

# Лабораторная работа №2

Журавлев Георгий Иванович

## Цель работы

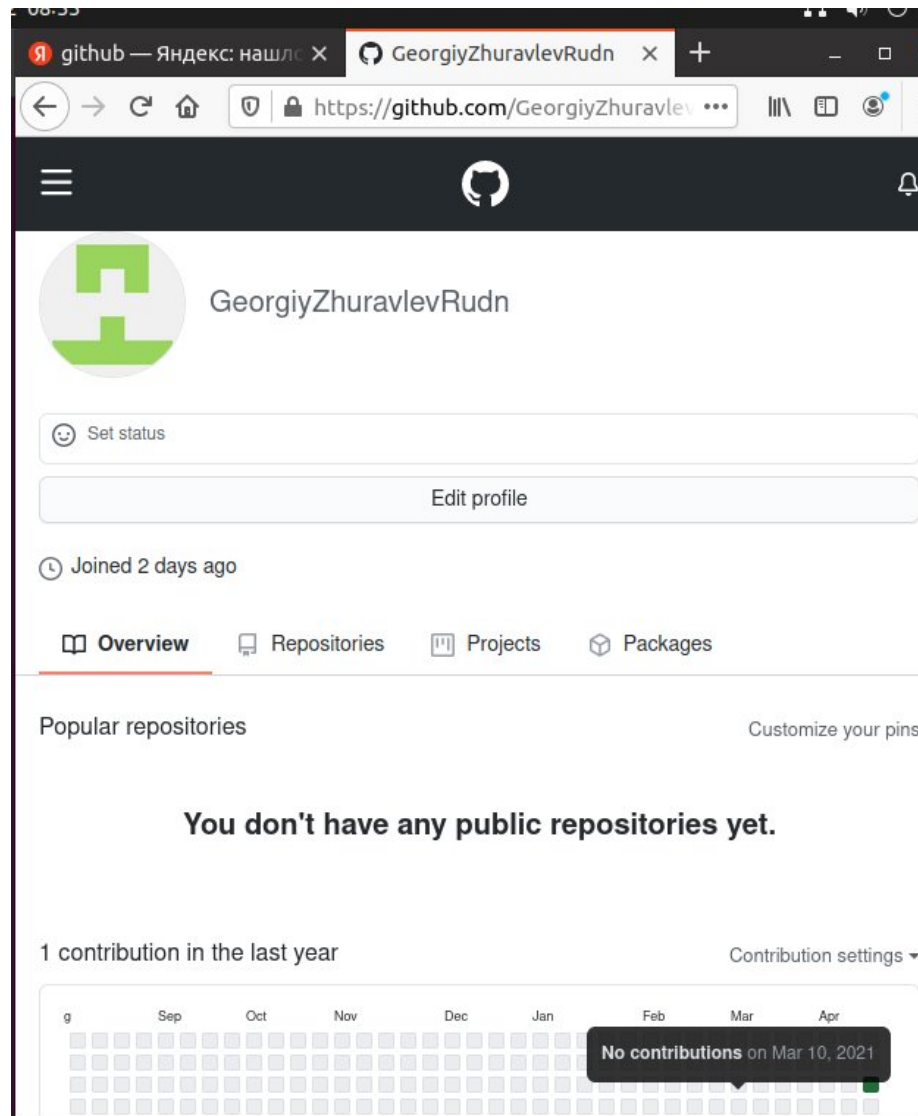
Получить опыт работы с Git. Создать аккаунт; подключить репозиторий к Github; Пройти первичную конфигурацию; провести конфигурацию git-flow.

## Задание

- Сделать отчет по предыдущей работе в формате Markdown.
- Предоставить в 3-х форматах: pdf, md and docx.

## Выполнение лабораторной работы

### 1. Создаем аккаунт github



2

```
georgiyzhuravlev@georgiy:~$ ssh-keygen -t rsa -b 4096 -C "georgiyzhuravlev@mail.ru"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/georgiyzhuravlev/.ssh/id_rsa):
Created directory '/home/georgiyzhuravlev/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match. Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/georgiyzhuravlev/.ssh/id_rsa.
Your public key has been saved in /home/georgiyzhuravlev/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:Ad5RTg/GDl+R7MycnxaanSEvxt/btg2TvdHWu9P7jTw george.zhuravlev@mail.ru
The key's randomart image is:
+----[RSA 4096]-----+
|  .o=..o |
|  o.=.o+ |
|  o+.*.. |
|   .o B o |
|  S  . B = |
|   *.Bo |
|   .+0.o |
|          |
+----+-----+

```

Blocked users

Interaction limits

## SSH keys

This is a list of SSH keys associated with your account. Remove any you do not recognize.



### newkey

SHA256: Ad5RTg/GD1+R7MycnxaanSEvxt  
/btg2TvDHWu9P7jTw

SSH

Added on Apr 22, 2021

Never used — Read/write

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH problems](#).

## GPG keys

There are no GPG keys associated with your account.

Learn how to [generate a GPG key and add it to your account](#).

### 3. Добавляем сгенерированный ключ.

```
georgiyzhuravlev@georgiy:~/Rudnrep$ git init
Initialized empty Git repository in /home/georgiyzhuravlev/Rudnrep/
georgiyzhuravlev@georgiy:~/Rudnrep$ git add .
georgiyzhuravlev@georgiy:~/Rudnrep$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track new files)
georgiyzhuravlev@georgiy:~/Rudnrep$ echo test>file1.txt
georgiyzhuravlev@georgiy:~/Rudnrep$ git add file1.txt
georgiyzhuravlev@georgiy:~/Rudnrep$ git commit -m "First file"
[master (root-commit) 184811e] First file
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt
georgiyzhuravlev@georgiy:~/Rudnrep$ git remote add origin https://github.com/georgiyzhuravlev/Rudnrep.git
georgiyzhuravlev@georgiy:~/Rudnrep$ git push -u origin master
```

### 4. Приступаем к созданию репозитория и файлов.

```

georgiyzhuravlev@georgiy:~/Rudnrep$ wget https://creativecommons.org/licenses/by/4.0/legalcode.txt -O LICENSE
--2021-04-22 10:37:02-- https://creativecommons.org/licenses/by/4.0/legalcode.txt
Resolving creativecommons.org (creativecommons.org)... 104.20.150.16, 104.20.151.16, 172.67.34.140, ...
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

2021-04-22 10:37:02 (118 MB/s) - 'LICENSE' saved [18657]

```

### 5. Добавляем лицензию

### 6. Добавляем игнорируемый файл ( с помощью vs code).

```

georgiyzhuravlev@georgiy:~/Rudnrep$ git ls-files
2020-2021/OS/lab02/file1.txt
LICENSE
README1.md
georgiyzhuravlev@georgiy:~/Rudnrep$ code .gitignore
georgiyzhuravlev@georgiy:~/Rudnrep$ git add README1.md
georgiyzhuravlev@georgiy:~/Rudnrep$ git commit -m "gitignore"
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
georgiyzhuravlev@georgiy:~/Rudnrep$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
georgiyzhuravlev@georgiy:~/Rudnrep$ code .gitignore
georgiyzhuravlev@georgiy:~/Rudnrep$ git push
Username for 'https://github.com': GeorgiyZhuravlevRudn
Password for 'https://GeorgiyZhuravlevRudn@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 6 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 6.08 KiB | 6.08 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/GeorgiyZhuravlevRudn/gzhuravlev.pfur
be80e17..1f58765 master -> master
georgiyzhuravlev@georgiy:~/Rudnrep$

```

```

georgiyzhuravlev@georgiy: ~/Rudnrep
Reading state information... Done
The following NEW packages will be installed:
  git-flow
0 upgraded, 1 newly installed, 0 to remove and 7 not upgraded.
Need to get 38,8 kB of archives.
After this operation, 330 kB of additional disk space will be used.
Get:1 http://de.archive.ubuntu.com/ubuntu focal/universe amd64 git-flow all 1.12.3-1 [38,8 kB]
Fetched 38,8 kB in 0s (96,5 kB/s)
Selecting previously unselected package git-flow.
(Reading database ... 169422 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) ...
georgiyzhuravlev@georgiy:~/Rudnrep$ 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/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [/home/georgiyzhuravlev/Rudnrep/.git/hooks]
georgiyzhuravlev@georgiy:~/Rudnrep$ git branch
* develop
  master
georgiyzhuravlev@georgiy:~/Rudnrep$

```

### 7. Инициализируем git flow.

```

georgiyzhuravlev@georgiy:~/Rudnrep$ echo "1.0.0" >> VERSION
georgiyzhuravlev@georgiy:~/Rudnrep$ git add .
georgiyzhuravlev@georgiy:~/Rudnrep$ git commit -am 'chore(main): add version'
[release/1.0.0 aa26c8f] chore(main): add version
1 file changed, 1 insertion(+)
 create mode 100644 VERSION
georgiyzhuravlev@georgiy:~/Rudnrep$ git flow release finish 1.0.0
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
Merge made by the 'recursive' strategy.
 VERSION | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 VERSION
Already on 'master'
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)
fatal: no tag message?
Fatal: Tagging failed. Please run finish again to retry.
georgiyzhuravlev@georgiy:~/Rudnrep$ git push --all
Username for 'https://github.com': GeorgiyZhuravlevRudn
Password for 'https://GeorgiyZhuravlevRudn@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 6 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 387 bytes | 387.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/GeorgiyZhuravlevRudn/gzhuravlev.pfur
 1f58765..e6465f9 master -> master
* [new branch]      develop -> develop
* [new branch]      release/1.0.0 -> release/1.0.0

```

### 8. Добавляем релиз и загружаем на Github.



```
* [new branch]      release/1.0.0 -> release/1.0.0
georgiyzhuravlev@georgiy:~/Rudnrep$ git push --tags
Username for 'https://github.com': GeorgiyZhuravlevRudn
Password for 'https://GeorgiyZhuravlevRudn@github.com':
Everything up-to-date
georgiyzhuravlev@georgiy:~/Rudnrep$
```

GeorgiyZhuravlevRudn / gzhuravlev.pfur

<> Code ⓘ Issues 🔗 Pull requests ▶ Actions 📁 Projects ⋮

Releases Tags

Edit release Delete

release Latest release

1.0.0 aa26c8f Compare ▼

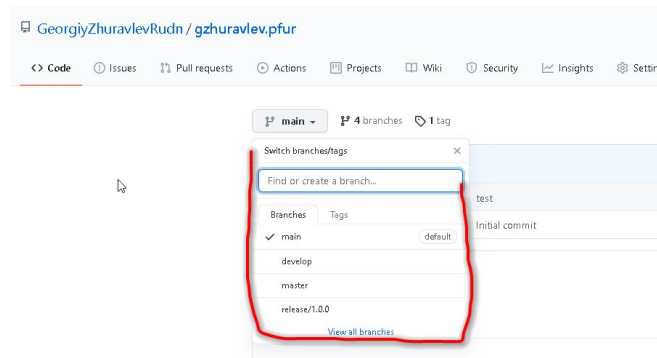
GeorgiyZhuravlevRudn released this now

release 1.0.0

▼ Assets 2

Source code (zip)

Source code (tar.gz)



### 9. Получившиеся результаты(ветви + данные).

## Вывод

Благодаря этой лабораторной работе, я научился: создавать репозитории, создавать файлы и проходить процесс подтверждения, загружать файлы в репозиторий git, делать релизы и создавать ветви.

## Контрольные вопросы

1. VCS – ПО для облегчения работы с изменяющейся информацией(хранит изменяющиеся версии; может быть изменена разными людьми, если проходит работа над совместным проектом). VCS предназначена для удобства использования в проектах, например один человек написал некоторый код, который хочет поместить в основную ветвь проекта, его коллеги могут проверить и сделать заметки в виде комментариев.
2. Хранилище – основное место хранения; Commit – подтверждение всех изменений и тп., для дальнейшей загрузки в репозиторий; History – история изменения; Рабочая копия – действительные папка с файлами.
3. Централизованные системы контроля представляют собой приложения типа клиент-сервер;( 1 основной репозиторий)(SVN) Распределенные системы контроля версий позволяют хранить репозиторий (его копию) у каждого разработчика, работающего с данной системой.( множество репозитория, взаимодействующих с сервером)(Git)
4. Создание репозитория ~> создание/добавление файлов в локальную файловую систему ~> добавление в локальный репозиторий ~> подтверждение добавления ~> загрузка в онлайн репозиторий.
5. •
6. •

7. Git add – adds a file to the staging area; git commit- opens chosen git editor or git commit -m “message”(commitment with a message); git push – pushes files to repository; git pull – downloads/ changes files from repository; git branch – shows branches; git status – status of your process; git rm “”- removes a file/es; and so on.
8. Создание файла в репозитории( лок.); загрузка файлов на удалённый репозиторий(удал.)
9. Ветви – специальные разделения дерева, которые нужны для удобства использования git, например для дальнейшего merg’a.
10. “git editor”.gitignore ~>git add .gitignore~>git commit; Игнорирование нужно для исключения ненужных файлов/ файловых систем/ тп. из области работы.