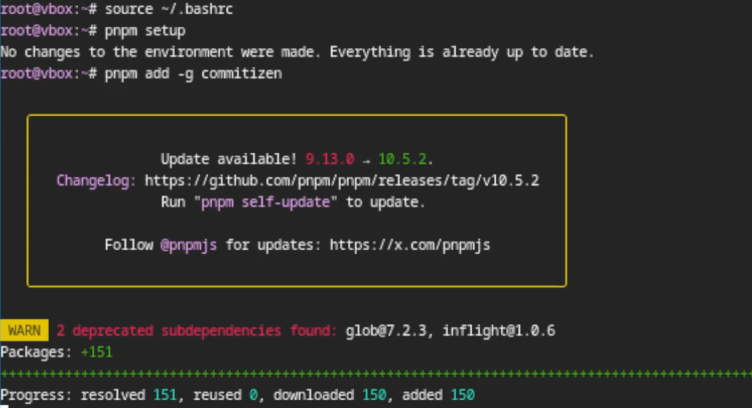


Рис. 5: Общепринятые коммиты

## 4.2 Практический сценарий использования git

Начинаю подключение репозитория к github. Создаю репозиторий на GitHub. Назыываю его git-extended (рис. 6).

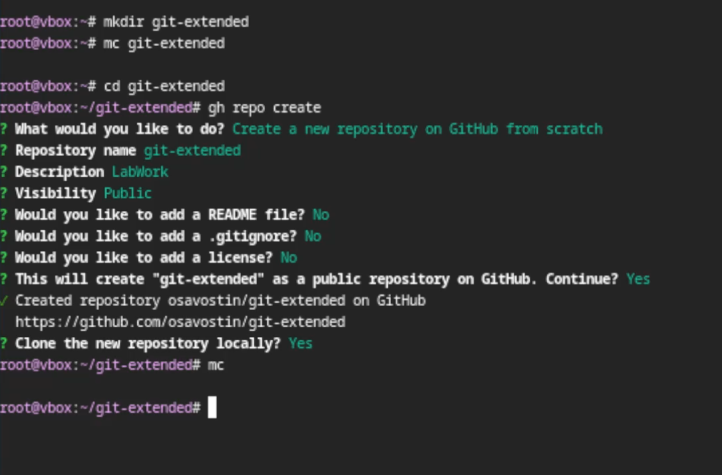


Рис. 6: Новый репозиторий

Делаю первый коммит (рис. 7).

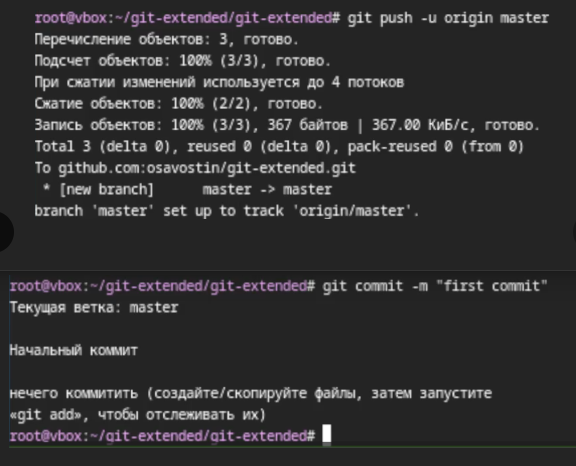


Рис. 7: Первый коммит, выкладываемый на гитхаб

Конфигурирую пакеты Node.js и заполняю несколько параметров пакета. Добавляю в файл package.json команду для формирования коммитов/. Вид package.json (рис. 8).

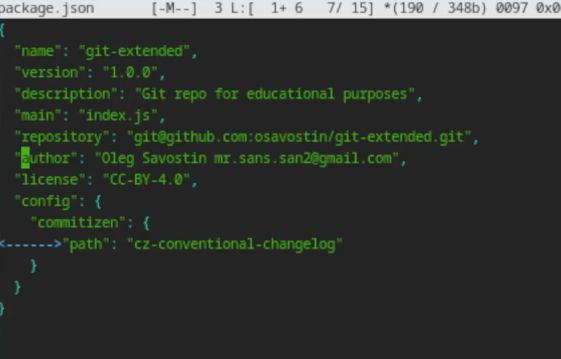


Рис. 8: package.json

Добавляю новые файлы, выполняю коммит cz и отправляю на github (рис. 9).

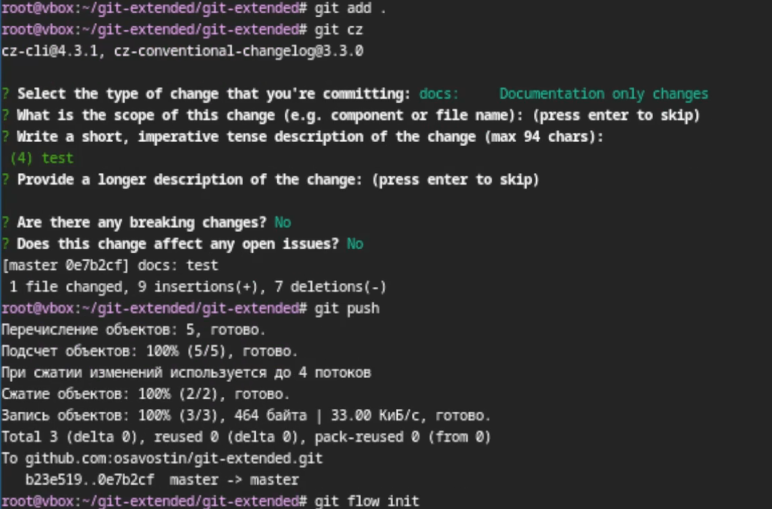


Рис. 9: Отправка на github

Теперь инициализирую git-flow, префикс для ярлыков установляю в v (рис. 10).

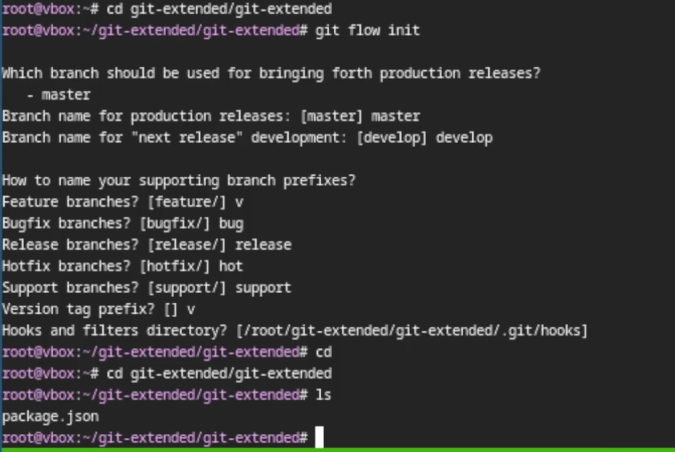


Рис. 10: git-flow

Проверяю, если я на ветке develop и загружаю весь репозиторий в хранилище. Устанавливаю внешнюю ветку как вышестоящую для этой ветки. Создаю релиз с версией 1.0.0 и создаю журнал изменений (рис. 11).

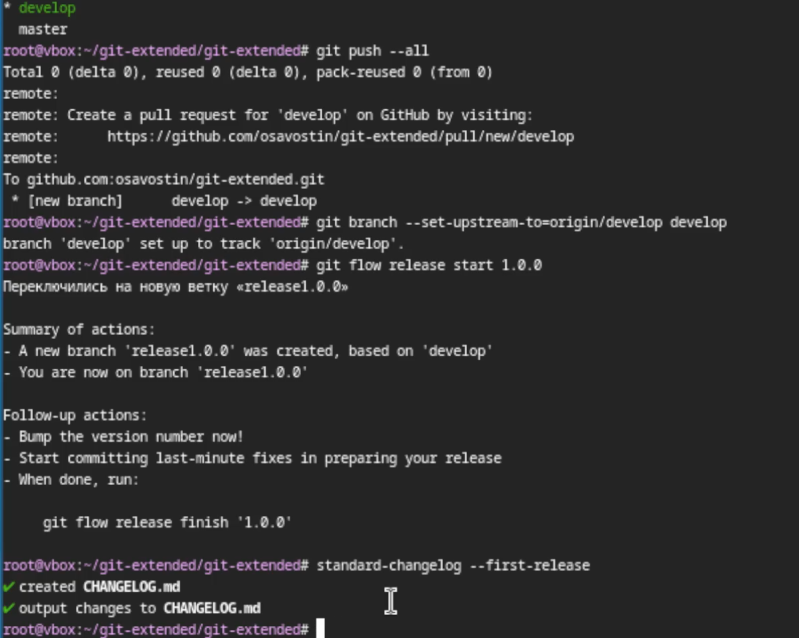


Рис. 11: git push –all

Добавляю журнал изменений в индекс и заливаю релизную ветку в основную ветку (рис. 12).

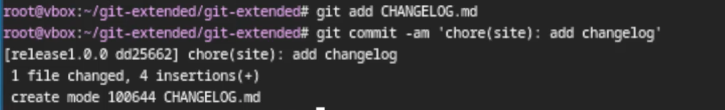


Рис. 12: Changelog

Отправляю все данные на гитхаб и создаю релиз на гитхаб (рис. 13).

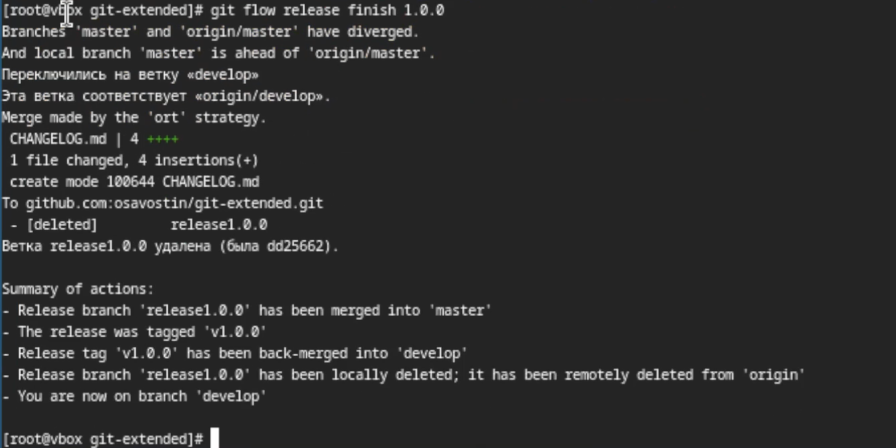


Рис. 13: Релиз на гитхабе

Приступаем ко второму этапу Разработка новой функциональности. Создаю ветку для новой функциональности. Объединяю ветку feature\_branch с develop. Создаю релиз с версией 1.2.3 (рис. 14).

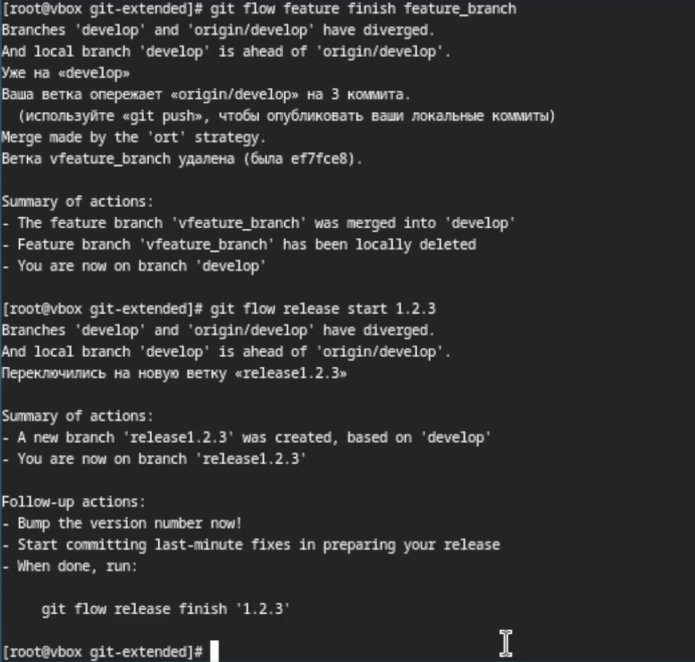


Рис. 14: Версия 1.2.3

Обновляю номер версии в файле package.json и устанавливаю её в 1.2.3 (рис. 15).

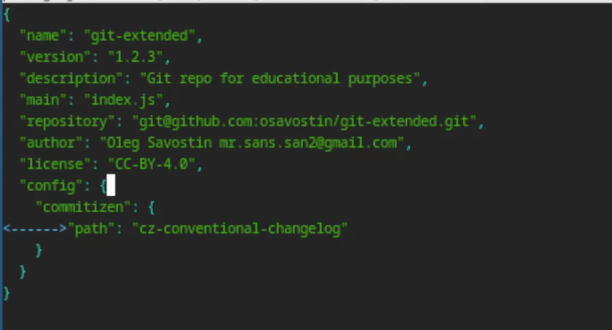


Рис. 15: package.json 1.2.3

Создаю журнал изменений и добавляю его в индекс(рис. 16) (рис. 17)

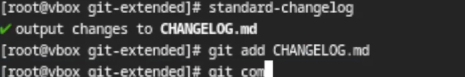


Рис. 16: Новый журнал изменений

Рис. 17: Журнал изменений

Рис. 17: Журнал изменений

Заливаю ветку в основную ветку и отправляю данные на гитхаб, после чего создаю релиз на гитхаб с комментарием из журнала изменений (рис. 18) (рис. 19) (рис. 20) .

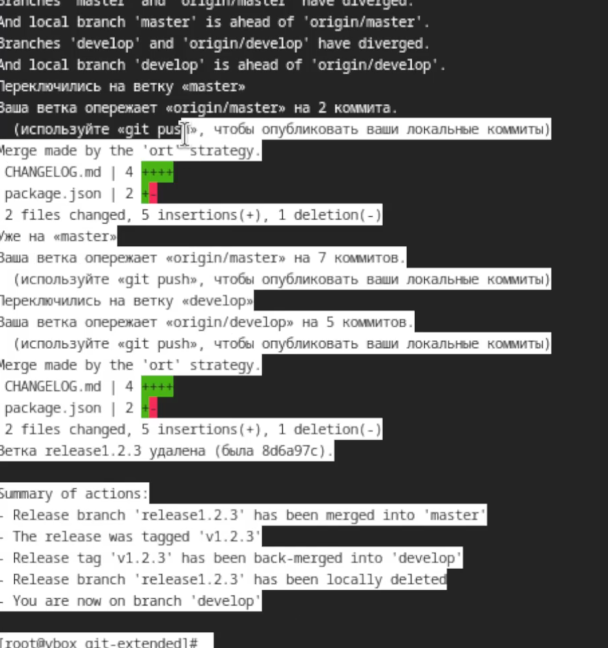


Рис. 18: Слияние новой ветки и релиз на гитхаб

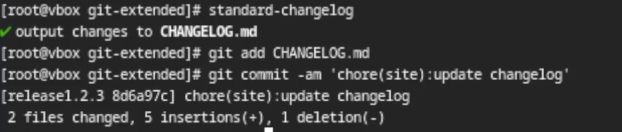


Рис. 19: Слияние новой ветки и релиз на гитхаб

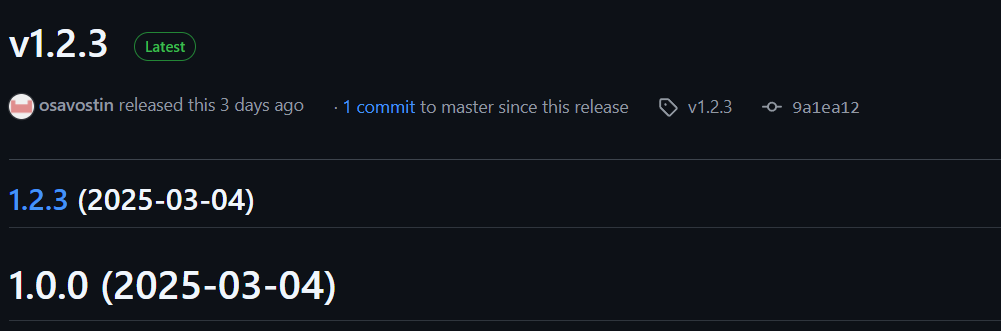


Рис. 20: Версии на гит

# 5 Выводы

Я получил навыки правильной работы с репозиториями git.

# Список литературы

Лабораторная работа номер 4