Лабораторная работа №4

Продвинутое использование git

Приспешкин Андрей Андреевич

Содержание

1 Цель работы **5**

2 Задание **6**

3 Выполнение лабораторной работы **7**

4 Выводы **15**

Список иллюстраций

3.1 Установка gitflow пакетным менеджером paru 7
3.2 Установка node.js 7
3.3 Установка pnpm 7
3.4 Добавление каталога с исполняемыми файлами 8
3.5 Установка программы commitizen 8
3.6 Установка программы standard-changelog 8
3.7 Создание нового репозитория 9
3.8 Выкладывание первого коммита на Github 10
3.9 Конфигурация для пакетов Node.js 10
3.10 Добавление команды для формирования коммитов 10
3.11 Выполнение коммита командой git cz 10
3.12 Конфигурация gitflow через git flow init 11
3.13 Проверка текущей ветки с помощью git branch 11
3.14 Установка внешней ветки и создание релиза 11
3.15 Создание журнала изменений и добавление его в индекс 11
3.16 Заливаем релизную ветку в основную 11
3.17 Создание релиза на github 12
3.18 Создание и объединение ветки для новой функциональности . . 12
3.19 Создание релиза с версией 1.2.3 12
3.20 Обновление номера нашего релиза 13
3.21 Создание журнала изменений и внесение этого журнала в индекс 13
3.22 Объединение релизной ветки и основной 13
3.23 Создание релиза на github утилитой gh 14

Список таблиц

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

Установим программное обеспечние gitflow node.js и pnpm(Рис.1, 2 и 3)

```
aaprispeshkin:[aaprispeshkin]:~$ paru gitflow
```

Рис. 3.1: Установка gitflow пакетным менеджером paru

```
aaprispeshkin:[aaprispeshkin]:~$ sudo pacman -S nodejs
warning: nodejs-21.6.2-1 is up to date -- reinstalling
resolving dependencies...
looking for conflicting packages...

Packages (1) nodejs-21.6.2-1

Total Installed Size:  47,95 MiB
Net Upgrade Size:       0,00 MiB
```

Рис. 3.2: Установка node.js

```
aaprispeshkin:[aaprispeshkin]:~$ sudo pacman -S pnpm
resolving dependencies...
looking for conflicting packages...

Packages (1) pnpm-8.15.2-2

Total Download Size:   1,26 MiB
Total Installed Size:  8,91 MiB
```

Рис. 3.3: Установка pnpm

Добавим каталог с исполняемыми файлами через pnpm setup и выполним команду source в каталоге bashrc(Рис.4)

```
aaprispeshkin:[aaprispeshkin]:~$ pnpm setup
Appended new lines to /home/aaprispeshkin/.bashrc

Next configuration changes were made:
export PNPM_HOME="/home/aaprispeshkin/.local/share/pnpm"
case ":$PATH:" in
  *":$PNPM_HOME:"*) ;;
  *) export PATH="$PNPM_HOME:$PATH" ;;
esac

To start using pnpm, run:
source /home/aaprispeshkin/.bashrc
aaprispeshkin:[aaprispeshkin]:~$ source ~/.bashrc
```

Рис. 3.4: Добавление каталога с исполняемыми файлами

Установим программы commitizen и standard-changelog через pnpm(Рис.5 и 6)

```
aaprispeshkin:[aaprispeshkin]:~$ pnpm add - g commitizen
```

Рис. 3.5: Установка программы commitizen

```
aaprispeshkin:[aaprispeshkin]:~$ pnpm add -g standard-changelog
```

Рис. 3.6: Установка программы standard-changelog

Создадим новый репозиторий на Github и назовём его git-extended(Рис.7)

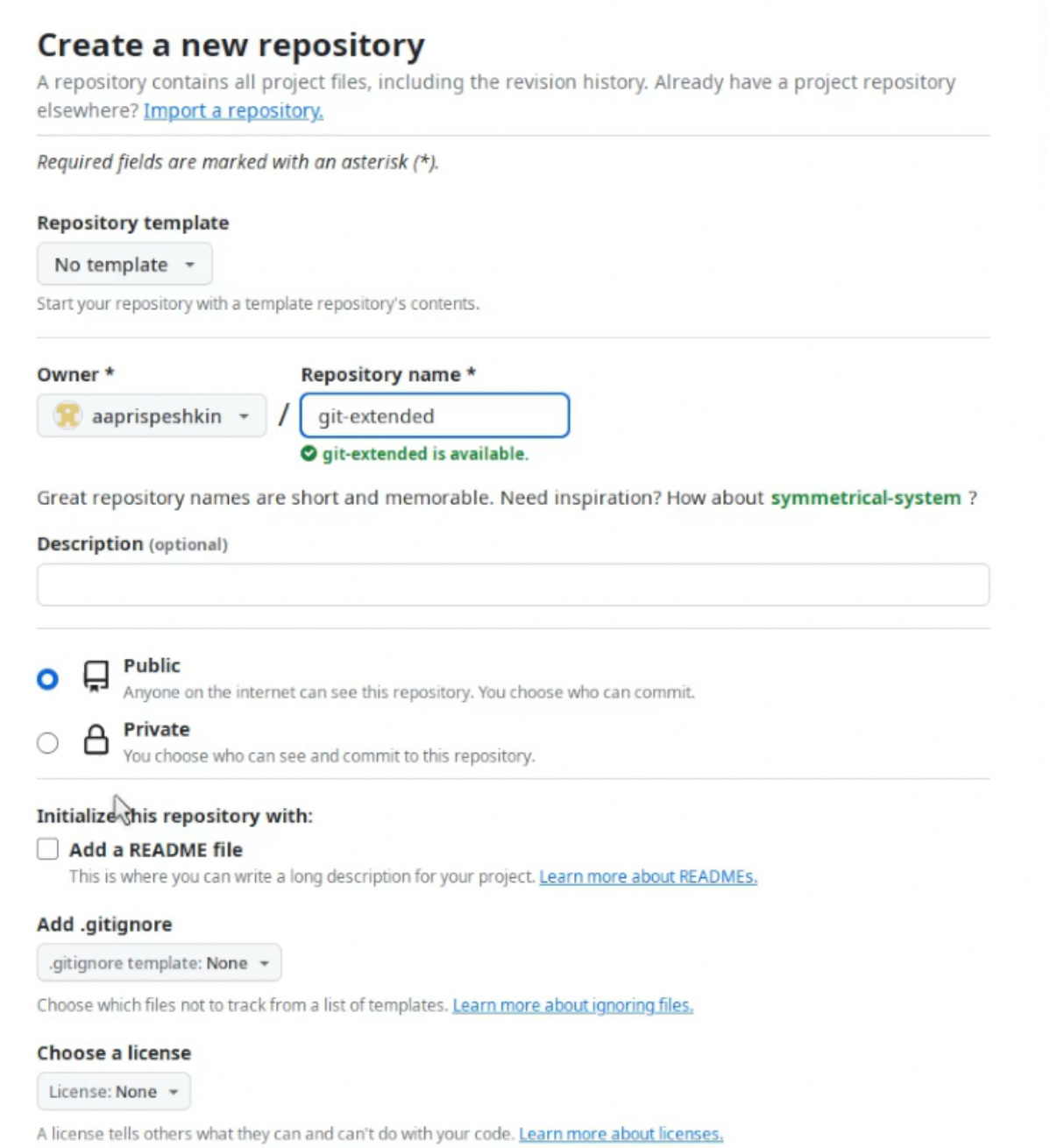

Рис. 3.7: Создание нового репозитория

Сделаем первый коммит и выложим его на Github(Рис.8)

```
aaprispeshkin:[aaprispeshkin]:~/git-extended$ echo "# git-extended" >> README.md
aaprispeshkin:[aaprispeshkin]:~/git-extended$ git init
Initialized empty Git repository in /home/aaprispeshkin/git-extended/.git/
aaprispeshkin:[aaprispeshkin]:~/git-extended$ git add README.md
aaprispeshkin:[aaprispeshkin]:~/git-extended$ git commit -m "first commit"
[master (root-commit) 33c9495] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
aaprispeshkin:[aaprispeshkin]:~/git-extended$ git remote add origin git@github.com:aaprispeshkin/git-extended.git
aaprispeshkin:[aaprispeshkin]:~/git-extended$ git push -u origin master
```

Рис. 3.8: Выкладывание первого коммита на Github

Сконфигурируем пакеты для Node.js через команду pnpm init(Рис.9)

```
aaprispeshkin:[aaprispeshkin]:~/git-extended$ pnpm init
```

Рис. 3.9: Конфигурация для пакетов Node.js

Добавим в package.json команду для формирования коммитов(Рис.10)

```
  "config": {
    "commitizen": {
      "path": "cz-conventional-changelog"
    }
  }
```

Рис. 3.10: Добавление команды для формирования коммитов

Выполним коммит командой git cz(Рис.11)

```
aaprispeshkin:[aaprispeshkin]:~/git-extended$ git cz
cz-cli@4.3.0, cz-conventional-changelog@3.3.0

? Select the type of change that you're committing: docs:     Documentation only changes
? What is the scope of this change (e.g. component or file name): (press enter to skip)
? Write a short, imperative tense description of the change (max 94 chars):
 (18) added package.json
? Provide a longer description of the change: (press enter to skip)

? Are there any breaking changes? No
? Does this change affect any open issues? (y/N)
```

Рис. 3.11: Выполнение коммита командой git cz

Начнём конфигурацию gitflow(Рис.12)

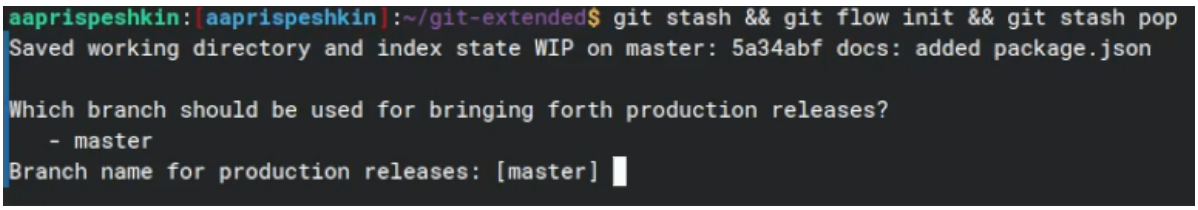

Рис. 3.12: Конфигурация gitflow через git flow init

Через git branch проверим что мы находимся на ветке develop(Рис.13)

Рис. 3.13: Проверка текущей ветки с помощью git branch

Установим внешнюю ветку как вышестоящую и создадим релиз с версией 1.0.0(Рис.14)

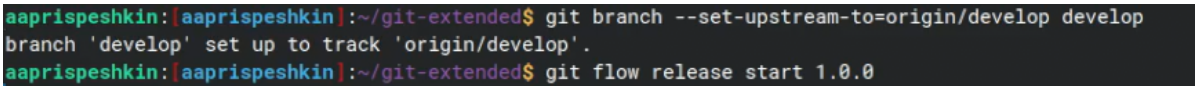

Рис. 3.14: Установка внешней ветки и создание релиза

Создадим журнал изменений и добавим его в индекс(Рис.15)

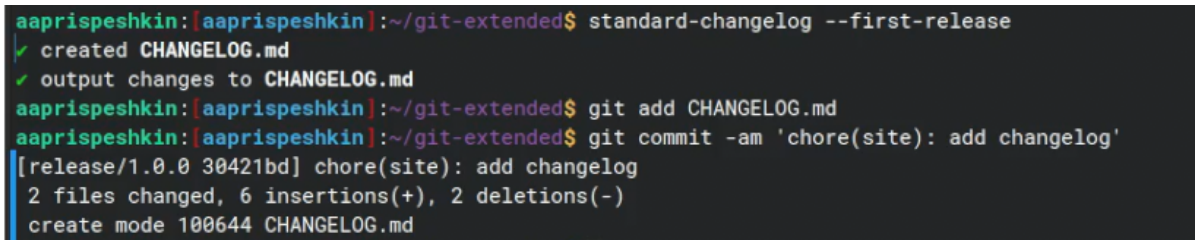

Рис. 3.15: Создание журнала изменений и добавление его в индекс

Зальём релизную ветку в основную ветку через git flow release finish и номер нашего релиза(Рис.16)

Рис. 3.16: Заливаем релизную ветку в основную

Создадим релиз на github утилитой gh(Рис.17)

```
aaprispeshkin:[aaprispeshkin]:~/git-extended$ gh release create v1.0.0 -F CHANGELOG.md
https://github.com/aaprispeshkin/git-extended/releases/tag/v1.0.0
```

Рис. 3.17: Создание релиза на github

Создадим ветку для новой функциональности и объединим её с веткой develop(Рис.18)

```
aaprispeshkin:[aaprispeshkin]:~/git-extended$ git stash && git flow feature start feature_branch && git stash pop
No local changes to save
Switched to a new branch 'feature/feature_branch'

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

Now, start committing on your feature. When done, use:

     git flow feature finish feature_branch

No stash entries found.
aaprispeshkin:[aaprispeshkin]:~/git-extended$ git stash && git flow feature finish feature_branch && git stash pop
```

Рис. 3.18: Создание и объединение ветки для новой функциональности

Создадим релиз с версией 1.2.3(Рис.19)

```
aaprispeshkin:[aaprispeshkin]:~/git-extended$ git stash && git flow release start 1.2.3 && git stash pop
No local changes to save
Switched to a new branch 'release/1.2.3'

Summary of actions:
- A new branch 'release/1.2.3' was created, based on 'develop'
- You are now on branch 'release/1.2.3'

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

     git flow release finish '1.2.3'

No stash entries found.
```

Рис. 3.19: Создание релиза с версией 1.2.3

Обновим номер нашего релиза в package.json(Рис.20)

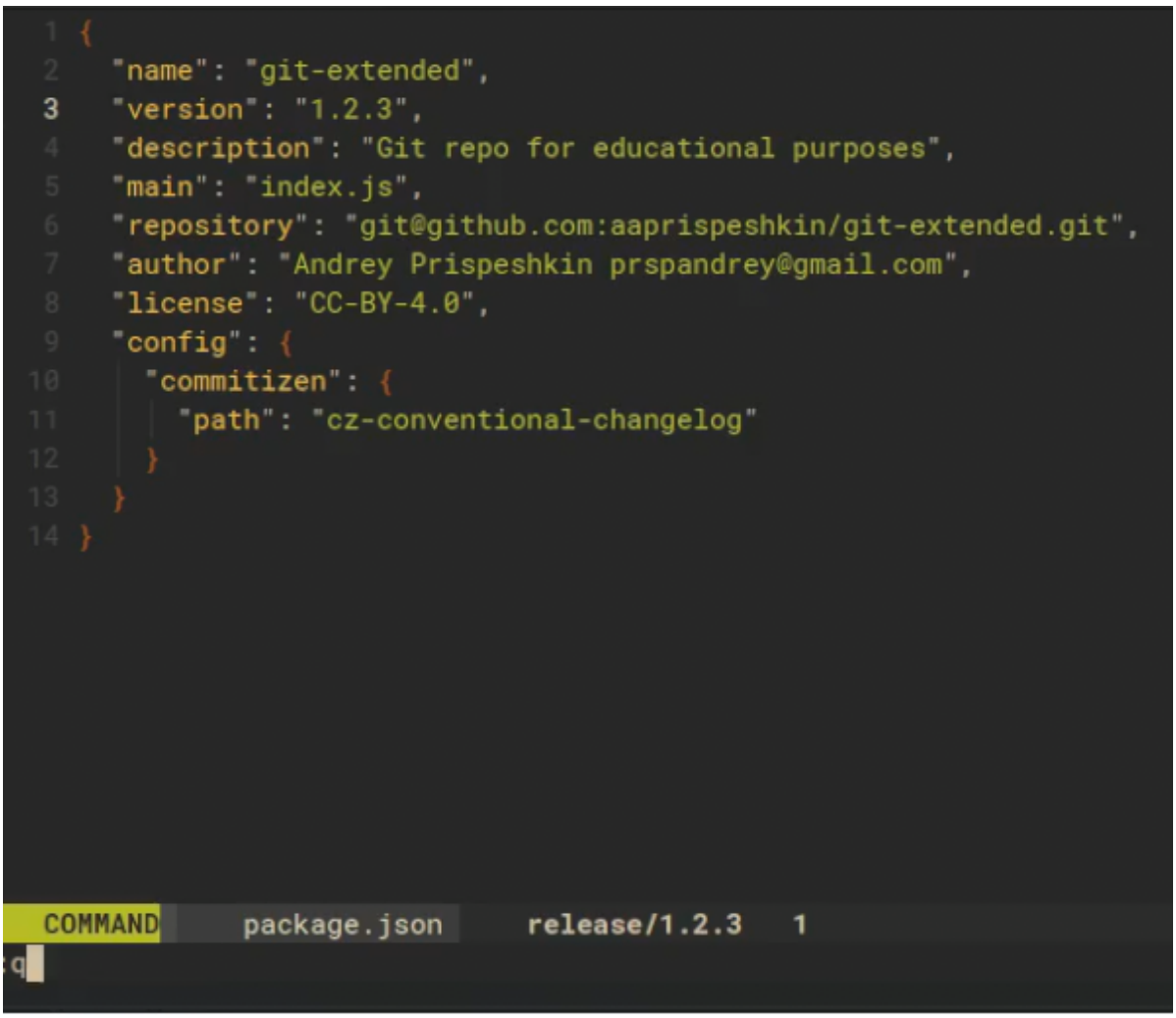

Рис. 3.20: Обновление номера нашего релиза

Создадим журнал изменений и внесём его в индекс(Рис.21)

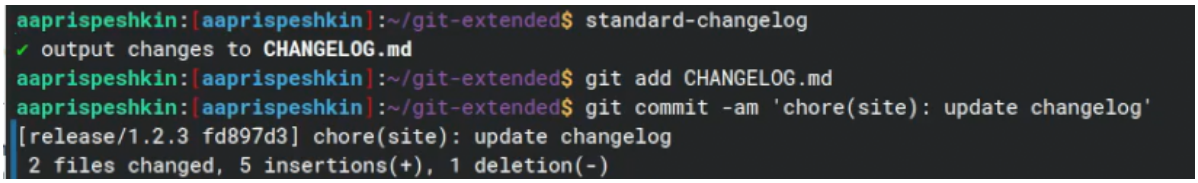

Рис. 3.21: Создание журнала изменений и внесение этого журнала в индекс

Зальём релизную ветку в основную ветку(Рис.22)

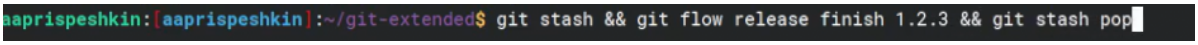

Рис. 3.22: Объединение релизной ветки и основной

Создадим релиз на github с комментарием из журнала изменений(Рис.23)

Рис. 3.23: Создание релиза на github утилитой gh

4 Выводы

Я получил навыки правильной работы с репозиториями git.