

Отчёт по лабораторной работе №4

Специальность: архитектура компьютеров

Комаров Владимир Артемович

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
3.1	Рабочий процесс Gitflow	7
4	Выполнение лабораторной работы	8
4.1	Установка программного обеспечения.	8
4.1.1	Установка git-flow	8
4.1.2	Установка и настройка node.js	8
4.1.3	Общепринятые коммиты.	9
4.2	Практический сценарий использования git.	9
4.2.1	Создание репозитория git.	9
4.2.2	Работа с репозиторием git.	15
5	Выводы	17
	Список литературы	18

Список иллюстраций

4.1	Установка gitflow в режиме суперпользователя	8
4.2	Установка приложений	8
4.3	Настройка node.js	9
4.4	Настройка программ	9
4.5	Создание репозитория, первый коммит	10
4.6	Настройка пакета	10
4.7	Изменения файла	10
4.8	Выполнение коммита	10
4.9	Команда push	11
4.10	Инициализация gitflow	11
4.11	Установка внешней ветки	12
4.12	Создание релиза и журнала изменений	13
4.13	Добавление релизной ветки в основную	14
4.14	Команды push -all и push -tags	15

Список таблиц

1 Цель работы

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 Теоретическое введение

3.1 Рабочий процесс Gitflow

Рабочий процесс Gitflow Workflow. Будем описывать его с использованием пакета git-flow. Общая информация Gitflow Workflow опубликована и популяризована Винсентом Дриссенем. Gitflow Workflow предполагает выстраивание строгой модели ветвления с учётом выпуска проекта. Данная модель отлично подходит для организации рабочего процесса на основе релизов. Работа по модели Gitflow включает создание отдельной ветки для исправлений ошибок в рабочей среде. Последовательность действий при работе по модели Gitflow: Из ветки master создаётся ветка develop. Из ветки develop создаётся ветка release. Из ветки develop создаются ветки feature. Когда работа над веткой feature завершена, она сливается с веткой develop. Когда работа над веткой релиза release завершена, она сливается в ветки develop и master. Если в master обнаружена проблема, из master создаётся ветка hotfix. Когда работа над веткой исправления hotfix завершена, она сливается в ветки develop и master.

4 Выполнение лабораторной работы

4.1 Установка программного обеспечения.

4.1.1 Установка git-flow

1. Открываем терминал и входим в режим суперпользователя, устанавливаем gitflow. (рис. 4.1).

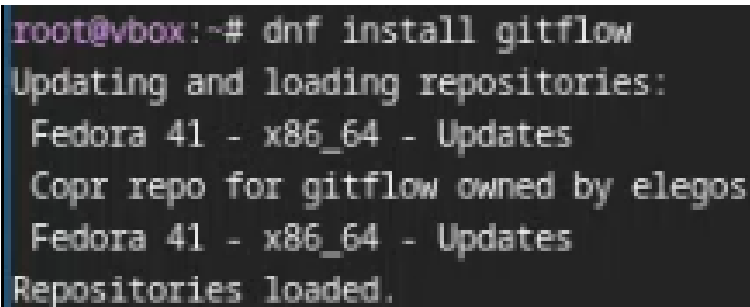


```
vakomarov@vbox:~$ dnf copr enable elegos/gitflow
```

Рис. 4.1: Установка gitflow в режиме суперпользователя

4.1.2 Установка и настройка node.js

2. Устанавливаем rpm и nodejs. (рис. 4.2).



```
root@vbox:~# dnf install gitflow
Updating and loading repositories:
Fedora 41 - x86_64 - Updates
Copr repo for gitflow owned by elegos
Fedora 41 - x86_64 - Updates
Repositories loaded.
```

Рис. 4.2: Установка приложений

3. Настраиваем nodejs. (рис. 4.3).


```
root@vbox:~# dnf install nodejs
Updating and loading repositories:
Repositories loaded.
```

Рис. 4.3: Настройка node.js

4.1.3 Общепринятые коммиты.

4. Настраиваем commitizen и standard-changelog (рис. 4.4).

```
root@vbox:~# dnf install pnpm
Updating and loading repositories:
Repositories loaded.
Package
Installing:
```

Рис. 4.4: Настройка программ

4.2 Практический сценарий использования git.

4.2.1 Создание репозитория git.

5. Создаем репозиторий git, настраиваем его и делаем в него первый коммит. (рис. 4.5).

```

root@vbox:~# pnpm setup
Appended new lines to /root/.bashrc

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

```

Рис. 4.5: Создание репозитория, первый коммит

6. Настраиваем пакет файлов nodejs (рис. 4.6). В файле package.json меняем необходимые данные. (рис. 4.7).

```

root@vbox:~# source ~/.bashrc

```

Рис. 4.6: Настройка пакета

```

root@vbox:~# pnpm add -g commitizen

```

Update available! 9.13.0 -> 10.6.1.
 Changelog: <https://github.com/pnpm/pnpm/releases/tag/v10.6.1>
 Run "pnpm self-update" to update.
 Follow @pnpmjs for updates: <https://x.com/pnpmjs>

Рис. 4.7: Изменения файла

7. Выполняем коммит (рис. 4.8). Выкладываем на github. (рис. 4.9).

```

root@vbox:~# pnpm add -g standart-changelog

```

Рис. 4.8: Выполнение коммита

```

vakomarov@vbox: ~/git-extended$ git commit -m "first commit"
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
vakomarov@vbox: ~/git-extended$ git push -u origin master
branch 'master' set up to track 'origin/master'.
Everything up-to-date
vakomarov@vbox: ~/git-extended$ git commit -m "first commit"
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
vakomarov@vbox: ~/git-extended$ git remote add origin git@github.com:kerfarion/git-extended.git
error: remote origin already exists.
vakomarov@vbox: ~/git-extended$ npm init
Wrote to /home/vakomarov/git-extended/package.json

{
  "name": "git-extended",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "keywords": [],

```

Рис. 4.9: Команда push

8. Инициализируем gitflow, проверяем, на какой ветке мы находимся в данный момент, после чего загружаем весь репозиторий в хранилище. (рис. 4.10).

```

vakomarov@vbox: ~/git-extended$ git add
vakomarov@vbox: ~/git-extended$ git cz
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) package.json
? Write a short, imperative tense description of the change (max 80 chars):
(9) to config
? Provide a longer description of the change: (press enter to skip)

? Are there any breaking changes? Yes
? A BREAKING CHANGE commit requires a body. Please enter a longer description of the commit itself:
-
? Describe the breaking changes:

? Does this change affect any open issues? Yes
? Add issue references (e.g. "fix #123", "re #123").:
fix #123
[master c2505a4] feat(package.json): to config
1 file changed, 14 insertions(+)
create mode 100644 package.json
vakomarov@vbox: ~/git-extended$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.17 KiB | 399.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:kerfarion/git-extended.git
2060aac..c2505a4 master -> master
vakomarov@vbox: ~/git-extended$

```

Рис. 4.10: Инициализация gitflow

9. Устанавливаем внешнюю ветку как вышестоящую для этой ветки (рис. 4.11).

Создаем релиз с версией 1.0.0 и журнал изменений. (рис. 4.12).

```
vakomarov@vbox:~/git-extended$ git flow init

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

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

Рис. 4.11: Установка внешней ветки

```

akomarov@vbox: ~/git-extended$ standard-changelog --first-release
# created CHANGELOG.md
# output changes to CHANGELOG.md
akomarov@vbox: ~/git-extended$ git add CHANGELOG.md
akomarov@vbox: ~/git-extended$ git commit -am 'chore(site): add changelog'
releasel.0.0 64844e6] chore(site): add changelog
1 file changed, 10 insertions(+)
create mode 100644 CHANGELOG.md
akomarov@vbox: ~/git-extended$ git flow release finish 1.0.0
ranches 'develop' and 'origin/develop' have diverged.
nd local branch 'develop' is ahead of 'origin/develop'.
witched to branch 'master'
our branch is up to date with 'origin/master'.
erge made by the 'ort' strategy.
CHANGELOG.md | 10 ++++++++
package.json | 2 +-
2 files changed, 11 insertions(+), 1 deletion(-)
create mode 100644 CHANGELOG.md
lready on 'master'
our branch is ahead of 'origin/master' by 3 commits.
(use "git push" to publish your local commits)
atal: no tag message?
atal: Tagging failed. Please run finish again to retry.
akomarov@vbox: ~/git-extended$ git flow release finish -m "Release version 1.0.0" 1.0.0
ranches 'master' and 'origin/master' have diverged.
nd local branch 'master' is ahead of 'origin/master'.
ranches 'develop' and 'origin/develop' have diverged.
nd local branch 'develop' is ahead of 'origin/develop'.
lready on 'master'
our branch is ahead of 'origin/master' by 3 commits.
(use "git push" to publish your local commits)
witched to branch 'develop'
our branch is ahead of 'origin/develop' by 1 commit.
(use "git push" to publish your local commits)
erge made by the 'ort' strategy.
CHANGELOG.md | 10 ++++++++
1 file changed, 10 insertions(+)
create mode 100644 CHANGELOG.md
eleted branch releasel.0.0 (was 64844e6).

Summary of actions:
Release branch 'releasel.0.0' has been merged into 'master'
The release was tagged 'v1.0.0'
Release tag 'v1.0.0' has been back-merged into 'develop'
Release branch 'releasel.0.0' has been locally deleted
You are now on branch 'develop'

```

Рис. 4.12: Создание релиза и журнала изменений

10. Заливаем релизную ветку в основную, добавляем журнал изменений в индекс, после чего заливаем релизную ветку в основную. (рис. 4.13).

```
vakomarov@vbox:~/git-extended$ git push --all
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 4 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 3.83 KiB | 3.83 MiB/s, done.
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:kerfarion/git-extended.git
   c2505a4..8954411 develop -> develop
   c2505a4..651bd22 master -> master
vakomarov@vbox:~/git-extended$ git push --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 177 bytes | 177.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
40 github.com:kerfarion/git-extended.git
   * [new tag]           v1.0.0 -> v1.0.0
vakomarov@vbox:~/git-extended$
```

Рис. 4.13: Добавление релизной ветки в основную

11. Отправляем данные и теги на гитхаб. (рис. 4.14).

```

vakomarov@vbox:~/git-extended$ gh release create v1.0.0 -F CHANGELOG.md
https://github.com/kerfarion/git-extended/releases/tag/v1.0.0
vakomarov@vbox:~/git-extended$ git flow feature start feathure_branche
Switched to a new branch 'featurefeathure_branche'

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

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

    git flow feature finish feathure_branche

vakomarov@vbox:~/git-extended$ git flow feature finish feature_branch
Fatal: Branch 'featurefeathure_branch' does not exist and is required.
vakomarov@vbox:~/git-extended$ git flow feature finish feature_branche
Fatal: Branch 'featurefeathure_branche' does not exist and is required.
vakomarov@vbox:~/git-extended$ git flow feature finish feathure_branche
Switched to branch 'develop'
Your branch is up to date with 'origin/develop'.
Already up to date.
Deleted branch featurefeathure_branche (was 8954411).

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

vakomarov@vbox:~/git-extended$ git flow release start 1.2.3
Switched to a new branch 'release1.2.3'

Summary of actions:
- A new branch 'release1.2.3' was created, based on 'develop'
- You are now on branch 'release1.2.3'

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'

vakomarov@vbox:~/git-extended$ standard-changelog
✓ output changes to CHANGELOG.md
vakomarov@vbox:~/git-extended$ git add CHANGELOG.md
vakomarov@vbox:~/git-extended$ git commit -am 'chore(site): update changelog'
[release1.2.3 ffd6966] chore(site): update changelog
2 files changed, 5 insertions(+), 1 deletion(-)
vakomarov@vbox:~/git-extended$

```

Рис. 4.14: Команды push –all и push –tags

4.2.2 Работа с репозиторием git.

12. Создаем релиз на гитхабе. Создаем ветку для новой функциональности.
13. Объединяем новую ветку с develop. Создаём релиз с версией 1.2.3.

14. Изменяем номер версии в файле package.json.
15. Заливаем релизную ветку в основную. Отправляем данные на гитхаб.
16. Создаём релиз на гитхабе с комментарием из журнала изменений.

5 Выводы

В процессе выполнения лабораторной работы я приобрел навыки правильной работы с репозиториями git.

Список литературы