

Лабораторная работа №4

Продвинутое использование git

Буллер Т. А.

8 марта 2024

Российский университет дружбы народов, Москва, Россия

Информация

- Буллер Татьяна Александровна
- студент группы НБИбд-01-23
- Российский университет дружбы народов

Вводная часть

- Система контроля версий git
- Менеджер пакетов rpm
- Методы создания шаблонных коммитов и версий релизов

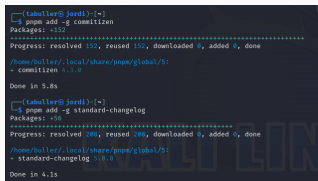
- Получение навыков правильной работы с репозиториями git.

- Система контроля версий `git`
- Менеджер пакетов `pip`
- Программы `commitizen` и `standard-changelog` для создания шаблонных коммитов
- Процессор `pandoc` для входного формата `Markdown`
- Результирующие форматы
 - `pdf`
 - `html`
- Автоматизация процесса создания: `Makefile`

Выполнение лабораторной работы

Добавление общепринятых коммитов.

С помощью утилиты `pnpm` добавляем две программы: `standard changelog` и `commitizen`.



```
(tabuller@jordi)~$ pnpm add -g commitizen
Packages: +152
Progress: resolved 152, reused 152, downloaded 0, added 0, done
/home/buller/.local/share/pnpm/global/5:
+ commitizen 4.1.0
Done in 5.8s

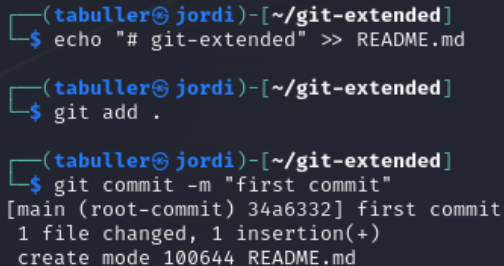
(tabuller@jordi)~$ pnpm add -g standard-changelog
Packages: +56
Progress: resolved 288, reused 288, downloaded 0, added 0, done
/home/buller/.local/share/pnpm/global/5:
+ standard-changelog 5.0.0
Done in 4.1s
```

Рис. 1: Добавление программ с помощью `pnpm`

Программы добавлены. В дальнейшем они будут использоваться для добавления коммитов и создания лога изменений.

Создание первого коммита в репозитории.

Репозиторий создаем через сайт github, после чего добавляем его на виртуальную машину. Для первого коммита создадим README-файл и добавим его:

A terminal window with a dark background and light blue text. It shows three commands being executed in a shell. The first command creates a README.md file with the content "# git-extended". The second command stages the file for commit. The third command commits the file with the message "first commit". The output shows the commit hash 34a6332, the commit message, and details about the file changes.

```
(tabuller@jordi)-[~/git-extended]
$ echo "# git-extended" >> README.md

(tabuller@jordi)-[~/git-extended]
$ git add .

(tabuller@jordi)-[~/git-extended]
$ git commit -m "first commit"
[main (root-commit) 34a6332] first commit
1 file changed, 1 insertion(+)
create mode 100644 README.md
```

Рис. 2: Первый коммит в тестовый репозиторий

Создание первого коммита в репозитории.

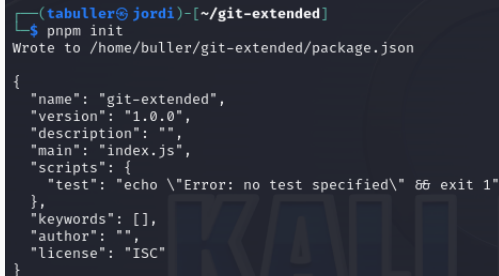
```
(tabuller@jordi)-[~/git-extended]
$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 881 bytes | 881.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:sarykush/git-extended.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

Рис. 3: Отправление первого коммита

Файл отправлен с первым коммитом, работу можно продолжать дальше.

Конфигурация общепринятых коммитов.

Введем команду `pnpm init`. Вывод команды демонстрирует содержание файла и его место в системе.



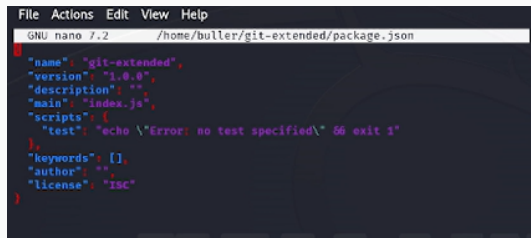
```
(tabuller@jordi)-[~/git-extended]
$ pnpm init
Wrote to /home/buller/git-extended/package.json

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

Рис. 4: `pnpm init`

Конфигурация общепринятых коммитов.

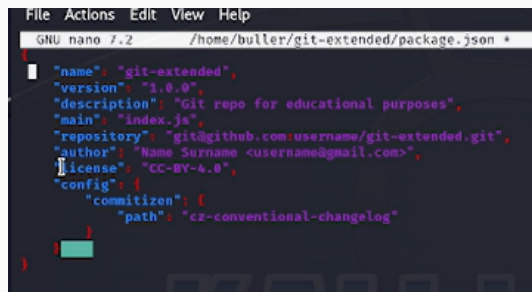
Откроем файл, адрес которого видим в выводе команды, и отредактируем его так, как указано в задании лабораторной работы:



```
File Actions Edit View Help
GNU nano 7.2 /home/buller/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": [],
"author": "",
"license": "ISC"
}
```

Рис. 5: Изначальный файл package.json

Конфигурация общепринятых коммитов.

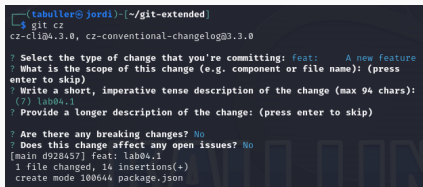


```
File Actions Edit View Help
GNU nano 7.2 /home/buller/git-extended/package.json *
{
  "name": "git-extended",
  "version": "1.0.0",
  "description": "Git repo for educational purposes",
  "main": "index.js",
  "repository": "git@github.com:username/git-extended.git",
  "author": "Name Surname <username@gmail.com>",
  "license": "CC-BY-4.0",
  "config": {
    "commitizen": {
      "path": "cz-conventional-changelog"
    }
  }
}
```

Рис. 6: Отредактированный файл package.json

Конфигурация общепринятых коммитов.

После редактирования файла необходимо добавить файлы в репозиторий и выполнить коммит с помощью установленного ранее скрипта.



```
(tabuller@jordi) - [~/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)
? Write a short, imperative tense description of the change (max 94 chars):
  (7) lab04.1
? 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 d928457] feat: lab04.1
1 file changed, 14 insertions(+)
create mode 100644 package.json
```

Рис. 7: Использование скрипта cz для коммита

Коммит успешно настроен и отправлен.

Инициализируем git-flow и проверим, что мы находимся на нужной ветке (develop):

```
(tabuller@jordi)-[~/git-extended]
$ git flow init

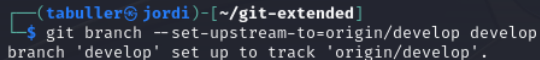
Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

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/buller/git-extended/.git/hooks]

(tabuller@jordi)-[~/git-extended]
$ git branch
* develop
  main
```

Рис. 8: Инициализация git-flow

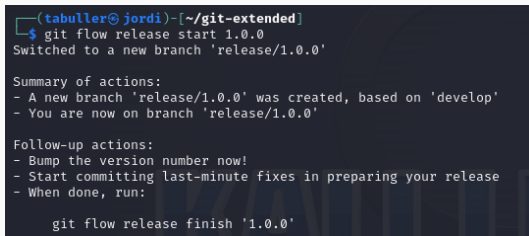
Названия веток оставляем по умолчанию, для версий добавляем перфикс v: каждая новая версия будет выглядеть как vNN.NN.NN. Отправляем изменения на гитхаб. Следующее, что нужно сделать - установить внешнюю ветку как вышестоящую для этой ветки.

A terminal window with a dark background. The prompt is `(tabuller@jordi)-[~/git-extended]`. The command entered is `$ git branch --set-upstream-to=origin/develop develop`. The output is `branch 'develop' set up to track 'origin/develop'.`

```
(tabuller@jordi)-[~/git-extended]
$ git branch --set-upstream-to=origin/develop develop
branch 'develop' set up to track 'origin/develop'.
```

Рис. 9: Установка вышестоящей ветки

После того, как работа с ветками закончена, создаем новый релиз: 1.0.0



```
(tabuller@jordi)-[~/git-extended]
$ git flow release start 1.0.0
Switched to a new branch '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. 10: Создание первого релиза

Конфигурация git-flow.

Далее настраиваем релиз: создаем журнал изменений, добавляем его в индекс и заливаем релиз в основную ветку.

```
(tabuller@jordi)-[~/git-extended]
$ standard-changelog --first-release
✓ created CHANGELOG.md
✓ output changes to CHANGELOG.md

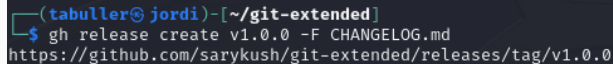
(tabuller@jordi)-[~/git-extended]
$ git add CHANGELOG.md

(tabuller@jordi)-[~/git-extended]
$ git commit -am 'chore(site): add changelog'
[release/1.0.0 91b9a1b] chore(site): add changelog
1 file changed, 9 insertions(+)
create mode 100644 CHANGELOG.md

(tabuller@jordi)-[~/git-extended]
$ git flow release finish 1.0.0
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
Merge made by the 'ort' strategy.
CHANGELOG.md | 9 ++++++++
1 file changed, 9 insertions(+)
create mode 100644 CHANGELOG.md
Already on 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)
```

Рис. 11: Работа с релизом

После того, как отправили данные, создаем релиз на гитхаб и получаем ссылку на него:

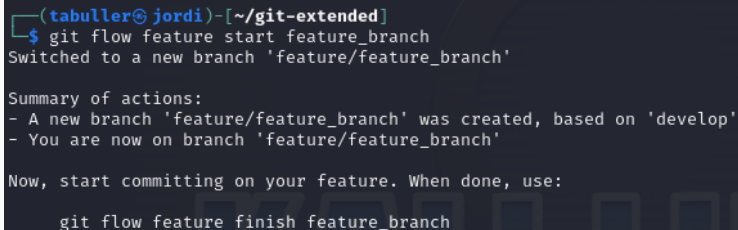
A terminal window with a dark background. The prompt is `(tabuller@jordi)-[~/git-extended]`. The command entered is `$ gh release create v1.0.0 -F CHANGELOG.md`. The output is the URL `https://github.com/sarykush/git-extended/releases/tag/v1.0.0`.

```
(tabuller@jordi)-[~/git-extended]  
$ gh release create v1.0.0 -F CHANGELOG.md  
https://github.com/sarykush/git-extended/releases/tag/v1.0.0
```

Рис. 12: Создание итогового релиза

Разработка новой функциональности.

Создадим ветку для новой функциональности:



```
(tabuller@jordi)-[~/git-extended]
$ git flow feature start feature_branch
Switched to a new branch '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'

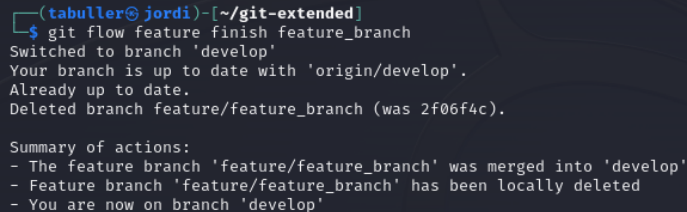
Now, start committing on your feature. When done, use:

    git flow feature finish feature_branch
```

Рис. 13: Создание новой ветки

Разработка новой функциональности.

Далее работа с гит продолжается как обычно. Для тестового репозитория никакую новую функциональность разрабатывать не будем, переходим сразу к следующему шагу:



```
(tabuller@jordi)-[~/git-extended]
$ git flow feature finish feature_branch
Switched to branch 'develop'
Your branch is up to date with 'origin/develop'.
Already up to date.
Deleted branch feature/feature_branch (was 2f06f4c).

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. 14: Заккрытие ветки

Созданная нами ветка объединилась с веткой develop.

Создание релиза.

Создадим релиз с версией 1.2.3 и оказываемся на ветке release/1.2.3:

```
(tabuller@jordi)-[~/git-extended]
$ git flow release start 1.2.3
Switched to a new branch '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'

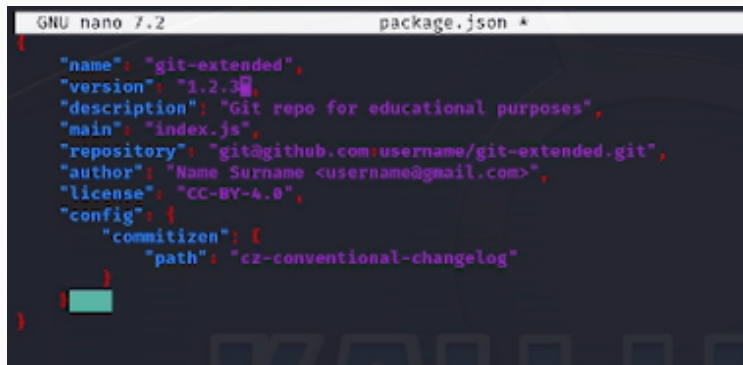
Follow-up actions:
- Bump the version number now!
- Start committing last-minute fixes in preparing your release
- When done, run:

    git flow release finish '1.2.3'
```

Рис. 15: Создание нового релиза

Создание релиза.

В файле `package.json` обновляем номер версии. Этот файл находится прямо в репозитории, в котором мы работаем, и копирует в себе данные, которые мы настроили на первых этапах:



```
GNU nano 7.2 package.json *
{
  "name": "git-extended",
  "version": "1.2.3",
  "description": "Git repo for educational purposes",
  "main": "index.js",
  "repository": "git@github.com:username/git-extended.git",
  "author": "Name Surname <username@gmail.com>",
  "license": "CC-BY-4.0",
  "config": {
    "commitizen": {
      "path": "cz-conventional-changelog"
    }
  }
}
```

Рис. 16: Редактирование json файла

Создание релиза.

Когда редактирование завершено, обновляем журнал изменений и добавляем его на гитхаб:

```
(tabuller@jordi)-[~/git-extended]
$ nano package.json

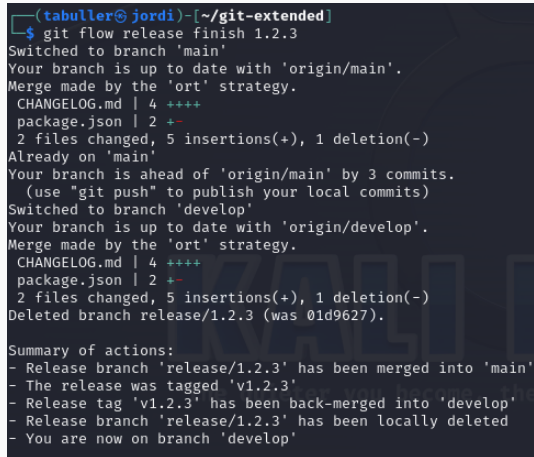
(tabuller@jordi)-[~/git-extended]
$ standard-changelog
✓ output changes to CHANGELOG.md

(tabuller@jordi)-[~/git-extended]
$ git add CHANGELOG.md

(tabuller@jordi)-[~/git-extended]
$ git commit -am 'chore(site): update changelog'
[release/1.2.3 01d9627] chore(site): update changelog
2 files changed, 5 insertions(+), 1 deletion(-)
```

Рис. 17: Обновление журнала изменений

Заливаем релиз в основную ветку, после чего отправляем данные на гитхаб:

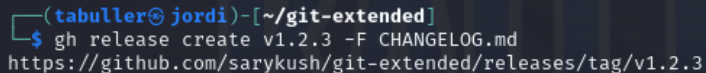


```
(tabuller@jordi)~[~/git-extended]
$ git flow release finish 1.2.3
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
Merge made by the 'ort' strategy.
  CHANGELOG.md | 4 ++++
  package.json | 2 +-
  2 files changed, 5 insertions(+), 1 deletion(-)
Already on 'main'
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)
Switched to branch 'develop'
Your branch is up to date with 'origin/develop'.
Merge made by the 'ort' strategy.
  CHANGELOG.md | 4 ++++
  package.json | 2 +-
  2 files changed, 5 insertions(+), 1 deletion(-)
Deleted branch release/1.2.3 (was 01d9627).

Summary of actions:
- Release branch 'release/1.2.3' has been merged into 'main'
- The release was tagged 'v1.2.3'
- Release tag 'v1.2.3' has been back-merged into 'develop'
- Release branch 'release/1.2.3' has been locally deleted
- You are now on branch 'develop'
```

Рис. 18: Завершение работы с релизом

Последним шагом закроем релиз и создадим на гитхаб новую версию - 1.2.3:

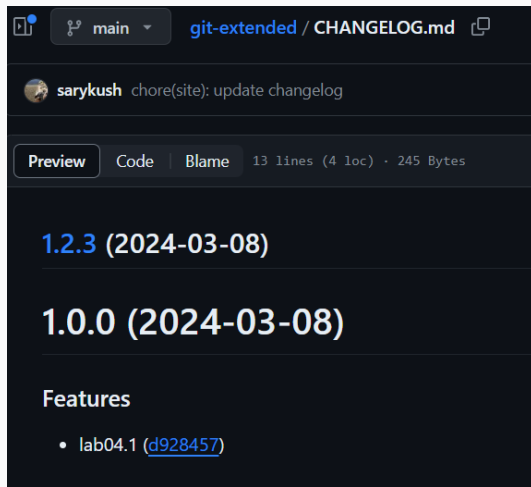
A terminal window with a dark background. The prompt is `(tabuller@jordi)-[~/git-extended]`. The command entered is `$ gh release create v1.2.3 -F CHANGELOG.md`. Below the command, the GitHub release URL is displayed: `https://github.com/sarykush/git-extended/releases/tag/v1.2.3`.

```
(tabuller@jordi)-[~/git-extended]  
$ gh release create v1.2.3 -F CHANGELOG.md  
https://github.com/sarykush/git-extended/releases/tag/v1.2.3
```

Рис. 19: Завершение работы с релизом

Создание релиза.

Проверим журнал изменений в репозитории на сайте гитхаб - видим обе версии, следовательно, работа выполнена успешно.



Выводы

Получены навыки правильной работы с репозиториями git, выполнена работа для тестового репозитория и дальнейшие преобразования для основного репозитория курса.