

# **Лабораторная работа №2**

**Первоначальная настройка Git**

Козлов Всеволод Павлович

# Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	19
	Список литературы	20

## Список иллюстраций

3.1	Установка Git . . . . .	7
3.2	Установка gh . . . . .	8
3.3	Имя и почта пользователя . . . . .	8
3.4	Настройка utf-8 . . . . .	9
3.5	Создание GPG ключа . . . . .	9
3.6	Экспорт ключа . . . . .	10
3.7	Копирование ключа . . . . .	10
3.8	Передача ключа на Github . . . . .	11
3.9	Настройка автоматических подписей . . . . .	11
3.10	Задание начальной ветви, настройка autocrlf и safecrlf . . . . .	12
3.11	Создание ключа SSH через rsa . . . . .	12
3.12	Создание SSH ключа через ed25519 . . . . .	13
3.13	Авторизация на gh . . . . .	13
3.14	Удаление лишних файлов, создание каталогов . . . . .	14
3.15	Отправка файлов на сервер: add, commit . . . . .	14
3.16	Отправка файлов на сервер: push . . . . .	15

## **Список таблиц**

# 1 Цель работы

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

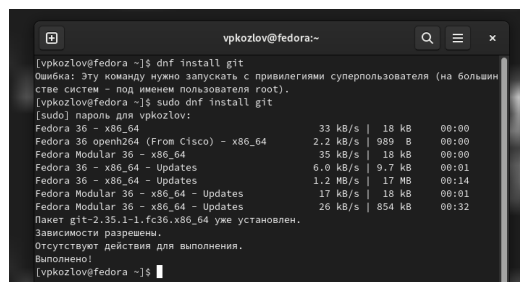
## 2 Задание

- 1) Создать базовую конфигурацию для работы с git.
- 2) Создать ключ SSH.
- 3) Создать ключ PGP.
- 4) Зарегистрироваться на Github.
- 5) Создать локальный каталог для выполнения заданий по предмету.

### 3 Выполнение лабораторной работы

#### 1) Установка программного обеспечения:

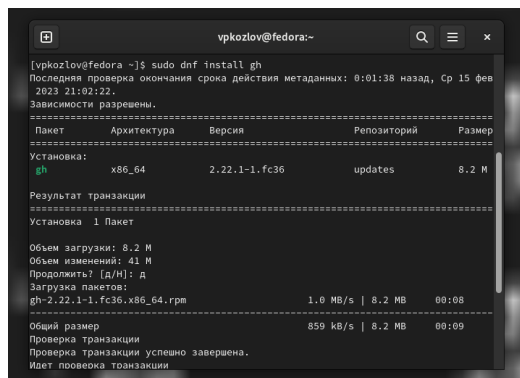
Произвел установку Git через терминал (рис. [3.1])



```
vpkozlov@fedora:~  
[vpkozlov@fedora ~]$ dnf install git  
Ошибка: Эту команду нужно запускать с привилегиями суперпользователя (на большин  
стве систем - под именем пользователя root).  
[vpkozlov@fedora ~]$ sudo dnf install git  
[sudo] пароль для vpkozlov:  
Fedora 36 - x86_64 33 kB/s | 18 kB 00:00  
Fedora 36 openh264 (From Cisco) - x86_64 2.2 kB/s | 989 B 00:00  
Fedora Modular 36 - x86_64 35 kB/s | 18 kB 00:00  
Fedora 36 - x86_64 - Updates 6.9 kB/s | 9.7 kB 00:01  
Fedora 36 - x86_64 - Updates 1.2 MB/s | 17 MB 00:14  
Fedora Modular 36 - x86_64 - Updates 17 kB/s | 18 kB 00:01  
Fedora Modular 36 - x86_64 - Updates 26 kB/s | 854 kB 00:32  
Пакет git-2.35.1-1.fc36.x86_64 уже установлен.  
Зависимости разрешены.  
Отсутствуют действия для выполнения.  
Выполнено!  
[vpkozlov@fedora ~]$
```

Рис. 3.1: Установка Git

Произвел установку gh (рис. [3.2])



```
vpkzlov@fedora:~  
[vpkzlov@fedora ~]$ sudo dnf install gh  
Последняя проверка окончания срока действия метаданных: 0:01:38 назад, Ср 15 фев 2023 21:02:22.  
Зависимости разрешены.  
=====
```

Пакет	Архитектура	Версия	Репозиторий	Размер
gh	x86_64	2.22.1-1.fc36	updates	8.2 М

```
=====
```

Результат транзакции

Установка 1 Пакет

Объем загрузки: 8.2 М  
Объем изменений: 41 М  
Продолжить? [Д/Н]: Д  
Загрузка пакетов:  
gh-2.22.1-1.fc36.x86\_64.rpm 1.0 MB/s | 8.2 MB 00:08

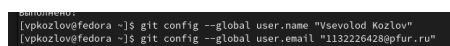
Общий размер 859 kB/s | 8.2 MB 00:09

Проверка транзакции  
Проверка транзакции успешно завершена.  
Идет проверка транзакции

Рис. 3.2: Установка gh

## 2) Базовая настройка Git:

Задал имя и почту пользователя (рис. [3.3])



```
[vpkzlov@fedora ~]$ git config --global user.name "Vsevolod Kozlov"  
[vpkzlov@fedora ~]$ git config --global user.email "1132226428@pfur.ru"
```

Рис. 3.3: Имя и почта пользователя

Настроил utf-8 в выводе сообщений git (рис. [3.4])



```
[vpkozlov@fedora ~]$ git config --global user.email "1132226428@pfur.ru"
[vpkozlov@fedora ~]$ git config --global core.quotepath false
```

Рис. 3.4: Настройка utf-8

Создал GPG ключ (рис. 3.5)

```
pub rsa4096 2023-02-15 [SC]
    BA1A150F8EACA30EC73554C0ED473B4D36790ETA
uid                               Vsevolod <1132226428@pfur.ru>
sub rsa4096 2023-02-15 [E]
[vpkozlov@fedora ~]$
```

Рис. 3.5: Создание GPG ключа

Экспортировал GPG ключ (рис. [3.6])

```
[vpkozlov@fedora ~]$ gpg --list-secret-keys --keyid-format LONG
gpg: проверка таблицы доверия
gpg: marginals needed: 3 completes needed: 1 trust model: gpg
gpg: глубина: 0 достоверных: 1 подписанных: 0 доверие: 0-, 0q, 0n, 0m, 0f
y, 1u
/home/vpkozlov/.gnupg/pubring.kbx
-----
sec   rsa4096/ED47384D36790E7A 2023-02-15 [SC]
      BA1A15DF8EACA30EC73554C0ED47384D36790E7A
uid     [  вскоментно ] Vsevolod <i132226428@pfur.ru>
ssb     rsa4096/486653E773790EDC 2023-02-15 [E]

[vpkozlov@fedora ~]$ gpg --armor --export <PGP Fingerprint>
bash: синтаксическая ошибка рядом с неожиданным маркером «newline»
[vpkozlov@fedora ~]$ gpg --armor --export <PGP Fingerprint>
bash: синтаксическая ошибка рядом с неожиданным маркером «newline»
[vpkozlov@fedora ~]$ gpg --armor --export BA1A15DF8EACA30EC73554C0ED47384D36790E7A
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGPTIm0BEAcwpt4eMMpy1c+q3ybPg8GDCKnPowu8DjTH0HbBj2YrcC3jjuI
JR3eyEXMKKh3C141FpuzWBC9FnNQ0PD22fjWVmy8ZtUK0E9k6chSt/tm0bNe+o40
```

Рис. 3.6: Экспорт ключа

Скопировал GPG ключ (рис. @[fig:007])

```
[vpkozlov@fedora ~]$ gpg --armor --export BA1A15DF8EACA30EC73554C0ED47384D36790E7A | xclip -sel clip
```

Рис. 3.7: Копирование ключа

Передал GPG ключ на Github (рис. [3.8])

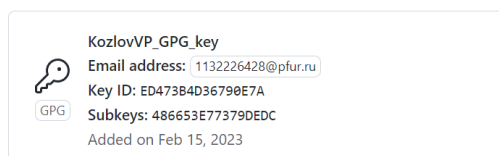


Рис. 3.8: Передача ключа на Github

Настройка автоматических подписей коммитов git (рис. [3.9])

```
[vpkozlov@fedora ~]$ git config --global user.signingkey BA1A15DF8EACA30EC73554C0ED473B4D36790E7A
[vpkozlov@fedora ~]$ git config --global commit.gpgsign true
[vpkozlov@fedora ~]$ git config --global gpg.program $(which gpg2)
```

Рис. 3.9: Настройка автоматических подписей

Задал имя начальной ветви, настроил параметры autocrlf и safecrlf (рис. [3.10])

```

[vpkozlov@fedora ~]$ git config --global init.defaultBranch master
[vpkozlov@fedora ~]$ git config --global core.autocrlf input
[vpkozlov@fedora ~]$ git config --global core.safecrlf warn
[vpkozlov@fedora ~]$

```

Рис. 3.10: Задание начальной ветви, настройка autocrlf и safecrlf

### 3) Создание ключа SSH:

Создал SSH ключ по алгоритму rsa размером 4096 бит (рис. [3.11])

```

[vpkozlov@fedora ~]$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/vpkozlov/.ssh/id_rsa):
/home/vpkozlov/.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/vpkozlov/.ssh/id_rsa
Your public key has been saved in /home/vpkozlov/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Dm]WXYUM+eqizGNlgQoar263NTcgFtuRljxKZTAzicc vpkozlov@fedora
The key's randomart image is:
+----[RSA 4096]-----+
|Xo. .+..|
|++E+ ..o.|
|o=+. .o|
|=.o.o..|
|++ o+.o.S|
|o .os .o|
| .o + .|
|..+o.o o|
|oo+..|
+----[SHA256]-----+

```

Рис. 3.11: Создание ключа SSH через rsa

Создал SSH ключ по алгоритму ed25519 (рис. [3.12])

```
[vpkozlov@fedora ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/vpkozlov/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vpkozlov/.ssh/id_ed25519
Your public key has been saved in /home/vpkozlov/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:9MT0zW9hKJwRdLS9QgZVomV085Wvx3LSyZQzp1kQhU vpkozlov@fedora
The key's randomart image is:
+--[ED25519 256]--+
|
|      =%*+.E. |
|      .+..0ooo |
|      .  =+..+ |
|      . ....=+ |
|      So.o. o B |
|      += . +. |
|      . oo . |
|      . o o . |
|      . oo.o |
|      +-----+ [SHA256]
[vpkozlov@fedora ~]$
```

Рис. 3.12: Создание SSH ключа через ed25519

4) Создание ключа PGP - сделано выше (рис. 5-9)

5) Регистрация на Github - произведена ранее

6) Настройка gh:

Авторизировался на gh (рис. [3.13])

```
[vpkozlov@fedora ~]$ gh auth login
? What account do you want to log into? GitHub.com
? What is your preferred protocol for Git operations? SSH
? Upload your SSH public key to your GitHub account? /home/vpkozlov/.ssh/id_rsa.pub
? Title for your SSH key: GH
? How would you like to authenticate GitHub CLI? Login with a web browser

First copy your one-time code: 28A3-8331
Press Enter to open github.com in your browser...
✓ Authentication complete.
- gh config set -h github.com git_protocol ssh
✓ Configured git protocol
✓ Uploaded the SSH key to your GitHub account: /home/vpkozlov/.ssh/id_rsa.pub
  Logged in as KozlovVP123
[vpkozlov@fedora ~]$
#### [Parent][PBackgroundParent] Error: RunMessage(msgname=PMessagePort::Msg_C
lose) Channel closing: too late to send/recvd, messages will be lost
```

Рис. 3.13: Авторизация на gh

7) Создание репозитория курса на основе шаблона:

Потребовалась помощь Татьяны Рефатовны, забыл сделать скриншот

## 8) Настройка каталога курса:

Удалил лишние файлы и создал необходимые каталоги (рис. [3.14])

```
vpkozlov@fedora os-intro]$ cd ~/work/study/2022-2023/"Операционные системы"/os-intro
vpkozlov@fedora os-intro]$ rm package.json
vpkozlov@fedora os-intro]$ echo os-intro > COURSE
vpkozlov@fedora os-intro]$ make
vpkozlov@fedora os-intro]$
```

Рис. 3.14: Удаление лишних файлов, создание каталогов

Отправил файлы на сервер (add, commit) (рис. [3.15])

```
vpkozlov@fedora os-intro]$ make
vpkozlov@fedora os-intro]$ git add .
vpkozlov@fedora os-intro]$ git commit -am 'feat(main): make course structure'
[master d5aefc1] feat(main): make course structure
361 files changed, 100327 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
```

Рис. 3.15: Отправка файлов на сервер: add, commit

Отправил файлы на сервер (push) (рис. [3.16])

```
[vpkozlov@fedora os-intro]$ git push
Перечисление объектов: 40, готово.
Подсчет объектов: 100% (40/40), готово.
Сжатие объектов: 100% (30/30), готово.
Запись объектов: 100% (30/30), 343.05 КиБ | 2.38 МиБ/с, готово.
Всего 38 (изменений 4), повторно использовано 0 (изменений 0), повторно использова
но пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:KozlovVP123/study_2022-2023_os-intro.git
 f51d995..d5aefc1 master -> master
[vpkozlov@fedora os-intro]$
```

Рис. 3.16: Отправка файлов на сервер: push

Ответы на контрольные вопросы:

- 1) Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются?

Система контроля версий — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое. Системы контроля версий (Version Control System, VCS) применяются для:

- Хранение полной истории изменений
- причин всех производимых изменений
- Откат изменений, если что-то пошло не так
- Поиск причины и ответственного за появления ошибок в программе
- Совместная работа группы над одним проектом
- Возможность изменять код, не мешая работе других пользователей

- 2) Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.

Репозиторий - хранилище версий - в нем хранятся все документы вместе с историей их изменения и другой служебной информацией. Commit — отслежива-

ние изменений, сохраняет разницу в изменениях Рабочая копия - копия проекта, связанная с репозиторием (текущее состояние файлов проекта, основанное на версии из хранилища (обычно на последней)) История хранит все изменения в проекте и позволяет при необходимости обратиться к нужным данным.

- 3) Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.

Централизованные VCS (Subversion; CVS; TFS; VAULT; AccuRev):

- Одно основное хранилище всего проекта
- Каждый пользователь копирует себе необходимые ему файлы из этого репозитория, изменяет и, затем, добавляет свои изменения обратно

Децентрализованные VCS (Git; Mercurial; Bazaar):

- У каждого пользователя свой вариант (возможно не один) репозитория
- Присутствует возможность добавлять и забирать изменения из любого репозитория [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. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным.

- 4) Опишите действия с VCS при единоличной работе с хранилищем

Сначала создаем и подключаем удаленный репозиторий. Затем по мере изменения проекта отправлять эти изменения на сервер.

- 5) Опишите порядок работы с общим хранилищем VCS.

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.



6) Каковы основные задачи, решаемые инструментальным средством git?

Первая — хранить информацию о всех изменениях в вашем коде, начиная с самой первой строчки, а вторая — обеспечение удобства командной работы над кодом.

7) Назовите и дайте краткую характеристику командам 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 имена_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 изменения и все изменённые файлы: `git commit -am 'Описание коммита'` – сохранить добавленные изменения с внесением комментария через встроенный редактор `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 с текущим деревом: `git merge --no-ff имя_ветки`
- удаление ветки: – удаление локальной уже слитой с основным деревом ветки: `git branch -d имя_ветки` – принудительное удаление локальной ветки: `git branch -D имя_ветки` – удаление ветки с центрального репозитория: `git push origin :имя_ветки`

8) Приведите примеры использования при работе с локальным и удалённым репозиториями.

`git push –all (push origin master/любой branch)`

9) Что такое и зачем могут быть нужны ветви (branches)?

Ветвление («ветка», branch) — один из параллельных участков истории в одном хранилище, исходящих из одной версии (точки ветвления). [3] • Обычно есть главная ветка (master), или ствол (trunk). • Между ветками, то есть их концами, возможно слияние. Используются для разработки новых функций.

10) Как и зачем можно игнорировать некоторые файлы при commit?

Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется 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.

## 4 Выводы

Изучил идеологию и применение средств контроля версий, освоил умения по работе с git.

## **Список литературы**