

Отчет по лабораторной работе №4

Дисциплина: Операционные системы

Иванов Сергей Владимирович

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
4	Выводы	14

Список иллюстраций

3.1	Активация Corp	6
3.2	Установка git-flow	6
3.3	Установка Node.js	7
3.4	Настройка Node.js	7
3.5	Commitizen	7
3.6	Standard-changelog	7
3.7	Создание репозитория	8
3.8	Клонирование	8
3.9	Отправка на сервер	8
3.10	Конфигурируем коммит	9
3.11	Отправка коммита	9
3.12	Конфигурация git-flow	9
3.13	develop	9
3.14	Загрузка репозитория	10
3.15	Создаем релиз	10
3.16	Журнал изменений	10
3.17	Заливаем релизную ветку	10
3.18	Отправка	11
3.19	Создаем релиз	11
3.20	Новая ветка	11
3.21	Создаем релиз	12
3.22	Обновляем номер версии	12
3.23	Создаем релиз	12
3.24	Заливаем ветку	12
3.25	Отправляем данные	13
3.26	Создаем релиз с комментарием	13

1 Цель работы

1. Получение навыков правильной работы с репозиториями git.

2 Задание

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

3 Выполнение лабораторной работы

Установим git-flow из коллекции репозиторийев Copr. Активируем репозиторий Copr. (рис. 1).

```
[svivanov1@svivanov1 ~]$ sudo dnf copr enable elegos/gitflow
[sudo] пароль для svivanov1:
Включение репозитория Copr. Обратите внимание, что этот репозиторий не является частью основного дистрибутива, и качество может отличаться.

Проект Fedora не имеет какого-либо влияния на содержимое этого репозитория за рамками правил, описанных в Вопросах и Ответах Copr в <https://docs.pagure.org/copr.copr/user_documentation.html#what-i-can-build-in-copr>, а качество и безопасность пакетов не поддерживаются на каком-либо уровне.

Не отправляйте сообщения об ошибках этих пакетов в Fedora Bugzilla. В случае возникновения проблем обращайтесь к владельцу этого репозитория.

Do you really want to enable copr.fedorainfracloud.org/elegos/gitflow? [y/N]: y
Репозиторий успешно подключен.
```

Рис. 3.1: Активация Copr

Установим git-flow - `dnf install gitflow` (рис. 2).

```
[svivanov1@svivanov1 ~]$ sudo dnf install gitflow
Copr repo for gitflow owned by elegos                2.5 kB/s | 1.5 kB    00:00
Fedora 39 - x86_64 - Updates                          16 kB/s | 18 kB     00:01
Fedora 39 - x86_64 - Updates                        2.0 MB/s | 3.6 MB    00:01
Зависимости разрешены.

=====
Пакет      Архитектура  Версия      Репозиторий                                     Размер
=====
Установка:
gitflow    x86_64       1.12.3-1.fc34  copr:copr.fedorainfracloud.org:elegos:gitflow  57 k
```

Рис. 3.2: Установка git-flow

Установим Node.js (рис. 3, 4).

```
[svivanov1@svivanov1 ~]$ sudo dnf install nodejs
Последняя проверка окончания срока действия метаданных: 0:01:28 назад, Ср 28 фев 2024 18:47:14.
Зависимости разрешены.

=====
Пакет      Архитектура  Версия      Репозиторий                                     Размер
=====
Установка:
nodejs     x86_64       1:20.12.0-3.fc39  fedora                                           42 k
```

```
[svivanov1@svivanov1 ~]$ sudo dnf install pnpm
Последняя проверка окончания срока действия метаданных: 0:02:43 назад, Ср 28 фев 2024 18:47:14.
Зависимости разрешены.
=====
Пакет      Архитектура  Версия      Репозиторий      Размер
=====
Установка:
pnpm       noarch       8.12.0-1.fc39 updates         2.6 М
```

Рис. 3.3: Установка Node.js

Настроим Node.js. Запустим 'pnpm setup' и перелогинимся (рис. 5).

```
[svivanov1@svivanov1 ~]$ pnpm setup
Appended new lines to /home/svivanov1/.bashrc

Next configuration changes were made:
export PNPM_HOME="/home/svivanov1/.local/share/pnpm"
case ":$PATH:" in
  *"$PNPM_HOME:") ;;
  *) export PATH="$PNPM_HOME:$PATH" ;;
esac

To start using pnpm, run:
source /home/svivanov1/.bashrc
[svivanov1@svivanov1 ~]$ source ~/.bashrc
[svivanov1@svivanov1 ~]$ source /home/svivanov1/.bashrc
```

Рис. 3.4: Настройка Node.js

Установим программу для помощи в форматировании коммитов. (рис. 6).

```
[svivanov1@svivanov1 ~]$ pnpm add -g commitizen

Update available! 8.12.0 → 8.15.4.
Changelog: https://github.com/pnpm/pnpm/releases/tag/v8.15.4
Run "pnpm add -g pnpm" to update.

Follow @pnpmjs for updates: https://twitter.com/pnpmjs

Packages: +152
+++++
Progress: resolved 152, reused 0, downloaded 152, added 152, done
Downloading registry.npmjs.org/typescript/5.3.3: 5.76 MB/5.76 MB, done

/home/svivanov1/.local/share/pnpm/global/5:
+ commitizen 4.3.0
```

Рис. 3.5: Commitizen

Установим программу для помощи в создании логов. (рис. 7).

```
[svivanov1@svivanov1 ~]$ pnpm add -g standard-changelog
Packages: +56
+++++
Progress: resolved 208, reused 152, downloaded 56, added 56, done

/home/svivanov1/.local/share/pnpm/global/5:
+ standard-changelog 5.0.0

Done in 4.5s
```

Рис. 3.6: Standard-changelog

Создадим репозиторий 'git-extended' (рис. 8).

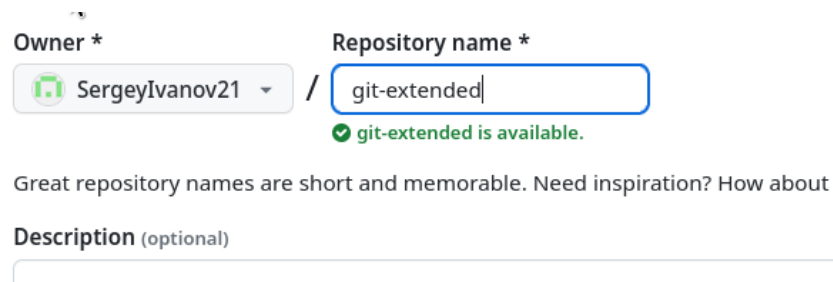


Рис. 3.7: Создание репозитория

Клонируем репозиторий и создаем файл README.md (рис. 9).

```
[svivanov1@svivanov1 ~]$ git clone --recursive https://github.com/SergeyIvanov21/git-extended.git
Клонирование в «git-extended»...
warning: Похоже, что вы клонировали пустой репозиторий.
[svivanov1@svivanov1 ~]$ touch README.md
```

Рис. 3.8: Клонирование

Делаем коммит и выкладываем на github (рис. 10).

```
[svivanov1@svivanov1 git-extended]$ git add .
[svivanov1@svivanov1 git-extended]$ git commit -m "first commit"
[main (корневой коммит) 0862eaf] first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.md
[svivanov1@svivanov1 git-extended]$ git remote add origin https://github.com/SergeyIvanov21/git-extended.git
error: внешний репозиторий origin уже существует
[svivanov1@svivanov1 git-extended]$ git push -u origin main
Перечисление объектов: 3, готово.
Подсчет объектов: 100% (3/3), готово.
Запись объектов: 100% (3/3), 894 байта | 894.00 КиБ/с, готово.
Всего 3 (изменений 0), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
To https://github.com/SergeyIvanov21/git-extended.git
 * [new branch]      main -> main
```

Рис. 3.9: Отправка на сервер

Сконфигурируем формат коммитов. Для этого добавим в файл package.json команду для формирования коммитов (рис. 11).


```
GNU nano 7.2 package.json
{
  "name": "git-extended",
  "version": "1.0.0",
  "description": "Git repo for educational purposes",
  "main": "index.js",
  "keywords": [],
  "repository": "git@github.com:SergeyIvanov21/git-extended.git",
  "author": "Sergey Ivanov 1sergeiivanov1@mail.ru",
  "license": "CC-BY-4.0"
  "config": {
    "commitizen": {
      "path": "cz-conventional-changelog"
    }
  }
}
```

Рис. 3.10: Конфигурируем коммит

Добавим новые файлы, выполним коммит, отправим на github (рис. 12).

```
[svivanov1@svivanov1 git-extended]$ git add .
[svivanov1@svivanov1 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) reame.md
? Write a short, imperative tense description of the change (max 84 chars):
(8) add 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 92d42d4] feat(ream.md): add file
1 file changed, 12 insertions(+)
create mode 100644 package.json
```

Рис. 3.11: Отправка коммита

Инициализируем git-flow, префикс для ярлыков установим в v (рис. 13).

```
[svivanov1@svivanov1 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/svivanov1/git-extended/.git/hooks]
```

Рис. 3.12: Конфигурация git-flow

Проверим, что мы на ветке develop (рис. 14).

```
[svivanov1@svivanov1 git-extended]$ git branch
* develop
  main
[svivanov1@svivanov1 git-extended]$
```

Рис. 3.13: develop

Загрузим весь репозиторий в хранилище (рис. 15).

```
[svivanov1@svivanov1 git-extended]$ git push --all
Перечисление объектов: 4, готово.
Подсчет объектов: 100% (4/4), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (3/3), готово.
Запись объектов: 100% (3/3), 1.11 КиБ | 1.11 МБ/с, готово.
Всего 3 (изменений 0), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
To https://github.com/SergeyIvanov21/git-extended.git
   0862eaf..92d42d4  main -> main
   * [new branch]      develop -> develop
```

Рис. 3.14: Загрузка репозитория

Установим внешнюю ветку как вышестоящую для этой ветки и создадим релиз с версией 1.0.0 (рис. 16).

```
[svivanov1@svivanov1 git-extended]$ git branch --set-upstream-to=origin/develop develop
branch 'develop' set up to track 'origin/develop'.
[svivanov1@svivanov1 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'
```

Рис. 3.15: Создаем релиз

Создадим журнал изменений и добавим журнал изменений в индекс (рис. 17).

```
[svivanov1@svivanov1 git-extended]$ standard-changelog --first-release
✓ created CHANGELOG.md
✓ output changes to CHANGELOG.md
[svivanov1@svivanov1 git-extended]$ git add CHANGELOG.md
[svivanov1@svivanov1 git-extended]$ git commit -am 'chore(site): add changelog'
[release/1.0.0 57560e4] chore(site): add changelog
1 file changed, 9 insertions(+)
create mode 100644 CHANGELOG.md
```

Рис. 3.16: Журнал изменений

Зальём релизную ветку в основную ветку (рис. 18).

```
[svivanov1@svivanov1 git-extended]$ git flow release finish 1.0.0
Переключились на ветку «main»
Эта ветка соответствует «origin/main».
Merge made by the 'ort' strategy.
 CHANGELOG.md | 9 ++++++++
 1 file changed, 9 insertions(+)
 create mode 100644 CHANGELOG.md
Уже на «main»
Ваша ветка опережает «origin/main» на 2 коммита.
(используйте «git push», чтобы опубликовать ваши локальные коммиты)
```

Рис. 3.17: Заливаем релизную ветку

Отправим данные на github (рис. 19).

```
[svivanov1@svivanov1 git-extended]$ git push --all
Перечисление объектов: 6, готово.
Подсчет объектов: 100% (6/6), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (5/5), готово.
Запись объектов: 100% (5/5), 2.88 КиБ | 2.88 МиБ/с, готово.
Всего 5 (изменений 0), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
To https://github.com/SergeyIvanov21/git-extended.git
   92d42d4..842a5c9  develop -> develop
   92d42d4..f430216  main -> main
[svivanov1@svivanov1 git-extended]$ git push --tags
Перечисление объектов: 1, готово.
Подсчет объектов: 100% (1/1), готово.
Запись объектов: 100% (1/1), 170 байтов | 170.00 КиБ/с, готово.
Всего 1 (изменений 0), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
To https://github.com/SergeyIvanov21/git-extended.git
 * [new tag]           v1.0.0 -> v1.0.0
```

Рис. 3.18: Отправка

Создадим релиз на github. (рис. 20).

```
[svivanov1@svivanov1 git-extended]$ gh release create v1.0.0 -F CHANGELOG.md
https://github.com/SergeyIvanov21/git-extended/releases/tag/v1.0.0
[svivanov1@svivanov1 git-extended]$
```

Рис. 3.19: Создаем релиз

Создадим ветку для новой функциональности, объединим ветку feature_branch с develop. (рис. 21).

```
[svivanov1@svivanov1 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

[svivanov1@svivanov1 git-extended]$ git flow feature finish feature_branch
Переключились на ветку «develop»
Эта ветка соответствует «origin/develop».
Уже актуально.
Ветка feature/feature_branch удалена (была 842a5c9).

Summary of actions:
- The feature branch 'feature/feature_branch' was merged into 'develop'
- Feature branch 'feature/feature_branch' has been locally deleted
- You are now on branch 'develop'
```

Рис. 3.20: Новая ветка

Создадим релиз с версией 1.2.3. (рис. 22).

```
[svivanov1@svivanov1 git-extended]$ git flow release start 1.2.3
Переключились на новую ветку «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'
```

Рис. 3.21: Создаем релиз

Обновим номер версии в файле package.json. Установим её в 1.2.3. (рис. 23).

```
{
  "name": "git-extended",
  "version": "1.2.3",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  }
}
```

Рис. 3.22: Обновляем номер версии

Создадим журнал изменений (рис. 24).

```
[svivanov1@svivanov1 git-extended]$ standard-changelog
✓ output changes to CHANGELOG.md
[svivanov1@svivanov1 git-extended]$ git add CHANGELOG.md
[svivanov1@svivanov1 git-extended]$ git commit -am 'chore(site): update changelog'
[release/1.2.3 9776e44] chore(site): update changelog
2 files changed, 5 insertions(+), 1 deletion(-)
```

Рис. 3.23: Создаем релиз

Добавим журнал изменений в индекс и зальём релизную ветку в основную ветку. (рис. 25).

```
[svivanov1@svivanov1 git-extended]$ git commit -am 'chore(site): update changelog'
[release/1.2.3 9776e44] chore(site): update changelog
2 files changed, 5 insertions(+), 1 deletion(-)
[svivanov1@svivanov1 git-extended]$ git flow release finish 1.2.3
Переключились на ветку «main»
Эта ветка соответствует «origin/main».
Merge made by the 'ort' strategy.
CHANGELOG.md | 4 ++++
```

Рис. 3.24: Заливаем ветку

Отправим данные на github. (рис. 26).

```
[svivanov1@svivanov1 git-extended]$ git push --all
Перечисление объектов: 9, готово.
Подсчет объектов: 100% (9/9), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 2.85 КиБ | 2.85 МБ/с, готово.
Всего 6 (изменений 2), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/SergeyIvanov21/git-extended.git
   842a5c9..9eee999  develop -> develop
   f430216..66200d7  main -> main
[svivanov1@svivanov1 git-extended]$ git push --tags
Перечисление объектов: 1, готово.
Подсчет объектов: 100% (1/1), готово.
Запись объектов: 100% (1/1), 172 байта | 172.00 КиБ/с, готово.
Всего 1 (изменений 0), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
To https://github.com/SergeyIvanov21/git-extended.git
 * [new tag]         v1.2.3 -> v1.2.3
```

Рис. 3.25: Отправляем данные

Создадим релиз на github с комментарием из журнала изменений (рис. 27)

```
[svivanov1@svivanov1 git-extended]$ gh release create v1.2.3 -F CHANGELOG.md
https://github.com/SergeyIvanov21/git-extended/releases/tag/v1.2.3
[svivanov1@svivanov1 git-extended]$
```

Рис. 3.26: Создаем релиз с комментарием

4 Выводы

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