

Отчет по лабораторной работе №2

Операционные системы

Комаров Владимир Артемович.

Российский университет дружбы народов, Москва, Россия

Информация

- Комаров Владимир Артемович
- НКАбд-02-2024 № Студенческого билета: 1132246757
- Российский университет дружбы народов
- https://github.com/kerfarion/study_2024-2025_os-intro

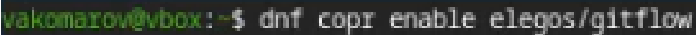


Получение навыков правильной работы с репозиториями git.

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

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`.

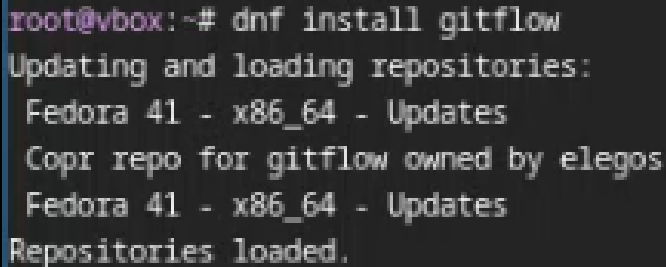
1. Открываем терминал и входим в режим суперпользователя, устанавливаем gitflow.



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

Рис. 1: Установка gitflow в режиме суперпользователя

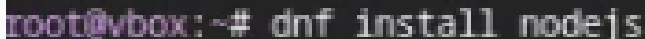
2. Устанавливаем rpm и nodejs.

A terminal window with a dark background and light-colored text. The prompt is 'root@vbox:~#'. The command 'dnf install gitflow' has been entered. The output shows the process of updating and loading repositories, listing 'Fedora 41 - x86_64 - Updates' and 'Copr repo for gitflow owned by elegos', and finally stating 'Repositories loaded.'

```
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.
```

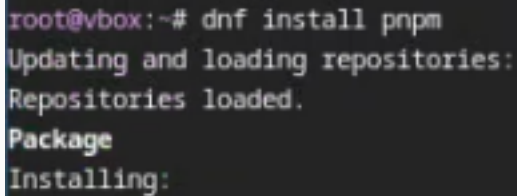
Рис. 2: Установка приложений

3. Настраиваем nodejs.

A terminal window with a dark background and light-colored text. The prompt is 'root@vbox:~#'. The command 'dnf install nodejs' has been entered.

```
root@vbox:~# dnf install nodejs
```

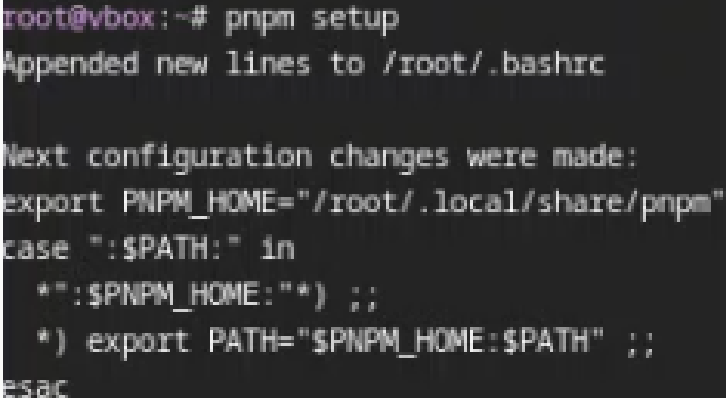
4. Настраиваем commitizen и standard-changelog.



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

Рис. 4: Настройка программ

5. Создаем репозиторий git, настраиваем его и делаем в него первый коммит.

A terminal window with a dark background and light-colored text. The prompt is 'root@vbox:~#'. The command 'pnpm setup' has been executed. The output shows that new lines were appended to '/root/.bashrc' and lists the configuration changes made to the environment variables.

```
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. 5: Создание репозитория, первый коммит

6. Настраиваем пакет файлов nodejs. В файле package.json меняем необходимые данные.

```
root@vbox:~# source ~/.bashrc
```

Рис. 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. 7: Изменения файла

7. Выполняем коммит. Выкладываем на github.

```
oot@vbox:~# npm add -g standart-changeloc
```

Рис. 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": ""
}
```

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

```
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$ 1
```

9. Устанавливаем внешнюю ветку как вышестоящую для этой ветки. Создаем релиз с версией 1.0.0 и журнал изменений.

```
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. 11: Установка внешней ветки

```
vakomarov@vbox:~/git-extended$ standard-changelog --first-release
created CHANGELOG.md
output changes to CHANGELOG.md
vakomarov@vbox:~/git-extended$ git add CHANGELOG.md
```


10. Заливаем релизную ветку в основную, добавляем журнал изменений в индекс, после чего заливаем релизную ветку в основную.

```
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)
To github.com:kerfarion/git-extended.git
   * [new tag]         v1.0.0 -> v1.0.0
vakomarov@vbox:~/git-extended$
```

11. Отправляем данные и теги на гитха.

```
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 'releasel.2.3'

Summary of actions:
- A new branch 'releasel.2.3' was created, based on 'develop'
- You are now on branch 'releasel.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'
[releasel.2.3 ffd6966] chore(site): update changelog
2 files changed, 5 insertions(+), 1 deletion(-)
vakomarov@vbox:~/git-extended$
```

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.

1. Кулябов Д. С. Введение в операционную систему UNIX - Лекция.
2. Таненбаум Э., Бос Х. Современные операционные системы. - 4-е изд. -СПб. : Питер, 2015. - 1120 с.

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