

Лабораторная работа №2

Отчёт

Сергеев Д. О.

08 марта 2025

Российский университет дружбы народов, Москва, Россия

Информация

- Сергеев Даниил Олегович
- Студент
- Направление: Прикладная информатика
- Российский университет дружбы народов
- 1132246837@pfur.ru

Цель работы

Изучить идеологию и применение средств контроля версий. Освоить умения по работе с git.

Задание

- Создать базовую конфигурацию для работы с git.
- Создать ключ SSH.
- Создать ключ PGP.
- Настроить подписи git.
- Зарегистрироваться на Github.
- Создать локальный каталог для выполнения заданий по предмету.

Выполнение лабораторной работы

Установка необходимого ПО (git, gh)

Установим git и gh.

```
foot
[dosergeev@vbox ~]$ dnf install git
Для выполнения запрошенной операции требуются привилегии суперпользователя. Пожалуйста, войдите в систему как пользователь с повышенными правами или используйте опции "--assumeno" или "--downloadonly", чтобы выполнить команду без изменения состояния системы.
[dosergeev@vbox ~]$ sudo dnf install git
[sudo] пароль для dosergeev:
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет "git-2.48.1-1.fc41.x86_64" уже установлен.

Нечего делать.
[dosergeev@vbox ~]$ sudo dnf install gh
Обновление и загрузка репозитория:
Репозитории загружены.
Пакет "gh-2.65.0-1.fc41.x86_64" уже установлен.

Нечего делать.
[dosergeev@vbox ~]$
```

Рис. 1: Процесс установки git и gh.

Необходимые пакеты уже установлены, поэтому продолжим.

Зададим имя и почту владельца репозитория, настроим utf-8 в выводе сообщений git, зададим имя начальной ветки master, параметр autocrlf и safecrlf.

```
[dosergeev@vbox ~]$ git config --global user.name "Daniel Sergeev"
[dosergeev@vbox ~]$ git config --global user.email "den.sergeev17@yandex.ru"
[dosergeev@vbox ~]$ git config --global core.quotepath false
[dosergeev@vbox ~]$ git config --global init.defaultBranch master
[dosergeev@vbox ~]$ git config --global core.autocrlf input
[dosergeev@vbox ~]$ git config --global core.safecrlf warn
[dosergeev@vbox ~]$
```

Рис. 2: Настройка параметров git.

Создание ключа ssh

Создадим ключ ssh по алгоритму ed25519.

```
[dosergeev@vbox ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/dosergeev/.ssh/id_ed25519):
/home/dosergeev/.ssh/id_ed25519 already exists.
Overwrite (y/n)? y
Enter passphrase for "/home/dosergeev/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/dosergeev/.ssh/id_ed25519
Your public key has been saved in /home/dosergeev/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:v3AUI4rS8XfHc5zCUF1YP1Do1HsEZMX+ftfEfN2XFT8 dosergeev@vbox
The key's randomart image is:
+--[ED25519 256]--+
|      .oo..oo*+|
|      .. o .+ +|
|      . . B o oEB|
|      . + . o X o *B|
|      . o o S = B * o|
|      . . . = o = oo|
|      . +      +|
|      o .      |
|      .        |
+-----[SHA256]-----+
[dosergeev@vbox ~]$
```

Рис. 3: Генерация ключа ssh.

Выведем ключ, скопируем и вставим в гит.



```
[doseergeev@vbox ~]$ cat ~/.ssh/id_ed25519.pub  
ssh-ed25519 AAAAC3NzaC11ZDI1NTE5AAAAIF3j9F4WtnPcPK1WuzebDn98sutuJdUBPoIj0rKQAM61 doseergeev@vbox  
[doseergeev@vbox ~]$
```

Рис. 4: Копирование ключа ssh.

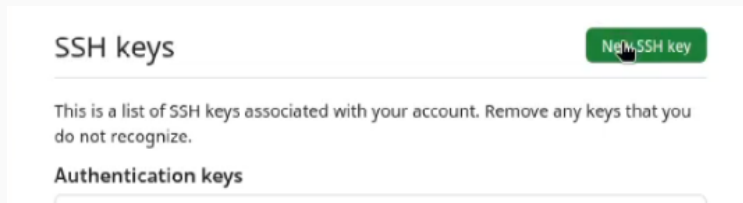


Рис. 5: Добавляем ключ ssh в гите.

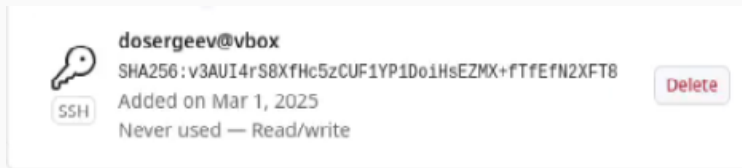


Рис. 6: Вид ключа ssh в гите.

сгенерируем ключ с помощью команды `gpg --full-generate-key`. В предложенных вариантах выберем: - тип RSA and RSA. - размер 4096. - срок действия по умолчанию. - имя. - адрес электронной почты. - без комментария.

```
obs@obs-gnome: ~/gpg --full-generate-key
gpg (GnuPG) 2.4.5; Copyright (C) 2024 g10 Code Gent
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Выберите тип ключа.
(1) RSA and RSA
(2) RSA and ElGamal
(3) DSA (sign only)
(4) RSA (sign only)
(5) ECC (sign and encrypt) "Default"
(10) ECC (только для подписи)
(14) Existing key from cache
Как выбрать? 1
Ключи ключей RSA могут быть от 1024 до 4096.
Какой размер ключа Вам необходим? (1024) 4096
Запрошенный размер ключа - 4096 бит
Выберите срок действия ключа.
0 = не ограничен
<0> = срок действия ключа - в дней
<0w> = срок действия ключа - в недель
<0m> = срок действия ключа - в месяцев
<0y> = срок действия ключа - в лет
Срок действия ключа? (0) 0
Срок действия ключа не ограничен
Вос. нулею? (y/N) y

GnuPG должен составить идентификатор пользователя для идентификации ключа.
Ваше полное имя: Daria
Адрес электронной почты: den.seguren17@yandex.ru
Примечание:
Вы выбрали следующий идентификатор пользователя:
'Daria' <den.seguren17@yandex.ru>
Создать (Y/n), (C)Примечание, (E)Адрес, (O)Примечание, (Q)Вывод
```

Рис. 7: Генерация ключа gpg.

Добавление ключа gpg в GitHub

Выведем список ключей.

```
[dosergeev@vbox ~]$ gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: глубина: 0 достоверных: 2 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f, 2u
[keyboard]
-----
sec   rsa4096/05C705DF9D20ECD1 2025-03-01 [SC]
      32678AE5CFD81A6FDE0D7AA905C705DF9D20ECD1
uid           [ абсолютно ] Daniel <den.sergeev17@yandex.ru>
ssb   rsa4096/F0C676A33E6CBC16 2025-03-01 [E]

sec   rsa4096/D26A655795F05626 2025-03-01 [SC]
      89D5B4E2E3851038DF030508D26A655795F05626
uid           [ абсолютно ] Daniel <den.sergeev17@yandex.ru>
ssb   rsa4096/8A934455760D3D71 2025-03-01 [E]

[dosergeev@vbox ~]$
```

Рис. 8: Список ключей gpg.

Скопируем сгенерированный ключ в буфер обмена с помощью его отпечатка

```
[dosergeev@vbox ~]$ gpg --armor --export 05C705DF9D20ECD1
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGfDI1MBEADLI dGauIEC4Q32Km2PxSouqshU4EURnwmNtq0PVQ+Q1JdgDbXb
AgWYdTauUvdKE7yI66n1B0bxvtYqh4t50sgDHqng1H7J/nh+u3U+UerCj8Ua+7Ny
Oq/WE42gIlqyx4454P0RzaNerMAIucc1ro/PRH87CA+fX+UFc5Pa584EUsh/FJum
qKkoTh/hxADIj2MhBXqdzSL8kh5Qk+dDnGQ6qHMOuvBsD2CKfXOkpKrxGo7Hhr06
LsQ4qokJN0F5F2zqIPerzvZYNBfQpWPMpru9Uvb/NrMzgDqjaKPPHB0nCmZ2P6/R
ACa7LOCevQqFinFStBZvIgQzNJc5/+E1kEXHzUfY0tdynu00dYz115Jqu9+IPsqk
bZ22bwsJ2V12nYpI+eT71TT4ZV10AQJbeXJyDhmf0i13JL7ZXBad3kcuNDHbWmk4
```

Рис. 9: Копирование ключа gpg.

Добавим ключ gpg в Github.

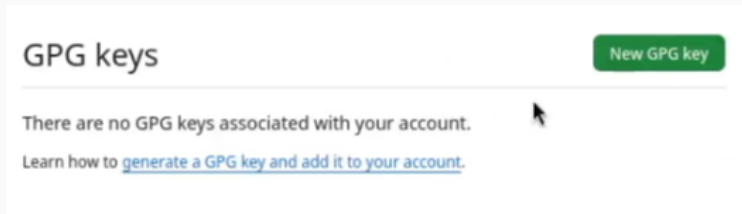


Рис. 10: Добавляем ключ gpg в гите.

Add new GPG key

Title

Key

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQIwTVOmYwobcWf5qEVCsXEm0N+adKROEKjYjY5VmoQh3Xh0X7kZqieeJk
CQ
w+rWOECtOCK3BUiLb7eB4EejTpGeutXnd2aXtD6LjZ5uqy4eOEvIsJjR5KugCfHP
CD9YL65UI3sBOSLtS0YgptIcv6odciDH3DRJ6As/6ojZlPj/kqst0ZzUwhVbs/Z9
WbExcWPP7YrnA6RHv+ptm2tmROQgwHW7ckX0KANAQOQWYY/1G1GPkyOgx
17whrtI
n0I3TyHNP1XnoaKpNMjXrBPfCsV3Sy5SEXT3eoFZlGai2JfSjx6WISiZtt+WeDIC
2+ze5byNpjzrxjGotxae3j880ph+QIR3EYAl6Gn20+M6My8SpafriDHVlcY9r3g=
=P5mp
-----END PGP PUBLIC KEY BLOCK-----
```

Add GPG key

Рис. 11: Вводим в окно для ключа gpg.

GPG keys

[New GPG key](#)

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



GPG

Delete

Email address: den.sergeev17@yandex.ru

Key ID: 05C705DF9D20ECD1

Subkeys: F0C676A33E6CBC16

Added on Mar 1, 2025

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

Рис. 12: Вид ключа gpg в гите.

Указываем git-у применять адрес почты при подписи коммитов.

```
[dosergeev@vbox ~]$ git config --global user.signingkey 05C705DF9D20EC  
01  
[dosergeev@vbox ~]$ git config --global commit.gpgsign true  
[dosergeev@vbox ~]$ git config --global gpg.program $(which gpg2)  
[dosergeev@vbox ~]$
```

Рис. 13: Настройка подписей коммитов git.

Настройка автоматических подписей коммитов git

Авторизуемся через браузер.

```
[dosergeev@vbox ~]$ gh auth login
? Where do you use GitHub? GitHub.com
? What is your preferred protocol for Git operations on this host? HTTP
? Authenticate Git with your GitHub credentials? Yes
? How would you like to authenticate GitHub CLI? [Use arrows to move,
type to filter]
> Login with a web browser
  Paste an authentication token
```

Рис. 14: Выбираем вход через браузер.

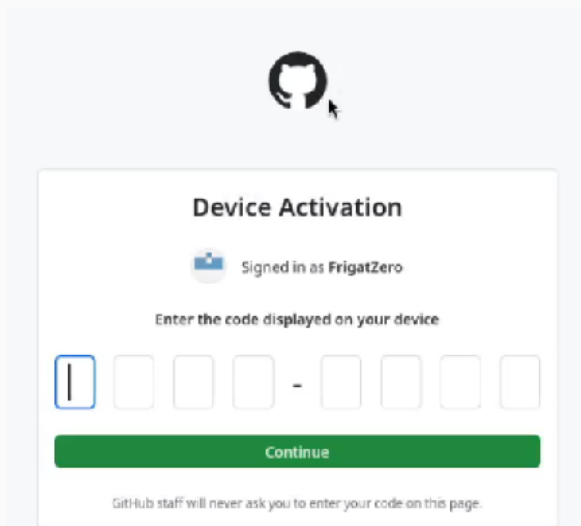


Рис. 15: Вводим ключ.

Создадим репозиторий гит и каталог курса.

```
[dosergeev@vbox ~]$ mkdir -p ~/work/study/2024-2025/os  
[dosergeev@vbox ~]$ cd ~/work/study/2024-2025/os  
[dosergeev@vbox os]$ gh repo create study_2024-2025_os-intro --template=y  
amadharma/course-directory-student-template --public  
✓ Created repository FrigatZero/study_2024-2025_os-intro on GitHub  
https://github.com/FrigatZero/study_2024-2025_os-intro  
[dosergeev@vbox os]$
```

Рис. 16: Создание репозитория курса.


```
[dosergeev@vbox os]$ git clone --recursive git@github.com:FrigatZero/study_2024-2025_os-intro.git
Клонирование в «study_2024-2025_os-intro»...
The authenticity of host 'github.com (140.82.121.4)' can't be established
.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvUkr4UvCOqU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
```

Рис. 17: Клонирование удалённого репозитория в локальный.

Создание репозитория курса на GitHub

Перейдем в каталог курса и удалим лишние файлы. Создадим необходимые каталоги с помощью make prepare.

```
[dosergeev@vbox ~]$ cd ~/work/study/2024-2025/os/study_2024-2025_os-intro
/
[dosergeev@vbox study_2024-2025_os-intro]$ rm package.json
[dosergeev@vbox study_2024-2025_os-intro]$ make
Usage:
  make <target>

Targets:
  list           List of courses
  prepare        Generate directories structure
  submodule      Update submodules

[dosergeev@vbox study_2024-2025_os-intro]$ make prepare
[dosergeev@vbox study_2024-2025_os-intro]$ ls
CHANGELOG.md  labs      prepare      README.en.md  template
config        LICENSE   presentation  README.git-flow.md
COURSE        Makefile  project-personal  README.md
[dosergeev@vbox study_2024-2025_os-intro]$
```

Рис. 18: Удаление лишних файлов и создание необходимых каталогов make.

Создание репозитория курса на GitHub

Отправим файлы на сервер.

```
[dosergeev@vbox study_2024-2025_os-intro]$ git add .  
[dosergeev@vbox study_2024-2025_os-intro]$ git commit -am 'feat(main): make course stucture'
```

Рис. 19: Сохраняем изменения.

```
Перечисление объектов: 40, готово.  
Подсчет объектов: 100% (40/40), готово.  
При сжатии изменений используется до 4 потоков  
Сжатие объектов: 100% (30/30), готово.  
Запись объектов: 100% (38/38), 342.32 КиБ | 1.47 МБ/с, готово.  
Total 38 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)  
remote: Resolving deltas: 100% (4/4), completed with 1 local object.  
To github.com:FrigatZero/study_2024-2025_os-intro.git  
    b52ba75..1642e1f  master -> master  
[dosergeev@vbox study_2024-2025_os-intro]$
```

Рис. 20: Отправляем файлы на сервер.

Ответы на контрольные вопросы

1. Системы контроля версий - это инструменты, позволяющие организовать работу над проектом разработки, выкладывать его обновления и контролировать релизы и изменения кода. Они предназначены для отслеживания изменений, защиты исходного кода от удаления и изменения, возможности отката изменений и командной работы для 10 и более человек.

2.

- Хранилище - репозиторий в котором хранятся файлы проекта в различных версиях.
- Commit - комментарий внесённых изменений в репозитории.
- История - история изменений файлов проекта.
- Рабочая копия - копия, созданная из определенной версии репозитория, которую модифицирует разработчик.

3. Централизованные системы контроля версий имеют единый сервер под хранение проекта, для изменения которых необходимо скачать необходимые файлы, изменить и вернуть обратно на сервер. Пример централизованной VCS: Subversion. Децентрализованные системы полностью копируют удалённый репозиторий в локальный. При этом внесенные изменения отправляются на сервер в качестве новой версии. Пример: git.

4. Создается репозиторий для работы с проектом, при необходимости файлы обновляются локально и отправляются на сервер в качестве новой версии.

5.Репозиторий копируется локально. После внесения изменений файлы загружаются на сервер в качестве отдельной версии. После этого измененная ветка может быть объединена с текущей и отправлена в релиз.

6.Хранение файлов проекта, отслеживание версий, защита от изменений, работа в команде.

7.

- `git clone` – клонирование проекта с сервера в указанный локальный репозиторий.
- `git add` – добавляет все изменённые или созданные файлы или каталоги.
- `git commit` – сохраняет изменения репозитория с комментарием.
- `git push` – загружает добавленные изменения на сервер
- `git pull` – получить последние изменения с сервера
- `git rm` – удаляет файл из индекса репозитория
- `git status` – просматривает список измененных файлов в текущем репозитории

8.С локальным: `git add .` , `git commit -am commit`, `git push` - добавление всех измененных файлов в текущем каталоге, их сохранение с комментарием `commit`, `git pull` - получаем последние изменения с удалённого репозитория. С удалённым: `git push` - отправляем сохраненные изменения на сервер.

9.Ветки - это различные версии исходного репозитория, являющиеся копиями с внесенными изменениями. Нужны для параллельной работы над проектом. Могут быть объединены для внесения изменений. Их можно проигнорировать, добавив имя в `.gitignore`.

10.Мы можем игнорировать некоторые файлы при коммитах, когда не хотим добавлять их в удаленный репозиторий для чистоты или упрощения.

Вывод

В результате выполнения лабораторной работы я изучил как применять средства контроля версий для работы с удалённым репозиторием и освоил умения по работе с git.