

# Отчёт по лабораторной работе № 2

Дисциплина: Операционные системы

Перелыгин Сергей Викторович

# Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
4	Контрольные вопросы:	13
5	Выводы	17

## Список иллюстраций

3.1	Рисунок 1 . . . . .	6
3.2	Рисунок 2 . . . . .	7
3.3	Рисунок 3 . . . . .	8
3.4	Рисунок 4 . . . . .	9
3.5	Рисунок 5 . . . . .	9
3.6	Рисунок 6 . . . . .	10
3.7	Рисунок 7 . . . . .	10
3.8	Рисунок 8 . . . . .	11
3.9	Рисунок 9 . . . . .	11
3.10	Рисунок 10 . . . . .	12

# 1 Цель работы

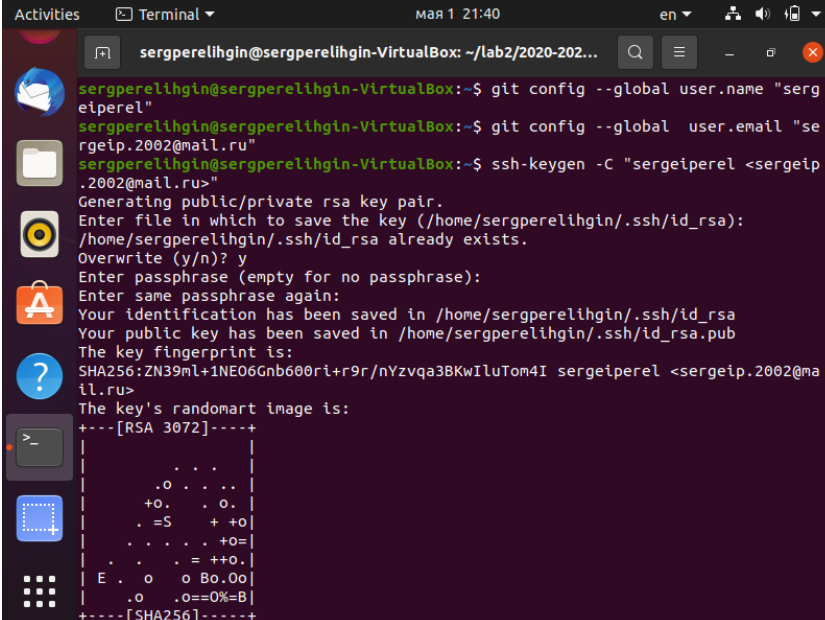
Изучить идеологию и применение средств контроля версий.

## 2 Задание

- Сделать отчёт по предыдущей лабораторной работе в формате Markdown.
- В качестве отчёта предоставить отчёты в 3 форматах: pdf, docx и md.

### 3 Выполнение лабораторной работы

1. Сначала нужно зарегистрироваться на гитхабе.
2. Далее настроим систему контроля версий git.
3. Сначала сделаем предварительную конфигурацию, указав имя и email владельца репозитория: (рис. 1)



```
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-202...
sergperelihgin@sergperelihgin-VirtualBox:~$ git config --global user.name "sergeiperel"
sergperelihgin@sergperelihgin-VirtualBox:~$ git config --global user.email "sergeip.2002@mail.ru"
sergperelihgin@sergperelihgin-VirtualBox:~$ ssh-keygen -C "sergeiperel <sergeip.2002@mail.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/sergperelihgin/.ssh/id_rsa):
/home/sergperelihgin/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/sergperelihgin/.ssh/id_rsa
Your public key has been saved in /home/sergperelihgin/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:ZN39ml+1NE06Gnb600ri+r9r/nYzvqa3BKwIluTom4I sergeiperel <sergeip.2002@mail.ru>
The key's randomart image is:
+---[RSA 3072]-----+
|
| .o . . .
| +o. . o.
| . =S . + +o|
| . . . . +o=|
| . . . . = ++O.|
| E . o o Bo.Oo|
| .o .o =O%=B|
+---[SHA256]-----+
```

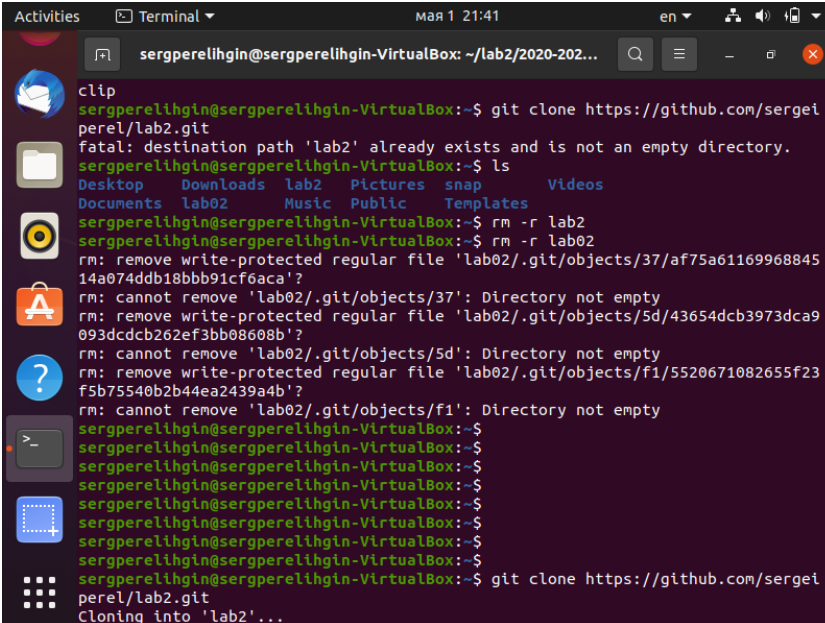
Рис. 3.1: Рисунок 1

Это команды `git config --global user.name "Имя Фамилия"` `git config --global user.email "work@mail"`.

4. Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый). На рисунке 1

это команда `ssh-keygen -C "Имя Фамилия"`.

- После этого заходим на гитхаб в раздел “Настройки”, где нужно создать SSH-ключ и, скопировав его командой `cat ~/.ssh/id_rsa.pub | xclip -sel clip` в буфер обмена, добавить его на гитхаб.
- После этого в разделе “Мои репозитории” создать новую репозиторию и назвать ее.
- Введем команду `git clone https://github.com/sergeiperel/lab2.git`, чтобы стянуть все на консоль компьютера.



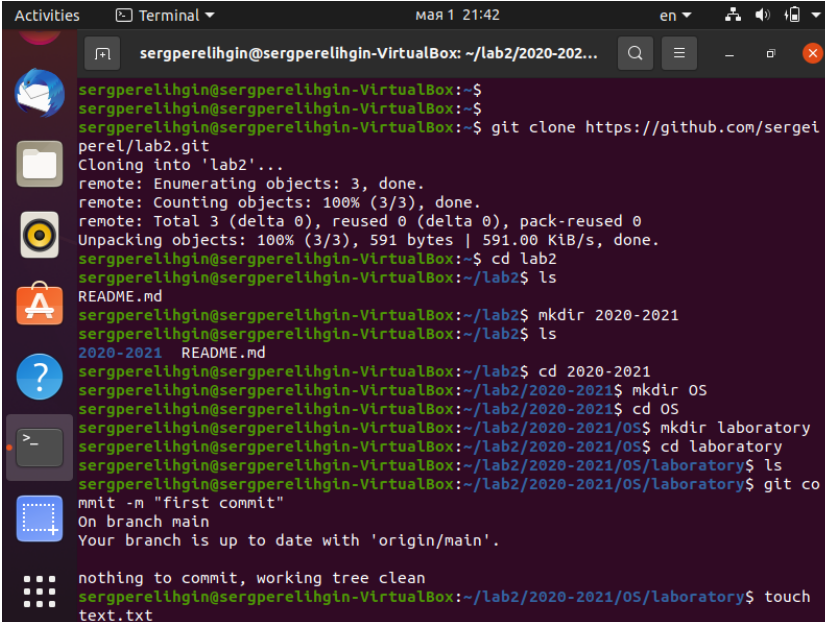
```
Activities Terminal мая 1 21:41 en
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-202...

clip
sergperelihgin@sergperelihgin-VirtualBox:~$ git clone https://github.com/sergeiperel/lab2.git
fatal: destination path 'lab2' already exists and is not an empty directory.
sergperelihgin@sergperelihgin-VirtualBox:~$ ls
Desktop  Downloads  lab2  Pictures  snap  Videos
Documents  lab02  Music  Public  Templates
sergperelihgin@sergperelihgin-VirtualBox:~$ rm -r lab2
sergperelihgin@sergperelihgin-VirtualBox:~$ rm -r lab02
rm: remove write-protected regular file 'lab02/.git/objects/37/af75a6116996884514a074ddb18bbb91cf6aca'?
rm: cannot remove 'lab02/.git/objects/37': Directory not empty
rm: remove write-protected regular file 'lab02/.git/objects/5d/43654dcb3973dca9093dcdcb262ef3bb08608b'?
rm: cannot remove 'lab02/.git/objects/5d': Directory not empty
rm: remove write-protected regular file 'lab02/.git/objects/f1/5520671082655f23f5b75540b2b44ea2439a4b'?
rm: cannot remove 'lab02/.git/objects/f1': Directory not empty
sergperelihgin@sergperelihgin-VirtualBox:~$
sergperelihgin@sergperelihgin-VirtualBox:~$
sergperelihgin@sergperelihgin-VirtualBox:~$
sergperelihgin@sergperelihgin-VirtualBox:~$
sergperelihgin@sergperelihgin-VirtualBox:~$
sergperelihgin@sergperelihgin-VirtualBox:~$
sergperelihgin@sergperelihgin-VirtualBox:~$
sergperelihgin@sergperelihgin-VirtualBox:~$ git clone https://github.com/sergeiperel/lab2.git
Cloning into 'lab2'...
```

Рис. 3.2: Рисунок 2

- Далее создадим несколько папок(рис. 3).
- Затем сделаем первый коммит с помощью команды `git commit -m "first commit"`(рис. 3).
- Создадим файл `text.txt` и добавим команду `git add`. Применяем команду `git push` и отправляем его на гитхаб(рис. 4).

11. Далее добавим файл лицензии(рис. 4 и 5).
12. Далее добавим шаблон игнорируемых файлов. Просмотрим список имеющихся шаблонов: `curl -L -s https://www.gitignore.io/api/list` Затем скачаем шаблон, например, для C: `curl -L -s https://www.gitignore.io/api/c >> .gitignore` (рис. 6 и 7)



```
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-2021...
sergperelihgin@sergperelihgin-VirtualBox:~$
sergperelihgin@sergperelihgin-VirtualBox:~$ git clone https://github.com/sergei
perel/lab2.git
Cloning into 'lab2'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 591 bytes | 591.00 KiB/s, done.
sergperelihgin@sergperelihgin-VirtualBox:~$ cd lab2
sergperelihgin@sergperelihgin-VirtualBox:~/lab2$ ls
README.md
sergperelihgin@sergperelihgin-VirtualBox:~/lab2$ mkdir 2020-2021
sergperelihgin@sergperelihgin-VirtualBox:~/lab2$ ls
2020-2021  README.md
sergperelihgin@sergperelihgin-VirtualBox:~/lab2$ cd 2020-2021
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021$ mkdir OS
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021$ cd OS
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS$ mkdir laboratory
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS$ cd laboratory
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ ls
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ git co
mmitt -m "first commit"
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ touch
text.txt
```

Рис. 3.3: Рисунок 3



```
Activities Terminal МАР 1 21:42 en
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-202...

sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ git add .
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ git commit -m "first commit"
[main f6b2801] first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 2020-2021/OS/laboratory/text.txt
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ git push
Username for 'https://github.com': sergeiperel
Password for 'https://sergeiperel@github.com':
remote: Invalid username or password.
fatal: Authentication failed for 'https://github.com/sergeiperel/lab2.git/'
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ git push
Username for 'https://github.com': sergeiperel
Password for 'https://sergeiperel@github.com':
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (6/6), 403 bytes | 403.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0)
To https://github.com/sergeiperel/lab2.git
90ae4a4..f6b2801 main -> main
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ wget https://creativecommons.org/licenses/by/4.0/legalcode.txt -O LICENSE
--2021-05-01 21:05:53-- https://creativecommons.org/licenses/by/4.0/legalcode.
```

Рис. 3.4: Рисунок 4

```
Activities Terminal МАР 1 21:42 en
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-202...

https://creativecommons.org/licenses/by/4.0/legalcode.txt -O LICENSE
--2021-05-01 21:05:53-- https://creativecommons.org/licenses/by/4.0/legalcode.txt
Resolving creativecommons.org (creativecommons.org)... 104.20.150.16, 172.67.34.140, 104.20.151.16, ...
Connecting to creativecommons.org (creativecommons.org)|104.20.150.16|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/plain]
Saving to: 'LICENSE'

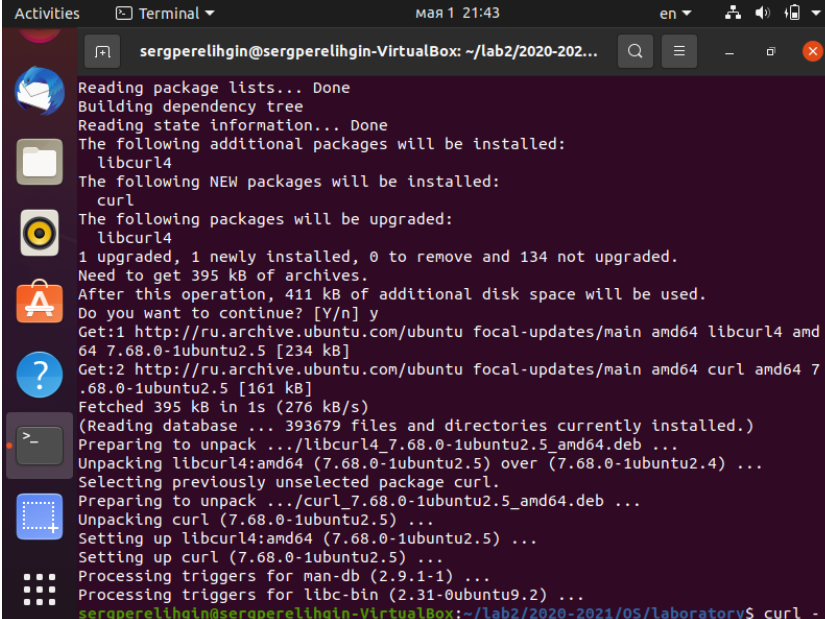
LICENSE [ <=> ] 18,22K --.-KB/s in 0s

2021-05-01 21:05:55 (158 MB/s) - 'LICENSE' saved [18657]

sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ curl -L -s https://www.gitignore.io/api/list
Command 'curl' not found, but can be installed with:
sudo apt install curl

sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/OS/laboratory$ sudo apt install curl
[sudo] password for sergperelihgin:
Sorry, try again.
[sudo] password for sergperelihgin:
Reading package lists... Done
Building dependency tree
```

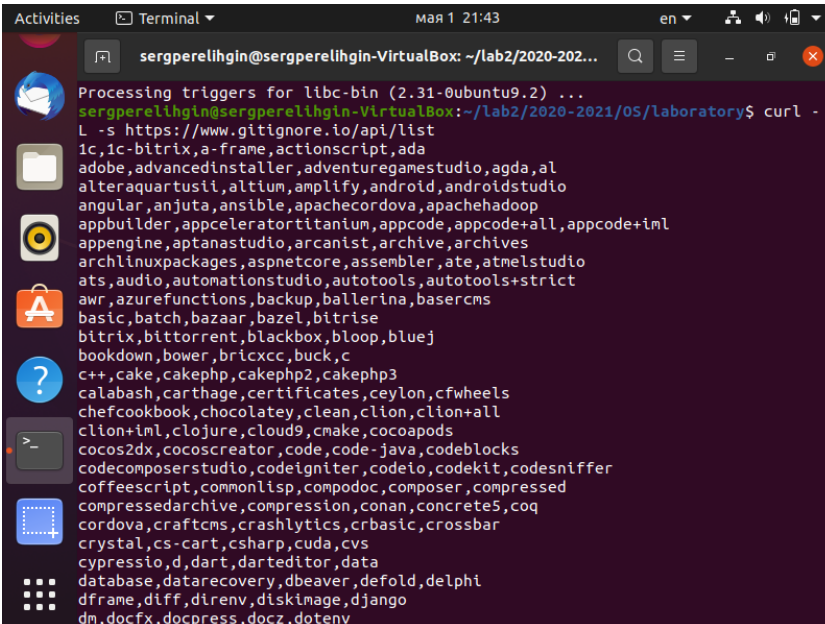
Рис. 3.5: Рисунок 5



```
Activities Terminal 3 мая 1 21:43 en
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-202...

Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libcurl4
The following NEW packages will be installed:
  curl
The following packages will be upgraded:
  libcurl4
1 upgraded, 1 newly installed, 0 to remove and 134 not upgraded.
Need to get 395 kB of archives.
After this operation, 411 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ru.archive.ubuntu.com/ubuntu focal-updates/main amd64 libcurl4 amd64 7.68.0-1ubuntu2.5 [234 kB]
Get:2 http://ru.archive.ubuntu.com/ubuntu focal-updates/main amd64 curl amd64 7.68.0-1ubuntu2.5 [161 kB]
Fetched 395 kB in 1s (276 kB/s)
(Reading database ... 393679 files and directories currently installed.)
Preparing to unpack .../libcurl4_7.68.0-1ubuntu2.5_amd64.deb ...
Unpacking libcurl4:amd64 (7.68.0-1ubuntu2.5) over (7.68.0-1ubuntu2.4) ...
Selecting previously unselected package curl.
Preparing to unpack .../curl_7.68.0-1ubuntu2.5_amd64.deb ...
Unpacking curl (7.68.0-1ubuntu2.5) ...
Setting up libcurl4:amd64 (7.68.0-1ubuntu2.5) ...
Setting up curl (7.68.0-1ubuntu2.5) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ curl -
```

Рис. 3.6: Рисунок 6



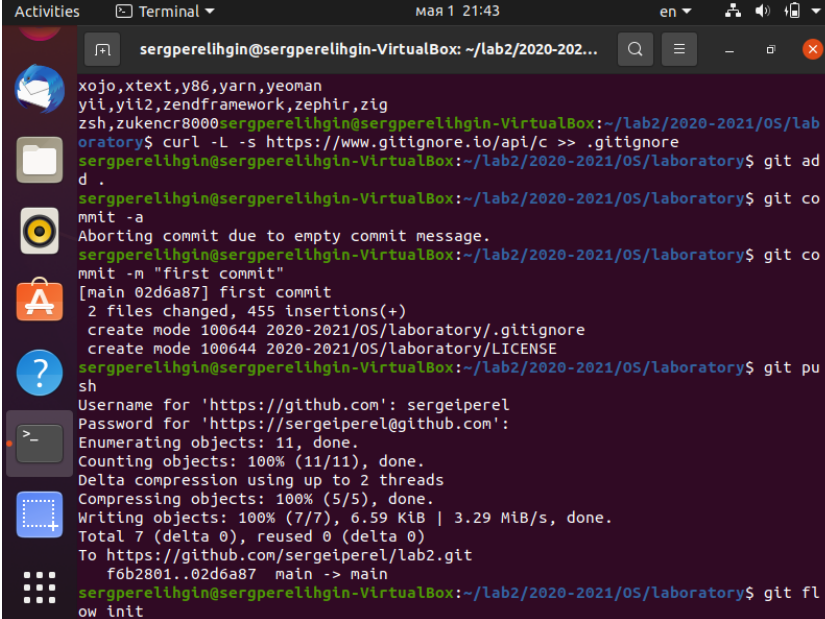
```
Activities Terminal 3 мая 1 21:43 en
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-202...

Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ curl -L -s https://www.gitignore.io/api/list
1c,1c-bitrix,a-frame,actionsript,ada
adobe,advancedinstaller,adventuregamestudio,agda,al
alteraquartusii,altium,amplify,android,androidstudio
angular,anjuta,ansible,apachecordova,apachehadoop
appbuilder,appcelerator titanium,appcode,appcode+all,appcode+iml
appengine,aptanastudio,arcanist,archive,archives
archlinuxpackages,aspnetcore,assembler,ate,atmelstudio
ats,audio,automationstudio,autotools,autotools+strict
avr,azurefunctions,backup,ballerina,basercms
basic,batch,bazaar,bazel,bitrise
bitrix,bittorrent,blackbox,bloop,bluej
bookdown,bower,bricxcc,buck,c
c++,cake,cakephp,cakephp2,cakephp3
calabash,carthage,certificates,ceylon,cfwheels
chefcookbook,chocolatey,clean,clion,clion+all
clion+iml,clojure,cloud9,cmake,cocoapods
cocos2dx,cocoscreator,code,code-java,codeblocks
codecomposerstudio,codeigniter,codeio,codekit,codesniffer
coffeescript,commonlisp,compodoc,composer,compressed
compressedarchive,compression,conan,concrete5,coq
cordova,craftcms,crashlytics,crbasic,crossbar
crystal,cs-cart,csharp,cuda,cvs
cypressio,d,dart,darteditor,data
database,datarecovery,dbeaver,defold,delphi
dframe,diff,direnv,diskimage,django
dm,docfx,docpress,docz,dotenv
```

Рис. 3.7: Рисунок 7

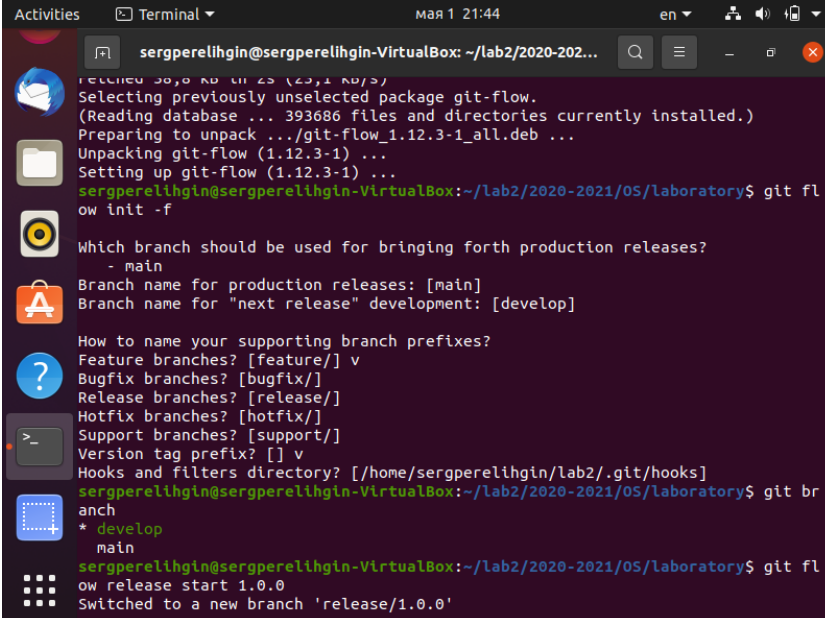
- Добавим новые файлы с помощью команды `git add .`, выполним коммит (`git commit -a`) и отправим на github (`git push`). Это представлено на рисунке 8.

14. Далее будем работать с командой `git flow` (рис. 9 и 10).
15. Затем заходим на гитхаб и публикуем релиз.



```
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-202...
xoho,xtext,y86,yarn,yeoman
yii,yii2,zendframework,zephir,zig
zsh,zukencr8000sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/lab
oratory$ curl -L -s https://www.gitignore.io/api/c >> .gitignore
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ git ad
d .
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ git co
mmit -a
Aborting commit due to empty commit message.
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ git co
mmit -m "first commit"
[main 02d6a87] first commit
2 files changed, 455 insertions(+)
create mode 100644 2020-2021/05/laboratory/.gitignore
create mode 100644 2020-2021/05/laboratory/LICENSE
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ git pu
sh
Username for 'https://github.com': sergeiperel
Password for 'https://sergeiperel@github.com':
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 2 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (7/7), 6.59 KiB | 3.29 MiB/s, done.
Total 7 (delta 0), reused 0 (delta 0)
To https://github.com/sergeiperel/lab2.git
f6b2801..02d6a87  main -> main
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ git fl
ow init
```

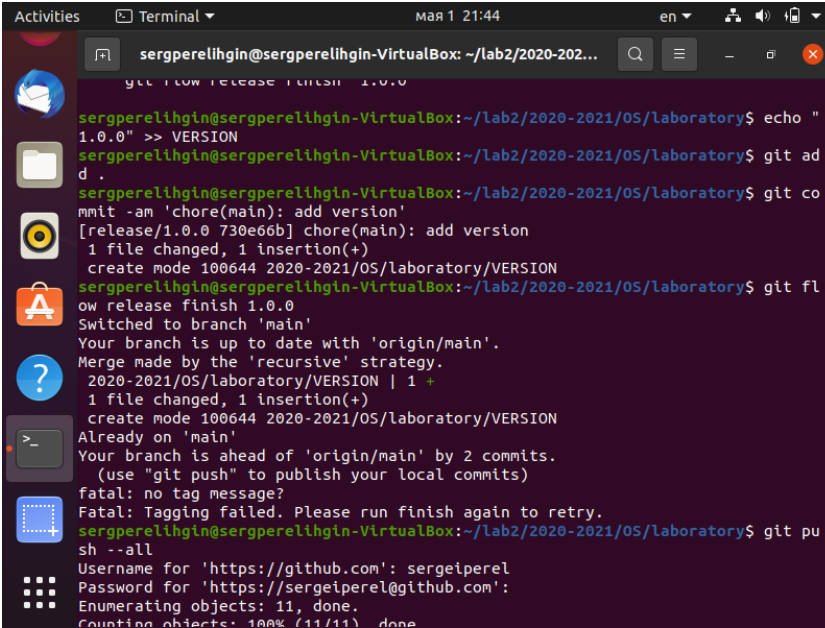
Рис. 3.8: Рисунок 8



```
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-202...
Selecting previously unselected package git-flow.
(Reading database ... 393686 files and directories currently installed.)
Preparing to unpack .../git-flow_1.12.3-1_all.deb ...
Unpacking git-flow (1.12.3-1) ...
Setting up git-flow (1.12.3-1) ...
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ git fl
ow init -f
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/] v
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? [] v
Hooks and filters directory? [/home/sergperelihgin/lab2/.git/hooks]
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ git br
anch
* develop
main
sergperelihgin@sergperelihgin-VirtualBox:~/lab2/2020-2021/05/laboratory$ git fl
ow release start 1.0.0
Switched to a new branch 'release/1.0.0'
```

Рис. 3.9: Рисунок 9



```
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-2021/OS/laboratory$ echo "1.0.0" >> VERSION
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-2021/OS/laboratory$ git add .
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-2021/OS/laboratory$ git commit -am 'chore(main): add version'
[release/1.0.0 730e66b] chore(main): add version
1 file changed, 1 insertion(+)
create mode 100644 2020-2021/OS/laboratory/VERSION
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-2021/OS/laboratory$ git merge release/1.0.0
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
Merge made by the 'recursive' strategy.
2020-2021/OS/laboratory/VERSION | 1 +
1 file changed, 1 insertion(+)
create mode 100644 2020-2021/OS/laboratory/VERSION
Already on 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)
sergperelihgin@sergperelihgin-VirtualBox: ~/lab2/2020-2021/OS/laboratory$ git push --all
Username for 'https://github.com': sergeiperel
Password for 'https://sergeiperel@github.com':
Enumerating objects: 11, done.
Counting objects: 100% (11/11) done
```

Рис. 3.10: Рисунок 10

## 4 Контрольные вопросы:

- 1) Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды `git` различными опциями. Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом.
- 2) В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером. Участник проекта (пользователь) перед началом работы посредством определённых команд получает нужную ему версию файлов. После внесения изменений, пользователь размещает новую версию в хранилище. При этом предыдущие версии не удаляются из центрального хранилища и к ним можно вернуться в любой момент. Сервер может сохранять неполную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных. Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить.

- 3) Централизованные системы — это системы, которые используют архитектуру клиент / сервер, где один или несколько клиентских узлов напрямую подключены к центральному серверу. Пример - Wikipedia. В децентрализованных системах каждый узел принимает свое собственное решение. Конечное поведение системы является совокупностью решений отдельных узлов. Пример — Bitcoin. В классических системах контроля версий используется централизованная модель, предполагающая наличие единого репозитория для хранения файлов. Выполнение большинства функций по управлению версиями осуществляется специальным сервером.
- 4) Создадим локальный репозиторий. Сначала сделаем предварительную конфигурацию, указав имя и email владельца репозитория: `git config --global user.name "Имя Фамилия"` `git config --global user.email "work@mail"` и настроив utf-8 в выводе сообщений: `git config --global quotePath false` Для инициализации локального репозитория, расположенного, например, в каталоге `~/tutorial`, необходимо ввести в командной строке: `cd mkdir tutorial cd tutorial git init`
- 5) Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый): `ssh-keygen -C "Имя Фамилия work@mail"` Ключи сохраняться в каталоге `~/.ssh/`. Скопировав из локальной консоли ключ в буфер обмена `cat ~/.ssh/id_rsa.pub | xclip -sel clip` вставляем ключ в появившееся на сайте поле.
- 6) У Git две основных задачи: первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.
- 7) Основные команды git: Наиболее часто используемые команды git: – создание основного дерева репозитория: `git init` – получение обновлений (изменений) текущего дерева из центрального репозитория: `git pull` – отправка всех произведённых изменений локального дерева в центральный репозиторий: `git`

push – просмотр списка изменённых файлов в текущей директории: `git status` –  
 просмотр текущих изменений: `git diff` – сохранение текущих изменений: – добавить  
 все изменённые и/или созданные файлы и/или каталоги: `git add .` – добавить  
 конкретные изменённые и/или созданные файлы и/или каталоги: `git add имя_`  
`на_файлов` – удалить файл и/или каталог из индекса репозитория (при этом  
 файл и/или каталог остаётся в локальной директории): `git rm имена_файлов`  
 – сохранение добавленных изменений: – сохранить все добавленные измене-  
 ния и все изменённые файлы: `git commit -am 'Описание коммита'` – сохранить  
 добавленные изменения с внесением комментария через встроенный редак-  
 тор: `git commit` – создание новой ветки, базирующейся на текущей: `git checkout`  
`-b имя_ветки` – переключение на некоторую ветку: `git checkout имя_ветки` (при  
 переключении на ветку, которой ещё нет в локальном репозитории, она будет  
 создана и связана с удалённой) – отправка изменений конкретной ветки в  
 центральный репозиторий: `git push origin имя_ветки` – слияние ветки текущим  
 деревом: `git merge --no-ff имя_ветки` – удаление ветки: – удаление локальной уже  
 слитой с основным деревом ветки: `git branch -d имя_ветки` – принудительное уда-  
 ление локальной ветки: `git branch -D имя_ветки` – удаление ветки с центрального  
 репозитория: `git push origin :имя_ветки`

- 8) Использование `git` при работе с локальными репозиториями (добавления тек-  
 стового документа в локальный репозиторий): `git add hello.txt` `git commit`  
`-am 'Новый файл'`
- 9) Проблемы, которые решают ветки `git`: • нужно постоянно создавать архивы с  
 рабочим кодом • сложно “переключаться” между архивами • сложно перетас-  
 кивать изменения между архивами • легко что-то напутать или потерять
- 10) Во время работы над проектом так или иначе могут создаваться файлы, которые  
 не требуются добавлять в последствии в репозиторий. Например, временные  
 файлы, создаваемые редакторами, или объектные файлы, создаваемые  
 компиляторами. Можно прописать шаблоны игнорируемых при добавлении

в репозиторий типов файлов в файл .gitignore с помощью сервисов. Для этого сначала нужно получить списки имеющихся шаблонов: `curl -L -s https://www.gitignore.io/api/list` Затем скачать шаблон, например, для C и C++ `curl -L -s https://www.gitignore.io/api/c >> .gitignore` `curl -L -s https://www.gitignore.io/api/c++ >> .gitignore`



## 5 Выводы

Вывод: в этой лабораторной работе я изучил идеологию и применение средств контроля версий.