

Лабораторная работа №4

Презентация

Мосолов А.Д.

09 марта 2024

Российский университет дружбы народов, Москва, Россия

- Мосолов Александр Денисович
- Студент, НПИбд02-23
- Российский университет дружбы народов
- 1132236128@pfur.ru

Получение навыков правильной работы с репозиториями git.

Выполнить работу для тестового репозитория.

Преобразовать рабочий репозиторий в репозиторий с git-flow и conventional commits.

Enable the copr repository

Установка из коллекции репозитория Copr

```
[root@alnos05 ~]# dnf copr enable elegos/gitflow  
Включение репозитория Copr. Обратите внимание, что этот репозиторий  
не является частью основного дистрибутива, и качество может отличаться.
```

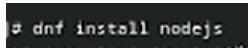
Рис. 1: Enable the copr repository

Установка gitflow

```
[root@elnos05 ~]# dnf install gitflow  
Copr repo for gitflow owned by elegos 2.4 kB/s |
```

Рис. 2: Install gitflow

Установка Node.js



```
# dnf install nodejs
```

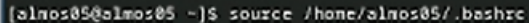
Рис. 3: Установка Node.js

Установка pnpm с помощью wget

```
[almos05@almos05 ~]$ wget -qO- https://get.pnpm.io/install.sh | sh -  
==> Downloading pnpm binaries 8.15.4  
WARN using --force I sure hope you know what you are doing  
Copying pnpm CLI from /tmp/tmp.SvqTY7X91H/pnpm to /home/almos05/.local/s
```

Рис. 4: Установка pnpm

Запускаем rprn и выполняем

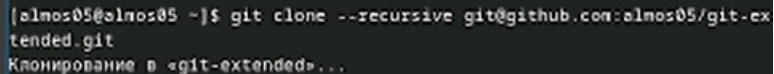
A terminal window showing a command being executed. The prompt is [almos05@almos05 ~]\$. The command is source /home/almos05/.bashrc. The output is not visible.

```
[almos05@almos05 ~]$ source /home/almos05/.bashrc
```

Рис. 5: Запускаем rprn

Клонируем репозиторий

Создаем пустой репозиторий и клонируем его

A terminal window with a dark background and light-colored text. The prompt is [almos05@almos05 ~]. The command entered is git clone --recursive git@github.com:almos05/git-extended.git. The output shows the cloning progress, with the text "Клонирование в «git-extended»..." visible.

```
[almos05@almos05 ~]$ git clone --recursive git@github.com:almos05/git-extended.git
Клонирование в «git-extended»...
```

Рис. 6: Клонируем репозиторий

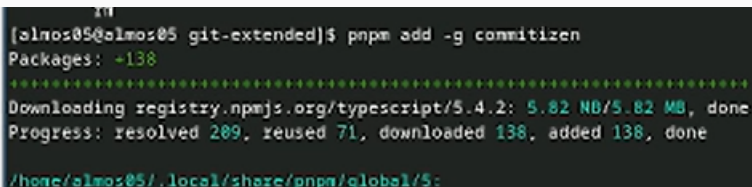
standard-changelog используется для помощи в создании логов

```
[almos05@almos05 git-extended]$ pnpm add -g standard-changelog
Packages: +71
Progress: resolved 71, reused 0, downloaded 71, added 71, done

/home/almos05/.local/share/pnpm/global/5:
+ standard-changelog 5.0.0
```

Рис. 7: Standard-changelog

Commitizen используется для помощи в форматировании коммитов

A terminal window showing the command 'pnpm add -g commitizen' being executed. The output shows that 138 packages were added, including TypeScript 5.4.2, which was downloaded from the registry. The progress bar indicates that 289 packages were resolved, 71 were reused, 138 were downloaded, and 138 were added. The final line shows the installation path: /home/almos05/.local/share/pnpm/global/5:

```
[almos05@almos05 git-extended]$ pnpm add -g commitizen
Packages: +138
+-----+
Downloading registry.npmjs.org/typescript/5.4.2: 5.82 MB/5.82 MB, done
Progress: resolved 289, reused 71, downloaded 138, added 138, done

/home/almos05/.local/share/pnpm/global/5:
```

Рис. 8: Commitizen

Первый коммит

Создаем пустой файл README.md, фиксируем изменения и делаем первый КОММИТ

```
[almos05@almos05 git-extended]$ touch README.md  
[almos05@almos05 git-extended]$ git add .  
[almos05@almos05 git-extended]$ git commit -m "first commit"  
[main (корневой коммит) 993240c] first commit  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 README.md
```

Рис. 9: Первый коммит

Делаем git push в ветку main

```
[almos05@almos05 git-extended]$ git push -u origin main
Перечисление объектов: 3, готово.
Подсчет объектов: 100% (3/3), готово.
Запись объектов: 100% (3/3), 874 байта | 874.00 КиБ/с, готово.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:almos05/git-extended.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

Рис. 10: git push в ветку main

Конфигурация для пакетов Node.js

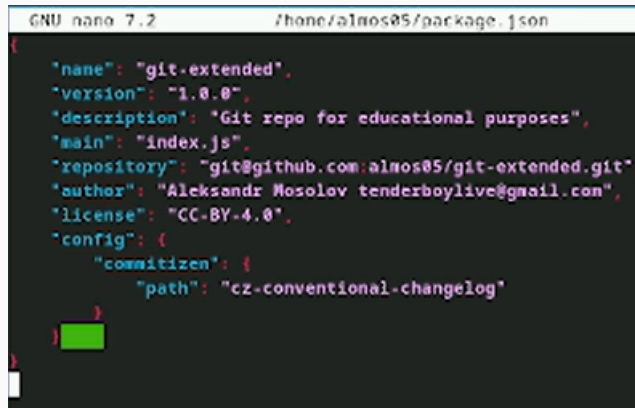
```
[almos05@almos05 ~]$ pnpm init
Wrote to /home/almos05/package.json

{
  "name": "almos05",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
[almos05@almos05 ~]$
```

Рис. 11: pnpm init

Изменяем package.json

Изменяем package.json



```
GNU nano 7.2 /home/almos05/package.json
{
  "name": "git-extended",
  "version": "1.0.0",
  "description": "Git repo for educational purposes",
  "main": "index.js",
  "repository": "git@github.com:almos05/git-extended.git",
  "author": "Aleksandr Mosolov tenderboylove@gmail.com",
  "license": "CC-BY-4.0",
  "config": {
    "commitizen": {
      "path": "cz-conventional-changelog"
    }
  }
}
```

Рис. 12: Изменяем package.json

Добавим новые файлы:

```
git add .
```

Выполним коммит:

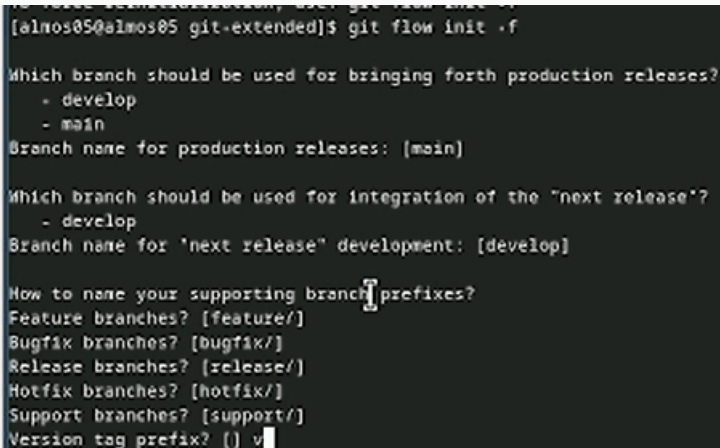
```
git cz
```

Отправим на github:

```
git push
```

Инициализируем git-flow

Инициализируем git-flow, префикс для ярлыков установим в v



```
[almos05@almos05 git-extended]$ git flow init .f

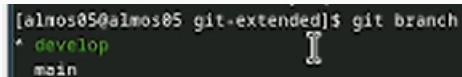
Which branch should be used for bringing forth production releases?
  - develop
  - main
Branch name for production releases: [main]

Which branch should be used for integration of the "next release"?
  - develop
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. 13: Инициализируем git-flow

Проверьте, что Вы на ветке develop

A terminal window with a dark background. The prompt is [almos05@almos05 git-extended]\$ and the command is git branch. The output shows two branches: develop (highlighted in green) and main. A cursor is positioned at the end of the line.

```
[almos05@almos05 git-extended]$ git branch
* develop
main
```

Рис. 14: Ветка develop

Загружаем весь репозиторий в хранилище

```
[almos05@almos05 git-extended]$ git push --all  
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)  
remote:
```

Рис. 15: git push -all

Внешняя ветка становится вышестоящей

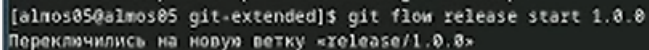
Установим внешнюю ветку как вышестоящую для этой ветки

```
[alnos05@alnos05 git-extended]$ git branch --set-upstream-to=origin/develop develop  
branch 'develop' set up to track 'origin/develop'.  
[alnos05@alnos05 git-extended]$
```

Рис. 16: Внешняя ветка становится вышестоящей

Создадим релиз с версией 1.0.0

Создадим релиз с версией 1.0.0



```
[almos05@almos05 git-extended]$ git flow release start 1.0.0  
Переключились на новую ветку «release/1.0.0»
```

Рис. 17: Создадим релиз с версией 1.0.0

Добавим журнал изменений в индекс

Создадим журнал изменений
standard-changelog –first-release

И добавим журнал изменений в индекс

```
[almos05@almos05 git-extended]$ git add CHANGELOG.md  
[almos05@almos05 git-extended]$ git commit -m 'chore(site): add changelog'  
[release/1.0.0 1ec0408] chore(site): add changelog  
1 file changed, 9 insertions(+)  
create mode 100644 CHANGELOG.md
```

Рис. 18: Добавим журнал изменений в индекс

Отправим данные на github

Зальём релизную ветку в основную ветку

git flow release finish 1.0.0

Отправим данные на github

```
[alnos05@alnos05 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), 2.80 КиБ | 2.80 КиБ/с, готово.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:alnos05/git-extended.git
  9028aa5..6400471 develop -> develop
  9028aa5..00220f1 main -> main
[alnos05@alnos05 git-extended]$ git push --tags
Перечисление объектов: 1, готово.
Подсчет объектов: 100% (1/1), готово.
Запись объектов: 100% (1/1), 165 байтов | 165.00 КиБ/с, готово.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:alnos05/git-extended.git
  * [new tag]          v1.0.0 -> v1.0.0
[alnos05@alnos05 git-extended]$
```

Рис. 19: Отправим данные на github

Создадим релиз на github

Создадим релиз на github. Для этого будем использовать утилиты работы с github

```
[almos05@almos05 git-extended]$ gh release create v1.0.0 -F CHANGELOG.m
https://github.com/almos05/git-extended/releases/tag/v1.0.0
[almos05@almos05 git-extended]$
```

Рис. 20: Создадим релиз на github

Создадим ветку для новой функциональности

Создадим ветку для новой функциональности



```
[almos05@almos05 git-extended]$ git flow feature start feature_branch
Переключились на новую ветку «feature/feature_branch»

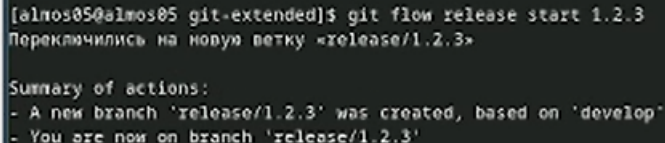
Summary of actions:
- A new branch 'feature/feature_branch' was created, based on 'develop'
- You are now on branch 'feature/feature_branch'

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

Рис. 21: Создадим ветку для новой функциональности

Создадим релиз с версией 1.2.3

Создадим релиз с версией 1.2.3



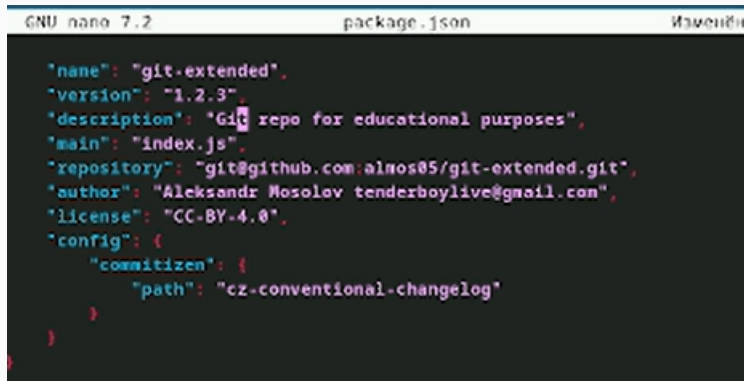
```
[almos05@almos05 git-extended]$ git flow release start 1.2.3
Переключились на новую ветку «release/1.2.3»

Summary of actions:
- A new branch 'release/1.2.3' was created, based on 'develop'
- You are now on branch 'release/1.2.3'
```

Рис. 22: Создадим релиз с версией 1.2.3

Обновите номер версии в файле package.

Обновите номер версии в файле package.json. Установите её в 1.2.3

A screenshot of a terminal window with a dark background. The title bar at the top shows 'GNU nano 7.2', 'package.json', and 'Измeнeн'. The main area displays the content of the package.json file in a syntax-highlighted format. The JSON object includes fields for name, version, description, main, repository, author, license, and config. The 'version' field is set to '1.2.3'.

```
GNU nano 7.2 package.json Измeнeн

{
  "name": "git-extended",
  "version": "1.2.3",
  "description": "Git repo for educational purposes",
  "main": "index.js",
  "repository": "git@github.com:alnos05/git-extended.git",
  "author": "Aleksandr Mosolev tenderboylive@gmail.com",
  "license": "CC-BY-4.0",
  "config": {
    "commitizen": {
      "path": "cz-conventional-changelog"
    }
  }
}
```

Рис. 23: Обновите номер версии в файле package.json

Создадим журнал изменений

Создадим журнал изменений

```
[almos05@almos05 git-extended]$ standard-changelog  
✓ output changes to CHANGELOG.md  
[almos05@almos05 git-extended]$ git add CHANGELOG.md
```

Рис. 24: Создадим журнал изменений

Добавим журнал изменений в индекс

Добавим журнал изменений в индекс

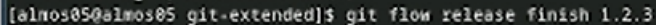
git add CHANGELOG.md, сделаем новый КОММИТ

```
[almos05@almos05 git-extended]$ git commit -am 'chore(site): update changelog'
[feature/feature_branch 5212aac] chore(site): update changelog
1 file changed, 12 insertions(+)
[almos05@almos05 git-extended]$
```

Рис. 25: Добавим журнал изменений в индекс

Зальём релизную ветку в основную ветку

Зальём релизную ветку в основную ветку



```
[almos05@almos05 git-extended]$ git flow release finish 1.2.3
```

Рис. 26: Зальём релизную ветку в основную ветку

Отправим данные на github

Отправим данные на github

```
[almos05@almos05 git-extended]$ git push --all
gПеречисление объектов: 7, готово.
Подсчет объектов: 100% (7/7), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (5/5), готово.
Запись объектов: 100% (5/5), 2.82 КиБ | 2.82 МБ/с, готово.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
it To github.com:almos05/git-extended.git
    6400471..e34069d develop -> develop
    00220f1..a8c6d30 main -> main
* [new branch]      feature/feature_branch -> feature/feature_branch
[almos05@almos05 git-extended]$ git push --tags
```

Рис. 27: Отправим данные на github

Создадим релиз на github

Зальём релизную ветку в основную Создадим релиз на github с комментарием из журнала изменений]):

```
[almos05@almos05 git-extended]$ gh release create v1.2.3 -F CHANGELOG.md  
https://github.com/almos05/git-extended/releases/tag/v1.2.3  
[almos05@almos05 git-extended]$
```

Рис. 28: Создадим релиз на github с комментарием из журнала изменений

В ходе работы мы получили навыки правильной работы с репозиториями git.