

Отчёт по лабораторной работе №4

Специальность: архитектура компьютеров

Кудинец Максим Антонович

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.	10
4.2	Практический сценарий использования git.	10
4.2.1	Создание репозитория git.	10
4.2.2	Работа с репозиторием git.	15
5	Выводы	18
	Список литературы	19

Список иллюстраций

4.1	Установка gitflow в режиме суперпользователя	8
4.2	Установка приложений	9
4.3	Настройка node.js	9
4.4	Настройка программ	10
4.5	Создание репозитория, первый коммит	11
4.6	Настройка пакета	11
4.7	Изменения файла	12
4.8	Выполнение коммита	12
4.9	Команда push	13
4.10	Инициализация gitflow	13
4.11	Установка внешней ветки	14
4.12	Создание релиза и журнала изменений	14
4.13	Добавление релизной ветки в основную	14
4.14	Команды push -all и push -tags	14
4.15	Создание новой ветки	15
4.16	Объединение веток, создание релиза с более новой версией	15
4.17	Изменение номера версии с 1.0.0 на 1.2.3	16
4.18	Добавление релизной ветки в основную	16
4.19	Отправка данных на гитхаб	17
4.20	Создание релиза на гитхабе с необходимым комментарием	17

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

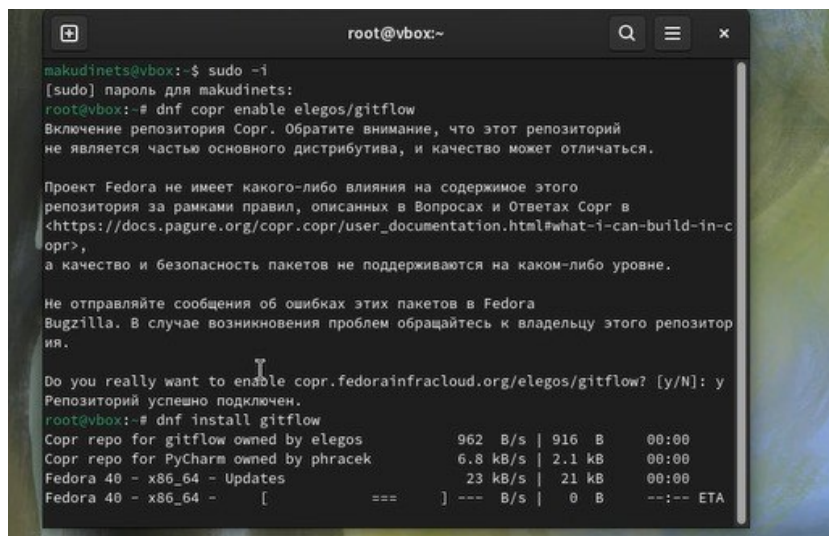
Рабочий процесс Gitflow Workflow. Будем описывать его с использованием пакета git-flow. Общая информация Gitflow Workflow опубликована и популяризована Винсентом Дриссенем. Gitflow Workflow предполагает выстраивание строгой модели ветвления с учётом выпуска проекта. Данная модель отлично подходит для 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).



```
makudinets@vbox:~$ sudo -i
[sudo] пароль для makudinets:
root@vbox:~# dnf copr enable elegos/gitflow
Включение репозитория Copr. Обратите внимание, что этот репозиторий
не является частью основного дистрибутива, и качество может отличаться.

Проект Fedora не имеет какого-либо влияния на содержимое этого
репозитория за рамками правил, описанных в Вопросах и Ответах Copr в
<https://docs.pagure.org/copr.copr/user_documentation.html#what-i-can-build-in-copr>,
а качество и безопасность пакетов не поддерживаются на каком-либо уровне.

Не отправляйте сообщения об ошибках этих пакетов в Fedora
Bugzilla. В случае возникновения проблем обращайтесь к владельцу этого репозитория.

Do you really want to enable copr.fedorainfracloud.org/elegos/gitflow? [y/N]: y
Репозиторий успешно подключен.
root@vbox:~# dnf install gitflow
Copr repo for gitflow owned by elegos          962 B/s | 916 B      00:00
Copr repo for PyCharm owned by phracek         6.8 kB/s | 2.1 kB   00:00
Fedora 40 - x86_64 - Updates                   23 kB/s | 21 kB     00:00
Fedora 40 - x86_64 - [=====] --- B/s | 0 B      --:-- ETA
```

Рис. 4.1: Установка gitflow в режиме суперпользователя

4.1.2 Установка и настройка node.js

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


```
root@vbox:~  
Идет проверка транзакции  
Тест транзакции проведен успешно.  
Выполнение транзакции  
Запуск скрипглета: nodejs-1:20.18.2-2.fc40.x86_64 1/1  
Подготовка : 1/1  
Установка : nodejs-docs-1:20.18.2-2.fc40.noarch 1/5  
Установка : nodejs-libs-1:20.18.2-2.fc40.x86_64 2/5  
Установка : nodejs-full-i18n-1:20.18.2-2.fc40.x86_64 3/5  
Установка : nodejs-npm-1:10.8.2-1.20.18.2.2.fc40.x86_64 4/5  
Установка : nodejs-1:20.18.2-2.fc40.x86_64 5/5  
Запуск скрипглета: nodejs-1:20.18.2-2.fc40.x86_64 5/5  
  
Установлен:  
nodejs-1:20.18.2-2.fc40.x86_64  
nodejs-docs-1:20.18.2-2.fc40.noarch  
nodejs-full-i18n-1:20.18.2-2.fc40.x86_64  
nodejs-libs-1:20.18.2-2.fc40.x86_64  
nodejs-npm-1:10.8.2-1.20.18.2.2.fc40.x86_64  
  
Выполнено!  
root@vbox:~# dnf install pnpm  
Последняя проверка окончания срока действия метаданных: 0:01:59 назад, Пн 24 фев 2025 20:37:28.
```

Рис. 4.2: Установка приложений

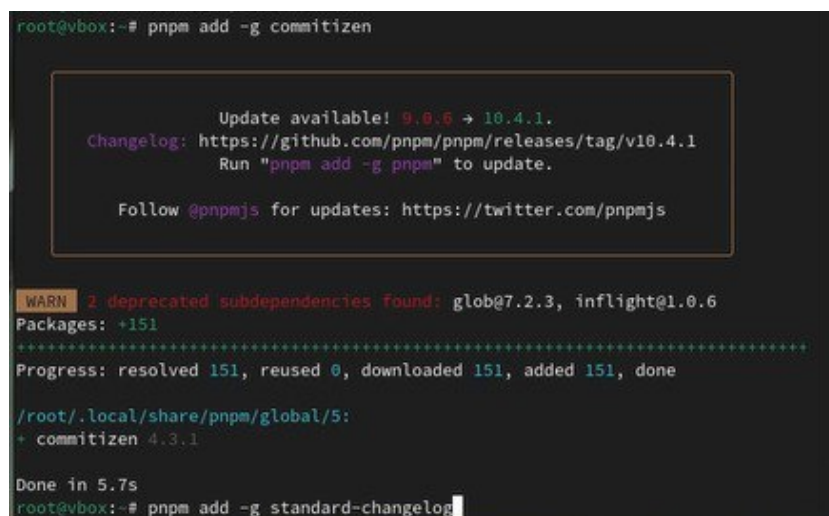
3. Настраиваем nodejs. (рис. 4.3).

```
root@vbox:~  
Тест транзакции проведен успешно.  
Выполнение транзакции  
Подготовка : 1/1  
Установка : pnpm-9.0.6-1.fc40.noarch 1/1  
Запуск скрипглета: pnpm-9.0.6-1.fc40.noarch 1/1  
  
Установлен:  
pnpm-9.0.6-1.fc40.noarch  
  
Выполнено!  
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  
  
To start using pnpm, run:  
source /root/.bashrc  
root@vbox:~# source ~/.bashrc  
root@vbox:~#
```

Рис. 4.3: Настройка node.js

4.1.3 Общепринятые коммиты.

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



```
root@vbox:~# pnpm add -g commitizen

Update available! 9.0.6 → 10.4.1.
Changelog: https://github.com/pnpm/pnpm/releases/tag/v10.4.1
Run "pnpm add -g pnpm" to update.

Follow @pnpmjs for updates: https://twitter.com/pnpmjs

WARN 2 deprecated subdependencies found: glob@7.2.3, inflight@1.0.6
Packages: +151
Progress: resolved 151, reused 0, downloaded 151, added 151, done

/root/.local/share/pnpm/global/5:
+ commitizen 4.3.1

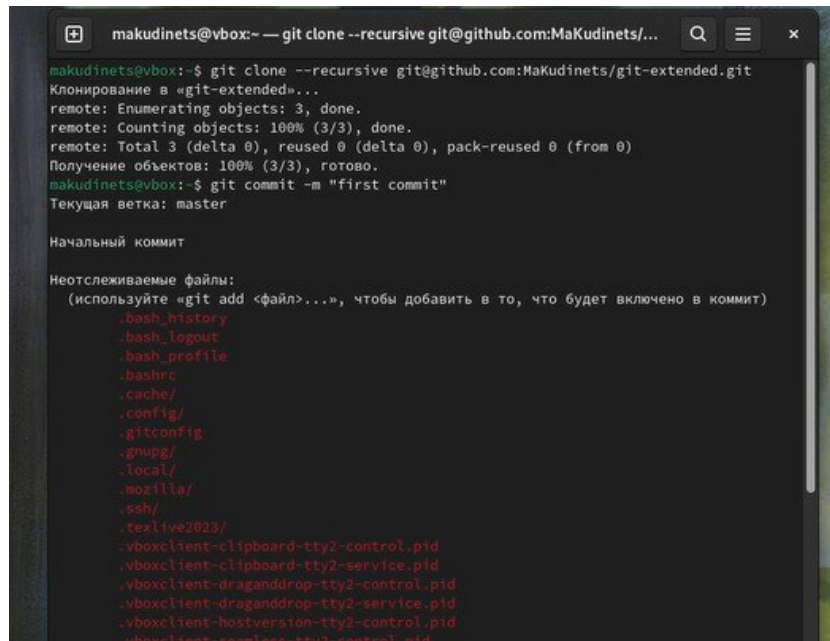
Done in 5.7s
root@vbox:~# pnpm add -g standard-changelog
```

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).



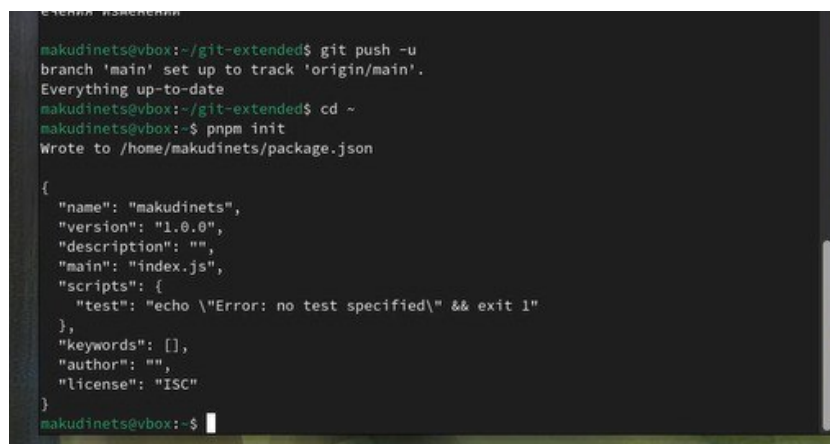
```
makudinets@vbox:~ -- git clone --recursive git@github.com:MaKudinets/...
makudinets@vbox:~$ git clone --recursive git@github.com:MaKudinets/git-extended.git
Клонирование в «git-extended»...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Получение объектов: 100% (3/3), готово.
makudinets@vbox:~$ git commit -m "first commit"
Текущая ветка: master

Начальный коммит

Неотслеживаемые файлы:
(используйте «git add <файл>...», чтобы добавить в то, что будет включено в коммит)
.bash_history
.bash_logout
.bash_profile
.bashrc
.cache/
.config/
.gitconfig
.gnupg/
.local/
.mozilla/
.ssh/
.texlive2023/
.vboxclient-clipboard-tty2-control.pid
.vboxclient-clipboard-tty2-service.pid
.vboxclient-draganddrop-tty2-control.pid
.vboxclient-draganddrop-tty2-service.pid
.vboxclient-hostversion-tty2-control.pid
.vboxclient-usb-lcd-control.pid
```

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).



```
makudinets@vbox:~/git-extended$ git push -u
branch 'main' set up to track 'origin/main'.
Everything up-to-date
makudinets@vbox:~/git-extended$ cd ~
makudinets@vbox:~$ pnpm init
Wrote to /home/makudinets/package.json

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

Рис. 4.6: Настройка пакета

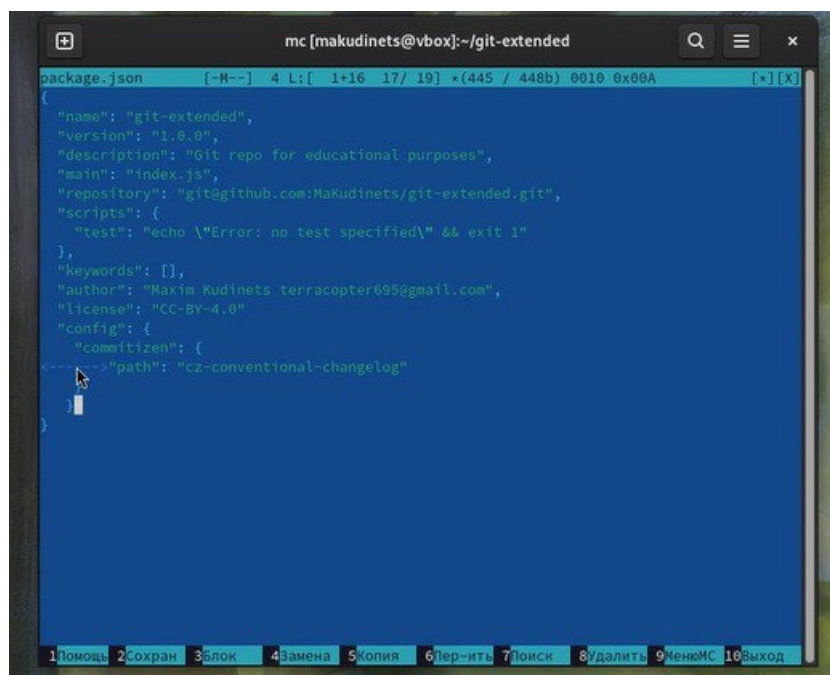


Рис. 4.7: Изменения файла

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

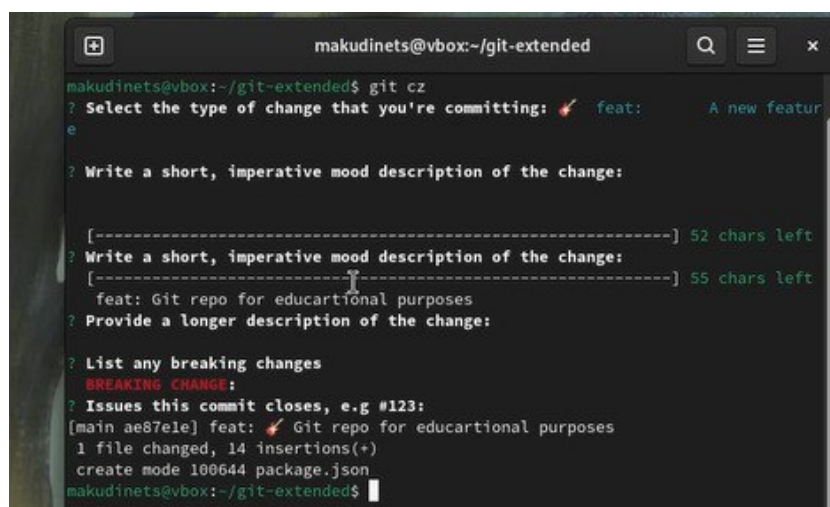


Рис. 4.8: Выполнение коммита

```
makudinets@vbox:~/git-extended$ git push
Перечисление объектов: 4, готово.
Подсчет объектов: 100% (4/4), готово.
При сжатии изменений используется до 10 потоков
Сжатие объектов: 100% (3/3), готово.
Запись объектов: 100% (3/3), 1.17 КиБ | 1.17 МБ/с, готово.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:Makudinets/git-extended.git
   aab92fc..ae87ele  main -> main
makudinets@vbox:~/git-extended$ git flow init

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

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).

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

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main] main
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/makudinets/git-extended/.git/hooks]
makudinets@vbox:~/git-extended$ git branch
* develop
  main
makudinets@vbox:~/git-extended$
makudinets@vbox:~/git-extended$ git push --all
```

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).

```

makudinets@vbox:~/git-extended$ git branch --set-upstream-to=origin/develop deve
lop
branch 'develop' set up to track 'origin/develop'.
makudinets@vbox:~/git-extended$ git flow release start 1.0.0
Переключились на новую ветку «releasel.0.0»

Summary of actions:
- A new branch 'releasel.0.0' was created, based on 'develop'
- You are now on branch 'releasel.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'

makudinets@vbox:~/git-extended$ standard-changelog --first-release

```

Рис. 4.11: Установка внешней ветки

```

makudinets@vbox:~/git-extended$ git flow release start 1.0.0
Fatal: There is an existing release branch '1.0.0'. Finish that one first.
makudinets@vbox:~/git-extended$ ls
CHANGELOG.md package.json README.md
makudinets@vbox:~/git-extended$ git add CHANGELOG.md

```

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).

```

makudinets@vbox:~$ cd ~/git-extended
makudinets@vbox:~/git-extended$ git add CHANGELOG.md
makudinets@vbox:~/git-extended$ git commit -am 'chore(site): add changelog'
[releasel.0.0 e316c2d] chore(site): add changelog
1 file changed, 10 insertions(+)
create mode 100644 CHANGELOG.md
makudinets@vbox:~/git-extended$ git flow release finish 1.0.0

```

Рис. 4.13: Добавление релизной ветки в основную

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

```

makudinets@vbox:~/git-extended$ git push --all
Перечисление объектов: 5, готово.
Подсчет объектов: 100% (5/5), готово.
При сжатии изменений используется до 10 потоков
Сжатие объектов: 100% (4/4), готово.
Запись объектов: 100% (4/4), 1.97 Киб | 1.97 Миб/с, готово.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:MaKudinets/git-extended.git
 ae87e1e..5c88867 main -> main
 * [new branch]      releasel.0.0 -> releasel.0.0
makudinets@vbox:~/git-extended$ git push --tags

```

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. (рис. 4.15).

```
makudinets@vbox:~/git-extended$ gh release create v1.0.0 -F CHANGELOG.md
https://github.com/MaKudinets/git-extended/releases/tag/v1.0.0
makudinets@vbox:~/git-extended$ git flow feature start feature_branch
Переключились на новую ветку «featurefeature_branch»

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

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

    git flow feature finish feature_branch
```

Рис. 4.15: Создание новой ветки

13. Объединяем новую ветку с develop. Создаём релиз с версией 1.2.3. (рис. 4.16).

```
makudinets@vbox:~/git-extended$ git flow feature finish feature_branch
Переключились на ветку «develop»
Эта ветка соответствует «origin/develop».
Уже актуально.
Ветка featurefeature_branch удалена (была ae87e1e).

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

makudinets@vbox:~/git-extended$ git flow release start 1.2.3
Fatal: There is an existing release branch '1.0.0'. Finish that one first.
makudinets@vbox:~/git-extended$
```

Рис. 4.16: Объединение веток, создание релиза с более новой версией

14. Изменяем номер версии в файле package.json. (рис. 4.17).

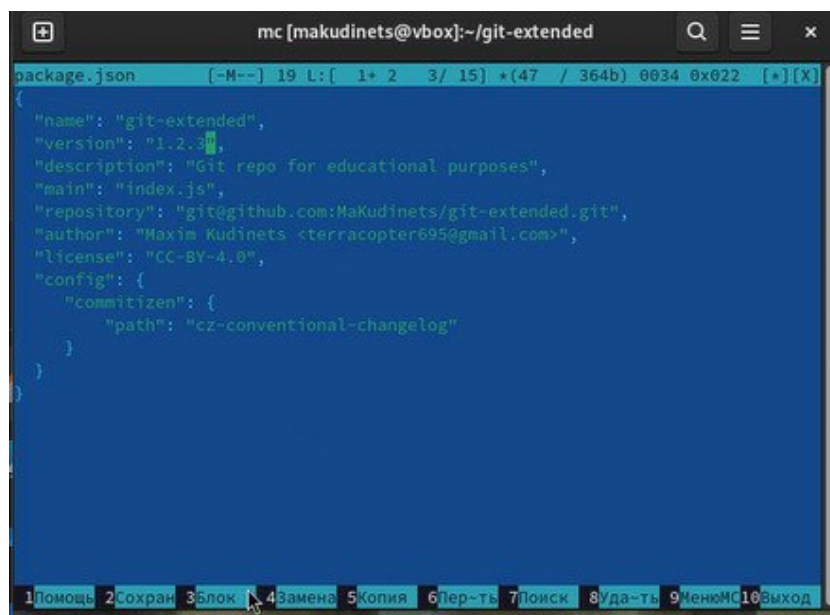


Рис. 4.17: Изменение номера версии с 1.0.0 на 1.2.3

15. Заливаем релизную ветку в основную (рис. 4.18). Отправляем данные на гитхаб. (рис. 4.19).

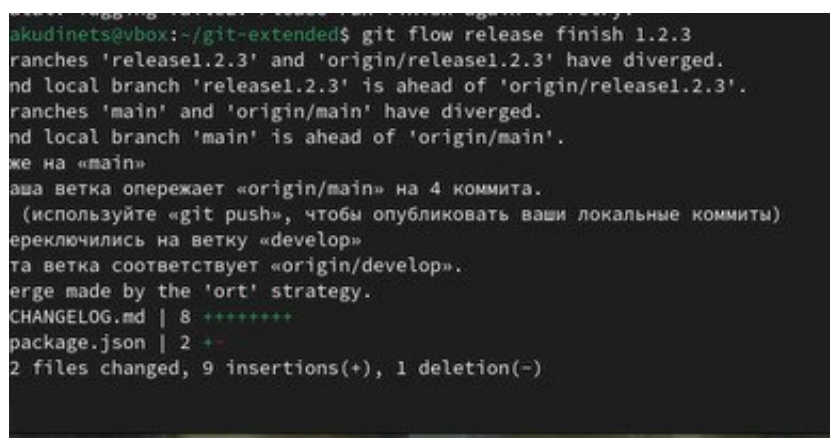


Рис. 4.18: Добавление релизной ветки в основную


```

kudinets@vbox:~/git-extended$ git push --all
счисление объектов: 12, готово.
счет объектов: 100% (12/12), готово.
и сжатии изменений используется до 10 потоков
ание объектов: 100% (9/9), готово.
пись объектов: 100% (9/9), 3.93 КиБ | 3.93 МиБ/с, готово.
tal 9 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
ote: Resolving deltas: 100% (2/2), completed with 1 local object.
github.com:MaKudinets/git-extended.git
714dfcc..68c887f develop -> develop
5c88867..7e6e62d main -> main
kudinets@vbox:~/git-extended$ git push --tags

```

Рис. 4.19: Отправка данных на гитхаб

16. Создаём релиз на гитхабе с комментарием из журнала изменений. (рис. 4.20).

```

makudinets@vbox:~/git-extended$ gh release create v1.2.3 -F CHANGELOG.md
https://github.com/MaKudinets/git-extended/releases/tag/v1.2.3
makudinets@vbox:~/git-extended$

```

Рис. 4.20: Создание релиза на гитхабе с необходимым комментарием

5 Выводы

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

Список литературы