

Лабораторная работа №4

Markdown

Комягин Андрей Николаевич

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	11
	Список литературы	12

Список иллюстраций

3.1	Установка gitfloy	7
3.2	pnpm	7
3.3	Настройка репозитория	8
3.4	git	8
3.5	Параметры пакета	9
3.6	Коммит	9
3.7	git-flow	9
3.8	git-flow	10
3.9	Релиз 1.2.3	10
3.10	Отправка	10

Список таблиц

1 Цель работы

Получение навыков продвинутой 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

Установка gitflow.(рис. 3.1).

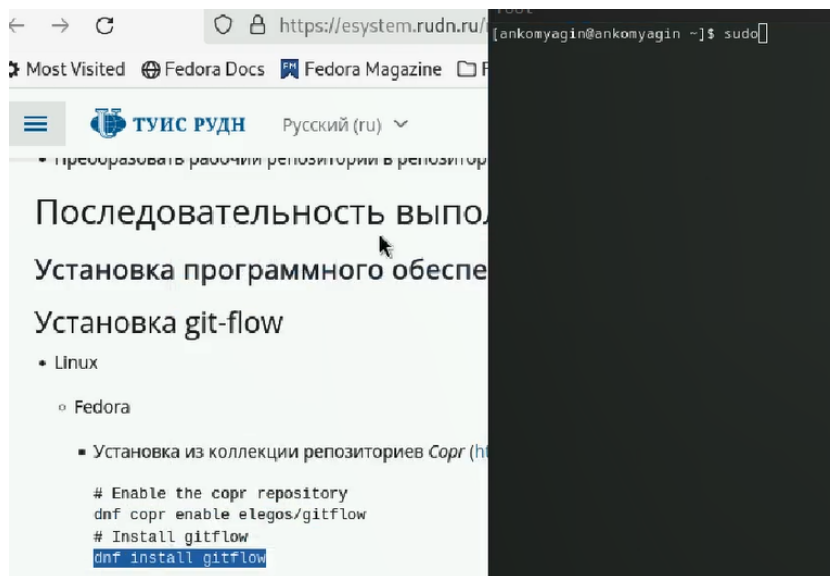


Рис. 3.1: Установка gitflow

Установим и откроем npnm, настроим node.js (рис. 3.2).

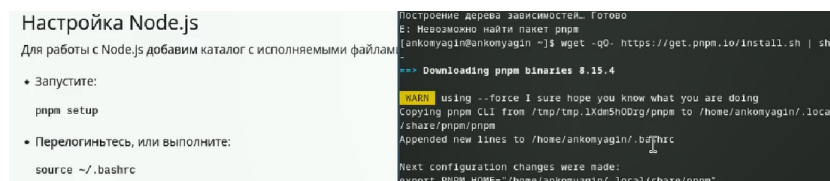


Рис. 3.2: npnm

Далее создаём репозиторий и клонируем его, создаем там файл, делаем коммит (рис. 3.3)

```
[ankomyagin@ankomyagin ~]$ cd work/
[ankomyagin@ankomyagin work]$ git clone --recurse git@github.com:ANKomyagin/git-extended.git
Клонирование в «git-extended»...
warning: Похоже, что вы клонировали пустой репозиторий.
[ankomyagin@ankomyagin work]$ cd
ANKomyagin.github.io/ git-extended/
blog/ study/
[ankomyagin@ankomyagin work]$ cd git-extended/
[ankomyagin@ankomyagin git-extended]$ touch text.txt
[ankomyagin@ankomyagin git-extended]$ git commit -m "first commit"
Текущая ветка: main

Начальный коммит

Неотслеживаемые файлы:
(используйте «git add <файл>...», чтобы добавить в то, что будет включено в коммит)
text.txt

индекс пуст, но есть неотслеживаемые файлы
(используйте «git add», чтобы проиндексировать их)
[ankomyagin@ankomyagin git-extended]$ git add .
[ankomyagin@ankomyagin git-extended]$ git commit -m "first commit"
[main (корневой коммит) 5c567c7] first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 text.txt
[ankomyagin@ankomyagin git-extended]$
```

Рис. 3.3: Настройка репозитория

Отправляем изменения на git (рис. 3.4).

```
[ankomyagin@ankomyagin git-extended]$ git remote add origin git@github.com:ANKomyagin/git-extended.git
error: внешний репозиторий origin уже существует
[ankomyagin@ankomyagin git-extended]$ git push -u origin master
error: src refspec master ничего не соответствует
error: не удалось отправить некоторые ссылки в «github.com:ANKomyagin/git-extended.git»
[ankomyagin@ankomyagin git-extended]$ git push -u origin main
Перечисление объектов: 3, готово.
Подсчет объектов: 100% (3/3), готово.
Запись объектов: 100% (3/3), 866 байтов | 866.00 КиБ/с, готово.
Всего 3 (изменений 0), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
To github.com:ANKomyagin/git-extended.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
[ankomyagin@ankomyagin git-extended]$
```

Рис. 3.4: git

Изменяем параметры пакета, меняем лицензию и сформируем формат коммитов (рис. 3.5).


```
{
  "name": "git-extended",
  "version": "1.0.0",
  "description": "Git repo for educational purposes",
  "main": "index.js",
  "repository": "git@github.com:username/git-extended.git",
  "author": "Andrey Komyagin Komyagin12345@mail.ru",
  "license": "CC-BY-4.0",
  "config": {
    "commitizen": {
      "path": "cz-conventional-changelog"
    }
  }
}
```

Рис. 3.5: Параметры пакета

Добавляем файлы, выполняем коммит cz и затем отправляем (рис. 3.6).

```
[ankomyagin@ankomyagin git-extended]$ nano package.json
[ankomyagin@ankomyagin git-extended]$ git add .
[ankomyagin@ankomyagin git-extended]$ git cz
cz-cli@4.3.0, cz-conventional-changelog@3.3.0

? Select the type of change that you're committing: (Use arrow keys)
> feat:      A new feature
  fix:       A bug fix
  docs:      Documentation only changes
  style:     Changes that do not affect the meaning of the code (white-space,
formatting, missing semi-colons, etc)
  refactor:  A code change that neither fixes a bug nor adds a feature
  perf:      A code change that improves performance
(Move up and down to reveal more choices)
```

Рис. 3.6: Коммит

Инициализируем git-flow, с префиксом v (рис. 3.7).

```
[ankomyagin@ankomyagin 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
```

Рис. 3.7: git-flow

Проводим операцию над веткой, релизом и журналом изменений (рис. 3.8).

- Установите внешнюю ветку как вышестоящую для этой ветки:
`git branch --set-upstream-to=origin/develop develop`
- Создадим релиз с версией 1.0.0
`git flow release start 1.0.0`
- Создадим журнал изменений
`standard-changelog --first-release`
- Добавим журнал изменений в индекс
`git add CHANGELOG.md`
`git commit -am 'chore(site): add changelog'`

Рис. 3.8: git-flow

Отправим изменения на github

Разработаем новую функциональность. Создадим релиз 1.2.3(рис. 3.9).

- Создадим релиз с версией 1.2.3:
`git flow release start 1.2.3`
- Обновите номер версии в файле `package.json`. Установите её в 1.2.3.
- Создадим журнал изменений
`standard-changelog`
- Добавим журнал изменений в индекс
`git add CHANGELOG.md`
`git commit -am 'chore(site): update changelog'`

Рис. 3.9: Релиз 1.2.3

Отправляем данные на git, создадим релиз изменений (рис. 3.10).

```
Подсчет объектов: 100% (1/1), готово.
Запись объектов: 100% (1/1), 160 байтов | 160.00 КиБ/с, готово.
Всего 1 (изменений 0), повторно использовано 0 (изменений 0), повторно использовано
котов 0
To github.com:ANKomyagin/git-extended.git
 * [new tag]          v1.2.3 -> v1.2.3
[ankomyagin@ankomyagin git-extended]$ gh release create v1.2.3 -F CHANGELOG.md
```

Рис. 3.10: Отправка

4 Выводы

В ходе выполнения лабораторной работы я получил навыки правильной работы с репозиториями git.

Список литературы

Туис, курс Архитектура компьютера и операционные системы