

Отчёт по лабораторной работе №4

Отчет

Кичигина Полина Евгеньевна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	18

Список иллюстраций

3.1	Устанавливаем git-flow	7
3.2	Устанавливаем Node.js	7
3.3	Настраиваем Node.js	8
3.4	commitizen	8
3.5	standard-changelog	8
3.6	Создаем репозиторий	9
3.7	Коммитим и выкладываем	10
3.8	Редактируем файл	11
3.9	Редактируем файл, коммитим и топуем	11
3.10	Инициализируем	12
3.11	Загружаем и устанавливаем ветку	13
3.12	Создаем релиз	13
3.13	Создаем журнал	14
3.14	Отправляем на github	14
3.15	Разрабатываем новую функциональность и объединяем ветки	15
3.16	Создаем релиз	16
3.17	Создаем и добавляем	16
3.18	Создаем	17

Список таблиц

1 Цель работы

Получение навыков правильной работы с репозиториями git.

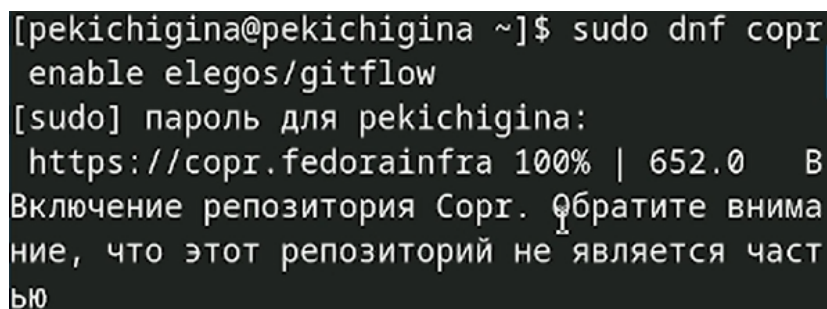
2 Задание

Выполнить работу для тестового репозитория. Преобразовать рабочий репозиторий в репозиторий с git-flow и conventional commits.

3 Выполнение лабораторной работы

1. Установка git-flow

Установка из коллекции репозитория Copr (рис. 3.1)

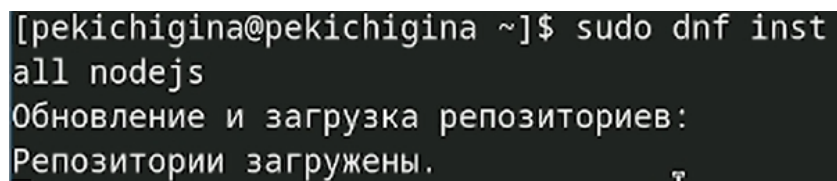


```
[pekichigina@pekichigina ~]$ sudo dnf copr
enable elegos/gitflow
[sudo] пароль для pekichigina:
https://copr.fedorainfra 100% | 652.0 В
Включение репозитория Copr. Обратите внима
ние, что этот репозиторий не является част
ью
```

Рис. 3.1: Устанавливаем git-flow

2. Установка Node.js

На Node.js базируется программное обеспечение для семантического версионирования и общепринятых коммитов (рис. 3.2)



```
[pekichigina@pekichigina ~]$ sudo dnf inst
all nodejs
Обновление и загрузка репозитория:
Репозитории загружены.
```

Рис. 3.2: Устанавливаем Node.js

3. Настройка Node.js

Для работы с Node.js добавим каталог с исполняемыми файлами, устанавливаемыми yarn, в переменную PATH. Перелогиньтесь (рис. 3.3)

```
[pekichigina@pekichigina ~]$ pnpm setup
Appended new lines to /home/pekichigina/.bashrc

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

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

Рис. 3.3: Настраиваем Node.js

4. Общепринятые коммиты

Данная программа используется для помощи в форматировании коммитов (рис. 3.4)

```
[pekichigina@pekichigina ~]$ pnpm add -g
commitizen
Progress: resolved 1, reused 0, downloaded 0, added 0
```

Рис. 3.4: commitizen

Данная программа используется для помощи в создании логов (рис. 3.5)

```
[pekichigina@pekichigina ~]$ pnpm add -g
standard-changelog
Progress: resolved 0, reused 1, downloaded 0, added 0
Progress: resolved 1, reused 1, downloaded 0, added 0
Progress: resolved 2, reused 1, downloaded 0, added 0
```

Рис. 3.5: standard-changelog

5. Практический сценарий использования git

Создайте репозиторий на GitHub. Для примера назовём его git-extended (рис. 3.6)

Рис. 3.6: Создаем репозиторий

Делаем первый коммит и выкладываем на github (рис. 3.7)

```

[pekichigina@pekichigina git-extended]$ git commit -m "first commit"
[main (корневой коммит) cd2ba50] first commit
1 file changed, 1 insertion(+)
create mode 100644 README.md
[pekichigina@pekichigina git-extended]$ git remote add origin git@github.com:PolinaEK/git-extended.git
error: внешний репозиторий origin уже существует
[pekichigina@pekichigina git-extended]$ git push -u origin main
Перечисление объектов: 3, готово.
Подсчет объектов: 100% (3/3), готово.
Запись объектов: 100% (3/3), 878 байтов | 878.00 КиБ/с, готово.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:PolinaEK/git-extended.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'

```

Рис. 3.7: Коммитим и выкладываем

6. 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.

Сконфигурируем формат коммитов. Для этого добавим в файл package.json команду для формирования коммитов (рис. 3.8)

```
GNU nano 8.1 package.json
{
  "name": "git-extended",
  "version": "1.0.0",
  "description": "Git repo for educational purposes",
  "main": "index.js",
  "repository": "git@github.com:PolinaEK/git-extended.git",
  "author": "Polina Kichigina <polina12071972@gmail.com>",
  "license": "CC-BY-4.0",
  "config": {
    "commitizen": {
      "path": "cz-conventional-changelog"
    }
  }
}
```

Рис. 3.8: Редактируем файл

Добавим новые файлы, выполним коммит и отправим на github (рис. 3.9)

```
z
cz-cli@4.3.1, 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) readme.md
? Write a short, imperative tense description
of the change (max 83 chars):
(10) added file
? Provide a longer description of the change:
(press enter to skip)

? Are there any breaking changes? No
? Does this change affect any open issues? No

[main 5056bf3] feat(readme.md): added file
1 file changed, 14 insertions(+)
create mode 100644 package.json
[pekichigina@pekichigina git-extended]$ git p
ush
```

Рис. 3.9: Редактируем файл, коммитим и топравляем

7. Конфигурация git-flow

Инициализируем git-flow. Префикс для ярлыков установим в v. Проверьте, что Вы на ветке develop(рис. 3.10)

```
[pekichigina@pekichigina git-extended]$ git flow init

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

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

Рис. 3.10: Инициализируем

Загрузите весь репозиторий в хранилище и установите внешнюю ветку как вышестоящую для этой ветки(рис. 3.11)

```
[pekichigina@pekichigina git-extended]$ git push
--all
Total 0 (delta 0), reused 0 (delta 0), pack-reuse
d 0 (from 0)
remote:
remote: Create a pull request for 'develop' on Gi
tHub by visiting:
remote:      https://github.com/PolinaEK/git-exte
nded/pull/new/develop
remote:
To github.com:PolinaEK/git-extended.git
 * [new branch]      develop -> develop
[pekichigina@pekichigina git-extended]$ git branc
h --set-upstream-to=origin/develop develop
branch 'develop' set up to track 'origin/develop'
```

Рис. 3.11: Загружаем и устанавливаем ветку

Создадим релиз с версией 1.0.0(рис. 3.12)

```
[pekichigina@pekichigina git-extended]$ git f
low 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, b
ased 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 prepa
ring your release
- When done, run:

    git flow release finish '1.0.0'
```

Рис. 3.12: Создаем релиз

Создадим журнал изменений и добавим журнал изменений в индекс(рис. 3.13)


```

[pekichigina@pekichigina git-extended]
$ standard-changelog --first-release
✓ created CHANGELOG.md
✓ output changes to CHANGELOG.md
[pekichigina@pekichigina git-extended]
$ git add CHANGELOG.md
[pekichigina@pekichigina git-extended]
[pekichigina@pekichigina git-extended]
$ git commit -am 'chore(site): add changelog'
[release/1.0.0 68239ec] chore(site): add changelog
1 file changed, 9 insertions(+)
create mode 100644 CHANGELOG.md

```

Рис. 3.13: 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. 3.14)

```

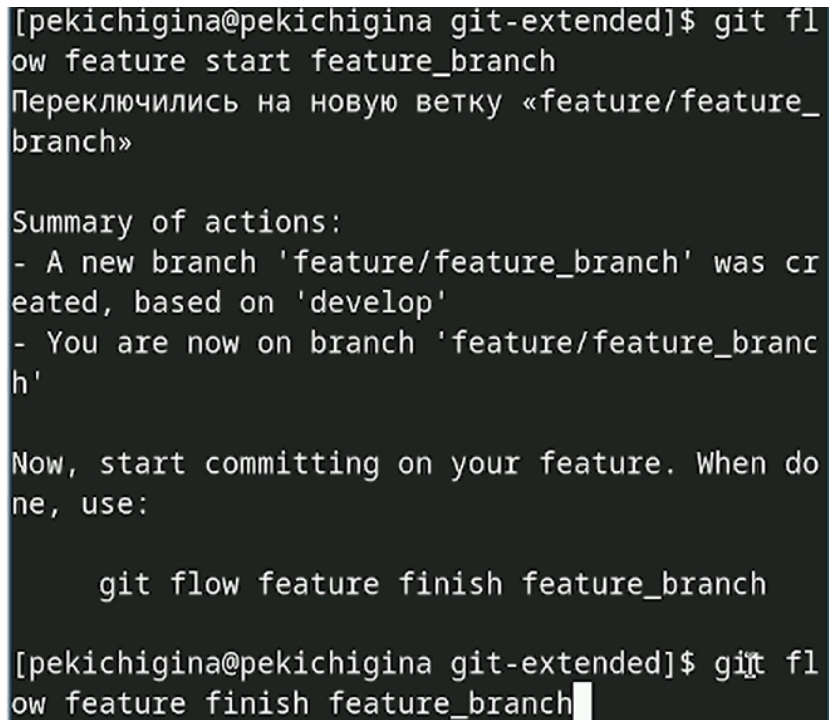
[pekichigina@pekichigina git-extended]$ git push --tags
Перечисление объектов: 1, готово.
Подсчет объектов: 100% (1/1), готово.
Запись объектов: 100% (1/1), 161 байт | 161.00 КиБ/с, готово.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:PolinaEK/git-extended.git
 * [new tag]          v1.0.0 -> v1.0.0
[pekichigina@pekichigina git-extended]$ [pekichigina@pekichigina git-extended]$ gh release create v1.0.0 -F CHANGELOG.md
https://github.com/PolinaEK/git-extended/releases/tag/v1.0.0

```

Рис. 3.14: Отправляем на github

8. Работа с репозиторием git

Создадим ветку для новой функциональности. Далее, продолжаем работу с git как обычно. По окончании разработки новой функциональности следующим шагом следует объединить ветку feature_branch с develop (рис. 3.15)



```
[pekichigina@pekichigina git-extended]$ git flow feature start feature_branch
Переключились на новую ветку «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

[pekichigina@pekichigina git-extended]$ git flow feature finish feature_branch
```

Рис. 3.15: Разрабатываем новую функциональность и объединяем ветки

9. 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. 3.16)

```

{
  "name": "git-extended",
  "version": "1.2.3",
  "description": "Git repo for educational",
  "main": "index.js",
  "repository": "git@github.com:PolinaEK/g",
  "author": "Polina Kichigina polina120719",
  "license": "CC-BY-4.0",
  "config": {
    "commitizen": {
      "path": "cz-conventional-changel
    }
  }
}

```

Рис. 3.16: Создаем релиз

Создадим журнал изменений и добавим журнал изменений в индекс (рис. 3.17)

```

[pekichigina@pekichigina git-extended]
$ standard-changelog
✓ output changes to CHANGELOG.md
[pekichigina@pekichigina git-extended]
$ git add CHANGELOG.md
[pekichigina@pekichigina git-extended]
[pekichigina@pekichigina git-extended]
$ git commit -am 'chore(site): update
changelog'
[release/1.2.3 c4a73e1] chore(site): u
pdate changelog
 2 files changed, 5 insertions(+), 1 d
eletion(-)

```

Рис. 3.17: Создаем и добавляем

Создадим релиз на github с комментарием из журнала изменений (рис. 3.18)


```
[pekichigina@pekichigina git-extended]  
$ gh release create v1.2.3 -F CHANGELOG.md  
https://github.com/PolinaEK/git-extended/releases/tag/v1.2.3
```

Рис. 3.18: Создаем

4 Выводы

Мы получили навыки правильной работы с репозиториями git.