

Отчёт по лабораторной работе №3

Простейший вариант

Тимур Ринатович Каримов

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	14
	Список литературы	15

Список иллюстраций

4.1	Установка gitflow	8
4.2	Установка nodejs	8
4.3	Установка npm	8
4.4	Запуск npm	8
4.5	Выполнение команды source	8
4.6	Выполнение команды commitizen	9
4.7	Выполнение команды standard-changelog	9
4.8	Пустой репозиторий	9
4.9	Коммит и отправка на GitHub	10
4.10	Изменения данных в package.json	10
4.11	Добавление новых файлов и выполнение коммит	10
4.12	Отправка файлов	11
4.13	Инициализация git-flow	11
4.14	Проверка ветки	11
4.15	Отправка репозитория	11
4.16	Создание релиза 1.0.0	11
4.17	Создание журнала изменений	11
4.18	Добавление журнала изменений в индекс	12
4.19	Отправка релизной ветки в основную	12
4.20	Отправка данных на сервер	12
4.21	Отправка данных на сервер	12
4.22	Создание релиза на GitHub	12
4.23	Создание ветки для новой функциональности	12
4.24	Создание релиза git-flow	13
4.25	Обновление номера версии в файле package.json	13
4.26	Создание журнала изменений	13
4.27	Создание релиз на github	13

Список таблиц

3.1	Описание некоторых каталогов файловой системы GNU Linux . . .	7
-----	---	---

1 Цель работы

Освоение навыков правильной работы с репозиториями git.

2 Задание

Установка git-flow Установка Node.js Настройка Node.js Общепринятые коммиты Создание репозитория git Работа с репозиторием git

3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы. Например, в табл. 3.1 приведено краткое описание стандартных каталогов Unix.

Таблица 3.1: Описание некоторых каталогов файловой системы GNU Linux	
Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно про Unix см. в [1–4].

4 Выполнение лабораторной работы

Установим *gitflow* (рис. 4.1)

```
[karimov_li@vbox ~]$ sudo dnf install gitflow
Обновление и загрузка репозитория:
Fedora 41 - x86_64 - Updates          100% | 28.3 KiB/s | 23.3 KiB | 00m01s
Copr repo for gitflow owned by elegos  100% | 2.6 KiB/s | 2.4 KiB | 00m01s
```

Рис. 4.1: Установка gitflow

Установим *nodejs* (рис. 4.2).

```
[karimov_li@vbox ~]$ sudo dnf install nodejs
```

Рис. 4.2: Установка nodejs

Установим *pnpm* (рис. 4.3).

```
[karimov_li@vbox ~]$ sudo dnf install pnpm
Обновление и загрузка репозитория:
Репозитории загружены.
```

Рис. 4.3: Установка pnpm

Запустим *pnpm* (рис. 4.4).

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

Рис. 4.4: Запуск pnpm

Выполним команду *source ~/.bashrc* (рис. 4.5).

```
[karimov_li@vbox ~]$ source ~/.bashrc
[karimov_li@vbox ~]$
```

Рис. 4.5: Выполнение команды source

Введем программу для помощи в форматировании коммитов (рис. 4.6).

```
[karimov_li@vbox ~]$ npm add -g commitizen
```

Рис. 4.6: Выполнение команды commitizen

Введем программу для помощи в создании логов (рис. 4.7).

```
[karimov_li@vbox ~]$ npm add -g standard-changelog  
WARN 2 deprecated subdependencies found: glob@7.2.3, inflight@1.0.6  
Packages: +39  
Progress: resolved 198, reused 151, downloaded 39, added 22
```

Рис. 4.7: Выполнение команды standard-changelog

Создадим пустой репозиторий *git-extended* (рис. 4.8).

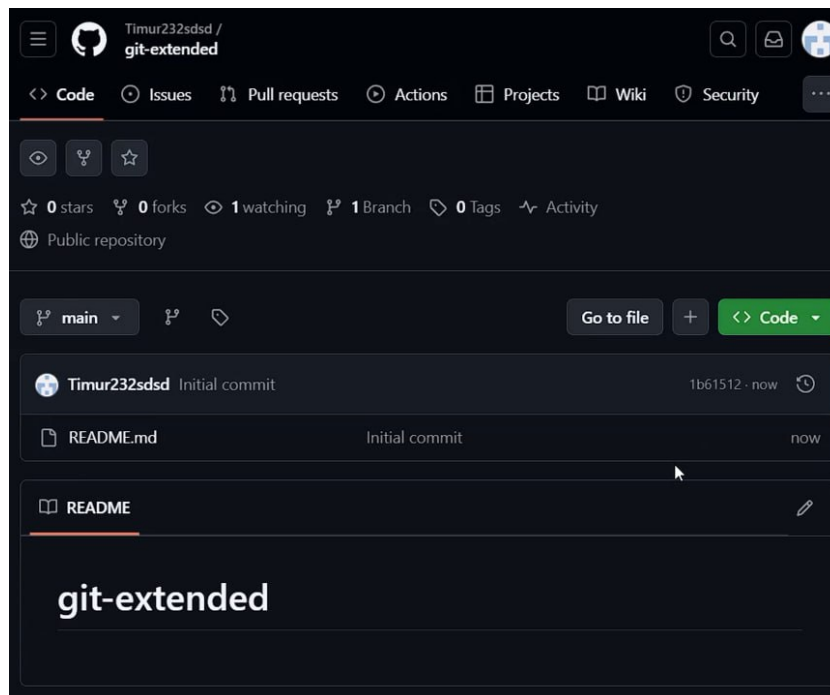


Рис. 4.8: Пустой репозиторий

Сделаем первый коммит и выложим на *GitHub* (рис. 4.9).

```
[karimov_li@vbox git-ex]$ git commit -m "first commit"
[master (корневой коммит) 37f5fb9] first commit
1 file changed, 1 insertion(+)
create mode 160000 git-extended
[karimov_li@vbox git-ex]$ git remote add origin git@github.com:<username>/git-extended.git
bash: username: Нет такого файла или каталога
[karimov_li@vbox git-ex]$ git remote add origin git@github.com:Timur232sdsd/git-extended.git
[karimov_li@vbox git-ex]$ git push -u origin master
Перечисление объектов: 2, готово.
Подсчет объектов: 100% (2/2), готово.
Запись объектов: 100% (2/2), 856 байтов | 428.00 КиБ/с, готово.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/Timur232sdsd/git-extended/pull/new/master
remote:
To github.com:Timur232sdsd/git-extended.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

Рис. 4.9: Коммит и отправка на GitHub

Добавим в файл package.json команду для формирования коммитов (рис. 4.10).

```
foot
package.json  [-----] 18 L: [ 1+ 6 7/ 17] *(208 / 395b) 0032 0x020 [*][X]
{
  "name": "git-extended",
  "version": "1.0.0",
  "description": "Git with the conventional commit",
  "main": "index.js",
  "dependencies": {
    "conventional-commits-parser": "3.2.4"
  },
  "scripts": {
    "test": "echo \"Error: no test specified\" & exit 1",
    "commit": "cz"
  },
  "config": {
    "commitizen": {
      "path": "cz-conventional-changelog"
    }
  },
  "devDependencies": {
    "commitizen": "4.3.1"
  }
}
```

Рис. 4.10: Изменения данных в package.json

Добавим новые файлы и выполним коммит (рис. 4.11).

```
[karimov_li@vbox git-ex]$ git add .
[karimov_li@vbox git-ex]$ git cz
cz-cli@4.3.1, cz-conventional-changelog@3.3.0

? Select the type of change that you're committing:
1: feat: a new feature
2: fix: a bug fix
3: docs: changes to documentation
4: style: formatting, missing semi-colons, etc;
5: test: adding missing tests, correcting existing tests
6: chore: updating dependencies, configuration files, etc;
7: revert: revert the last commit with a similar message
8: ignore: I don't want to commit this
9: quit: to exit the commitizen CLI
```

Рис. 4.11: Добавление новых файлов и выполнение коммит

Отправим файлы на GitHub (рис. 4.12).

```
[karimov_li@vbox git-ex]$ git push
Everything up-to-date
[karimov_li@vbox git-ex]$
```

Рис. 4.12: Отправка файлов

Инициализируем git-flow (рис. 4.13).

```
Everything up-to-date
[karimov_li@vbox git-ex]$ git flow init
Which branch should be used for bringing forth production releases?
master
```

Рис. 4.13: Инициализация git-flow

Проверим, что мы на ветке *develop* (рис. 4.14).

```
[karimov_li@vbox git-ex]$ git branch
* develop
  master
```

Рис. 4.14: Проверка ветки

Загрузим весь репозиторий в хранилище (рис. 4.15).

```
master
[karimov_li@vbox git-ex]$ git push --all
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
```

Рис. 4.15: Отправка репозитория

Создадим релиз с версией 1.0.0 (рис. 4.16).

```
[karimov_li@vbox git-ex]$ git flow release start 1.0.0
Переключились на новую ветку «release/1.0.0»
```

Рис. 4.16: Создание релиза 1.0.0

Создадим журнал изменений (рис. 4.17).

```
[karimov_li@vbox git-ex]$ standard-changelog --first-release
✓ created CHANGELOG.md
✓ output changes to CHANGELOG.md
```

Рис. 4.17: Создание журнала изменений

Добавим журнал изменений в индекс (рис. 4.18).

```
[karimov_li@vbox git-ex]$ git commit -am 'chore(site): add changelog'
Текущая ветка: release/1.0.0
ничего коммитить, нет изменений в рабочем каталоге
```

Рис. 4.18: Добавление журнала изменений в индекс

Зальём релизную ветку в основную ветку (рис. 4.19).

```
[karimov_li@vbox git-ex]$ git flow release finish 1.0.0
Переключились на ветку «master»
Эта ветка соответствует «origin/master».
Merge made by the 'ort' strategy.
```

Рис. 4.19: Отправка релизной ветки в основную

Отправим данные на *GitHub* (рис. 4.20) и (рис. 4.21).

```
[karimov_li@vbox git-ex]$ git push --all
Перечисление объектов: 5, готово.
Подсчет объектов: 100% (5/5), готово.
Сжатие объектов: 100% (3/3), готово.
```

Рис. 4.20: Отправка данных на сервер

```
[karimov_li@vbox git-ex]$ git push --tags
Everything up-to-date
```

Рис. 4.21: Отправка данных на сервер

Создадим релиз на github. Для этого будем использовать утилиты работы с github (рис. 4.22).

```
Everything up-to-date
[karimov_li@vbox git-ex]$ gh release create v1.0.0 -F CHANGELOG.md
https://github.com/Timur232sdsd/git-extended/releases/tag/v1.0.0
[karimov_li@vbox git-ex]$
```

Рис. 4.22: Создание релиза на GitHub

Создадим ветку для новой функциональности (рис. 4.23).

```
[karimov_li@vbox git-ex]$ git flow feature finish feature_branch
Переключились на ветку «develop»
Эта ветка соответствует «origin/develop».
```

Рис. 4.23: Создание ветки для новой функциональности

Создадим релиз git-flow (рис. 4.24).

```
[karimov_li@vbox git-ex]$ git flow release start 1.2.3
Fatal: There is an existing release branch '1.0.0'. Finish that one first.
[karimov_li@vbox git-ex]$ ls
git-extended node_modules package.json npm-lock.yaml
```

Рис. 4.24: Создание релиза git-flow

Обновим номер версии в файле package.json (рис. 4.25).

```
foot
package.json [-M--] 19 L: [ 1+ 2 3/ 17] *(47 / 396b) 0034 0x022 [*][X
{
  "name": "git-extended",
  "version": "1.2.3",
  "description": "git-flow for educational purposes",
  "main": "index.js",
  "repository": "github:karimov-li/git-extended.git",
  "author": "Timur232sdsd <timur232sdsd@gmail.com>",
  "license": "MIT",
  "scripts": {
    "test": "echo \"Error: no test specified\" & exit 1"
  },
  "keywords": [
    "git"
  ],
  "devDependencies": {}
}
```

Рис. 4.25: Обновление номера версии в файле package.json

Создадим журнал изменений (рис. 4.26).

```
[karimov_li@vbox git-ex]$ standard-changelog
✓ created CHANGELOG.md
✓ output changes to CHANGELOG.md
```

Рис. 4.26: Создание журнала изменений

Добавим журнал изменений в индекс и создадим релиз на github с комментарием из журнала изменений (рис. 4.27).

```
[karimov_li@vbox git-ex]$ gh release create v1.2.3 -F CHANGELOG.md
https://github.com/Timur232sdsd/git-extended/releases/tag/v1.2.3
```

Рис. 4.27: Создание релиза на github

5 Выводы

Работа продемонстрировала эффективность использования современных инструментов разработки, таких как `git-flow`, `commitizen`, `standard-changelog` и `npnm`, для автоматизации и стандартизации процессов разработки. Все задачи были выполнены успешно, репозиторий настроен для дальнейшей работы, а процесс создания коммитов, управления ветками и выпуска релизов стал более структурированным и удобным.

Список литературы

1. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.
2. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.