

# **Лабораторная работы 2**

**Первоначальная настройка git**

Богданюк Анна Васильевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
<b>5</b>	<b>Выводы</b>	<b>14</b>

# Список иллюстраций

4.1	Имя и email владельца репозитория . . . . .	9
4.2	utf-8 . . . . .	9
4.3	Имя начальной ветки . . . . .	9
4.4	autocrlf . . . . .	9
4.5	safecrlf . . . . .	9
4.6	Алгоритм rsa . . . . .	10
4.7	Алгоритм ed25519 . . . . .	10
4.8	Ключ pgr . . . . .	10
4.9	Passphrase . . . . .	10
4.10	Список ключей . . . . .	11
4.11	Копирование ключа . . . . .	11
4.12	New GPG key . . . . .	11
4.13	Подпись коммитов . . . . .	11
4.14	gh . . . . .	12
4.15	Настройка gh . . . . .	12
4.16	Создание репозитория . . . . .	12
4.17	Клонирование репозитория . . . . .	12
4.18	Удаление файла . . . . .	12
4.19	Создание каталогов . . . . .	13
4.20	Отправление файлов на сервер . . . . .	13

## Список таблиц

# 1 Цель работы

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

## 2 Задание

1. Базовая настройка git
2. Создание ssh ключа
3. Верификация коммитов с помощью PGP
4. Проверка коммитов в Git
5. Шаблон рабочего пространства

### 3 Теоретическое введение

Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет их фиксировать, совмещать изменения, произведённые разными участниками проекта, производить откат к любой более ранней версии проекта, если это требуется. 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 не полную версию изменённых файлов, а производить так называемую дельта-компрессию — сохранять только изменения между последовательными версиями, что позволяет уменьшить объём хранимых данных. Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблокировать файлы для изменения. В зави-

симости от настроек блокировка не позволяет другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, привилегированный доступ только одному пользователю, работающему с файлом.



## 4 Выполнение лабораторной работы

Задаю имя и email владельца репозитория (рис. 4.1).

```
avbogdanyuk@Bogdanyuk:~$ git config --global user.name "Anna Bogdanyuk"  
avbogdanyuk@Bogdanyuk:~$ git config --global user.email "silverblood2606@gmail.com"
```

Рис. 4.1: Имя и email владельца репозитория

Настраиваю utf-8 в выводе сообщений git (рис. 4.2).

```
avbogdanyuk@Bogdanyuk:~$ git config --global core.quotePath false  
avbogdanyuk@Bogdanyuk:~$
```

Рис. 4.2: utf-8

Настраиваю верификацию и подписание коммитов git. Задаю имя начальной ветки (master) (рис. 4.3).

```
avbogdanyuk@Bogdanyuk:~$ git config --global init.defaultBranch master
```

Рис. 4.3: Имя начальной ветки

Устанавливаю параметр autocrlf (рис. 4.4).

```
avbogdanyuk@Bogdanyuk:~$ git config --global core.autocrlf  
input
```

Рис. 4.4: autocrlf

Устанавливаю параметр safecrlf (рис. 4.5).

```
avbogdanyuk@Bogdanyuk:~$ git config --global core.safecrlf warn
```

Рис. 4.5: safecrlf

Создаю ключ ssh по алгоритму rsa с ключём размером 4096 бит (рис. 4.6).

```
avbogdanyuk@Bogdanyuk:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
```

Рис. 4.6: Алгоритм rsa

По алгоритму ed25519 (рис. 4.7).

```
avbogdanyuk@Bogdanyuk:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/avbogdanyuk/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/avbogdanyuk/.ssh/id_ed25519
Your public key has been saved in /home/avbogdanyuk/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:Ar3Mxr8kVn1Ibrqf4EMh5eL8YuMh2ta/3U1UvJ+HSMU avbogdanyuk@Bogdanyuk
The key's randomart image is:
+--[ED25519 256]--+
|
|..o..o|
|oO.O+..|
|oBoS.=...E|
|.O+.O..=+|
|..+O=O.+|
|O.O==O=O O|
|...O.+*++..|
+-----[SHA256]-----+
```

Рис. 4.7: Алгоритм ed25519

Генерируем ключ gpg (рис. 4.8).

```
avbogdanyuk@Bogdanyuk:~$ gpg --full-generate-key
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```

Рис. 4.8: Ключ gpg

Ввожу passphrase, чтобы защитить мой ключ, учетная запись на github осталась с прошлого семестра (рис. 4.9).

```
Please enter the passphrase to
protect your new key

Passphrase: _____

<OK> <Cancel>
```

Рис. 4.9: Passphrase

Вывожу список ключей и копирую отпечаток приватного ключа (рис. 4.10).

```
avbogdanyuk@Bogdanyuk:~$ gpg --list-secret-keys --keyid-format LONG
gpg: checking the trustdb
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u
/home/avbogdanyuk/.gnupg/pubring.kbx
-----
sec   rsa3072/55E9ACD3528412D9 2024-02-25 [SC]
      9B980894BC1374A4E5D12B1A55E9ACD3528412D9
uid           [ultimate] Anna Bogdanyuk <silverblood2606@gmail.com>
ssb   rsa3072/039AE62AAE64CE1F 2024-02-25 [E]
```

Рис. 4.10: Список ключей

Копирую сгенерированный PGP ключ в буфер обмена (рис. 4.11).

```
avbogdanyuk@Bogdanyuk:~$ gpg --armor --export 55E9ACD3528412D9 | xclip -s
```

Рис. 4.11: Копирование ключа

Перехожу в настройки Github, нажимаю на кнопку New GPG key и вставляю полученный ключ в поле ввода (рис. 4.12).

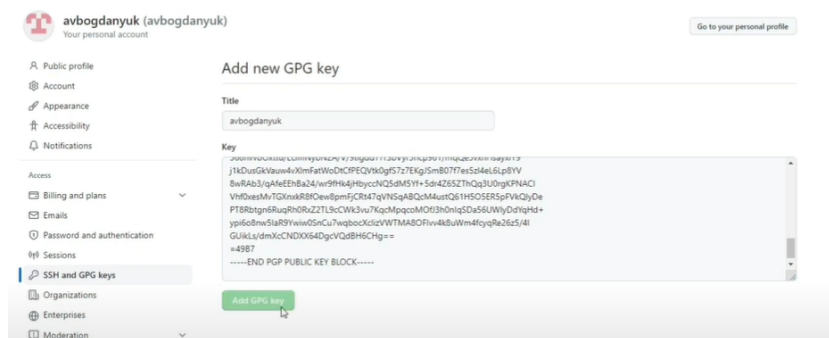


Рис. 4.12: New GPG key

Использую введенных email, указываю Git применять его при подписи коммитов (рис. 4.13).

```
avbogdanyuk@Bogdanyuk:~$ git config --global user.signingkey 55E9ACD3528412D9
avbogdanyuk@Bogdanyuk:~$ git config --global gpg.program $(which gpg2)
```

Рис. 4.13: Подпись коммитов

Устанавливаю gh (рис. 4.14).

```
avbogdanyuk@bogdanyuk:~$ sudo snap install gh # version 2.6.0-15-g1a10fd5a
Setup snap "gh" (502) security profile
```

Рис. 4.14: gh

Авторизуюсь через браузер (рис. 4.15).

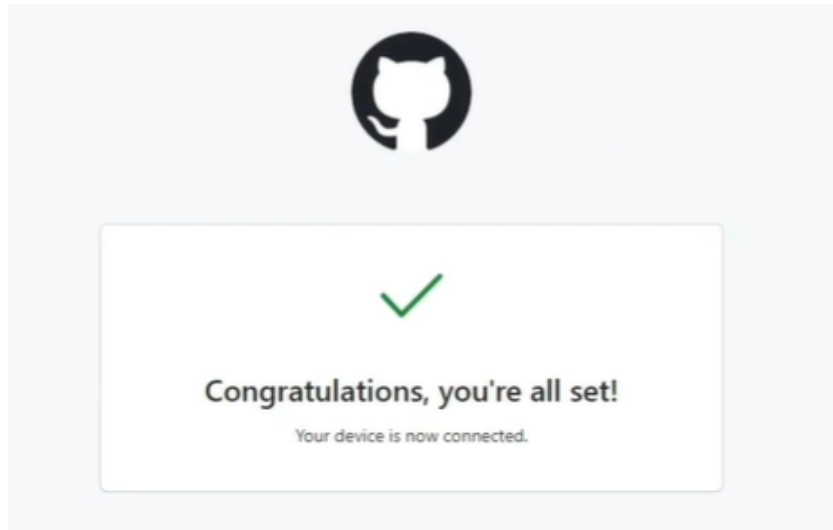


Рис. 4.15: Настройка gh

Создаю репозиторий курса на основе шаблона (рис. 4.16).

```
avbogdanyuk@bogdanyuk:~$ mkdir -p ~/work/study/2023-2024/Операционные системы
avbogdanyuk@bogdanyuk:~$ cd ~/work/study/2023-2024/Операционные системы
avbogdanyuk@bogdanyuk:~/work/study/2023-2024/Операционные системы$ gh repo create study_2023-2024-os-intro --template=yamadharma/course-directory-student-template --public
2024/02/25 20:14:15.912080 cmd_run.go:1055: WARNING: cannot start document portal: dial unix /run/user/1000/bus: connect
: no such file or directory
Created repository avbogdanyuk/study_2023-2024-os-intro on GitHub
```

Рис. 4.16: Создание репозитория

Клонирую репозиторий с шаблона (рис. 4.17).

```
avbogdanyuk@bogdanyuk:~/work/study/2023-2024/Операционные системы$ git clone --recursive git@github.com:avbogdanyuk/study_2023-2024-os-intro.git os-intro
Cloning into 'os-intro'...
remote: Enumerating objects: 32, done.
remote: Counting objects: 100% (32/32), done.
remote: Compressing objects: 100% (31/31), done.
remote: Total 32 (delta 1), reused 18 (delta 0), pack-reused 0
Receiving objects: 100% (32/32), 18.60 KiB | 6.20 MiB/s, done.
```

Рис. 4.17: Клонирование репозитория

Перехожу в каталог курса и удаляю лишний файл (рис. 4.18).

```
avbogdanyuk@bogdanyuk:~/work/study/2023-2024/Операционные системы$ cd os-intro
avbogdanyuk@bogdanyuk:~/work/study/2023-2024/Операционные системы/os-intro$ rm package.json
```

Рис. 4.18: Удаление файла

Создаю необходимые каталоги (рис. 4.19).

```
avbogdanyuk@bogdanyuk:~/work/study/2023-2024/Операционные системы/os-intro$ echo os-intro > COURSE
avbogdanyuk@bogdanyuk:~/work/study/2023-2024/Операционные системы/os-intro$ make
```

Рис. 4.19: Создание каталогов

Отправляю файлы на сервер (рис. 4.20).

```
avbogdanyuk@bogdanyuk:~/work/study/2023-2024/Операционные системы/os-intro$ git add .
avbogdanyuk@bogdanyuk:~/work/study/2023-2024/Операционные системы/os-intro$ git commit -am 'feat(main): make course structure'
[master cbd7090] feat(main): make course structure
2 files changed, 1 insertion(+), 14 deletions(-)
delete mode 100644 package.json
avbogdanyuk@bogdanyuk:~/work/study/2023-2024/Операционные системы/os-intro$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 296 bytes | 296.00 KiB/s, done.
```

Рис. 4.20: Отправление файлов на сервер

## 5 Выводы

В ходе выполнения лабораторной работы были изучены идеология и применение средств контроля версий и были освоены умения по работе с git.