

Отчёт по лабораторной работе №4

Операционные системы

Еремина Оксана Андреевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выводы	13
	Список литературы	14

Список таблиц

Список иллюстраций

3.1	Установка git-flow	8
3.2	Установка node.js	8
3.3	Настройка node.js	8
3.4	Commitizen	9
3.5	standard-changelog	9
3.6	Создание репозитория	9
3.7	Клонирование репозитория	10
3.8	Создание первого коммита	10
3.9	Конфигурация для пакетов Node.js	10
3.10	Параметры пакета	11
3.11	Работа с git	11
3.12	Инициализация git-flow	11
3.13	Загрузка в хранилище	12
3.14	Установка внешней ветки	12

1 Цель работы

Целью данной лабораторной работы является 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 Теоретическое введение

Рабочий процесс Gitflow Workflow. Будем описывать его с использованием пакета git-flow.

- Gitflow Workflow опубликована и популяризована Винсентом Дриссенем.
- 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.

1. Установка программного обеспечения

- Установка git-flow

Так как у меня ubuntu, то я устанавливаю ПО с помощью другой команды.
(рис.1)

```
oaeremina@oaeremina:~$ sudo apt-get install git-flow
[sudo] пароль для oaeremina:
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Уже установлен пакет git-flow самой новой версии (1.12.3-3).
Следующие пакеты устанавливались автоматически и больше не требуются:
  libappstream-glib8 libflashrom1 libftdi1-2 libfuse2 liblvm2
Для их удаления используйте «sudo apt autoremove».
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 0, и 25 пакетов не обновлено.
```

Рис. 3.1: Установка git-flow

- Установка Node.js

То же самое делаю и для node.js (рис.2)

```
oaeremina@oaeremina:~$ sudo apt install nodejs
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Уже установлен пакет nodejs самой новой версии (12.22.9~dfsg-1ubuntu1)
Следующие пакеты устанавливались автоматически и больше не требуются:
  libappstream-glib8 libflashrom1 libftdi1-2 libfuse2 liblvm2
Для их удаления используйте «sudo apt autoremove».
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 0, и 25 пакетов не обновлено.
oaeremina@oaeremina:~$
```

Рис. 3.2: Установка node.js

2. Настройка Node.js

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

```
oaeremina@oaeremina:~$ export PNPM_HOME="/home/oaeremina/.local/share/pnpm"
oaeremina@oaeremina:~$ export PATH="$PNPM_HOME:$PATH"
oaeremina@oaeremina:~$ echo $PATH
/home/oaeremina/.local/share/pnpm:/home/oaeremina/.local/share/pnpm:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
```

Рис. 3.3: Настройка node.js

3. Общепринятые коммиты

- commitizen

Данная программа используется для помощи в форматировании коммитов. При этом устанавливается скрипт git-cz, который я буду использовать для коммитов. (рис.4)

```
oaoeremina@oaoeremina:~$ npm add -g commitizen
Packages: +1
+
Progress: resolved 208, reused 199, downloaded 0, added 0, done
Done in 8.1s
```

Рис. 3.4: Commitizen

- standard-changelog

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

```
oaoeremina@oaoeremina:~$ npm add -g standard-changelog
Packages: +1
+
Progress: resolved 208, reused 199, downloaded 0, added 0, done
Done in 4.4s
```

Рис. 3.5: standard-changelog

- Практический сценарий использования git

Создаю репозиторий с именем git-extended. (рис.6)

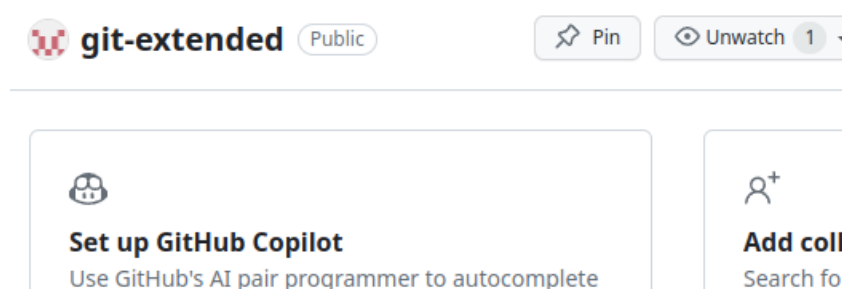


Рис. 3.6: Создание репозитория

Переходу в папку work и клонирую репозиторий в папку (рис.7)

```
oeremina@oeremina:~$ cd work
oeremina@oeremina:~/work$ git clone --recursive git@github.com:oeremina/git-extended.git
Клонирование в «git-extended»...
warning: Похоже, что вы клонировали пустой репозиторий.
```

Рис. 3.7: Клонирование репозитория

Делаю первый коммит и выкладываю на github. (рис.8)

```
oeremina@oeremina:~/work/git-extended$ echo "# git-extended" >> README.md
oeremina@oeremina:~/work/git-extended$ git init
Периинициализирован существующий репозиторий Git в /home/oeremina/work/git-extended/.
git/
oeremina@oeremina:~/work/git-extended$ git add README.md
oeremina@oeremina:~/work/git-extended$ git commit -m "first commit"
[main (корневой коммит) d2573ae] first commit
1 file changed, 1 insertion(+)
create mode 100644 README.md
oeremina@oeremina:~/work/git-extended$ git branch -M main
oeremina@oeremina:~/work/git-extended$ git remote add origin git@github.com:oeremina/git-extended.git
error: внешний репозиторий origin уже существует
oeremina@oeremina:~/work/git-extended$ ^[[200~git push -u origin master
git: команда не найдена
oeremina@oeremina:~/work/git-extended$ git push -u origin main
Перечисление объектов: 3, готово.
Подсчет объектов: 100% (3/3), готово.
Запись объектов: 100% (3/3), 883 байта | 883.00 КиБ/с, готово.
Всего 3 (изменений 0), повторно использовано 0 (изменений 0), повторно использовано па
кетов 0
To github.com:oeremina/git-extended.git
 * [new branch]      main -> main
Ветка «main» отслеживает внешнюю ветку «main» из «origin».
```

Рис. 3.8: Создание первого коммита

Конфигурация для пакетов Node.js с помощью команды pnpm init (рис.9)

```
oeremina@oeremina:~/work/git-extended$ pnpm init
Wrote to /home/oeremina/work/git-extended/package.json

{
  "name": "git-extended",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

Рис. 3.9: Конфигурация для пакетов Node.js

Заполняю несколько параметров пакеты (рис.10)

```

1 {
2   "name": "git-extended",
3   "version": "1.0.0",
4   "description": "Git repo for educational purposes",
5   "main": "index.js",
6   "repository": "git@github.com:username/git-extended.git",
7   "author": "Name Surname <username@gmail.com>",
8   "license": "CC-BY-4.0",
9   "config": {
10     "commitizen": {
11       "path": "cz-conventional-changelog"
12     }
13   }
14 }

```

Рис. 3.10: Параметры пакета

Добавляю новые файлы, выполняю коммит и отправляю на github (рис.11)

```

oaeremina@oaeremina:~/work/git-extended$ git add .
oaeremina@oaeremina:~/work/git-extended$ git cz
cz-cli@4.3.0, cz-conventional-changelog@3.3.0

? Select the type of change that you're committing: feat:      A new feature
? What is the scope of this change (e.g. component or file name): (press enter to skip)

```

Рис. 3.11: Работа с git

Инициализирую git-flow (рис.12)

```

oaeremina@oaeremina:~/work/git-extended$ git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main] main
Branch name for "next release" development: [develop] develop

How to name your supporting branch prefixes?
Feature branches? [feature/] feature/
Bugfix branches? [bugfix/] bugfix/
Release branches? [release/] release/
Hotfix branches? [hotfix/] hotfix/
Support branches? [support/] support/
Version tag prefix? []
Hooks and filters directory? [/home/oaeremina/work/git-extended/.git/hooks]

```

Рис. 3.12: Инициализация git-flow

Проверяю, что нахожусь на ветке develop и загружаю весь репозиторий в хранилище (рис.13)

```

oaeremina@oaeremina:~/work/git-extended$ git branch
* develop
  main
oaeremina@oaeremina:~/work/git-extended$ git push --all
Всего 0 (изменений 0), повторно использовано 0 (изменений 0), повторно использо-
вано пакетов 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/oaeremina/git-extended/pull/new/develop
remote:
To github.com:oaeremina/git-extended.git
 * [new branch]      develop -> develop

```

Рис. 3.13: Загрузка в хранилище

Устанавливаю внешнюю ветку как вышестоящую для этой ветки (рис.14)

```

oaeremina@oaeremina:~/work/git-extended$ git branch --set-upstream-to=origin/de-
velop develop
Ветка «develop» отслеживает внешнюю ветку «develop» из «origin».
oaeremina@oaeremina:~/work/git-extended$ git flow release start 1.0.0
Переключились на новую ветку «release/1.0.0»

Summary of actions:
- A new branch 'release/1.0.0' was created, based on 'develop'
- You are now on branch 'release/1.0.0'

Follow-up actions:
- Bump the version number now!
- Start committing last-minute fixes in preparing your release
- When done, run:

    git flow release finish '1.0.0'

oaeremina@oaeremina:~/work/git-extended$ standard-changelog --first-release

```

Рис. 3.14: Установка внешней ветки

4 Выводы

При выполнении данной лабораторной работы я получила практические навыки правильной работы с репозиториями github

Список литературы

<https://esystem.rudn.ru/mod/page/view.php?id=1098794#org363eddd>